

AI-Powered Cybersecurity Building Intelligent Defense Systems

Techniques and Applications

```
S sudo pip install  
import  
import torch
```

```
from sklearn  
model.fit(X_train,  
y_train)  
predict(X_test)
```

ElMouatez Karbab

الأمن السيبراني المدعوم بالذكاء الاصطناعي: بناء أنظمة دفاع ذكية، التقنيات والتطبيقات

المعتز بالله قرباب

20 سبتمبر 2026

Version: 0.0.311-82d6c17d

المحتويات

3	1	المفاهيم الأساسية والأدوات الضرورية
5	1	أساسيات الأمن السيبراني المدعوم بالذكاء الاصطناعي
6	1.1	من القواعد القديمة إلى الذكاء الجديد
6	1.1.1	حدود التوقعات
7	2.1.1	ميزة الذكاء الاصطناعي: التحليل السلوكي
7	3.1.1	هرم الألم
	4.1.1	كيف يتعلم الذكاء الاصطناعي: الخاضع للإشراف مقابل غير
8		الخاضع للإشراف
9	2.1	مفاهيم الأمان الأساسية
9	1.2.1	ثالوث CIA
10	2.2.1	الدفاع في العمق
11	3.2.1	سطح الهجوم
11	3.1	فهم الخصم
11	1.3.1	سلسلة الهجوم السيبراني
12	2.3.1	إطار عمل ATT&CK MITRE
14	4.1	البيانات: وقود الذكاء الاصطناعي
14	1.4.1	أنواع البيانات الرئيسية
14	2.4.1	تحدي "البيانات الضخمة"
15	3.4.1	تحديات جودة البيانات والتحضير
17	4.4.1	من النص إلى الأرقام: استخراج الميزات
17	5.4.1	مثال عملي: تحضير البيانات للذكاء الاصطناعي
17	5.1	الذكاء الاصطناعي في العمل: مركز عمليات الأمان الحديث
18	1.5.1	SIEM و SOAR
19	2.5.1	الإيجابيات الكاذبة مقابل السلبيات الكاذبة
20	3.5.1	الذكاء الاصطناعي كمضاعف للقوة وقابلية التفسير (XAI)

25	Cybersecurity for Essentials ML	2
26	Definitions DL and ML, AI,	1.2
26	Artificial Intelligence (AI)	1.1.2
27	Machine Learning (ML)	2.1.2
27	Deep Learning (DL)	3.1.2
27	Paradigms ML Core	2.2
28	Supervised Learning	1.2.2
30	Unsupervised Learning	2.2.2
33	Semi-supervised Learning	3.2.2
35	Reinforcement Learning (RL)	4.2.2
36	Supervised Learning سير عمل	5.2.2
40	Families Model ML Key	3.2
40	Classical Machine Learning	1.3.2
44	Deep Learning أساسيات	2.3.2
46	Feature Engineering	3.3.2
49	تطبيع البيانات وقياسها	4.3.2
50	Representation Learning	5.3.2
52	Security for Metrics Evaluation	4.2
52	ما وراء الدقة البسيطة: مغالطة المعدل الأساسي	1.4.2
52	Classification: 1: مقاييس	2.4.2
53	Regression: 2: مقاييس	3.4.2
54	Clustering: 3: مقاييس	4.4.2
55	قانون التوازن: ضبط العتبة	5.4.2
55	Cybersecurity for ML in Challenges	5.2
55	Imbalanced Data	1.5.2
56	تحدي التشفير ("الظلام الدامس")	2.5.2
57	Concept Drift	3.5.2
58	Adversarial Examples	4.5.2
58	قابلية التفسير (Explainable AI - XAI)	5.5.2
58	Project Hands-on	6.2
59	الخطوة 1: تحميل البيانات	1.6.2
60	الخطوة 2: تقسيم البيانات	2.6.2
60	الخطوة 3: اختيار النموذج و Cross-Validation	3.6.2
61	الخطوة 4: ضبط المعلمات الفائقة	4.6.2
61	الخطوة 5: تدريب النموذج وتقييمه	5.6.2
62	الخطوة 6: أهمية الميزة	6.6.2
62	كود المشروع الكامل	7.6.2

70	2	الذكاء الاصطناعي التطبيقي للأمن السيبراني
72	3	Phishing Email Detector
73	1.3	صياغة المشكلة واكتساب البيانات (Problem Framing and Data)
73		 (Acquisition)
73	1.1.3	مصادر البيانات (Data Sources)
74	2.3	جمع البيانات والمعالجة المسبقة (Data Collection and Preprocess-)
74		 (ing
75	1.2.3	تنظيف النص (Text Cleaning)
75	2.2.3	الترميز والتوحيد القياسي (Tokenization and Standard-)
75		 (ization
75	3.2.3	استخراج الإشارات من رؤوس البريد الإلكتروني (Extract-)
78		 (ing Signals from Email Headers
78	3.3	هندسة الميزات لرسائل البريد الإلكتروني (Feature Engineering for)
78		 (Emails
79	1.3.3	الميزات النصية (Textual Features)
80	2.3.3	ميزات محدد موقع الموارد الموحد (URL Features)
80	3.3.3	ميزات البيانات الوصفية للبريد الإلكتروني (Email Meta-)
81		 (data Features
82	4.3	اختيار النموذج والتدريب (Model Selection and Training)
83	1.4.3	مصنفات التعلم الآلي الكلاسيكية (Classical ML Classifiers)
83	2.4.3	نماذج التعلم العميق البسيطة (Simple Deep Learning Mod-)
83		 (els
85	5.3	التعامل مع مجموعات البيانات غير المتوازنة (Handling Imbalanced)
85		 (Datasets
87	6.3	تقييم النموذج والتكرار (Model Evaluation and Iteration)
88	1.6.3	المقاييس الخاصة بالأمن (Security-Specific Metrics)
88	2.6.3	تفسير النتائج والتكرار (Interpreting Results and Iter-)
89		 (ation
91	7.3	اعتبارات النشر الأساسية (Basic Deployment Considerations)
91	1.7.3	نقاط التكامل (Integration Points)
92	2.7.3	الأداء وقابلية التوسع (Performance and Scalability)
100	4	اكتشاف اختراق الشبكات (Network Intrusion Detection)
101	1.4	المشكلة: اكتشاف اختراقات الشبكة (The Problem: Detecting)
101		 (Network Intrusions

101	1.1.4	أنواع الاختراق الشائعة (Common Intrusion Types)
	2.4	مصادر البيانات والأدوات: ملفات التقاط الحزم وبيانات التدفق (Data)
102		(Sources & Tools: PCAPs & Flow Data)
	1.2.4	التقاط حركة مرور الشبكة الخام (Raw Network Traffic)
103		(Captures - PCAP files)
	2.2.4	بيانات التدفق المهيكلية (Structured Flow Data - Net-)
103		(Flow, IPFIX, Zeek logs)
	3.2.4	مكتبات بايثون لبيانات الشبكة (Python Libraries for Net-)
104		(work Data)
	4.2.4	عرض عملي: تحليل ملف التقاط حزم باستخدام سكايب (Prac-)
104		(tical Demo: Parsing a PCAP File with Scapy)
	3.4	هندسة الميزات لبيانات الشبكة (Feature Engineering for Net-)
105		(work Data)
106	1.3.4	ميزات على مستوى الحزمة (Packet Level Features)
106	2.3.4	ميزات على مستوى التدفق (Flow Level Features)
	3.3.4	التحليل الخوارزمي: توليد ميزات على مستوى التدفق (Algorithm-)
107		(mic Breakdown: Generating Flow Level Features)
	4.4	التقنيات: الاكتشاف الخاضع للإشراف وغير الخاضع للإشراف (Tech-)
109		(niques: Supervised & Unsupervised Detection)
109	1.4.4	التصنيف الخاضع للإشراف (Supervised Classification)
	2.4.4	اكتشاف الشذوذ غير الخاضع للإشراف (Unsupervised Anomaly)
110		(Detection)
	5.4	مجموعات البيانات المفتوحة لاكتشاف الشبكات (Open Datasets for)
110		(Network ID)
	6.4	مشروع عملي: التعلم الآلي الكلاسيكي لنظام اكتشاف اختراق الشبكات
111		(Hands on Project: Classical ML for NIDS)
111	1.6.4	تحميل البيانات (Data Loading)
	2.6.4	توليد الميزات (مفاهيمي) (Feature Generation - Con-)
112		(ceptual)
112	3.6.4	تقسيم البيانات وقياسها (Data Splitting & Scaling)
112	4.6.4	تدريب النموذج (Model Training)
113	5.6.4	التقييم (Evaluation)
113	6.6.4	تحليل أهمية الميزات (Feature Importance Analysis)
121	5	التعلم العميق لكشف اختراق الشبكات
122	1.5	معماريات التعلم العميق

122	1.1.5	الشبكات العصبونية الالتفافية للبيانات الأولية
122	2.1.5	RNNs و LSTMs و GRUs
123	3.1.5	نماذج المحولات للتبعيات طويلة المدى
123	4.1.5	إعداد بيانات السلاسل الزمنية
124	2.5	مجموعات البيانات المفتوحة لـ NIDS Deep
124	1.2.5	مجموعة بيانات CTU-13
125	2.2.5	مجموعات بيانات أخرى
125	3.5	مشروع عملي
125	1.3.5	إعداد البيانات
125	2.3.5	تعريف نموذج PyTorch
126	3.3.5	التدريب باستخدام PyTorch
126	4.3.5	التقييم والتفسير

6 التعلم العميق لتحليل البرمجيات الخبيثة 134

135	1.6	المشكلة: تصنيف البرمجيات الخبيثة الجديدة وغير المعروفة
136	2.6	تقنيات التحليل الثابت
136	1.2.6	التصنيف القائم على الصور باستخدام الشبكات العصبونية الالتفافية
137	2.2.6	التحليل القائم على التسلسل
	3.2.6	التحليل القائم على الرسوم البيانية (مقدمة موجزة عن الشبكات العصبونية البيانية)
137		
138	3.6	تقنيات التحليل الديناميكي
138	1.3.6	استخراج الميزات السلوكية
139	2.3.6	نمذجة السلاسل الزمنية/التسلسل
140	3.3.6	التحليل الديناميكي القائم على الرسوم البيانية
140	4.6	مجموعات البيانات المفتوحة لتحليل البرمجيات الخبيثة
141	5.6	مشروع عملي: التعلم العميق لتصنيف البرمجيات الخبيثة
141	1.5.6	الخطوة 1: المعالجة المسبقة للبيانات
141	2.5.6	الخطوة 2: مجموعة بيانات PyTorch ومحمل البيانات
141	3.5.6	الخطوة 3: تعريف نموذج الشبكة العصبونية الالتفافية
141	4.5.6	الخطوة 4: التدريب والتقييم

7 الذكاء الاصطناعي لأمن شبكات المركبات 151

152	1.7	المشكلة: تأمين الشبكات داخل المركبات
152	1.1.7	ناقل Area (Controller CAN Network)
152	2.1.7	إيثرنت السيارات (Automotive Ethernet)
153	2.7	مصادر البيانات: بيانات القياس عن بعد داخل المركبة
153	3.7	التقنيات: تعلم الآلة والتعلم العميق لأمن المركبات

153	التعلم الخاضع للإشراف (Supervised Learning)	1.3.7
	الكشف عن الشذوذ غير الخاضع للإشراف (Unsupervised)	2.3.7
154	Anomaly Detection)	
154	نماذج التسلسل العميق (Deep Sequence Models)	3.3.7
154	اعتبارات النماذج خفيفة الوزن (TinyML)	4.3.7
155	استخراج الميزات من رسائل CAN	5.3.7
156	مشروع عملي: نظام كشف التسلسل لنقل CAN	4.7
156	إعداد البيانات	1.4.7
156	اختيار النموذج وتدريبه (الكشف عن الشذوذ)	2.4.7
156	تحديد العتبة والتنبيه	3.4.7

8 الذكاء الاصطناعي لأمن المستشعرات في الأنظمة الفيزيائية السيبرانية

164	حماية المستشعرات في الأنظمة الفيزيائية السيبرانية	1.8
165	الأنظمة الفيزيائية السيبرانية	1.1.8
165	الهجمات الشائعة	2.1.8
166	مصادر البيانات: بيانات المستشعرات	2.8
166	بيانات المستشعرات المختلفة (خام ومعالجة)	1.2.8
166	سجلات NMEA GPS	2.2.8
167	قراءات وحدة القياس بالقصور الذاتي (IMU)	3.2.8
167	آثار الهجوم المحاكاة	4.2.8
167	مجموعات البيانات المفتوحة لأمن الأنظمة الفيزيائية السيبرانية	3.8
168	التقنيات: التعلم الآلي والتعلم العميق لسلامة بيانات المستشعرات	4.8
168	كشف الشذوذ بدمج المستشعرات	1.4.8
169	الدمج العميق متعدد الوسائط	2.4.8
169	كشف القيم المتطرفة (الأساليب الإحصائية)	3.4.8
170	نماذج التحقق المتبادل	4.4.8
171	مشروع عملي: مراقب سلامة بيانات المستشعرات	5.8
171	الخطوة 1: إعداد البيانات	1.5.8
171	الخطوة 2: اختيار وبناء النموذج	2.5.8
171	الخطوة 3: تدريب النموذج	3.5.8
172	الخطوة 4: كشف الشذوذ والتقييم	4.5.8
174	الخطوة 5: التصور	5.5.8

9 المحولات لأمن البرمجة اللغوية العصبية المتقدم

181	المشكلة والتقنية	1.9
181	معالجة اللغات الطبيعية ونماذج اللغة الكبيرة	1.1.9

181	2.1.9	تضمنين النصوص: من الثابت إلى المدرك للسياق
182	3.1.9	صقل نماذج المحولات
183	2.9	التطبيقات في الأمن السيبراني
183	1.2.9	كشف التصيد والهندسة الاجتماعية
184	2.2.9	أتمتة استخبارات التهديدات
184	3.2.9	تحليل وصف الثغرات الأمنية
185	3.9	مجموعات البيانات المفتوحة لمعالجة اللغات الطبيعية في الأمن
185	4.9	مشروع عملي: محول لمعالجة اللغات الطبيعية في الأمن
186	1.4.9	الخطوة 1: إعداد البيانات والترميز
186	2.4.9	الخطوة 2: تحميل النموذج والصقل
186	3.4.9	الخطوة 3: التقييم
186	4.4.9	استكشاف مفاهيمي: هندسة التلقين

10 نماذج التسلسل لكشف الشذوذ الزمني

196	1.10	المشكلة: تحديد الأنماط المشبوهة عبر الزمن
196	1.1.10	التهديدات الداخلية Insider Threats
196	2.1.10	الحسابات المخترقة Compromised Accounts
197	2.10	التقنية: نماذج التسلسل المتقدمة للسلاسل الزمنية Time-Series
197	1.2.10	الشبكات العصبونية المتكررة (GRUs و LSTMs)
198	2.2.10	الشبكات العصبونية الالتفافية الزمنية (TCNs)
198	3.2.10	نماذج التسلسل القائمة على المحولات Transformer
	4.2.10	التفصيل الخوارزمي: تحويل سجلات الأمان إلى جلسات Ses-
198		sionizing
200	3.10	التطبيقات
200	1.3.10	تحليلات سلوك المستخدم (UBA)
200	2.3.10	الكشف عن نشاط التهديدات المتقدمة المستمرة (APT)
201	3.3.10	الكشف عن حقن العمليات Process Injection
201	4.3.10	الكشف عن القيادة والتحكم (C2)
201	4.10	مجموعات البيانات المفتوحة لتحليلات سلوك المستخدم UBA/التحليل الزمني
202	5.10	مشروع عملي: نماذج التسلسل لتحليلات سلوك المستخدم UBA
202	1.5.10	إعداد البيانات
203	2.5.10	تعريف نموذج LSTM
203	3.5.10	التدريب والكشف عن الشذوذ
203	4.5.10	إدارة انحراف المفهوم Concept Drift
204	5.5.10	التصوير المرئي

11 الطرق غير الخاضعة للإشراف لاكتشاف هجمات يوم الصفر 213

- 1.11 المشكلة: اكتشاف التهديدات غير المعروفة 214
- 2.11 مجموعة من الخوارزميات غير الخاضعة للإشراف 214
- 1.2.11 آلة متجه الدعم أحادية الفئة (OCSVM) 214
- 2.2.11 غابة العزل (Isolation Forest) 215
- 3.2.11 خوارزميات التجميع (Clustering Algorithms) 215
- 4.2.11 خوارزميات كشف الحالات الشاذة الكلاسيكية 216
- 5.2.11 المرز التلقائي (Autoencoder) لاكتشاف الحالات الشاذة 217
- 3.11 التطبيقات 218
- 4.11 مجموعات البيانات المفتوحة للكشف غير الخاضع للإشراف 219
- 5.11 مشروع عملي: كشف الحالات الشاذة غير الخاضع للإشراف 220
- 1.5.11 إعداد البيانات 220
- 2.5.11 تطبيق نماذج متعددة غير خاضعة للإشراف 221
- 3.5.11 دمج درجات الشذوذ 222
- 4.5.11 تحديد عتبات فعالة والتقييم 223
- 5.5.11 استراتيجيات لتقليل الإنذارات الكاذبة 223

12 GNNs لتحليل العلاقات 232

- 1.12 المشكلة: العلاقات المعقدة في الهجمات 233
- 2.12 التقنية: الشبكات العصبونية البيانية للأمن 233
- 1.2.12 الشبكات العصبونية البيانية (GNNs) 234
- 2.2.12 أنواع الشبكات العصبونية البيانية 234
- 3.2.12 الشبكات العصبونية للرسوم البيانية الفائقة (HNNs) 235
- 4.2.12 بناء الرسوم البيانية من البيانات الأمنية 235
- 3.12 التطبيقات 237
- 4.12 مجموعات البيانات المفتوحة لشبكات GNNs في الأمن 238
- 5.12 مشروع عملي: شبكات GNNs للرسوم البيانية الأمنية 239
- 1.5.12 الخطوة 1: بناء الرسم البياني 239
- 2.5.12 الخطوة 2: إعداد بيانات Geometric PyTorch 239
- 3.5.12 الخطوة 3: تعريف نموذج GNN 239
- 4.5.12 الخطوة 4: التدريب وكشف الشذوذ 241
- 5.5.12 الخطوة 5: التفسير 241

13 النماذج الاحتمالية لإدارة الثغرات 250

- 1.13 مشكلة التنبؤ بمخاطر الثغرات 251
- 2.13 التقنية: النماذج الاحتمالية وتعلم الآلة البايزي 251

251	1.2.13	النماذج الاحتمالية
252	2.2.13	الشبكات البايزية
252	3.2.13	تقنيات تعلم الآلة البايزي
253	4.2.13	مكتبات بايثون
254	3.13	التطبيقات
254	1.3.13	تحديد أولويات الثغرات
255	2.3.13	التنبؤ بثغرات سلسلة توريد البرمجيات
255	3.3.13	التسجيل الديناميكي للمخاطر للحوادث والأصول
255	4.13	مجموعات البيانات المفتوحة لإدارة الثغرات
256	5.13	مشروع عملي: تحديد أولويات المخاطر الاحتمالية
256	1.5.13	إعداد البيانات
256	2.5.13	بناء شبكة بايزية
257	3.5.13	عرض توضيحي لتحديد الأولويات
258	4.5.13	تحديد كمية عدم اليقين

3 عمليات الذكاء الاصطناعي في الأمن السيبراني 265

267	14	من النموذج الأولي إلى الإنتاج
268	1.14	1.1.14 تفعيل تعلم الآلة في الأمن
268	1.1.14	1.1.14 الانحراف المستمر للمفهوم (Continuous Drift) Concept
268	2.1.14	2.1.14 القدرة المنخفضة على تحمل الأخطاء (مخاطر عالية)
269	3.1.14	3.1.14 الهجمات العدائية على الذكاء الاصطناعي
269	4.1.14	4.1.14 الأداء في الوقت الفعلي
269	5.1.14	5.1.14 ندرة البيانات وعدم توازنها
269	6.1.14	6.1.14 الحاجة إلى قابلية التفسير
270	2.14	2.14 دورة حياة عمليات تعلم الآلة (MLOps)
270	1.2.14	1.2.14 عمليات تعلم الآلة (MLOps)
270	2.2.14	2.2.14 المكونات الرئيسية لعمليات تعلم الآلة (MLOps)
272	3.14	3.14 مراقبة النماذج والتنبيه
273	1.3.14	1.3.14 اكتشاف انحراف النموذج
273	2.3.14	2.3.14 مشكلات جودة البيانات
273	3.3.14	3.3.14 أنظمة التنبيه الآلية
274	4.14	4.14 استراتيجيات النشر
274	1.4.14	1.4.14 الحاويات (Containerization)

275	اختبار A/B	2.4.14
275	عمليات النشر الكاري (Canary Deployments)	3.4.14
275	عمليات النشر الزرقاء/الخضراء (Blue/Green Deployments)	4.4.14
276	استراتيجيات موثوقة للتراجع عن النماذج	5.4.14
276	قابلية المراقبة لأنظمة تعلم الآلة	5.14
276	التسجيل (Logging)	1.5.14
276	التتبع (Tracing)	2.5.14
277	المقاييس (Metrics)	3.5.14
277	ممارسات عمليات تعلم الآلة الآمنة	6.14
277	مصدر البيانات وسلامتها	1.6.14
277	تأمين خطوط أنابيب البيانات	2.6.14
277	توقيع النماذج	3.6.14
278	النشر الآمن للنماذج	4.6.14
278	الدفاعات ضد هجمات تسميم بيانات التدريب	5.6.14
278	التحكم في الوصول للبنية التحتية لتعلم الآلة	6.6.14
278	أمان سلسلة التوريد لتعلم الآلة	7.6.14
279	مشروع عملي: خط أنابيب MLOps أساسي	7.14
279	الخطوة 1: إعداد المشروع وإصدار البيانات باستخدام DVC	1.7.14
279	الخطوة 2: تتبع التجارب باستخدام MLflow	2.7.14
280	الخطوة 3: سجل النماذج	3.7.14
280	الخطوة 4: النشر الأساسي (مفاهيمي)	4.7.14
281	الخطوة 5: التدريب المستمر (مفاهيمي)	5.7.14

15 تقييم النموذج والتحقق المستمر

287	تقنيات التحقق المتقدمة	1.15
288	الاختبار الكاري	1.1.15
288	اختبار أ/ب	2.1.15
288	الاختبار الرجعي بالبيانات التاريخية	3.1.15
289	مراقبة الأداء في الوقت الفعلي مقابل الحقائق المؤكدة	4.1.15
289	اكتشاف انحراف المفهوم والتكيف معه	2.15
289	استراتيجيات أعمق لاكتشاف الانحراف	1.2.15
290	مشغلات إعادة التدريب الآلية والتعلم التكيفي	2.2.15
291	التحقق بوجود الإنسان في الحلقة	3.15
291	الإنسان في الحلقة وآليات التغذية الراجعة	1.3.15

292	2.3.15	التعلم النشط
292	4.15	الذكاء الاصطناعي القابل للتفسير للأمن
292	1.4.15	تقنيات للسياق الأمني
293	5.15	التكامل مع الأدوات الأمنية
293	1.5.15	تكمّل SIEM
295	2.5.15	تكمّل كتيبات التشغيل SOAR
295	6.15	مشروع عملي: اكتشاف الانحراف و XAI
296	1.6.15	محاكاة انحراف المفهوم
296	2.6.15	تنفيذ اكتشاف الانحراف
297	3.6.15	تطبيق XAI على التصنيفات الخاطئة
297	4.6.15	تكمّل SIEM/SOAR المفاهيمي

16 الخصوصية والأخلاقيات والحكمة في الذكاء الاصطناعي 304

304	1.16	إدارة المخاطر الأخلاقية
305	1.1.16	الآثار المترتبة على الاستخدام المزدوج
305	2.1.16	مخاوف الخصوصية
306	3.1.16	التحيز الخوارزمي
307	2.16	الأطر السياسية والقانونية
307	1.2.16	التنقل بين متطلبات الامتثال
307	2.2.16	اللائحة العامة لحماية البيانات (GDPR)
308	3.2.16	قانون خصوصية المستهلك في كاليفورنيا (CCPA)
308	4.2.16	توجيه أمن الشبكات والمعلومات 2 (NIS2)
308	5.2.16	قانون الذكاء الاصطناعي (الاتحاد الأوروبي)
309	6.2.16	الآثار القانونية لاتخاذ القرارات الآلية
309	3.16	أطر الحكمة
309	1.3.16	مبادئ الذكاء الاصطناعي المسؤول في الأمن السيبراني
310	2.3.16	تطوير "بطاقات التماذج"
310	3.3.16	"صحائف البيانات لمجموعات البيانات"
311	4.3.16	إجراء "تقييمات تأثير الذكاء الاصطناعي"
311	4.16	الاختبار الأحمر المسؤول للذكاء الاصطناعي
311	1.4.16	التحديد الاستباقي
311	2.4.16	تخطيط السيناريوهات
312	3.4.16	تحليل الأضرار

318	17 الاتجاهات المستقبلية والمواضيع المتقدمة
318	1.17 التعلم الفيدرالي للدفاع التعاوني
319	1.1.17 مفاهيم التعلم الفيدرالي
319	2.1.17 تطبيقات في الأمن السيبراني
320	3.1.17 التجميع الآمن والمتانة
320	4.1.17 عرض مفاهيمي: الكشف الفيدرالي عن الشذوذ
321	2.17 التعلم الآلي الخصومي
321	1.2.17 نظرة أعمق على الهجمات الخصومية
322	2.2.17 الدفاعات ضد الهجمات الخصومية
323	3.2.17 تحليل خوارزمي: طريقة إشارة التدرج السريعة
324	3.17 الذكاء الاصطناعي الآمن كميًا وتشفير ما بعد الكوانتم
324	4.17 وكلاء الأمن المستقلون
325	5.17 الذكاء الاصطناعي لتطوير البرمجيات الآمنة

```latex

## القسم 1

# المفاهيم الأساسية والأدوات الضرورية

يضع هذا الجزء الأساس الحاسم. سنقوم بإزالة الغموض عن التقاطع المثير بين الذكاء الاصطناعي AI والأمن السيبراني cybersecurity، ونقدم المفاهيم الأساسية من كلا المجالين بلغة واضحة، ونزودك بمجموعة أدوات تعلم الآلة machine learning الأساسية اللازمة لجميع المشاريع القادمة. ``

## باب 1

# أساسيات الأمن السيبراني المدعوم بالذكاء الاصطناعي

## أهداف التعلم

بنهاية هذا الفصل، ستكون قادراً على:

- شرح قيود الأمان القائم على التوقعات باستخدام "هرم الألم" Pyramid of Pain.
- مقارنة التعلم الخاضع للإشراف Supervised وغير الخاضع للإشراف Unsupervised في سياق أمني.
- تطبيق إطار سلسلة الهجوم السيبراني Cyber Kill Chain لتحليل الاختراقات في العالم الحقيقي.
- تقييم تحديات جودة البيانات وضرورة استخراج الميزات feature extraction.
- تقييم أداء نماذج الذكاء الاصطناعي باستخدام مصفوفة الارتباك Confusion Matrix وأنواع الأخطاء.

## مقدمة

عالم الأمن السيبراني في حالة تغير مستمر. في الماضي، كانت أنظمة الأمان تتبع قواعد صارمة، تشبه حارساً يتحقق من قائمة أسماء معتمدة عند الباب. إذا كان اسمك في القائمة، تدخل. وإلا،

يتم منعه.

اليوم، المهاجمون أسرع وأذكى ويستخدمون أدوات آلية لاختراق الأنظمة. إنهم لا يترقبون الباب فحسب، بل يسرقون المفاتيح، وينسخون بصمات الأصابع، ويدخلون من خلال النوافذ المفتوحة. لهذا السبب، لم تعد الدفاعات التقليدية كافية. يجب أن تنتقل من الأمان البسيط القائم على القواعد إلى الدفاع الذكي المدعوم بالذكاء الاصطناعي AI-powered defense. ولكن، لبناء هذه الأنظمة الذكية، يجب أن نفهم الأساسيات أولاً. لا يكون نموذج الذكاء الاصطناعي AI model جيداً إلا بمقدار جودة البيانات التي يتعلم منها. إذا لم تفهم كيف يحدث هجوم سيبراني cyber attack أو كيف تبدو بيانات الأمان، فلا يمكنك بناء أداة ذكاء اصطناعي فعالة. يجمع هذا الفصل بين المبادئ الأساسية للأمن السيبراني والواقع الحديث للذكاء الاصطناعي. إنه الجسر بين عالم "الأمان" Security وعالم "علم البيانات" Data Science.

## 1.1 من القواعد القديمة إلى الذكاء الجديد

لسنوات عديدة، اعتمد الأمن السيبراني على مبدأ بسيط: تحديد ما هو سيئ ومنعه. يُعرف هذا النهج باسم الكشف القائم على التوقيعات signature-based detection.

### 1.1.1 حدود التوقيعات

تعمل الأدوات التقليدية، مثل برامج مكافحة الفيروسات القديمة، من خلال البحث عن التوقيعات signatures. التوقيع هو نمط فريد، مثل بصمة رقمية. عند إنشاء ملف، يمكننا حساب "الهاش" Hash الخاص به (على سبيل المثال، MD5 أو SHA-256). يعمل هذا الهاش كمعرف فريد لذلك الملف. كيف يعمل:

- (1) تجد شركة الأمان فيروساً.
- (2) يقومون بحساب توقيع الفيروس (الهاش).
- (3) يضيفون هذا التوقيع إلى قاعدة بيانات.
- (4) يقوم برنامج مكافحة الفيروسات بفحص جهاز الكمبيوتر الخاص بك. إذا وجد ملفاً بنفس التوقيع بالضبط، فإنه يحذفه.

**المشكلة:** هذه الطريقة تفاعلية بحتة. إنها توقف فقط التهديدات التي شوهدت من قبل. المهاجمون يعرفون ذلك. يستخدمون البرمجيات الخبيثة متعددة الأشكال -Polymorphic Mal-ware وهي برمجيات خبيثة تغير شفرتها الخاصة قليلاً في كل مرة تنتشر فيها. هذا يغير الهاش، مما يغير "البصمة" بشكل فعال. بالنسبة لبرنامج مكافحة الفيروسات التقليدي، يبدو وكأنه ملف جديد وآمن تماماً.

### 2.1.1 ميزة الذكاء الاصطناعي: التحليل السلوكي

يحل الذكاء الاصطناعي (AI) والتعلم الآلي -Machine Learning (ML) هذه المشكلة عن طريق تغيير النهج. بدلاً من حفظ التوقعات "السيئة"، يتعلم الذكاء الاصطناعي السلوك Behavior. يراقب نظام الذكاء الاصطناعي الشبكة لأسابيع لإنشاء خط أساس Baseline. خط الأساس هو حساب لما يبدو عليه النشاط "الطبيعي".

• طبيعي: يسجل المستخدم "أ" الدخول عادةً في الساعة 9:00 صباحاً من لندن ويصل إلى قاعدة بيانات الموارد البشرية.

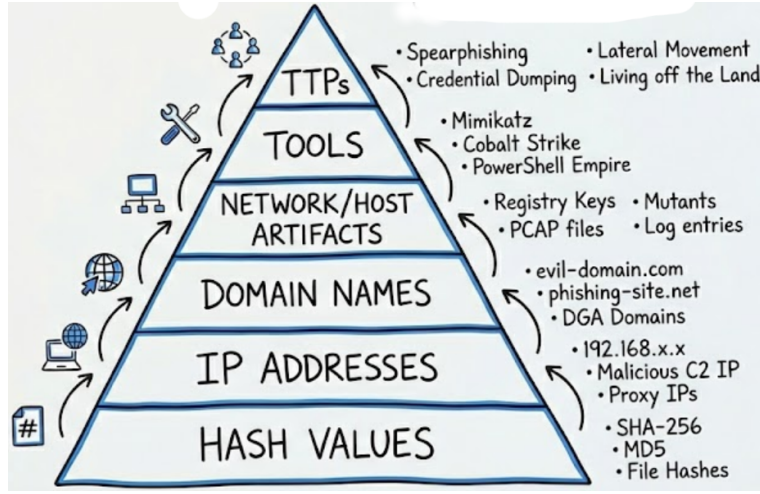
• شاذ: يسجل المستخدم "أ" الدخول فجأة في الساعة 3:00 صباحاً من بلد مختلف ويحاول الوصول إلى قاعدة بيانات المالية.

يكشف الذكاء الاصطناعي هذا التهديد ليس لأنه يمتلك "توقعاً" له، ولكن لأن السلوك ينحرف عن خط الأساس. هذا يسمح للذكاء الاصطناعي باكتشاف هجمات اليوم-صفر Zero-Day Attacks وهي هجمات تستغل ثغرات أمنية لا يعرفها أحد بعد.

### 3.1.1 هرم الألم

لفهم سبب حاجتنا إلى الذكاء الاصطناعي، ننظر إلى هرم الألم Pyramid of Pain، وهو مفهوم أنشأه خبير الأمن ديفيد بيانكو. يصنف المؤشرات بناءً على مقدار الألم الذي تسببه للمهاجم إذا قننا بحظرها.

• القاعدة (قيم الهاش وعناوين IP) من السهل على المهاجمين تغييرها. إذا حظرت عنوان IP، فإنهم ببساطة يشترون واحداً جديداً. هذا لا يسبب لهم أي ألم.



شكل 1.1: هرم الألم. يركز الذكاء الاصطناعي على القمة (السلوك)، بينما تركز الأدوات التقليدية على القاعدة (الهياكل).

- القمة (TTPs) - السلوك: هذه هي الطريقة التي يعمل بها المهاجم (على سبيل المثال، "تخمين كلمات المرور بسرعة"). من الصعب جداً على المهاجم أن يتخلى عن مهاراته وأساليبه. إذا اكتشفنا سلوكهم، فإننا نسبب لهم أقصى درجات الألم.
- الذكاء الاصطناعي قوي لأنه يعمل في قمة الهرم.

#### 4.1.1 كيف يتعلم الذكاء الاصطناعي: الخاضع للإشراف مقابل غير الخاضع للإشراف

لا يتعلم كل الذكاء الاصطناعي بنفس الطريقة. في الأمن السيبراني، نستخدم بشكل أساسي طريقتين:

(1) التعلم الخاضع للإشراف (التعلم بالمثل): Supervised Learning (Learning by Example)

- المفهوم: تخيل أنك تعلم طفلاً التعرف على التفاح. تعرض له فاكهة حمراء وتقول "تفاحة". تعرض له فاكهة صفراء وتقول "موزة".
- في الأمن: نقوم بتغذية الكمبيوتر ببيانات مصنفة: "هذا الملف هو برمجية خبيثة" و "هذا الملف آمن". يتعلم الذكاء الاصطناعي الفرق.



- الأفضل لـ: كشف البرمجيات الخبيثة وتصنيف الهجمات المعروفة.
- (2) التعلم غير الخاضع للإشراف (التعلم بالملاحظة): (Unsupervised Learning (Learn-ing by Observation)

- المفهوم: تخيل أنك تترك طفلاً في غابة. لا تعلمه أي شيء، لكنه في النهاية يلاحظ أن السناجب تتصرف بشكل مختلف عن الصخور.
- في الأمن: نقوم بتغذية الكمبيوتر بحركة مرور الشبكة الخام بدون تسميات. يقوم الذكاء الاصطناعي بتجميع الأنشطة المتشابهة معاً ويسلط الضوء على القيم الشاذة.
- الأفضل لـ: التهديدات الداخلية والبحث عن هجمات جديدة وغير معروفة.

## 2.1 مفاهيم الأمان الأساسية

قبل تطبيق الذكاء الاصطناعي، يجب أن نفهم قواعد اللعبة. يستخدم محترفو الأمن نماذج محددة لاتخاذ القرارات.

### 1.2.1 ثالوث CIA

ثالوث CIA Triad هو النموذج الأساسي لأمن المعلومات. كل ضابط أمني، وكل نموذج ذكاء اصطناعي تبنيه، سيحاول حماية أحد هذه الأهداف الثلاثة.

(1) السرية: Confidentiality ضمان الحفاظ على سرية البيانات. يجب أن يرى الأشخاص المصرح لهم فقط المعلومات الحساسة.

- مثال على هجوم: اختراق بيانات حيث تُسرق كلمات المرور.
- دفاع الذكاء الاصطناعي: كشف عمليات نقل البيانات غير المصرح بها (التسريب (Exfiltration).

(2) النزاهة: Integrity ضمان دقة البيانات وموثوقيتها. لا ينبغي لأحد أن يكون قادراً على تغيير أو حذف البيانات دون إذن.

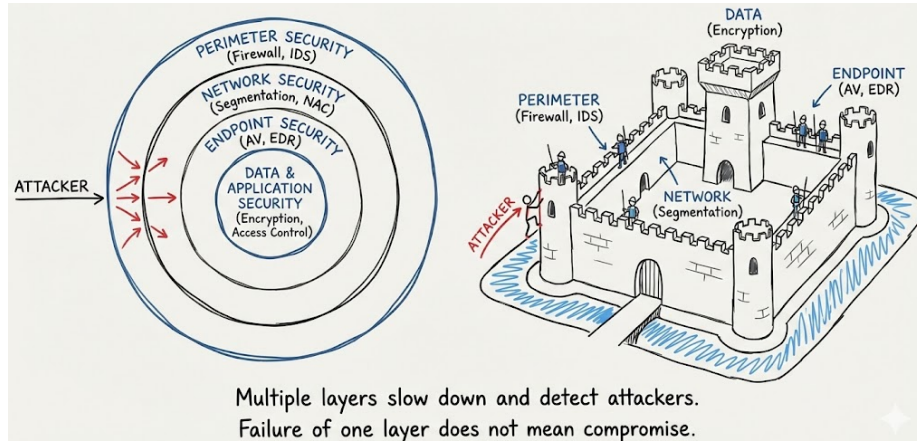
- مثال على هجوم: مهاجم يعدل معاملة بنكية من 10 دولارات إلى 10,000 دولار.
- دفاع الذكاء الاصطناعي: كشف التغييرات غير المصرح بها في ملفات النظام.

(3) التوافر: Availability ضمان عمل الأنظمة عند الحاجة.

- مثال على هجوم: برامج الفدية Ransomware التي تقفل أجهزة الكمبيوتر، أو هجوم حجب الخدمة الموزع DDoS الذي يعطل موقعاً إلكترونياً.
- دفاع الذكاء الاصطناعي: التنبؤ بفشل النظام أو تصفية تدفقات حركة المرور الخبيثة.

## 2.2.1 الدفاع في العمق

أداة أمان واحدة لا تكفي أبداً. يتبع الأمن مبدأ الدفاع في العمق Defense in Depth، المعروف أيضاً باسم "النهج متعدد الطبقات" Layered Approach. تخيل قلعة: لديها خندق، وجدار عال، وبوابة ثقيلة، وحراس في الداخل. إذا عبر المهاجم الخندق، يوقفه الجدار (انظر الشكل 2.1).



شكل 2.1: الدفاع في العمق ("البصلة" أو "القلعة")

في الأمن السيرياني، لدينا:

- (1) المحيط: Perimeter جدران الحماية Firewalls (توقف حركة مرور الشبكة).
- (2) نقطة النهاية: Endpoint برامج مكافحة الفيروسات Antivirus (توقف البرمجيات الخبيثة على أجهزة الكمبيوتر المحمولة).
- (3) البيانات: Data التشفير Encryption (يحمي الملفات في حالة سرقتها).

دور الذكاء الاصطناعي: الذكاء الاصطناعي ليس درعاً سحرياً يحل محل هذه الطبقات. بدلاً من ذلك، يقوم الذكاء الاصطناعي بمراقبة كل هذه الطبقات في وقت واحد لربط النقاط.

### 3.2.1 سطح الهجوم

سطح الهجوم attack surface هو مجموع كل النقاط التي يمكن للمهاجم أن يحاول الدخول من خلالها إلى نظامك.

• سطح الشبكة: Network Surface المنافذ المفتوحة على الخوادم، واتصالات الواي فاي، وتكوينات السحابة.

• السطح البشري: Human Surface غالباً ما يكون الموظفون الحلقة الأضعف. يستخدم المهاجمون الهندسة الاجتماعية Social Engineering (مثل رسائل التصيد الاحتيالي Phishing) لخداع الناس للكشف عن كلمات المرور.

• سطح البرمجيات: Software Surface كل تطبيق تقوم بتنصيبه قد يحتوي على أخطاء (ثغرات) يمكن للمهاجمين استخدامها.

### 3.1 فهم الخصم

لبناء دفاع جيد، يجب أن تفكر مثل المهاجم. المهاجمون الحديثون ليسوا مجرد قراصنة منعزلين؛ غالباً ما يكونون مجموعات منظمة تسمى التهديدات المتقدمة المستمرة Advanced Persistent Threats (APTs). تبقى هذه المجموعات في الشبكة لفترة طويلة (أشهر أو سنوات) لسرقة البيانات.

#### 1.3.1 سلسلة الهجوم السيبراني

طورتها شركة لوكهيد مارتين، يصف هذا النموذج الهجوم السيبراني كعملية خطية خطوة بخطوة.

(1) الاستطلاع: Reconnaissance البحث عن الهدف (إيجاد عناوين البريد الإلكتروني، فحص المنافذ المفتوحة).

(2) التسليح: Weaponization إنشاء ملف خبيث (على سبيل المثال، ملف PDF بداخله فيروس).

(3) التسليم: Delivery إرسال السلاح (على سبيل المثال، إرسال ملف PDF عبر البريد الإلكتروني).

(4) الاستغلال: Exploitation يعمل الفيروس ويستغل ضعفاً في البرمجيات.

(5) التنصيب: Installation يقوم المهاجم بتنصيب "باب خلفي" backdoor لضمان قدرته على العودة لاحقاً.

(6) القيادة والتحكم: (C2) (C2) Command and Control يتصل الكمبيوتر المصاب بخادم المهاجم لتلقي التعليمات.

(7) الإجراءات على الأهداف: Actions on Objectives سرقة البيانات، تدمير الأنظمة، أو تشفير الملفات للحصول على فدية.

إذا كسر المدافعون السلسلة في أي خطوة، يفشل الهجوم.

#### دراسة حالة: اختراق شركة تارجت (2013)

لفهم سلسلة الهجوم Kill Chain، دعونا نلقي نظرة على الاختراق الشهير لشركة التجزئة "تارجت".

- الاستطلاع: Reconnaissance وجد المهاجمون أن تارجت تستخدم مورداً خارجياً لأنظمة التدفئة وتكييف الهواء (HVAC) يدعى Fazio Mechani-cal Services.

- التسليم: Delivery أرسل المهاجمون بريداً إلكترونياً تصيداً phishing email يحتوي على برمجيات خبيثة malware إلى مورد HVAC، وليس إلى تارجت مباشرة.

- الاستغلال: Exploitation سرقت البرمجيات الخبيثة بيانات اعتماد تسجيل الدخول الخاصة بالمورد.

- الحركة الجانبية: Lateral Movement استخدم المهاجمون بيانات اعتماد المورد لتسجيل الدخول إلى بوابة موردي تارجت ثم انتقلوا إلى الشبكة الرئيسية.

- الإجراءات على الأهداف: Action on Objectives قاموا بتثبيت برمجيات خبيثة على أجهزة نقاط البيع (POS) Point-of-Sale وسرقوا 40 مليون رقم بطاقة أئتمان.

الدرس: نادراً ما يهاجم المهاجمون الباب الأمامي مباشرة. إنهم يتبعون سلسلة. يمكن للذكاء الاصطناعي اكتشاف الحركة غير العادية من "بوابة الموردين" إلى "الشبكة الرئيسية".

### 2.3.1 إطار عمل MITRE ATT&CK

بينما أن سلسلة الهجوم Kill Chain خطية، فإن إطار عمل MITRE ATT&CK أكثر تفصيلاً. يرمز إلى التكتيكات والتقنيات والمعرفة العامة للخصوم, *Adversarial Tactics*.

*Techniques, and Common Knowledge*. يعمل مثل "جدول دوري" لسلوك القرصنة (انظر الشكل 3.1).

| Reconnaissance                         | Resource Development          | Initial Access                    | Execution                              | Persistence                             | Privilege Escalation                    | Defense Evasion                         | Credential Access                     | Discovery                      | Lateral Movement                | Collection                  | Command and Control                   |
|----------------------------------------|-------------------------------|-----------------------------------|----------------------------------------|-----------------------------------------|-----------------------------------------|-----------------------------------------|---------------------------------------|--------------------------------|---------------------------------|-----------------------------|---------------------------------------|
| 11 techniques                          | 8 techniques                  | 11 techniques                     | 17 techniques                          | 23 techniques                           | 14 techniques                           | 47 techniques                           | 17 techniques                         | 34 techniques                  | 9 techniques                    | 17 techniques               | 18 techniques                         |
| Active Scanning (3)                    | Acquire Access                | Content Injection                 | Cloud Administration Command           | Account Manipulation (2)                | Abuse Elevation Control Mechanism (4)   | Abuse Elevation Control Mechanism (4)   | Adversary in-the-Middle (4)           | Account Discovery (4)          | Exploitation of Remote Services | Adversary in-the-Middle (4) | Application Layer Protocol (5)        |
| Gather Victim Host Information (4)     | Acquire Infrastructure (8)    | Drive-by Compromise               | Command and Scripting Interpreter (13) | Boots or Logon Autostart Execution (14) | Access Token Manipulation (3)           | Access Token Manipulation (3)           | Brute Force (4)                       | Application Window Discovery   | Internal Spearphishing          | Archive Collected Data (3)  | Communication Through Removable Media |
| Gather Victim Identity Information (3) | Compromise Accounts (3)       | Exploit Public-Facing Application | Container Administration Command       | Boots or Logon Autostart Execution (14) | Boots or Logon Autostart Execution (14) | Boots or Logon Autostart Execution (14) | Credentials from Password Stores (8)  | Cloud Infrastructure Discovery | Lateral Tool Transfer           | Audio Capture               | Content Injection                     |
| Gather Victim Network Information (6)  | Compromise Infrastructure (8) | External Remote Services          | Deploy Container                       | Cloud Application Integration           | Boots or Logon Autostart Execution (14) | Boots or Logon Autostart Execution (14) | Exploitation of Credential for Access | Cloud Service Dashboard        | Remote Session Hijacking (2)    | Automated Collection        | Data Exfiltration (3)                 |
| Gather Victim Org Information (4)      | Develop Capabilities (4)      | Establish Accounts (3)            | Hardware Additions                     | Compromise Host Software Library        | Boots or Logon Autostart Execution (14) | Boots or Logon Autostart Execution (14) | Exploitation of Credential for Access | Cloud Service Dashboard        | Remote Session Hijacking (2)    | Automated Collection        | Data Exfiltration (3)                 |
| Playing for Information (4)            | Obtain Capabilities (7)       | Flushing (4)                      | Exploitation for Client Execution      | Create or Modify System Processes (5)   | Boots or Logon Autostart Execution (14) | Boots or Logon Autostart Execution (14) | Exploitation of Credential for Access | Cloud Service Dashboard        | Remote Session Hijacking (2)    | Automated Collection        | Data Exfiltration (3)                 |
| Search Closed Sources (2)              | Stage Capabilities (8)        | Supply Chain Compromise (3)       | Input Injection                        | Create or Modify System Processes (5)   | Boots or Logon Autostart Execution (14) | Boots or Logon Autostart Execution (14) | Exploitation of Credential for Access | Cloud Service Dashboard        | Remote Session Hijacking (2)    | Automated Collection        | Data Exfiltration (3)                 |
| Search Open Technical Databases (5)    | Trusted Relationship          | Valid Accounts (4)                | Scheduled Task Job (3)                 | Event Triggered Execution (18)          | Event Triggered Execution (18)          | Event Triggered Execution (18)          | Exploitation of Credential for Access | Cloud Service Dashboard        | Remote Session Hijacking (2)    | Automated Collection        | Data Exfiltration (3)                 |
| Search Open Websites/Domains (3)       | Wi-Fi Networks                | Shared Modules                    | Shared Modules                         | Shared Modules                          | Shared Modules                          | Shared Modules                          | Exploitation of Credential for Access | Cloud Service Dashboard        | Remote Session Hijacking (2)    | Automated Collection        | Data Exfiltration (3)                 |
| Search Threat Vendor Data              |                               |                                   |                                        |                                         |                                         |                                         | Exploitation of Credential for Access | Cloud Service Dashboard        | Remote Session Hijacking (2)    | Automated Collection        | Data Exfiltration (3)                 |
| Search Victim-Owned Websites           |                               |                                   |                                        |                                         |                                         |                                         | Exploitation of Credential for Access | Cloud Service Dashboard        | Remote Session Hijacking (2)    | Automated Collection        | Data Exfiltration (3)                 |

شكل 3.1: مصفوفة ATT&CK MITRE

يقسم MITRE الهجمات إلى ثلاثة مستويات:

- (1) التكتيكات (الهدف): Tactics (The Goal) لماذا يفعل المهاجم هذا؟
- (2) التقنيات (الطريقة): Techniques (The Method) كيف يحققون الهدف؟
- (3) الإجراءات (التفاصيل): Procedures (The Specifics) ما هي الأداة المحددة التي استخدموها؟

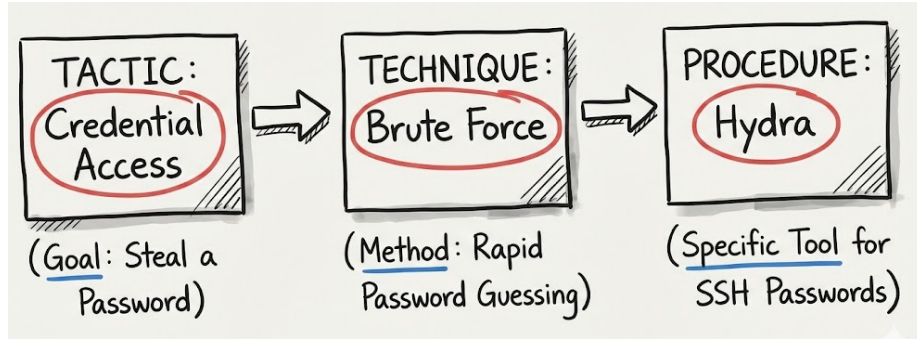
مثال:

• التكتيك: Tactic الوصول إلى بيانات الاعتماد Credential Access (الهدف هو سرقة كلمة مرور).

• التقنية: Technique القوة الغاشمة Brute Force (الطريقة هي تخمين كلمات المرور بسرعة).

• الإجراء: Procedure استخدام أداة تسمى "Hydra" لتخمين كلمات مرور SSH.

بالنسبة لبناء الذكاء الاصطناعي، يعد MITRE ضرورياً. إنه يوفر التسميات لبياناتنا. يمكننا تدريب نموذج ذكاء اصطناعي لتصنيف حدث معين على أنه "قوة غاشمة" أو "تصيد احتيالي". (انظر الشكل 4.1).



شكل 4.1: عرض مبسط لمصفوفة MITRE ATT&CK

## 4.1 البيانات: وقود الذكاء الاصطناعي

تعتمد فعالية أي نظام ذكاء اصطناعي في الأمن السيبراني بشكل مباشر على البيانات التي يستهلكها. البيانات عالية الجودة وذات الصلة والشاملة هي الوقود لبناء أنظمة دفاع ذكية. بدونها، حتى أكثر الخوارزميات تقدماً ستفشل. هذه البيانات، التي يشار إليها غالباً باسم القياس الأمني عن بعد security telemetry، يتم جمعها من مصادر مختلفة عبر البيئة الرقمية للمؤسسة.

### 1.4.1 أنواع البيانات الرئيسية

يتم تدريب نماذج الذكاء الاصطناعي في الأمن السيبراني على مجموعة واسعة من مصادر البيانات. لفهمها بسرعة، راجع الجدول 1.1 أدناه.

### 2.4.1 تحدي "البيانات الضخمة"

يمثل العمل مع بيانات الأمان العديد من التحديات الفريدة، والتي توصف غالباً بـ "V's" للبيانات الضخمة Big Data:

- **الحجم: Volume** تولد المؤسسات الحديثة تيرابايت من البيانات كل يوم. إنها أكثر من أن يقرأها البشر، ولهذا السبب نحتاج إلى الذكاء الاصطناعي.
- **السرعة: Velocity** تصل البيانات بسرعة عالية. تحدث الهجمات في أجزاء من الثانية، لذا يجب على النظام معالجة البيانات في الوقت الفعلي.

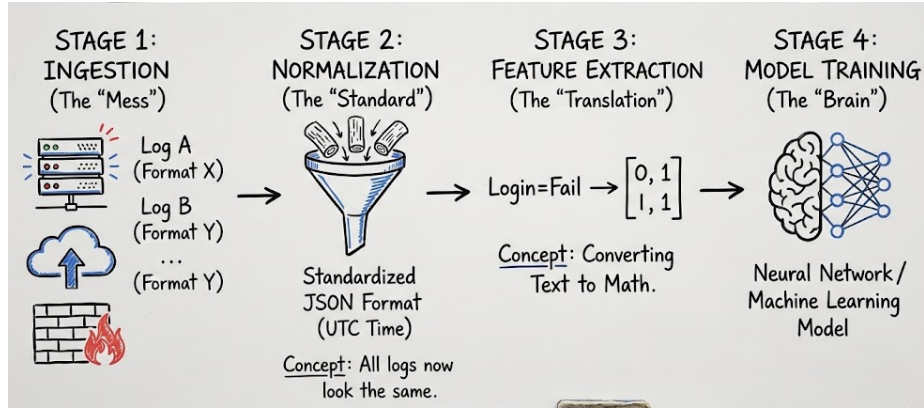
| مصدر البيانات                        | الوصف                                                                                                                                                                             |
|--------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| تدفقات الشبكة Network Flows          | "فاتورة الهاتف". ملخصات لحركة مرور الشبكة. تظهر من تحدث إلى من، ومتى، وكمية البيانات التي أرسلت، ولكن ليس ما قيل. سريعة وخفيفة الوزن.                                             |
| مخزّنات الحزم (PCAP) Packet Captures | "التسجيل الكامل". تحتوي على كل تفاصيل محادثة الشبكة (الرؤوس والمحتوى). مفصلة جداً ولكنها تنتج ملفات ضخمة. تستخدم في الغالب للتحقيق العميق، وليس للذكاء الاصطناعي في الوقت الفعلي. |
| سجلات نقاط النهاية Endpoint Logs     | "يوميات الجهاز". سجلات من أجهزة الكمبيوتر المحمولة والخوادم (مثل سجلات أحداث ويندوز Sysmon، Windows Event Logs). تظهر عمليات تسجيل الدخول، وإنشاء العمليات، وتغييرات الملفات.     |
| سجلات الويب Web Logs                 | "سجل الزوار". تسجل كل عنوان URL مطلوب من خادم الويب. ضرورية للكشف عن هجمات الويب مثل حقن SQL Injection.                                                                           |
| سجلات السحابة Cloud Logs             | "تدقيق واجهة برمجة التطبيقات API". تسجل النشاط في المنصات السحابية (AWS، Azure). تتعقب من أنشأ الخوادم أو غير الأذونات.                                                           |

جدول 1.1: مصادر بيانات الأمان الشائعة

- التنوع: Variety تأتي بيانات الأمان بأشكال وأحجام مختلفة كثيرة (مثل السجلات النصية من الخوادم، والحزم الخام من الشبكات، و JSON من السحابة). دمج هذه التنسيق المختلفة في لغة واحدة للذكاء الاصطناعي أمر صعب.
- عدم التوازن: Imbalance هذه مشكلة رئيسية لتدريب الذكاء الاصطناعي. في مجموعة بيانات مكونة من مليون حدث، قد يكون 999,990 "آمناً" و 10 فقط "هجمات". يكافئ الذكاء الاصطناعي لتعلم شكل الهجوم لأن الأمثلة نادرة.

### 3.4.1 تحديات جودة البيانات والتحضير

بناء نموذج ذكاء اصطناعي يشبه الطهي (إذا استخدمت مكونات سيئة، ستحصل على وجبة سيئة). في علم البيانات، نقول "قمامة تدخل، قمامة تخرج" Garbage In, Garbage Out. قبل استخدام البيانات، يجب أن نحل ثلاث مشاكل تتعلق بالجودة (انظر الشكل 5.1).



شكل 5.1: خط أنابيب بيانات الذكاء الاصطناعي السبراني. يجب تسوية البيانات وتحويلها إلى ميزات قبل أن يتمكن الذكاء الاصطناعي من التعلم منها.

## 1. التنوع والتسوية

تأتي أدوات الأمان من العديد من الشركات المختلفة (Palo Alto، Cisco، Microsoft)، وكلها تتحدث لغات مختلفة.

• الأداة "أ" تقول: "User: logged Admin in."

• الأداة "ب" تقول: "Account: Admin, Authentication Action: Success."

لا يمكن للذكاء الاصطناعي مقارنة هذين الحدثين. يجب أن نقوم بالتسوية Normalization، والتي تحول جميع التنسيقات المختلفة إلى تنسيق قياسي واحد (مثل JSON) حتى يتمكن الذكاء الاصطناعي من قراءتها باستمرار.

## 2. المزامنة الزمنية

يجب ترتيب الأحداث بشكل صحيح لفهم الهجوم.

• المشكلة: إذا تم ضبط جدار الحماية على توقيت لندن (UTC) وخادم على توقيت نيويورك (EST)، فلن تتطابق السجلات. قد يعتقد الذكاء الاصطناعي أن البيانات سُرقَت قبل أن يسجل المخترق الدخول.

• الحل: يجب مزامنة جميع الأجهزة مع توقيت قياسي، وعادة ما يكون التوقيت العالمي المنسق (UTC (Coordinated Universal Time)).



### 3. البيانات المفقودة والخصوصية

إذا فشل جهاز استشعار، يكون لدى الذكاء الاصطناعي "نقطة عمياء". علاوة على ذلك، يجب أن نضمن أن السجلات لا تحتوي على بيانات مستخدم حساسة (مثل كلمات المرور)، وإلا فقد يتعلمها الذكاء الاصطناعي ويكشفها عن طريق الخطأ لاحقاً.

#### 4.4.1 من النص إلى الأرقام: استخراج الميزات

بمجرد أن تكون البيانات نظيفة وموحدة، نواجه عقبة أخيرة: نماذج الذكاء الاصطناعي (مثل الشبكات العصبية Neural Networks) لا يمكنها قراءة النصوص. إنها تفهم الرياضيات. يجب علينا تحويل السجلات إلى أرقام. تسمى هذه العملية استخراج الميزات Feature Extraction.

| بيانات السجل الخام | → | ميزة للذكاء الاصطناعي (رقم) |
|--------------------|---|-----------------------------|
| "AM 10:00:01"      | → | 10 = hour_of_day            |
| "Status=Fail"      | → | 1 = is_failure              |
| "bytes=5000"       | → | 5000 = bytes_transferred    |

#### 5.4.1 مثال عملي: تحضير البيانات للذكاء الاصطناعي

فيما يلي مثال بلغة بايثون باستخدام مكتبة pandas. هذا الكود لا يكتشف هجوماً بشكل مباشر. بدلاً من ذلك، يقوم بإعداد البيانات (استخراج الميزات) حتى يتمكن نموذج الذكاء الاصطناعي من استخدامها لاحقاً. نريد حساب عدد الأخطاء التي تأتي من عنوان IP معين. قد يكون العدد المرتفع ميزة تشير إلى هجوم.

الشرح: لقد حولنا النص الخام إلى جدول من الأرقام. الميزة error\_count\_feature هي الآن مدخل رياضي. سيتعلم نموذج الذكاء الاصطناعي أن "0" طبيعي، لكن "200" مشبوه.

#### 5.1 الذكاء الاصطناعي في العمل: مركز عمليات الأمان الحديث

مركز عمليات الأمان Security Operations Center (SOC) هو المقر الرئيسي للدفاع. إنه غرفة (أو فريق افتراضي) حيث يعمل المحللون البشريون والآلات معاً (انظر الشكل 6.1).

---

```

1 import pandas as pd
2 import io
3
4 # 1. Simulate raw log data (CSV format)
5 # In the real world, this would be a large file.
6 log_data = """timestamp,source_ip,status_code
7 2025-08-16 10:00:01,192.168.1.10,200
8 2025-08-16 10:00:05,192.168.1.10,404
9 2025-08-16 10:00:10,192.168.1.10,404
10 2025-08-16 10:00:15,10.0.0.5,200
11 """
12
13 # 2. Load the data into a DataFrame
14 df = pd.read_csv(io.StringIO(log_data))
15
16 # 3. FEATURE EXTRACTION
17 # An AI cannot just read "404". It needs a count or a rate.
18 # Here, we convert raw rows into a statistical summary.
19 # Logic: Group by IP address, then count how many times '404' appears.
20
21 features = df.groupby('source_ip')['status_code'].apply(
22 lambda x: (x == 404).sum()
23).reset_index(name='error_count_feature')
24
25 print("Data ready for AI Model:")
26 print(features)
27
28 # Output:
29 # source_ip error_count_feature
30 # 0 10.0.0.5 0
31 # 1 192.168.1.10 2

```

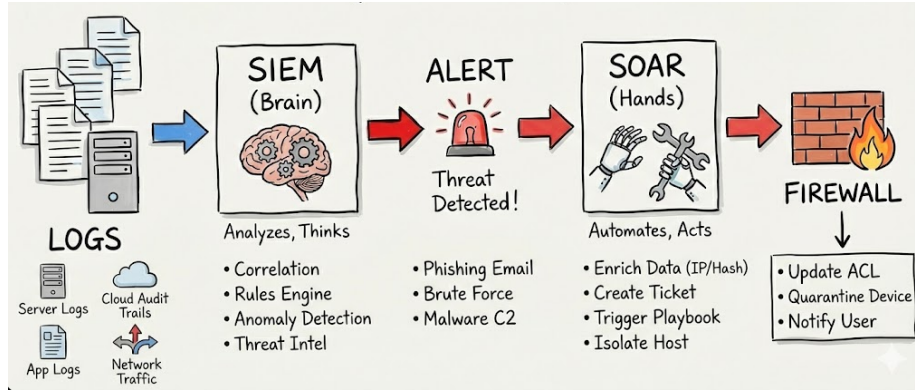
---

Listing 1: تحويل السجلات الخام إلى ميزات (تحضير البيانات)

## 1.5.1 SOAR و SIEM

يعتمد مركز عمليات الأمان على منصتين رئيسيتين:

- **SIEM (إدارة معلومات وأحداث الأمان):** Security Information and Event Management "الدماغ". يجمع السجلات من كل مكان ويستخدم الذكاء الاصطناعي لتحليلها.
- **SOAR (تنسيق وأتمتة واستجابة الأمان):** Security Orchestration, Automation, and Response "الأيدي". ينفذ الإجراءات. إذا قال الذكاء الاصطناعي "احظر هذا ال IP"، يقوم SOAR بتسجيل الدخول إلى جدار الحماية وحظره تلقائياً.



شكل 6.1: سير عمل مركز عمليات الأمان (SIEM) + (SOAR)

## 2.5.1 الإيجابيات الكاذبة مقابل السلبيات الكاذبة

عند بناء ذكاء اصطناعي لمركز عمليات الأمان، يجب أن نوازن بين نوعين من الأخطاء. يُقاس نجاح نموذج الذكاء الاصطناعي بأخطائه. تخيل ذلك باستخدام مصفوفة الارتباك Confusion Matrix (الجدول 1.2).

| آمن فعلياً                                         | خبيث فعلياً                                         |                                |
|----------------------------------------------------|-----------------------------------------------------|--------------------------------|
| إيجابي كاذب False Positive<br>(إنذار كاذب - ضوضاء) | إيجابي حقيقي True Positive<br>(أمسكاً بالقرصان!)    | تنبؤ الذكاء الاصطناعي: "تهديد" |
| سلبي حقيقي True Negative<br>(النظام هادئ)          | سلبي كاذب False Negative<br>(تعرضنا للاختراق - خرق) | تنبؤ الذكاء الاصطناعي: "آمن"   |

جدول 2.1: مصفوفة الارتباك

(1) الإيجابي الكاذب (خطأ من النوع الأول): False Positive (Type I Error)  
يطلق الذكاء الاصطناعي الإنذار، ولكن لا يوجد هجوم.

• النتيجة: يضع المحلل وقته في التحقيق. إذا حدث هذا كثيراً، يعاني المحللون من "إرهاق التنبيهات" Alert Fatigue ويتوقفون عن الاستماع إلى الذكاء الاصطناعي.

(2) السلبي الكاذب (خطأ من النوع الثاني): False Negative (Type II Error) يوجد  
هجوم، لكن الذكاء الاصطناعي يبقى صامتاً.

• النتيجة: تتعرض الشركة للاختراق. هذا هو السيناريو الخطير.

الهدف: نريد تقليل السلبيات الكاذبة (القبض على القراصنة) دون خلق الكثير من الإيجابيات الكاذبة (إزعاج المحللين).

### 3.5.1 الذكاء الاصطناعي كمضاعف للقوة وقابلية التفسير (XAI)

لا يحل الذكاء الاصطناعي محل المحللين البشريين؛ بل يعمل كمضاعف للقوة Force Multiplier. مضاعف القوة هو أداة تزيد بشكل كبير من إنتاجية النظام.

• بدون الذكاء الاصطناعي: يمكن للمحلل مراجعة 50 تنبيهاً يومياً.

• مع الذكاء الاصطناعي: يمكن للمحلل مراجعة 500 تنبيه يومياً لأن الذكاء الاصطناعي يقوم بتصفية الضوضاء ويسلط الضوء على البيانات الهامة.

ولكن، هناك تحد: مشكلة "الصندوق الأسود" Black Box. غالباً ما تعطي نماذج الذكاء الاصطناعي المعقدة (مثل التعلم العميق Deep Learning) إجابة دون تفسير.

• الذكاء الاصطناعي يقول: "احظر المستخدم أ".

• المحلل يسأل: "لماذا؟"

• الذكاء الاصطناعي: (صمت).

إذا لم يفهم المحللون لماذا اتخذ الذكاء الاصطناعي قراراً، فلن يثقوا به. أدى هذا إلى ظهور الذكاء الاصطناعي القابل للتفسير Explainable AI (XAI)، الذي يضمن أن النماذج تقدم أسباباً لقراراتها (على سبيل المثال، "احظر المستخدم أ لأنه سجل الدخول من بلد جديد").

## تحقق من الفهم

- (1) ما هو الفرق الرئيسي بين "التكتيك" و "التقنية" في إطار عمل MITRE ATT&CK؟
- (2) لماذا يجب تحويل بيانات السجل الخام إلى "ميزات" قبل استخدامها مع شبكة عصبية Neural Network؟
- (3) أيهما يعتبر بشكل عام أسوأ في بيئة أمنية مشددة: إيجابي كاذب أم سلبي كاذب؟ ولماذا؟

## خاتمة

في هذا الفصل، وضعنا الأساس لبقية الكتاب. استكشفنا كيف تحول مشهد الأمن السيبراني من التوقعات الثابتة إلى التحليل السلوكي الديناميكي المدعوم بالذكاء الاصطناعي. استخدمنا هرم الألم *Pyramid of Pain* لنرى لماذا يكون اكتشاف السلوك أكثر فعالية من حظر الأدوات البسيطة.

نظرنا أيضًا إلى المادة الخام للذكاء الاصطناعي: البيانات. اكتشفنا أن الذكاء الاصطناعي لا يمكنه ببساطة قراءة السجلات؛ يجب علينا التعامل مع مشاكل جودة البيانات *Data Quality* واستخراج الميزات لتحويل النص إلى رياضيات. أخيرًا، حددنا دور الذكاء الاصطناعي في مركز عمليات الأمن. تعلمنا أن الذكاء الاصطناعي يعمل كمضاعف للقوة، لكنه يتطلب قابلية التفسير *Explainability (XAI)* لكي يثق به البشر، واستخدمنا مصفوفة الارتباك *Confusion Matrix* لفهم التوازن بين الإنذارات الكاذبة والهجمات الفائقة.

مع هذا السياق، أنت الآن مستعد للانتقال من المفاهيم إلى الكود. في الفصل التالي، سنبدأ في استكشاف خوارزميات التعلم الآلي المحددة المستخدمة للكشف عن هذه التهديدات.

## المختصرات والمصطلحات

الذكاء الاصطناعي (Artificial Intelligence) AI هو مجال في علوم الحاسوب يركز على إنشاء أنظمة يمكنها أداء المهام التي تتطلب ذكاءً بشرياً. في هذا السياق، يشير إلى الأنظمة التي يمكنها تعلم السلوك الطبيعي، واكتشاف الانحرافات، وأتمتة الإجراءات الدفاعية.

الحالة الشاذة Anomaly هو انحراف عن خط الأساس للسلوك الطبيعي والمنشأ داخل بيئة تكنولوجيا المعلومات. يمكن أن تشير الحالات الشاذة Anomalies إلى تهديد أمني محتمل.

التهديد المتقدم المستمر APT (Advanced Persistent Threat) هو حملة سببرانية مطولة وشديدة الاستهداف حيث يتمكن المهاجم من الوصول غير المصرح به إلى شبكة ما ويظل غير مكتشف لفترة طويلة، عادةً لأغراض التجسس أو سرقة البيانات.

الأمثلة الخسومية Adversarial examples هي مدخلات خبيثة مُعدلة بشكل طفيف (مثل ملف برمجية خبيثة malware مع تغيير بضعة بايتات) مصممة خصيصاً ليتم تصنيفها بشكل خاطئ والتهرب من أنظمة الذكاء الاصطناعي AI الدفاعية.

سلسلة الهجوم السببراني Cyber Kill Chain هو إطار عمل يحدد المراحل المتسلسلة النموذجية لهجوم سببراني، ويشير إلى أن تعطيل الهجوم في أي مرحلة يمكن أن يمنع الخصم من تحقيق هدفه النهائي.

التزييف العميق Deepfakes هي مقاطع صوتية أو مرئية اصطناعية مُولدة بواسطة الذكاء الاصطناعي AI تُستخدم لانتحال شخصية أفراد موثوق بهم، غالباً في مخططات الهندسة الاجتماعية so-cial engineering المتطورة.

الاختبار الضبابي Fuzzing هي طريقة آلية يُستخدمها المهاجمون لاكتشاف ثغرات برمجية جديدة عن طريق إدخال بيانات غير صالحة أو غير متوقعة أو عشوائية إلى مدخلات البرنامج.

الشبكات التوليدية التنافسية (Generative Adversarial Networks) GANs هي نوع من نماذج الذكاء الاصطناعي AI يمكن للمهاجمين تدريبها لإنشاء سلاسل جديدة وفعالة من البرمجيات الخبيثة malware تلقائياً تكون غير مرئية للدفاعات القائمة على التوقعات.

أنظمة كشف التسلل IDS (Intrusion Detection Systems) هي أدوات أمنية تراقب أنشطة الشبكة أو النظام بحثاً عن إجراءات ضارة أو انتهاكات للسياسات. غالباً ما تعتمد الإصدارات التقليدية على التوقعات signatures.

مؤشرات الاختراق (IOCs (Indicators of Compromise هي أدلة دقيقة أو أنماط أو بيانات جنائية رقمية forensic data توجد في السجلات أو الملفات وتشير إلى احتمال وقوع خرق أمني.

تعلم الآلة (ML (Machine Learning هو مجال فرعي من الذكاء الاصطناعي AI يمكن أنظمة الكمبيوتر من التعلم من البيانات وتحديد الأنماط دون برمجة صريحة. وهو التكنولوجيا الأساسية وراء اكتشاف التهديدات القائم على السلوك.

إطار عمل MITRE ATT&CK هو قاعدة معرفية وإطار عمل متاح عالمياً لتكتيكات وتقنيات الخصوم في العالم الحقيقي، يستخدمه متخصصو الأمن لفهم ونمذجة سلوك المهاجمين.

البرمجيات الخبيثة متعددة الأشكال Polymorphic malware هي برمجيات خبيثة تغير شفرتها أو ميزاتها باستمرار لإنشاء توقيعات signatures جديدة وفريدة في كل مرة تتكاثر فيها، مما يجعل من الصعب على برامج مكافحة الفيروسات التقليدية اكتشافها.

إدارة معلومات وأحداث الأمن (SIEM (Security Information and Event Management هي منصة مركزية تقوم بجمع وتجميع وتحليل بيانات السجلات والتنبيهات المتعلقة بالأمن من جميع أنحاء البنية التحتية لتكنولوجيا المعلومات في المؤسسة.

تنسيق وأتمتة واستجابة الأمن (SOAR (Security Orchestration, Automation, and Response هي منصة تساعد المؤسسات على الاستجابة للتنبيهات الأمنية بطريقة آلية باستخدام مسارات عمل محددة مسبقاً أو ما يعرف بـ "كتيبات اللعب" playbooks لتنفيذ إجراءات الاستجابة. التوقيع Signature هو نمط فريد، مثل بصمة رقمية، يرتبط بقطعة معروفة من التعليمات البرمجية الخبيثة أو طريقة هجوم محددة، وتستخدمه الأدوات الأمنية التقليدية للكشف.

هجوم سلسلة التوريد Supply chain attack هي استراتيجية هجوم غير مباشرة حيث يقوم الخصوم باختراق مؤسسة ما عن طريق استهداف العناصر الأقل أماناً في شبكة مورديها، مثل بائع برامج تابع لجهة خارجية.

صيد التهديدات Threat Hunting هي الممارسة الاستباقية للبحث في شبكات وبيانات المؤسسة للعثور على التهديدات المتقدمة التي أفلتت من الدفاعات الأمنية الآلية الحالية.

هجوم يوم الصفر Zero-day attack هو هجوم سببراني يستغل ثغرة برمجية غير معروفة سابقاً لا يتوفر لها تصحيح أو إصلاح، مما يجعله فعالاً للغاية ضد الدفاعات التقليدية.



## باب 2

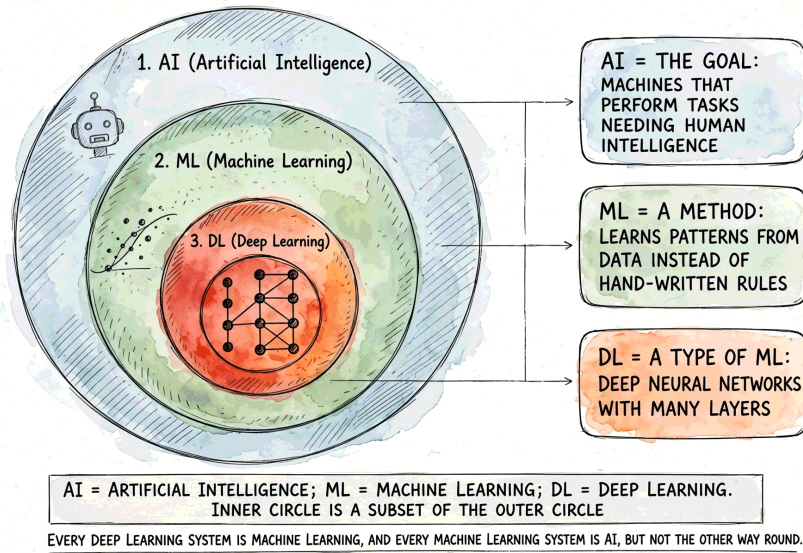
# Machine Learning Cybersecurity Essentials

### مقدمة

مرحباً بكم في جوهر أنظمة الدفاع الذكية الحديثة. يوفر هذا الفصل معرفة machine learning (ML) الأساسية التي يحتاجها كل محترف في cybersecurity. سنتجاوز الكلمات الطنانة ونبنى أساساً متيناً، بدءاً من الفروق الواضحة بين Artificial Intelligence و Machine Learning و Deep Learning. ستتعرف على نماذج ML الرئيسية ---supervised، unsupervised، semi-supervised، و reinforcement learning --- وسترى كيف ينطبق كل منها على مشاكل الأمن في العالم الحقيقي مثل تصنيف malware وكشف anomaly detection. سنستكشف بعد ذلك العائلات الرئيسية لنماذج ML، من الخوارزميات الكلاسيكية مثل Random Forest إلى معماريات deep learning المتقدمة مثل Convolutional Neural Networks و Recurrent Neural Networks. ينصب التركيز الحاسم في هذا الفصل على الجوانب العملية لبناء أنظمة ML للأمن. يتضمن ذلك العملية الحاسمة لـ feature engineering، وفن اختيار مقاييس التقييم التي تعكس حقاً فعالية النموذج في سياقات الأمن، ونظرة صادقة على التحديات الكبيرة التي ستواجهها، مثل imbalanced data و adversarial attacks. بنهاية هذا الفصل، ستكون قد أكملت مشروعاً عملياً، حيث ستقوم ببناء خط أنابيب أساسي لتصنيف التهديدات بدءاً من تحميل البيانات وصولاً إلى تقييم النموذج، مما يمنحك سير عمل ملموساً وقابلاً للتكرار لمشاريعك الخاصة.

## 1.2 AI، ML، و DL: إزالة الغموض عن الكلمات الطنانة

لبناء أنظمة دفاع ذكية، يجب علينا أولاً فهم لغة المجال. غالباً ما يتم استخدام مصطلحات AI- Deep Learning (AI)، Machine Learning (ML)، و Deep Learning (DL) بالتبادل، لكنها تمثل مفاهيم متميزة ذات علاقة هرمية واضحة.



شكل 1.2: العلاقة الهرمية: DL هو جزء من ML، و ML هو جزء من AI.

### 1.1.2 Artificial Intelligence (AI)

Artificial Intelligence (AI) هو المجال الأوسع. إنه فرع من علوم الكمبيوتر مخصص لإنشاء آلات أو أنظمة يمكنها أداء المهام التي تتطلب عادةً ذكاءً بشرياً. تشمل هذه المهام قدرات مثل الإدراك البصري، والتعرف على الكلام، واتخاذ القرار، وحل المشكلات، وفهم اللغة الطبيعية. الهدف النهائي لـ AI هو خلق ذكاء عام، ولكن في الممارسة العملية، فإن معظم أنظمة AI الحالية ضيقة النطاق، مما يعني أنها مصممة لأداء مهمة محددة بشكل جيد للغاية.

## Machine Learning (ML) 2.1.2

Machine Learning (ML) هو حقل فرعي من AI. إنه يمنح أجهزة الكمبيوتر القدرة على التعلم من البيانات دون أن تتم برمجتها بشكل صريح لكل سيناريو محتمل. بدلاً من كتابة القواعد يدوياً، تقوم خوارزمية ML بتحليل مجموعة بيانات للعثور على الأنماط. ثم تستخدم هذه الأنماط المكتسبة لعمل تنبؤات أو قرارات بشأن بيانات جديدة غير مرئية. بمرور الوقت، ومع معالجتها للزائد من البيانات، يتحسن أداء الخوارزمية في مهمتها المحددة.

## Deep Learning (DL) 3.1.2

Deep Learning (DL) هو حقل فرعي متخصص داخل machine learning. يستخدم نوعاً محدداً من الخوارزميات يسمى *artificial neural network*، وهو مستوحى من بنية الدماغ البشري. تحتوي هذه الشبكات على طبقات متعددة من العقد المترابطة، ويشير مصطلح "عميق" (Deep) إلى وجود العديد من هذه الطبقات. يعد Deep learning قوياً بشكل خاص للعثور على أنماط معقدة للغاية في مجموعات البيانات الكبيرة، وخاصة البيانات غير المهيكلة مثل الصور والصوت والنصوص. العلاقة بين هذه المفاهيم بسيطة:

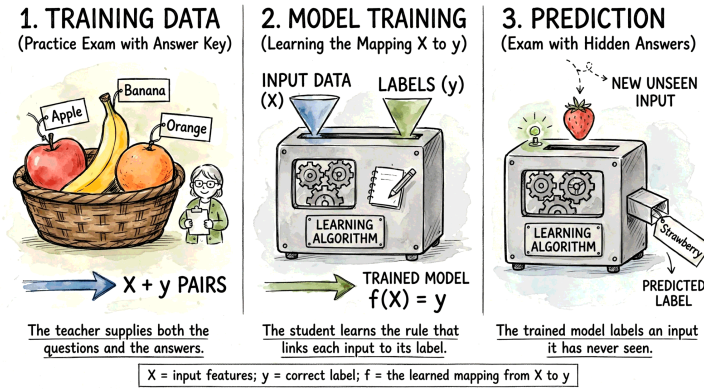
- AI هو الهدف العام لإنشاء آلات ذكية.
- ML هو طريقة أساسية لتحقيق AI من خلال التعلم من البيانات.
- DL هو نوع قوي وشائع من ML يستخدم deep neural networks لتحقيق أحدث النتائج في العديد من المجالات.

## 2.2 نماذج ML الأساسية: مجموعة أدوات الذكاء الاصطناعي الخاصة بك

لا يعد Machine learning تقنية واحدة بل مجموعة من الأساليب، أو النماذج، كل منها مناسب لأنواع مختلفة من المشاكل والبيانات. فهم هذه النماذج الأساسية أمر جوهري لاختيار الأداة المناسبة لمهمة cybersecurity.

## Supervised Learning 1.2.2

يعد Supervised learning أكثر نماذج ML شيوعاً (الشكل 3.2). تتعلم الخوارزمية من مجموعة بيانات حيث يقترن كل جزء من بيانات الإدخال بتسمية إخراج صحيحة. فكر في هذا مثل المعلم والطالب (الشكل 2.2). يزود المعلم الطالب بامتحانات تدريبية (بيانات الإدخال) جنباً إلى جنب مع مفتاح الإجابة (التسميات). هدف الطالب هو تعلم المنطق اللازم للوصول إلى الإجابات الصحيحة حتى يتمكن من اجتياز امتحان مستقبلي حيث تكون الإجابات مخفية.



شكل 2.2: Supervised Machine Learning: تعلم التعيين من الإدخال ( $X$ ) إلى الإخراج ( $y$ ).

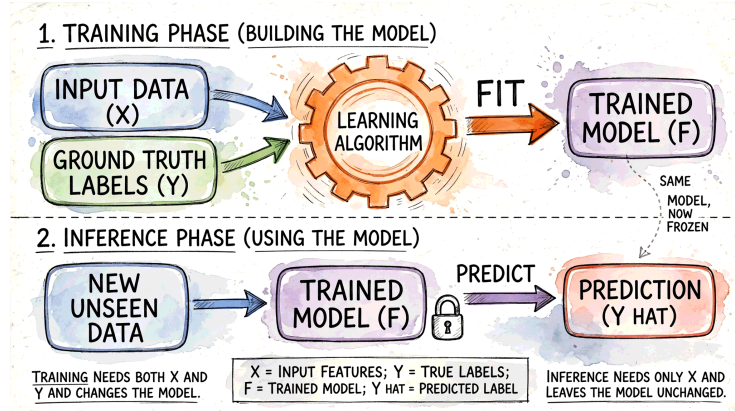
## تطبيقات في Cybersecurity

1. Classification (التنبؤ الفئوي) هذه هي مهمة التنبؤ بفئة منفصلة (الشكل 4.2).

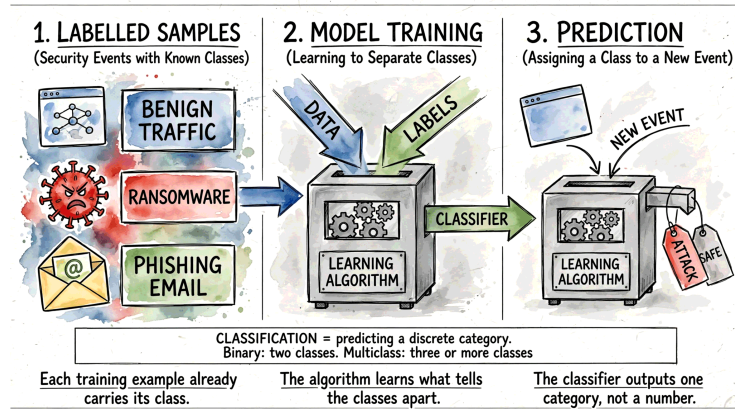
• Binary Classification: فئتان (مثلاً، "حميد" مقابل "Malware"، "Spam" مقابل "مقبول") (الشكل 5.2).

• Multiclass Classification: أكثر من فئتين (مثلاً، تحديد عائلة malware: "Ran-", "Worm"، "Spyware"، "Trojan"، "somware").

توضيح القائمة 2 كاشف Phishing بسيط. ينظر النموذج إلى ميزات عنوان URL لتصنيف العنوان إما كآمن (0) أو Phishing (1).



شكل 3.2: نظرة عامة على نظام Supervised Machine Learning.

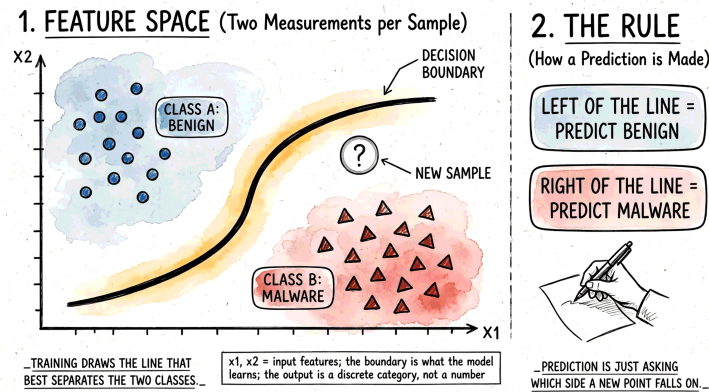


شكل 4.2: نظرة عامة على ML Classification.

2. Regression (التنبؤ العددي) هذه هي مهمة التنبؤ بقيمة عددية مستمرة (الشكل 6.2). في cybersecurity، يُستخدم هذا غالباً لتقييم المخاطر أو التنبؤ.

• أمثلة: التنبؤ بدرجة خطورة CVSS للثغرة الأمنية (0.0 إلى 0.10)، تقدير عدد الأيام حتى يتم عمل هندسة عكسية لتصحيح البرمجي، أو التنبؤ بإنتاجية الشبكة (الشكل 7.2).

تُظهر القائمة 3 نموذجاً يتنبأ بـ Risk Score. يتعلم النموذج أنه كلما زاد عدد الثغرات غير المصححة، ارتفعت درجة المخاطرة.



شكل 5.2: Classification: رسم حدود لفصل الفئات.

```

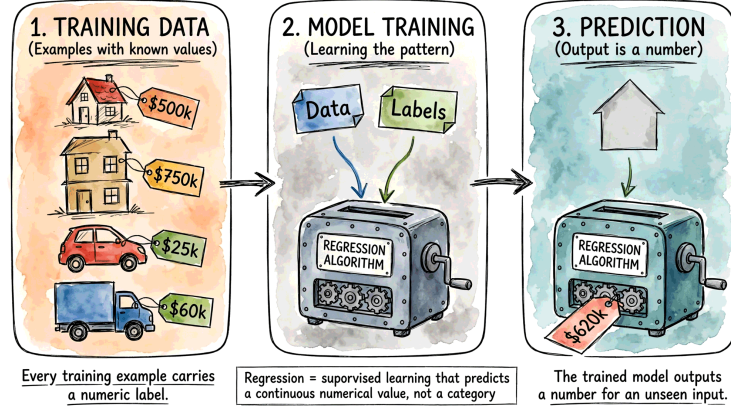
1 import numpy as np
2 from sklearn.linear_model import LogisticRegression
3
4 # --- CLASSIFICATION EXAMPLE: Phishing Detection ---
5
6 # 1. Dataset (Features: [URL Length, Number of '@' symbols])
7 # Long URLs and '@' symbols are suspicious.
8 X_train = np.array([
9 [15, 0], [22, 0], [18, 0], # Benign samples (Short, no '@')
10 [85, 1], [90, 2], [75, 1] # Phishing samples (Long, has '@')
11])
12
13 # Labels (0: Benign, 1: Phishing)
14 y_train = np.array([0, 0, 0, 1, 1, 1])
15
16 # 2. Train Model
17 clf = LogisticRegression()
18 clf.fit(X_train, y_train)
19
20 # 3. Predict on new data
21 new_email = np.array([[80, 1]]) # A long URL with an '@' symbol
22 prediction = clf.predict(new_email)
23
24 print(f"Classification Prediction: {prediction[0]} (1=Phishing, 0=Benign)")

```

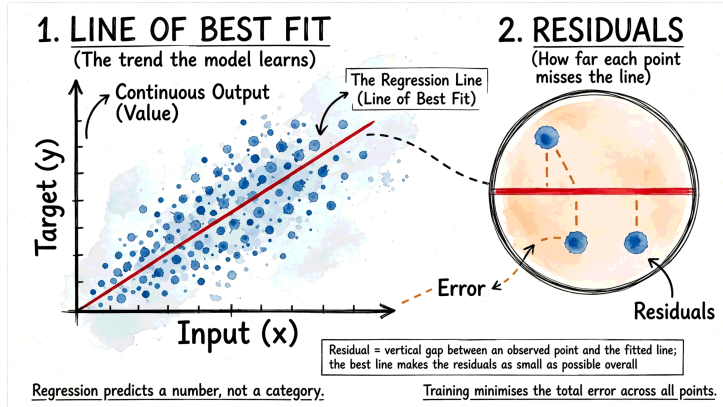
## Listing 2: Supervised Classification: التنبؤ بفئة (Phishing).

### 2.2.2 Unsupervised Learning

على النقيض من supervised learning، تعمل خوارزميات unsupervised learning مع البيانات التي ليس لها تسميات محددة مسبقاً (الشكل 8.2). فكر في هذا كمحقق يدخل مسرح جريمة دون معرفة الجريمة التي وقعت. يجب عليه النظر في الأدلة، والعثور على الروابط، وتحديد الأشياء الغريبة بمفرده تماماً.



شكل 6.2: نظرة عامة على Regression.



شكل 7.2: Regression: ملاءمة خط للتنبؤ بالاتجاهات العددية.

## تطبيقات في Cybersecurity

1. Anomaly Detection يتضمن ذلك تحديد نقاط البيانات التي تختلف اختلافاً كبيراً عن المعيار (الشكل 9.2). في الأمن، هذه هي الطريقة الأساسية للكشف عن *Zero-Day attacks* أو *Insider Threats*، حيث يكون نمط الهجوم غير معروف، لكن السلوك "غير طبيعي" بوضوح. تستخدم القائمة 4 خوارزمية *Isolation Forest*. تتعلم شكل سلوك تسجيل الدخول "العادي" وتضع علامة على محاولة القوة العاشمة العدوانية كشدوذ.

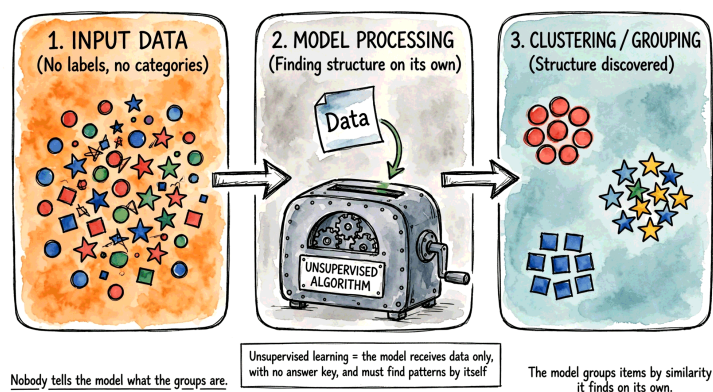


```

1 import numpy as np
2 from sklearn.linear_model import LinearRegression
3
4 # --- REGRESSION EXAMPLE: Risk Scoring ---
5
6 # 1. Dataset (Feature: [Count of Open Critical Vulnerabilities])
7 X_history = np.array([
8 [0], [1], [2], [5], [10]
9])
10
11 # Labels (Target Risk Score: 0.0 to 100.0)
12 # Notice the relationship: More vulnerabilities = Higher Score
13 y_history = np.array([10.0, 25.0, 40.0, 65.0, 95.0])
14
15 # 2. Train Model
16 reg = LinearRegression()
17 reg.fit(X_history, y_history)
18
19 # 3. Predict Risk for a system with 7 open vulnerabilities
20 new_system = np.array([[7]])
21 predicted_score = reg.predict(new_system)
22
23 print(f"Regression Prediction (Risk Score): {predicted_score[0]:.2f}")

```

### Listing 3: Supervised Regression: التنبؤ بقيمة مستمرة (Risk Score).

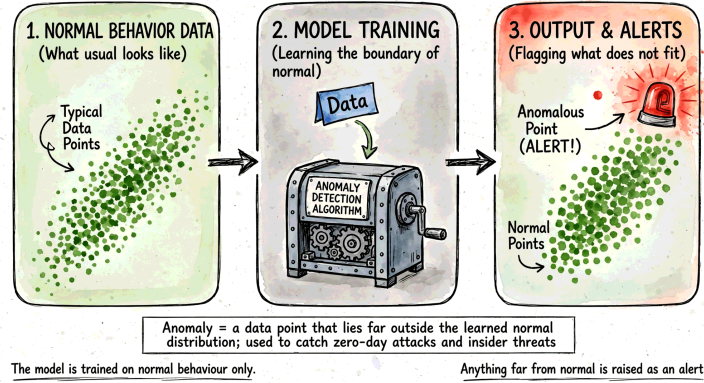


### شكل 8.2: Unsupervised Learning: يجد النموذج الهيكل في البيانات غير المسماة.

2. Clustering يقوم Clustering بتجميع نقاط البيانات المتشابهة معاً بناءً على ميزاتها (الشكل 10.2). يستخدم محللو الأمن هذا لتصنيف عينات malware الجديدة تلقائياً أو لتجميع المضيفين المخترقين في botnets. تستخدم القائمة 5 خوارزمية *K-Means* لفصل حركة مرور الشبكة إلى مجموعتين متميزتين دون إخبارها بماهية المجموعات.

3. Dimensionality Reduction يمكن أن تحتوي مجموعات بيانات الأمن على مئات الميزات (رؤوس الحزم، الأعلام، إنترويا المحولة)، مما يجعل معالجتها بطيئة وصعبة التصور. يقوم Dimensionality reduction بضغط هذه البيانات مع الاحتفاظ بالمعلومات المهمة (الشكل





## شكل 9.2: Anomaly Detection: تحديد القيم المتطرفة التي لا تتناسب مع التوزيع القياسي.

```

1 from sklearn.ensemble import IsolationForest
2 import numpy as np
3
4 # --- ANOMALY DETECTION: Insider Threat ---
5
6 # 1. Training Data (Normal User Behavior)
7 # Feature: [Number of File Accesses per Minute]
8 # Most users access about 5-10 files.
9 X_normal = np.array([[5], [6], [5], [7], [10], [8], [5]])
10
11 # 2. Train Model
12 # contamination=0.01 tells the model that anomalies are rare.
13 iso = IsolationForest(contamination=0.01, random_state=42)
14 iso.fit(X_normal)
15
16 # 3. Live Data Monitoring
17 # User A accesses 6 files (Normal)
18 # User B accesses 1000 files (Data Exfiltration Attempt)
19 X_live = np.array([[6], [1000]])
20
21 predictions = iso.predict(X_live)
22 # Output: 1 = Normal, -1 = Anomaly
23 print(f"Predictions: {predictions}")
24 # Expected: [1, -1] -> User B flagged as anomaly

```

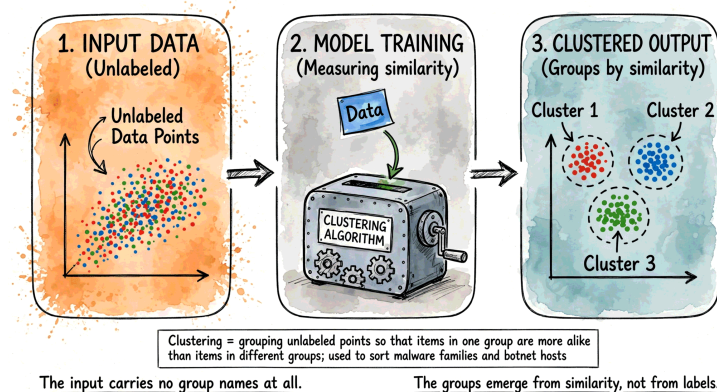
Listing 4: استخدام Isolation Forest للكشف عن أنماط الوصول غير الطبيعية للملفات.

(11.2).

تستخدم القائمة 6 تقنية *PCA (Principal Component Analysis)* لضغط مجموعة بيانات تحتوي على 5 ميزات إلى 2 فقط، مما يجعل من الممكن رسم بياني ثنائي الأبعاد.

### 3.2.2 Semi-supervised Learning

*Semi-supervised learning* هو نهج هجين يستخدم كمية صغيرة من البيانات المسماة جنباً إلى جنب مع كمية كبيرة من البيانات غير المسماة (الشكل 12.2).



شكل 10.2: Clustering: تجميع البيانات غير المسماة بناءً على التشابه.

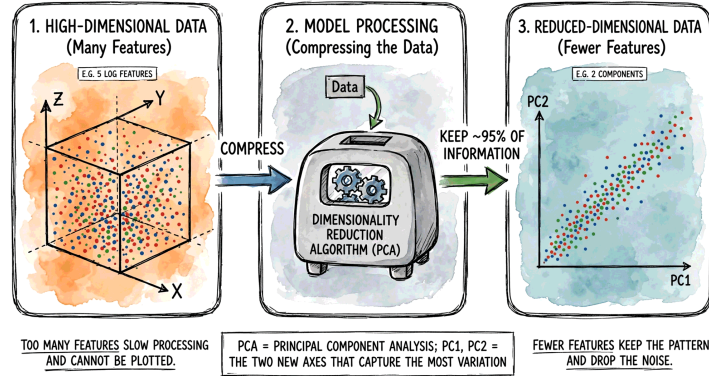
```

1 from sklearn.cluster import KMeans
2 import numpy as np
3
4 # --- CLUSTERING: Traffic Categorization ---
5
6 # 1. Unlabeled Data (Features: [Packet Size, Destination Port])
7 # We have mixed traffic, but we don't know what is what.
8 X_traffic = np.array([
9 [64, 53], [70, 53], [60, 53], # Small packets on Port 53 (DNS)
10 [1500, 443], [1450, 443], [1500, 443] # Large packets on Port 443 (HTTPS)
11])
12
13 # 2. Train Model
14 # We ask the model to find 2 distinct groups (clusters)
15 kmeans = KMeans(n_clusters=2, random_state=42)
16 kmeans.fit(X_traffic)
17
18 # 3. Analyze Results
19 print(f"Cluster Labels: {kmeans.labels_}")
20 # Expected: [0, 0, 0, 1, 1, 1]
21 # The model successfully grouped DNS traffic separately from HTTPS traffic.

```

## Listing 5: استخدام K-Means لتجميع حركة مرور الشبكة حسب التشابه.

في cybersecurity، يعد التحليل اليدوي للملف لتأكيد أنه malware مكلفاً وبطيئاً (يتطلب خبراء بشريين). ومع ذلك، فإن جمع الملفات الخام غير المحللة أمر سهل. يحل Semi-supervised learning هذه المشكلة باستخدام الملفات السيئة القليلة المعروفة لاستنتاج طبيعة الملفات غير المعروفة بناءً على تشابهها. توضح القائمة 7 تقنية *Label Spreading*. نضع علامة على الملفات غير المعروفة ب-1. ثم ينظر النموذج إلى هندسة البيانات وينشر التسميات المعروفة (0 أو 1) إلى أقرب الجيران غير المعروفين.



شكل 11.2: Dimensionality Reduction: ضغط البيانات المعقدة إلى أبعاد أقل.

```

1 from sklearn.decomposition import PCA
2 import numpy as np
3
4 # --- DIMENSIONALITY REDUCTION: Log Compression ---
5
6 # 1. High-Dimensional Data (5 Features per log entry)
7 # Imagine features: [CPU, RAM, Disk, Net_In, Net_Out]
8 X_logs = np.random.rand(10, 5)
9
10 print(f"Original Shape: {X_logs.shape} (10 samples, 5 features)")
11
12 # 2. Apply PCA
13 # We want to compress 5 features down to 2 "Principal Components"
14 pca = PCA(n_components=2)
15 X_compressed = pca.fit_transform(X_logs)
16
17 # 3. Result
18 print(f"Compressed Shape: {X_compressed.shape} (10 samples, 2 features)")
19 print(f"Information Retained: {np.sum(pca.explained_variance_ratio_):.2%}")
20 # We typically retain ~95% of the information while removing noise.

```

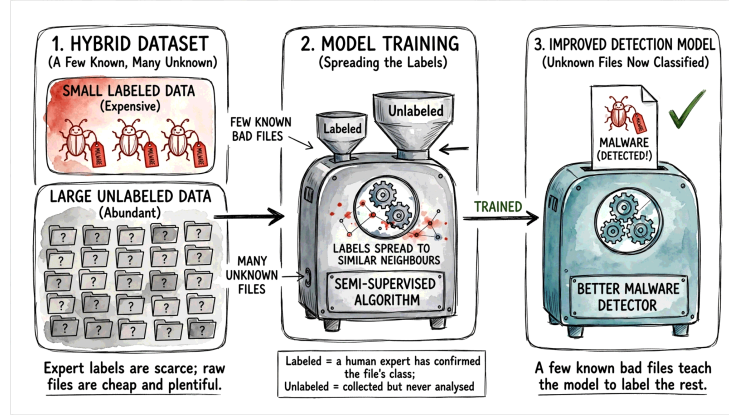
Listing 6: استخدام PCA لضغط بيانات السجلات عالية الأبعاد.

## 4.2.2 Reinforcement Learning (RL)

في reinforcement learning، يتعلم "العميل" اتخاذ القرارات عن طريق أداء إجراءات في بيئة لتحقيق هدف (الشكل 13.2).

على عكس الطرق الأخرى، لا يتم إخبار العميل \*بما\* يجب عليه فعله. يتعلم من خلال التجربة والخطأ. يتخذ إجراءً، ويتلقى *Reward* (نقاط إيجابية) أو *Penalty* (نقاط سلبية)، ويقوم بتحديث استراتيجيته لتعظيم المكافآت بمرور الوقت.

تحاكي القائمة 8 *Firewall Agent* بسيط. يتعلم ما إذا كان يجب "حظر" أو "السماح" بحركة المرور من نطاق IP معين. في البداية، يخمن بشكل عشوائي. بمرور الوقت، يتعلم أن حظر عناوين IP الضارة يمنح مكافأة (منع اختراق)، بينما حظر المستخدمين الصالحين يمنح عقوبة (شكوى المستخدم).



شكل 12.2: Semi-supervised Learning: نشر التسميات من البيانات المعروفة إلى غير المعروفة.

```

1 import numpy as np
2 from sklearn.semi_supervised import LabelSpreading
3
4 # --- SEMI-SUPERVISED EXAMPLE: Malware Label Propagation ---
5
6 # 1. Dataset (Feature: {File Entropy, File Size})
7 X = np.array([
8 [0.1, 100], [0.2, 120], # Cluster A (Likely Benign)
9 [0.9, 900], [0.8, 950], # Cluster B (Likely Malware)
10 [0.15, 110], # New File 1 (Unknown)
11 [0.85, 920] # New File 2 (Unknown)
12])
13
14 # 2. Labels (0: Benign, 1: Malware, -1: Unknown/Unlabeled)
15 # We only know the ground truth for the first 4 files.
16 y_mixed = np.array([0, 0, 1, 1, -1, -1])
17
18 # 3. Train Model (Label Spreading)
19 # The model measures similarity to guess the -1 labels.
20 model = LabelSpreading(kernel='knn', n_neighbors=2)
21 model.fit(X, y_mixed)
22
23 # 4. Results
24 print(f"Original Labels: {y_mixed}")
25 print(f"Inferred Labels: {model.transduction}")
26 # Expected: The last two files should become [0, 1] based on neighbors.

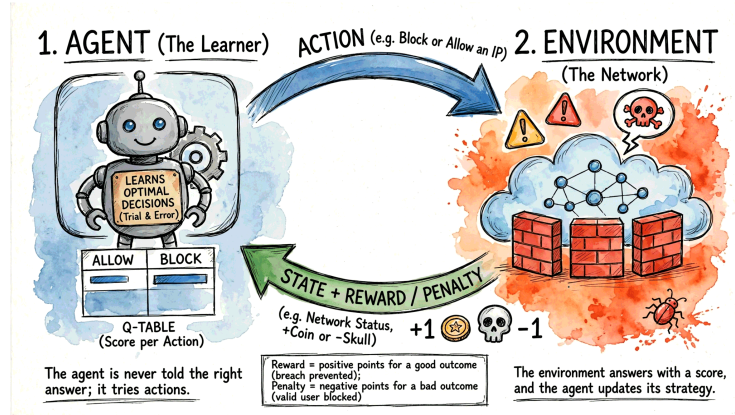
```

Listing 7: استخدام Label Spreading لتحديد الملفات غير المعروفة بناءً على أمثلة محدودة.

## 5.2.2 سير عمل Supervised Learning

يتبع مشروع supervised learning النموذجي سير عمل منظم. ترتيب هذه الخطوات أمر بالغ الأهمية: القيام بها خارج التسلسل (مثل قياس البيانات قبل تقسيمها) يمكن أن يؤدي إلى نتائج غير صالحة، وهي مشكلة تعرف بـ *data leakage*.

- (1) جمع البيانات: جمع مجموعة بيانات حيث يكون لكل مثال تسمية صحيحة. على سبيل المثال، جمع سجلات تدفق الشبكة، مع تسمية كل سجل بـ "حميد" أو "ضار".



شكل 13.2: Reinforcement Learning: حلقة العميل-البيئة-المكافأة.

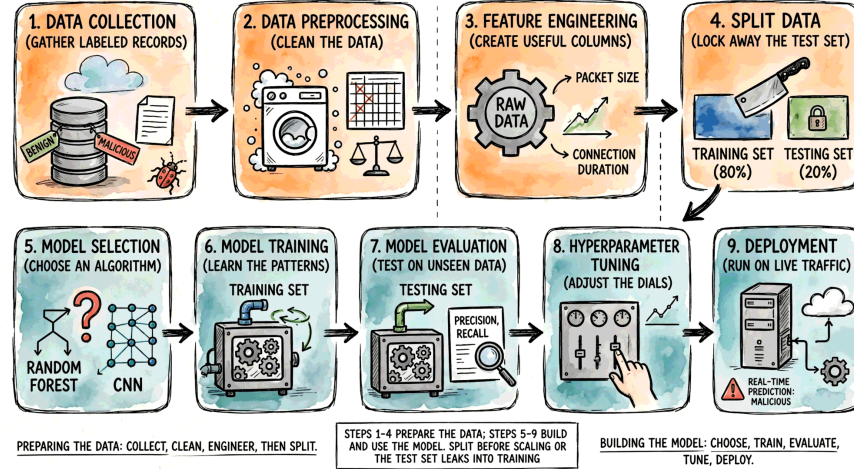
```

1 import numpy as np
2 import random
3
4 # --- RL EXAMPLE: Q-Learning Firewall Agent ---
5
6 # 1. Setup Environment
7 # Actions: 0 = Allow, 1 = Block
8 # State: A specific IP Address (e.g., "192.168.1.5")
9 q_table = { '192.168.1.5': [0.0, 0.0] } # [Value of Allow, Value of Block]
10
11 def get_reward(action, is_malicious):
12 if is_malicious and action == 1: return +10 # Good: Blocked attack
13 if is_malicious and action == 0: return -50 # Bad: Allowed attack (Breach)
14 if not is_malicious and action == 1: return -10 # Bad: Blocked user (Complaint)
15 return +5 # Good: Allowed user
16
17 # 2. Simulation Loop (Training)
18 ip = '192.168.1.5'
19 is_malicious = True # Ground truth (Agent doesn't know this yet!)
20 learning_rate = 0.5
21
22 print("--- Training Start ---")
23 for episode in range(5):
24 # Step A: Choose Action (Exploration vs Exploitation)
25 # Simple logic: mostly pick the action with highest current value
26 current_values = q_table[ip]
27 if random.random() < 0.5:
28 action = random.choice([0, 1]) # Explore (Random)
29 else:
30 action = np.argmax(current_values) # Exploit (Best known)
31
32 # Step B: Act and get Reward
33 reward = get_reward(action, is_malicious)
34
35 # Step C: Update Q-Table (Learn)
36 # New Value = Old Value + Learning Rate * Reward
37 old_value = q_table[ip][action]
38 q_table[ip][action] = old_value + (learning_rate * reward)
39
40 print(f"Ep {episode+1}: Action {'Block' if action==1 else 'Allow'} -> Reward {reward}")
41
42 print("--- Training End ---")
43 print(f"Final Q-Values for IP: {q_table[ip]}")
44 # Expected: Index 1 (Block) should have a much higher value than Index 0 (Allow).

```

Listing 8: حلقة Reinforcement Learning بسيطة لجدار حماية مستقل.

(2) تقسيم البيانات: قسّم مجموعة البيانات الخام فوراً إلى جزأين: *training set* و *testing set*. يتعلم النموذج من مجموعة التدريب فقط. يتم قتل مجموعة الاختبار واستخدامها فقط.



## شكل 14.2: سير عمل تعلم مشرف

للتقييم النهائي. التقسيم الشائع هو 80% للتدريب و 20% للاختبار.

```

1 import pandas as pd
2 from sklearn.model_selection import train_test_split
3
4 # 1. Data Collection (Mock Dataset)
5 # Features: Duration (sec), Source Bytes, Destination Bytes
6 data = {
7 'duration': [0.1, 5.2, 0.2, 12.0, 0.1, 0.5, 4.5, 1.2, 0.1, 3.0],
8 'src_bytes': [64, 5000, 128, 9000, 64, 256, 4500, 500, 64, 6000],
9 'dst_bytes': [1000, 0, 500, 0, 800, 1200, 0, 300, 950, 0],
10 'label': [0, 1, 0, 1, 0, 0, 1, 0, 0, 1] # 0: Benign, 1: Malicious
11 }
12 df = pd.DataFrame(data)
13
14 # 2. Data Splitting
15 X = df.drop('label', axis=1)
16 y = df['label']
17
18 # stratify=y ensures both sets maintain the same ratio of attacks
19 X_train, X_test, y_train, y_test = train_test_split(
20 X, y, test_size=0.2, random_state=42, stratify=y
21)

```

Listing 9: الخطوات 1 و 2: تحميل البيانات وتقسيمها فوراً لمنع التسرب.

(3) Feature Engineering: تحديد أو تحويل أو إنشاء خصائص عديدة، تسمى *features*، من البيانات الخام. على سبيل المثال، بدلاً من تغذية سجلات الحزم الخام في النموذج، قد تقوم بحساب "متوسط حجم الحزمة" أو "مدة الاتصال". تعتمد هذه الخطوة غالباً على المعرفة بـ *cybersecurity*.

(4) المعالجة المسبقة للبيانات: تنظيف بيانات التدريب وقياس الميزات العديدة (مثلاً، إجبار



القيم لتكون بين 0 و 1). بشكل حاسم، يجب عليك حساب معلمات القياس (مثل المتوسط) باستخدام بيانات التدريب فقط، ثم تطبيق تلك المعلمات نفسها على بيانات الاختبار.

```

1 from sklearn.preprocessing import StandardScaler
2
3 # 3. Feature Engineering
4 # Create a new feature: Bytes per Second (Transfer Rate)
5 # We apply the calculation to both sets independently
6 X_train['bytes_per_sec'] = X_train['src_bytes'] / (X_train['duration'] + 0.01)
7 X_test['bytes_per_sec'] = X_test['src_bytes'] / (X_test['duration'] + 0.01)
8
9 # 4. Data Preprocessing (Scaling)
10 scaler = StandardScaler()
11
12 # CRITICAL: We 'fit' the scaler ONLY on the Training data
13 X_train_scaled = scaler.fit_transform(X_train)
14
15 # We use those learned parameters to transform the Test data
16 X_test_scaled = scaler.transform(X_test)

```

Listing 10: الخطوات 3 و 4: هندسة الميزات وقياسها دون النظر إلى مجموعة الاختبار.

(5) اختيار النموذج: اختر خوارزمية machine learning مناسبة. على سبيل المثال، غالباً ما يكون مصنف Random Forest فعالاً للبيانات المهيكلة مثل شبكات الشبكة، بينما يكون Convolutional Neural Network (CNN) أفضل للبيانات غير المهيكلة مثل ثنائيات malware.

(6) ضبط المعلمات الفائقة: قبل التقييم النهائي، استخدم تقنية تسمى *Cross-Validation* على مجموعة التدريب للعثور على أفضل الإعدادات للنموذج (مثلاً، عدد الأشجار في Random Forest). يضمن ذلك تحسين النموذج دون اختلاس النظر إلى مجموعة الاختبار.

(7) تدريب النموذج: تدريب النموذج باستخدام أفضل المعلمات الفائقة على مجموعة بيانات التدريب الكاملة. تقوم الخوارزمية بضبط معلماتها الداخلية لتعيين ميزات الإدخال إلى تسميات الإخراج.

```

1 from sklearn.ensemble import RandomForestClassifier
2 from sklearn.model_selection import GridSearchCV
3
4 # 5. Model Selection (Random Forest)
5 rf = RandomForestClassifier(random_state=42)
6
7 # 6. Hyperparameter Tuning (using Cross-Validation on X_train)
8 param_grid = {'n_estimators': [10, 50], 'max_depth': [3, 5]}
9 grid_search = GridSearchCV(rf, param_grid, cv=2, scoring='f1')
10 grid_search.fit(X_train_scaled, y_train)
11
12 # 7. Model Training (Automatic via GridSearchCV)
13 best_model = grid_search.best_estimator_
14 print(f"Best Parameters: {grid_search.best_params_}")

```

Listing 11: الخطوات 5-7: اختيار وضبط وتدريب النموذج.

(8) تقييم النموذج: أخيراً، استخدم *testing set* المعزولة لتقييم الأداء. استخدم مقاييس

محددة مثل precision و recall و F1-score لقياس مدى جودة اكتشاف النموذج للتهديدات في البيانات التي لم يرها من قبل.

```
1 from sklearn.metrics import classification_report
2
3 # 8. Model Evaluation
4 # Predict using the locked-away X_test_scaled
5 y_pred = best_model.predict(X_test_scaled)
6
7 print("--- Final Test Set Evaluation ---")
8 print(classification_report(y_test, y_pred, target_names=['Benign', 'Malicious']))
```

Listing 12: الخطوة 8: التقييم النهائي على مجموعة الاختبار غير المرئية.

(9) النشر والمراقبة: دمج النموذج في نظام أمان مباشر لعمل تنبؤات في الوقت الفعلي. بمجرد نشره، يجب مراقبة النموذج باستمرار بحثاً عن تدهور الأداء الناجم عن تغيير أنماط الهجوم (concept drift).

```
1 # 9. Deployment (Simulation)
2 def analyze_packet(duration, src_bytes, dst_bytes):
3 # Create single-row DataFrame
4 live_data = pd.DataFrame([[duration, src_bytes, dst_bytes]],
5 columns=['duration', 'src_bytes', 'dst_bytes'])
6
7 # Repeat Feature Engineering
8 live_data['bytes_per_sec'] = live_data['src_bytes'] / (live_data['duration'] + 0.01)
9
10 # Scale using the TRAINING scaler
11 live_scaled = scaler.transform(live_data)
12
13 # Predict
14 prediction = best_model.predict(live_scaled)[0]
15 return "ALERT: Malicious" if prediction == 1 else "Benign"
16
17 # Simulate an incoming packet
18 print(analyze_packet(15.0, 10000, 0))
```

Listing 13: الخطوة 9: محاكاة نموذج منشور يقوم بعمل تنبؤ مباشر.

## 3.2 عائلات نماذج ML الرئيسية للأمن

تتناسب نماذج machine learning المختلفة مع أنواع مختلفة من بيانات ومهام الأمن. يمكننا تصنيفها على نطاق واسع إلى نماذج كلاسيكية، والتي تعمل بشكل جيد على البيانات المهيكلية (مثل السجلات)، ونماذج deep learning، التي تتفوق مع البيانات غير المهيكلية (مثل الكود الخлам أو النص).

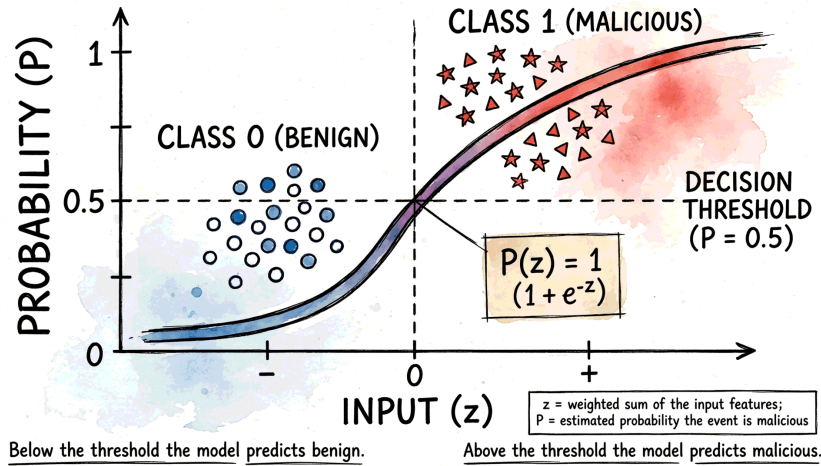
### 1.3.2 Classical Machine Learning

هذه النماذج هي المعيار الصناعي للعمل مع البيانات الجدولية. قبل القفز إلى الخوارزميات المعقدة، غالباً ما يبدأ محترفو الأمن هنا لإنشاء خط أساس للأداء.



## Logistic Regression

على الرغم من الاسم، هذه خوارزمية classification. إنها تقدر احتمالية أن ينتمي حدث ما إلى فئة معينة (مثلاً، "فرصة 75% أن يكون هذا spam"). إنها تعمل كنموذج خط أساس ممتاز لأنها سريعة وقابلة للتفسير وسهلة التنفيذ.



شكل 15.2: يستخدم Logistic Regression دالة sigmoid لتعيين التنبؤات إلى احتمالات بين 0 و 1.

```
from sklearn.linear_model import LogisticRegression

1. Initialize
clf = LogisticRegression()

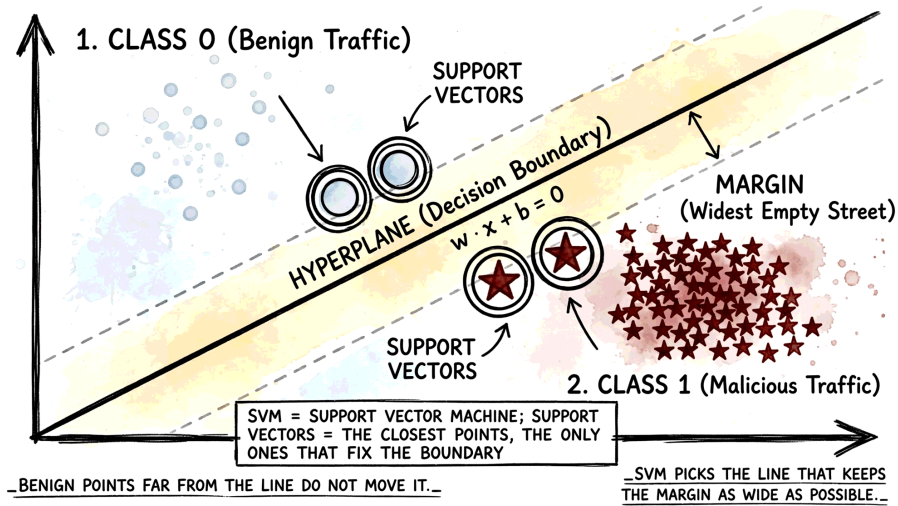
2. Train (Fit)
X_train: Matrix of features, y_train: Vector of labels
print("Training Logistic Regression...")
clf.fit(X_train, y_train)

3. Use
print("Prediction:", clf.predict([sample_data]))
```

## Listing 14: تنفيذ وتدريب Logistic Regression.

## Support Vector Machines (SVMs)

تعد SVMs قوية للتصنيف الثنائي. تخيل مجموعتين من نقاط البيانات (حميدة وضارة) على رسم بياني. يحاول SVM رسم خط مستقيم (أو مستوى مسطح في ثلاثي الأبعاد) يفصل بين هذه المجموعات بأكبر قدر ممكن من المساحة الفارغة، أو "الهامش"، بينهما.



شكل 16.2: يقوم SVM بتعظيم الهامش بين الفئتين.

```
from sklearn.svm import SVC

1. Initialize
Kernel='linear' creates a straight boundary; 'rbf' creates a curved one
clf = SVC(kernel='linear', C=1.0)

2. Train (Fit)
SVM training can be slow on very large datasets
print("Training SVM...")
clf.fit(X_train, y_train)
```

Listing 15: تنفيذ وتدريب SVM.

## Decision Trees

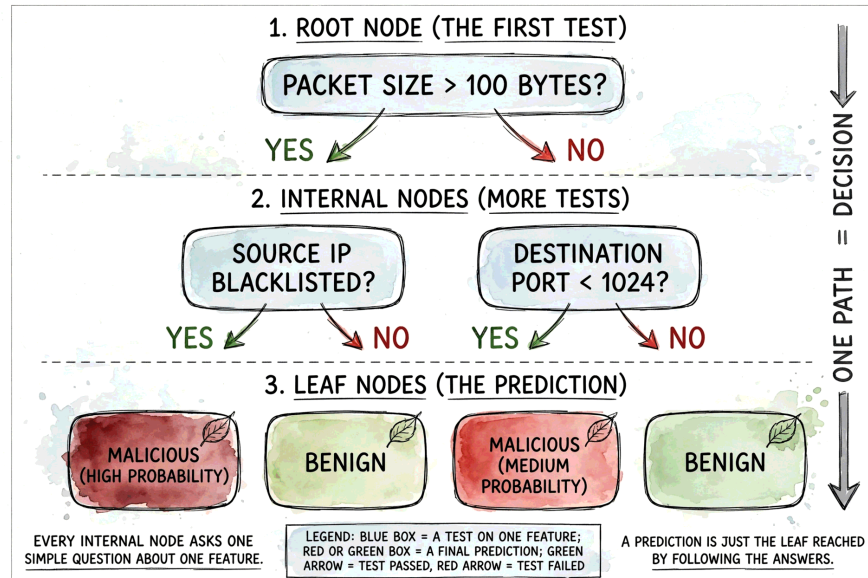
تستخدم هذه النماذج هيكلًا يشبه المخطط الانسيابي لاتخاذ القرارات. تمثل كل عقدة داخلية "اختبارًا" على ميزة (مثلاً، "هل حجم الحزمة < 100 بايت؟")، وتمثل كل عقدة ورقية تنبؤًا نهائيًا. إنها قابلة للتفسير للغاية ولكنها عرضة لـ "overfitting" (حفظ بيانات التدريب).

```
from sklearn.tree import DecisionTreeClassifier

1. Initialize
max_depth=5 restricts the tree growth to prevent overfitting
clf = DecisionTreeClassifier(max_depth=5)

2. Train (Fit)
print("Training Decision Tree...")
clf.fit(X_train, y_train)
```

Listing 16: تنفيذ وتدريب Decision Tree.



شكل 17.2: تقوم decision tree بعمل تنبؤات باتباع مسار من الاختبارات المنطقية.

## Ensemble Methods

تجمع هذه الطرق بين نماذج متعددة لتحسين الاستقرار والدقة.

- Random Forest: يبني مئات من decision trees على مجموعات فرعية عشوائية من البيانات ويأخذ متوسط تنبؤاتها.
- Gradient Boosting Machines (GBMs): خوارزميات مثل XGBoost تبني الأشجار بشكل تسلسلي، حيث تحاول كل شجرة جديدة إصلاح أخطاء الشجرة السابقة.

```
from sklearn.ensemble import RandomForestClassifier
from xgboost import XGBClassifier

1. Initialize
rf_model = RandomForestClassifier(n_estimators=100)
xgb_model = XGBClassifier(n_estimators=100, learning_rate=0.1)

2. Train (Fit)
Random Forest trains trees in parallel (fast)
rf_model.fit(X_train, y_train)

XGBoost trains trees sequentially (slower but often more accurate)
xgb_model.fit(X_train, y_train)
```

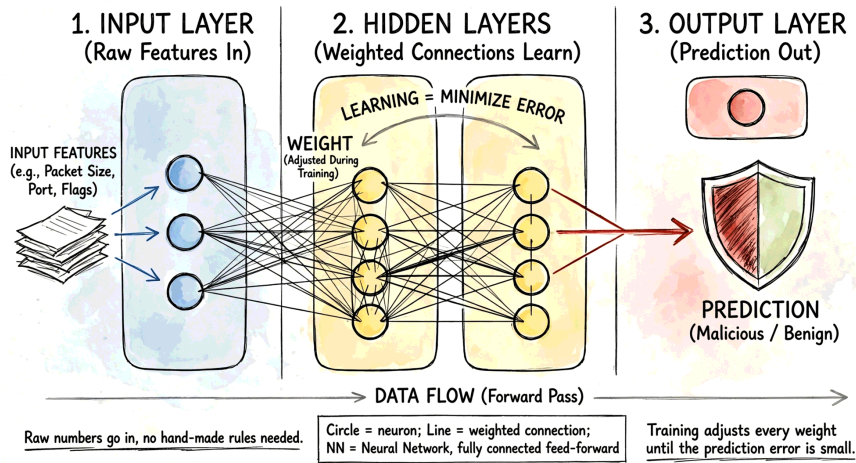
Listing 17: تدريب طرق Ensemble.

## 2.3.2 أساسيات Deep Learning

تستخدم نماذج Deep learning "الشبكات العصبية" لتعلم الأنماط المعقدة مباشرة من البيانات الخام غير المهيكلة دون هندسة ميزات يدوية.

### Neural Networks (NNs)

مستوحاة من الدماغ البشري، تتكون هذه من طبقات من "الخلايا العصبية" المترابطة. تتعلم الشبكة عن طريق ضبط "الأوزان" الرياضية للروابط بين الخلايا العصبية لتقليل الخطأ.



شكل 18.2: Feed-Forward Neural Network.

### Convolutional Neural Networks (CNNs)

صممت في الأصل لرؤية الكمبيوتر، تقوم CNNs بمسح البيانات بحثاً عن الأنماط المحلية. في الأمن، يمكننا تحويل ملف ثنائي malware إلى صورة ذات تدرج رمادي؛ ثم "تبحث" CNN عن القوام المرئي المقابل للكود الضار [44].

### LSTMs & Recurrent Neural Networks (RNNs)

تمتلك هذه الشبكات حلقة "ذاكرة" [32]، مما يسمح لها بمعالجة التسلسلات. هذا يجعلها مثالية لتحليل بيانات السلاسل الزمنية، مثل تسلسل مكالمات API التي يقوم بها البرنامج.

```

import torch
import torch.nn as nn
import torch.optim as optim

1. Define Network
model = nn.Sequential(nn.Linear(10, 64), nn.ReLU(), nn.Linear(64, 2))

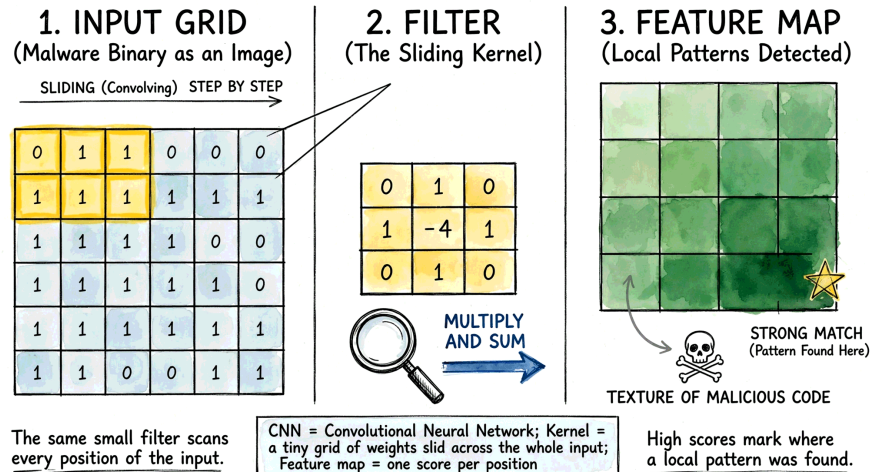
2. Define Loss and Optimizer
criterion = nn.CrossEntropyLoss()
optimizer = optim.Adam(model.parameters(), lr=0.001)

3. Training Step (Conceptual)
Dummy input: 5 samples, 10 features each
inputs = torch.randn(5, 10)
labels = torch.tensor([0, 1, 0, 1, 0])

optimizer.zero_grad() # Reset gradients
outputs = model(inputs) # Forward pass
loss = criterion(outputs, labels) # Calculate error
loss.backward() # Backward pass (calculate gradients)
optimizer.step() # Update weights
print(f"Loss: {loss.item():.4f}")

```

Listing 18: تعريف وتنفيذ خطوة تدريب واحدة لـ Neural Network.



شكل 19.2: تستخدم CNNs المرشحات لاكتشاف الأنماط المحلية في البيانات الشبكية بالشبكة.

```

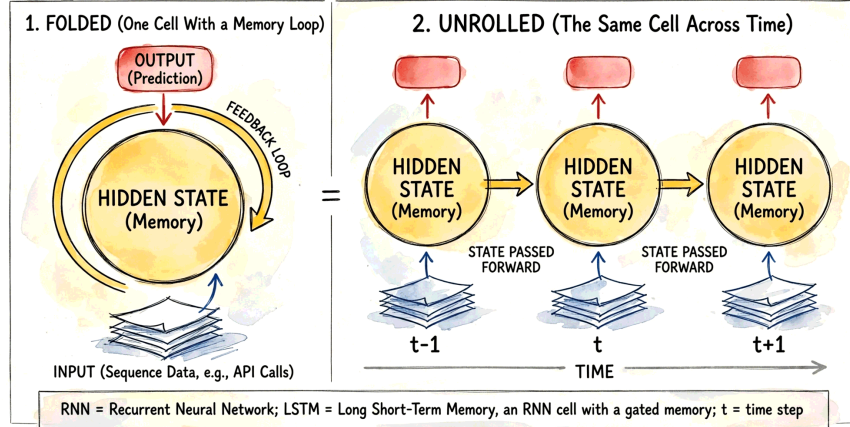
1D Conv Layer for sequence data (e.g., packet bytes)
conv_layer = nn.Conv1d(in_channels=1, out_channels=16, kernel_size=3)

Training Phase input preparation:
Input Shape: [Batch Size, Channels, Length]
e.g., 10 packets, 1 channel (raw byte), 1000 bytes long
input_data = torch.randn(10, 1, 1000)

Forward Pass: The filter slides over the sequence
feature_map = conv_layer(input_data)
print(f"Output shape: {feature_map.shape}")

```

Listing 19: معالجة البيانات من خلال Convolutional Layer.



The loop lets the cell remember what came before.

Unrolled, the loop is one cell reused at every step of the sequence.

شكل 20.2: تحافظ RNNs على حالة داخلية لمعالجة البيانات المتسلسلة.

```
LSTM Layer
lstm_layer = nn.LSTM(input_size=10, hidden_size=20, batch_first=True)

Training Phase input preparation:
Input Shape: [Batch Size, Sequence Length, Features per Event]
e.g., 5 sessions, 100 events long, 10 features per event
sequence_data = torch.randn(5, 100, 10)

Forward Pass
output contains features for every time step; hidden is the final state
output, (hidden, cell) = lstm_layer(sequence_data)
print(f"Encoded Sequence Shape: {output.shape}")
```

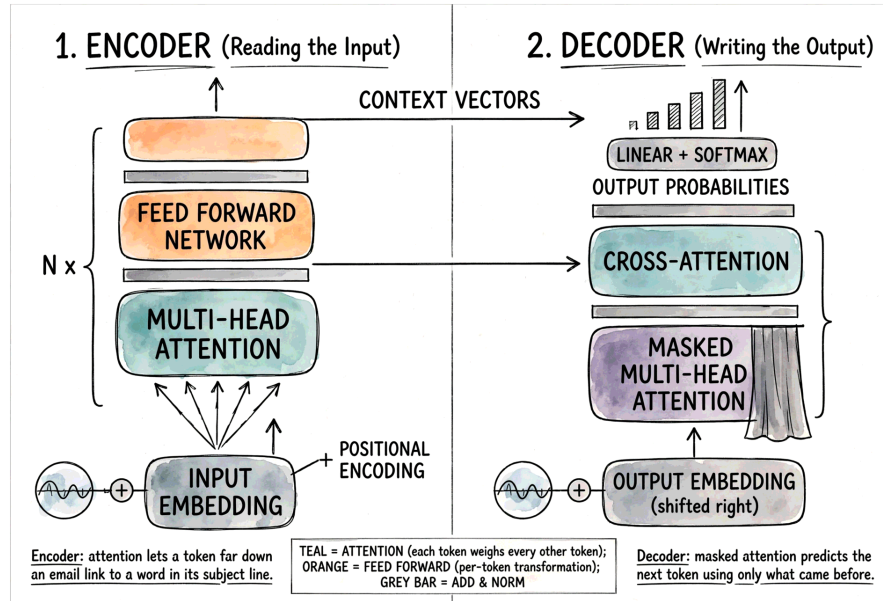
Listing 20: معالجة البيانات المتسلسلة من خلال LSTM.

### Transformer Architectures (مثل LLMs)

تستخدم Transformers [77] آلية تسمى Attention. يسمح هذا للنموذج بالتركيز ديناميكياً على الأجزاء ذات الصلة من تسلسل طويل --- على سبيل المثال، ربط "عنوان" URL مشبوه في نهاية نص البريد الإلكتروني بكلمة "عاجل" في سطر الموضوع --- بغض النظر عن المسافة بينهما.

### 3.3.2 Feature Engineering

Feature engineering هو فن استخدام المعرفة بالجمال لإنشاء ميزات عديدة تساعد الخوارزميات على التعلم. الميزة الجيدة تبسط المشكلة للنموذج.



شكل 21.2: معماريات Transformer.

```
from transformers import BertForSequenceClassification, Trainer, TrainingArguments

1. Load Pre-trained Model
model = BertForSequenceClassification.from_pretrained('bert-base-uncased')

2. Define Training Arguments
args = TrainingArguments(output_dir='./results', num_train_epochs=1)

3. Initialize Trainer (High-level API)
trainer = Trainer(
 model=model,
 args=args,
 train_dataset=train_dataset, # Assumes dataset is pre-loaded
 eval_dataset=test_dataset
)

4. Train
trainer.train()
```

Listing 21: ضبط دقيق لـ Transformer باستخدام واجهة برمجة تطبيقات Hugging Face Trainer.

## مميزات الأمن الشائعة

1. تحويل النص إلى أرقام (NLP) تحويل عنوان URL أو نص البريد الإلكتروني إلى أرقام. تتراوح الطرق من *Bag-of-Words* البسيط (عد تكرار الكلمات) إلى *Embeddings*.
- تظهر القائمة 22 تقنية *Bag-of-Words*. يتعلم النموذج أن كلمة "urgent" تظهر غالباً في phishing، بينما تظهر كلمة "meeting" في رسائل البريد الإلكتروني الحميدة.



---

```

1 from sklearn.feature_extraction.text import CountVectorizer
2 import pandas as pd
3
4 # --- TEXT EXAMPLE: Bag-of-Words for Email ---
5
6 # 1. Raw Data (Email Subjects)
7 emails = [
8 "Urgent: Update your password now", # Phishing
9 "Meeting agenda for tomorrow", # Benign
10 "Win a free iPhone instantly" # Phishing
11]
12
13 # 2. Transform to Numbers
14 # CountVectorizer creates a column for every unique word
15 vectorizer = CountVectorizer(stop_words='english')
16 X = vectorizer.fit_transform(emails)
17
18 # 3. View Result (The Feature Vector)
19 features = vectorizer.get_feature_names_out()
20 df_text = pd.DataFrame(X.toarray(), columns=features)
21
22 print(df_text)
23 # Result: A matrix where columns are words ('urgent', 'win', 'meeting')
24 # and values are counts (1 or 0).

```

---

## Listing 22: تحويل النص إلى متجهات عديدة باستخدام Bag-of-Words.

2. الميزات الشبكية والزمنية تجميع بيانات الحزم واستخراج الأنماط القائمة على الوقت. غالباً ما يعمل المهاجمون خارج ساعات العمل لتجنب الكشف، مما يجعل "ساعة اليوم" ميزة قوية. توضح القائمة 23 استخراج الساعة من طابع زمني وحساب البايتات في الثانية من بيانات التدفق.

---

```

1 import pandas as pd
2
3 # --- NETWORK/TEMPORAL EXAMPLE: Log Parsing ---
4
5 # 1. Raw Log Data
6 df = pd.DataFrame({
7 'timestamp': ['2026-01-15 03:15:00', '2026-01-15 14:30:00'],
8 'bytes_sent': [15000, 450],
9 'duration_sec': [5.0, 0.5]
10 })
11
12 # 2. Feature Engineering
13 # Extract 'Hour' (Attackers love 3 AM)
14 df['timestamp'] = pd.to_datetime(df['timestamp'])
15 df['hour'] = df['timestamp'].dt.hour
16
17 # Calculate Transfer Rate
18 df['transfer_rate'] = df['bytes_sent'] / df['duration_sec']
19
20 print(df[['hour', 'transfer_rate']])
21 # Result: Numerical features ready for analysis.

```

---

## Listing 23: استخراج الميزات الزمنية والإحصائية من سجلات الشبكة.

3. تحليل الملفات التنفيذية (الإترويا) استخراج الخصائص الثابتة من malware. المقياس الرئيسي هو Entropy (العشوائية). الكود الشرعي له هيكل (إترويا منخفضة)، بينما تبدو malware المعبأة أو المشفرة عشوائية (إترويا عالية). تحسب القائمة 24 إترويا شانون لرأس الملف.



---

```

1 import numpy as np
2 from scipy.stats import entropy
3 import collections
4
5 # --- MALWARE EXAMPLE: Calculating File Entropy ---
6
7 def calc_entropy(data_bytes):
8 # Count frequency of each byte (0-255)
9 counts = collections.Counter(data_bytes)
10 probs = [count / len(data_bytes) for count in counts.values()]
11 # Calculate Shannon Entropy
12 return entropy(probs, base=2)
13
14 # 1. Simulate Files
15 legitimate_header = b'\x00\x00\x00\x01\x00\x00\x00\x02' # Structured (Zeros)
16 encrypted_payload = b'\x8a\xf1\x23\x99\x1b\xcd\x5e\x4a' # Random
17
18 # 2. Compare Features
19 print(f"Legit Entropy: {calc_entropy(legitimate_header):.2f}")
20 # Expected: Low (e.g., 1.0) due to repetition
21
22 print(f"Malware Entropy: {calc_entropy(encrypted_payload):.2f}")
23 # Expected: High (e.g., 3.0) due to randomness

```

---

Listing 24: استخدام Entropy لتمييز الكود المهيكل عن malware المشفر.

## 4.3.2 تطبيع البيانات وقياسها

خطوة حاسمة، غالباً ما يتم تجاهلها، هي Normalization. غالباً ما يكون للميزات مقاييس مختلفة تماماً. على سبيل المثال، قد يكون "طول عنوان URL" 200، بينما "ساعة اليوم" 23 فقط. إذا تركنا هذه الأرقام كما هي، فسيقتض النماذج أن طول عنوان URL أكثر أهمية من الناحية الحسابية. لإصلاح ذلك، نقوم بقياس الميزات العددية إلى نطاق مشترك (عادة من 0 إلى 1). تُظهر القائمة 25 كيف أن الفشل في قياس البيانات يمكن أن يشوه الأرقام، وكيف يصلح MinMaxScaler ذلك.

---

```

1 from sklearn.preprocessing import MinMaxScaler
2 import pandas as pd
3
4 # --- SCALING EXAMPLE ---
5
6 # 1. Unscaled Data
7 # 'Bytes' is huge (10,000), 'Hour' is tiny (23).
8 df = pd.DataFrame({
9 'bytes': [100, 10000, 5000],
10 'hour': [1, 23, 12]
11 })
12
13 print("Original (Model will ignore 'hour'):")
14 print(df)
15
16 # 2. Apply Scaling
17 # Squeezes everything between 0.0 and 1.0
18 scaler = MinMaxScaler()
19 df_scaled = pd.DataFrame(scaler.fit_transform(df), columns=df.columns)
20
21 print("\nScaled (Model treats features equally):")
22 print(df_scaled.round(2))

```

---

Listing 25: تطبيع الميزات حتى لا تهيمن الأرقام الكبيرة على النموذج.

يوضح الكود في القائمة 26 نهج Manual Feature Engineering، موضحاً كيفية إنشاء ميزات الأمان وقياسها بشكل صحيح لنموذج كلاسيكي.

---

```

1 import pandas as pd
2 from sklearn.preprocessing import OneHotEncoder, MinMaxScaler
3
4 # 1. Raw Data (Web Logs with Timestamps)
5 df = pd.DataFrame({
6 'timestamp': ['2026-01-15 08:00:00', '2026-01-15 23:45:00', '2026-01-15 14:30:00'],
7 'method': ['GET', 'POST', 'GET'],
8 'url': ['/index.html', '/admin/login', '/api/data/' * 50] # Last one is a buffer overflow attempt
9 })
10
11 print("--- 1. Raw Data ---")
12 print(df)
13
14 # --- FEATURE ENGINEERING ---
15
16 # Feature 1: URL Length
17 # Context: Extremely long URLs often indicate Buffer Overflows or SQL Injection strings.
18 df['url_length'] = df['url'].apply(len)
19
20 # Feature 2: Dangerous Keywords
21 # Context: The presence of 'admin', 'root', or 'cmd' suggests targeted privilege escalation.
22 df['has_admin_keyword'] = df['url'].apply(lambda x: 1 if 'admin' in x else 0)
23
24 # Feature 3: Temporal Analysis (Request Hour)
25 # Context: Anomalous activity often occurs outside business hours (e.g., 3 AM).
26 df['timestamp'] = pd.to_datetime(df['timestamp'])
27 df['request_hour'] = df['timestamp'].dt.hour
28
29 # --- DATA PREPARATION (Encoding & Scaling) ---
30
31 # 4. One-Hot Encoding for Categorical Data
32 # We convert 'method' (GET, POST) into separate columns (method_GET, method_POST).
33 # We do NOT use 0, 1, 2 for categories, as that implies rank (POST > GET).
34 encoder = OneHotEncoder(sparse_output=False)
35 method_encoded = encoder.fit_transform(df[['method']])
36 method_df = pd.DataFrame(method_encoded, columns=encoder.get_feature_names_out(['method']))
37 df = pd.concat([df, method_df], axis=1)
38
39 # 5. Normalization (Scaling)
40 # We scale 'url_length' and 'request_hour' to the 0-1 range.
41 scaler = MinMaxScaler()
42 cols_to_scale = ['url_length', 'request_hour']
43 df[cols_to_scale] = scaler.fit_transform(df[cols_to_scale])
44
45 print("\n--- 2. Final Feature Vector (Ready for ML) ---")
46 # Drop the raw columns that the model cannot understand
47 final_df = df.drop(['timestamp', 'method', 'url'], axis=1)
48 print(final_df.round(2))

```

---

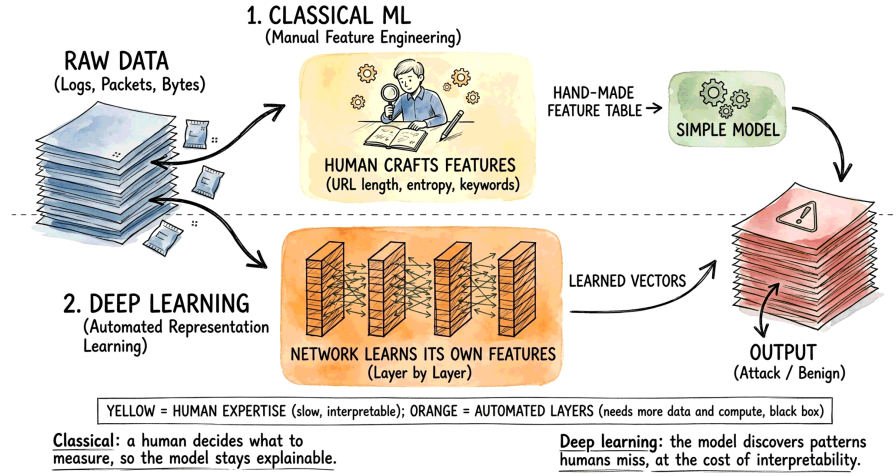
Listing 26: هندسة الميزات (طول عنوان URL، الكلمات الرئيسية، الوقت) وتطبيعها.

## 5.3.2 Representation Learning

بينما تعد هندسة الميزات قوية، إلا أنها تستغرق وقتاً طويلاً وتتطلب معرفة عميقة من الخبراء. Representation learning هو البديل الآلي المستخدم بواسطة Deep Learning.

بدلاً من أن يقوم الإنسان يدوياً بإخبار النموذج "ابحث عن السلسلة 'admin' أو 'احسب الإنتروبيا"، يقوم نموذج Deep Learning (مثل CNN أو Transformer) باستيعاب البيانات الخام و يتعلم أفضل الميزات بنفسه. يكتشف تلقائياً أن أنماط بايت معينة أو تسلسلات نصية مرتبطة بالهجمات.

- الميزة: يزيل الحاجة إلى صياغة الميزات اليدوية؛ يمكنه العثور على أنماط معقدة قد يفوتها البشر.
- العيوب: يتطلب المزيد من البيانات وقوة الحوسبة؛ النموذج الناتج عبارة عن "صندوق أسود" يصعب تفسيره.



شكل 22.2: هندسة الميزات اليدوية مقابل Representation Learning الآلي.

توضح القائمة 27 كيف تقوم Neural Network تلقائياً بتحويل مكالمات API للنظام الخام (أعداد صحيحة) إلى متجهات عددية كثيفة. على عكس مثال "Bag-of-Words" حيث قمنا بعد الكلمات، هنا يبدأ النموذج بأرقام عشوائية و يتعلم أفضل تمثيل أثناء التدريب.

```

1 import torch
2 import torch.nn as nn
3
4 # --- REPRESENTATION LEARNING EXAMPLE: API Call Embeddings ---
5
6 # 1. Raw Data (Sequence of System API Calls)
7 # Imagine we map calls to IDs: 0='NtOpenProcess', 1='NtWriteFile', 2='NtClose'
8 # A malware trace might look like: Open -> Write -> Close
9 raw_sequence = torch.tensor([[0, 1, 2]])
10
11 print(f"Raw Input (Human IDs): {raw_sequence}")
12
13 # 2. Define the Representation Layer (Embedding)
14 # We ask the Neural Network: "Learn a vector of size 3 for every API call."
15 # num_embeddings=10 (vocabulary size), embedding_dim=3 (feature size)
16 embedding_layer = nn.Embedding(num_embeddings=10, embedding_dim=3)
17
18 # 3. Automatic Feature Extraction
19 # The model converts the integers [0, 1, 2] into vectors automatically.
20 # Crucially, these numbers are 'parameters' that improve as the model trains.
21 learned_features = embedding_layer(raw_sequence)
22
23 print("\nLearned Representation (Model's Internal Features):")
24 # Each row represents one API call
25 print(learned_features.detach().numpy())
26
27 # Result:
28 # [[-0.45, 1.21, -0.05] <- The model's numerical view of 'NtOpenProcess'
29 # [0.88, -0.12, 0.33] <- The model's numerical view of 'NtWriteFile'
30 # ...]

```

Listing 27: استخدام Embedding Layer لتعلم التمثيلات العددية لمكالمات API تلقائياً.

## 4.2 مقاييس التقييم المصممة للأمن

قياس أداء النموذج بشكل صحيح أمر بالغ الأهمية. في cybersecurity، تكون تكلفة الخطأ—تفويت هجوم حقيقي أو حظر رئيس تنفيذي شرعي—مرتفعة للغاية. علاوة على ذلك، غالباً ما تفشل المقاييس القياسية لأن مجموعات بيانات الأمن غير متوازنة بشكل كبير.

### 1.4.2 ما وراء الدقة البسيطة: مغالطة المعدل الأساسي

في شبكة نموذجية، تكون الحزم الضارة نادرة. تخيل مجموعة بيانات تحتوي على 99,900 حزمة حميدة و 100 حزمة ضارة فقط. إذا تنبأ النموذج ببساطة بـ "حميد" لكل شيء، فإنه يحقق دقة 99.99%. ومع ذلك، فقد فشل في 100% من وظيفته لأنه فوت كل هجوم واحد. هذا هو السبب في أننا لا نعتمد أبداً على الدقة البسيطة لمهام الأمن.

### 2.4.2 نوع المقياس 1: مقاييس Classification

Classification هي المهمة الأكثر شيوعاً (مثلاً، جدار الحماية: حظر/سماع). لتقييمها، نقوم أولاً بتعيين النتائج في Confusion Matrix (الجدول 1.2).

| الفعلي: حميد                                | الفعلي: هجوم                              |            |
|---------------------------------------------|-------------------------------------------|------------|
| False Positive (FP)<br>(إنذار كاذب)         | True Positive (TP)<br>(هجوم تم اكتشافه)   | تنبؤ: هجوم |
| True Negative (TN)<br>(تم تجاهله بشكل صحيح) | False Negative (FN)<br>(اختراق تم تفويته) | تنبؤ: حميد |

جدول 1.2: Confusion Matrix.

من هذه المصفوفة، نحسب مقاييس محددة:

• Precision: عندما ينبه النظام، كم مرة يكون حقيقياً؟ (الدقة العالية تقلل من إجهاد المحلل).

$$Precision = \frac{TP}{TP + FP} \quad (1)$$

- Recall (Sensitivity): من بين جميع الهجمات الحقيقية، كم عدد الهجمات التي اكتشفناها؟ (الاستدعاء العالي يمنع الاختراقات).

$$Recall = \frac{TP}{TP + FN} \quad (2)$$

- F1-Score: المتوسط التوافقي ل Precision و Recall. الأفضل لمقارنة النماذج على البيانات غير المتوازنة.

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (3)$$

توضيح القائمة 28 تقييم نظام كشف التسلل (IDS).

```

1 from sklearn.metrics import classification_report, confusion_matrix
2 import numpy as np
3
4 # --- CLASSIFICATION EVALUATION: IDS ---
5
6 # 1. Ground Truth (0: Benign, 1: Attack)
7 # In reality, attacks are rare (imbalanced).
8 y_true = np.array([0, 0, 0, 0, 0, 0, 0, 1, 1, 1])
9
10 # 2. Model Predictions
11 # The model caught 2 attacks (TP), missed 1 (FN), and raised 1 false alarm (FP).
12 y_pred = np.array([0, 0, 0, 0, 0, 0, 1, 1, 1, 0])
13
14 # 3. Evaluation
15 print("Confusion Matrix:\n", confusion_matrix(y_true, y_pred))
16 print("\nDetailed Report:\n", classification_report(y_true, y_pred, target_names=['Benign', 'Attack']))
17
18 # Analysis:
19 # Precision (Attack) = 2/3 (0.67) -> Of 3 alerts, 2 were real.
20 # Recall (Attack) = 2/3 (0.67) -> Of 3 real attacks, we caught 2.

```

Listing 28: تقييم مصنف باستخدام تقرير التصنيف الخاص بـ Scikit-Learn.

## 3.4.2 نوع المقياس 2: مقاييس Regression

عند التنبؤ بقيم عديدة (مثلاً، التنبؤ بـ Risk Score من 0 إلى 10)، لا تنطبق "الدقة". بدلاً من ذلك، نقيس Error (المسافة بين التنبؤ والواقع).

- Mean Absolute Error (MAE): متوسط الفرق. مثلاً، "درجة المخاطرة المتوقعة لدينا عادة ما تكون خاطئة بمقدار 5.1 نقطة." سهل التفسير.

- Mean Squared Error (MSE): يقوم بتربيع الأخطاء قبل حساب المتوسط. هذا يعاقب بشدة الأخطاء الكبيرة. في الأمن، تفويت درجة المخاطرة بمقدار 5 نقاط أسوأ بكثير من تفويتها بمقدار نقطة واحدة، لذا غالباً ما يتم تفضيل MSE.

تقيم القائمة 29 نموذجاً يتنبأ بحدة الثغرة الأمنية (CVSS).

---

```

1 from sklearn.metrics import mean_absolute_error, mean_squared_error
2 import numpy as np
3
4 # --- REGRESSION EVALUATION: Risk Scoring ---
5
6 # 1. Actual CVSS Scores (0.0 to 10.0)
7 y_true = np.array([2.5, 5.0, 9.0, 4.0])
8
9 # 2. Predicted Scores
10 # The last prediction (4.0 vs 9.0) is a catastrophic error.
11 y_pred = np.array([2.0, 5.2, 4.0, 4.1])
12
13 # 3. Evaluation
14 mae = mean_absolute_error(y_true, y_pred)
15 mse = mean_squared_error(y_true, y_pred)
16
17 print(f"MAE: {mae:.2f}") # Result: 1.45 (Average error is small)
18 print(f"MSE: {mse:.2f}") # Result: 6.27 (High because of the one huge error)

```

---

Listing 29: تقييم أخطاء regression. لاحظ كيف يبرز MSE الخطأ الشاذ الكبير.

## 4.4.2 نوع المقياس 3: مقاييس Clustering

تقييم unsupervised learning أصعب لأننا غالباً ما نفتقر إلى التسميات "الصحيحة".

• Silhouette Score: يقيس مدى تمايز العناقيد. يتراوح من -1 إلى 1.

○ +1: عناقيد مثالية (كثيفة ومنفصلة جيداً).

○ 0: عناقيد متداخلة.

○ -1: نقاط بيانات مخصصة للعنقود الخطأ.

تقيم القائمة 30 مدى جودة تجميع النموذج لعائلات Malware.

---

```

1 from sklearn.metrics import silhouette_score
2 import numpy as np
3
4 # --- CLUSTERING EVALUATION: Malware Grouping ---
5
6 # 1. Feature Space (2D for simplicity: [File Size, Entropy])
7 X_features = np.array([
8 [100, 0.5], [105, 0.6], [102, 0.55], # Group A (e.g., Adware)
9 [500, 7.5], [510, 7.6], [505, 7.4] # Group B (e.g., Ransomware)
10])
11
12 # 2. Cluster Labels assigned by model (e.g., K-Means)
13 labels = np.array([0, 0, 0, 1, 1])
14
15 # 3. Evaluation (Internal Validity)
16 # Do these groups make mathematical sense?
17 score = silhouette_score(X_features, labels)
18
19 print(f"Silhouette Score: {score:.2f}")
20 # Result: ~0.98 (Very High). The groups are clearly separated.

```

---

Listing 30: استخدام Silhouette Score للتحقق من جودة العنقود دون حقيقة أساسية.

## 5.4.2 قانون التوازن: ضبط العتبة

لا يوجد نموذج "مثالي". يجب علينا اختيار مقايضة بناءً على تكلفة الأخطاء. يتم تنفيذ هذا الاختيار عن طريق ضبط **Decision Threshold**. تخرج معظم النماذج درجة احتمالية (0.0 إلى 0.1). بشكل افتراضي، نصنف أي شيء أعلى من 5.0 على أنه "Malicious". ومع ذلك، يمكننا تحويل هذا الخط:

- عتبة منخفضة (مثلاً،  $2.0 <$ ): إعداد "البارانويا". نصنف العناصر على أنها ضارة حتى لو كان النموذج مشبوهاً قليلاً فقط.

- النتيجة: Recall عالي (نفوت القليل جداً)، ولكن Precision منخفض (العديد من False Positives).

- حالة الاستخدام: Threat Hunting. يريد المحللون رؤية كل شيء مشبوه، حتى لو كان ذلك يخلق المزيد من العمل.

- عتبة عالية (مثلاً،  $9.0 <$ ): الإعداد "المحافظ". نتصرف فقط إذا كان النموذج واثقاً للغاية.

- النتيجة: Precision عالي (عدد قليل جداً من الإنذارات الكاذبة)، ولكن Recall أقل (قد نفوت هجمات دقيقة).

- حالة الاستخدام: الحظر الآلي (IPS). لا يمكننا تحمل حظر بريد الرئيس التنفيذي، لذلك نحظر فقط عندما نكون متأكدين بنسبة 99%.

توضح القائمة 31 كيف يغير تغيير العتبة بشكل كبير سلوك النموذج باستخدام نفس مجموعة التنبؤات.

## 5.2 تحديات ML في Cybersecurity

تطبيق machine learning في cybersecurity لا يخلو من الصعوبات. على عكس مجموعات البيانات الأكاديمية القياسية، فإن بيانات الأمن عداوية ومشفرة وغير مستقرة للغاية.

### 1.5.2 Imbalanced Data

عادة ما تكون حالات الهجوم أقل شيوعاً بكثير من الأحداث العادية (غالباً بنسبة 1:10,000). سيصبح النموذج المدرب على هذا كسولاً، حيث يتنبأ بـ "حميد" 100% من الوقت لتحقيق دقة عالية بينما يفوت كل هجوم.

الحل الممكن: نستخدم تقنيات مثل SMOTE (Synthetic Minority Over-sampling Technique) لتوليد أمثلة وهمية للهجمات، مما يجبر النموذج على الانتباه إليها.

---

```

1 import numpy as np
2 from sklearn.metrics import precision_score, recall_score, confusion_matrix
3
4 # --- THRESHOLD TUNING EXAMPLE ---
5
6 # 1. Ground Truth (10 Attacks, 10 Benign)
7 y_true = np.array([0]*10 + [1]*10)
8
9 # 2. Model Probabilities (0.0 to 1.0)
10 # The model is unsure about several items (scores around 0.4 - 0.6)
11 y_proba = np.array([
12 0.1, 0.2, 0.1, 0.1, 0.45, 0.55, 0.1, 0.1, 0.1, 0.1, # Benign samples
13 0.9, 0.8, 0.9, 0.95, 0.40, 0.60, 0.85, 0.9, 0.9, 0.9 # Malicious samples
14])
15
16 def evaluate_threshold(threshold):
17 # Apply threshold: If proba > threshold, predict 1 (Attack)
18 y_pred = (y_proba > threshold).astype(int)
19
20 prec = precision_score(y_true, y_pred)
21 rec = recall_score(y_true, y_pred)
22 tn, fp, fn, tp = confusion_matrix(y_true, y_pred).ravel()
23
24 print(f"--- Threshold: {threshold} ---")
25 print(f"Precision: {prec:.2f} | Recall: {rec:.2f}")
26 print(f"False Alarms (FP): {fp} | Missed Attacks (FN): {fn}\n")
27
28 # Scenario A: Paranoid Mode (Threat Hunting)
29 # We want to catch everything, even if we investigate false leads.
30 evaluate_threshold(0.3)
31
32 # Scenario B: Conservative Mode (Automated Blocking)
33 # We strictly avoid blocking legitimate users (Zero False Positives desired).
34 evaluate_threshold(0.8)
35
36 # OUTPUT ANALYSIS:
37 # Threshold 0.3: Recall is 0.90 (Caught 9/10), but we have 2 False Alarms.
38 # Threshold 0.8: Precision is 1.00 (0 False Alarms), but Recall drops to 0.80.

```

---

### Listing 31: توضيح كيف تتحكم Decision Threshold في المقايضة بين Precision و Recall.

---

```

1 from imblearn.over_sampling import SMOTE
2 from collections import Counter
3
4 # 1. Imbalanced Dataset (99 Benign, 1 Attack)
5 X = [[1.0, 2.0]] * 99 + [[9.0, 9.0]] * 1
6 y = [0] * 99 + [1] * 1
7
8 print(f"Original Counts: {Counter(y)}")
9 # Output: {0: 99, 1: 1}
10
11 # 2. Apply SMOTE
12 # It creates synthetic attacks by interpolating between existing ones
13 smote = SMOTE(k_neighbors=1) # k=i because we only have 1 sample!
14 X_res, y_res = smote.fit_resample(X, y)
15
16 print(f"Resampled Counts: {Counter(y_res)}")
17 # Output: {0: 99, 1: 99} -> Perfectly balanced!

```

---

### Listing 32: استخدام SMOTE لإصلاح عدم توازن الفئات قبل التدريب.

## 2.5.2 تحدي التشفير ("الظلام الدامس")

تاريخياً، كانت أدوات الأمن تفحص "حمولة" الخزم (مثلاً، البحث عن كلمة "malware" في نص HTML اليوم، أكثر من 90% من حركة مرور الويب مشفرة HTTPS/TLS (3.1)، مما يجعل الحمولات غير قابلة للقراءة.

الحل الممكن: تحول Machine Learning إلى Encrypted Traffic Analysis (ETA). بدلاً من قراءة المحتوى، تقوم النماذج بتحليل البيانات الوصفية:



• تسلسل أطوال الحزم.

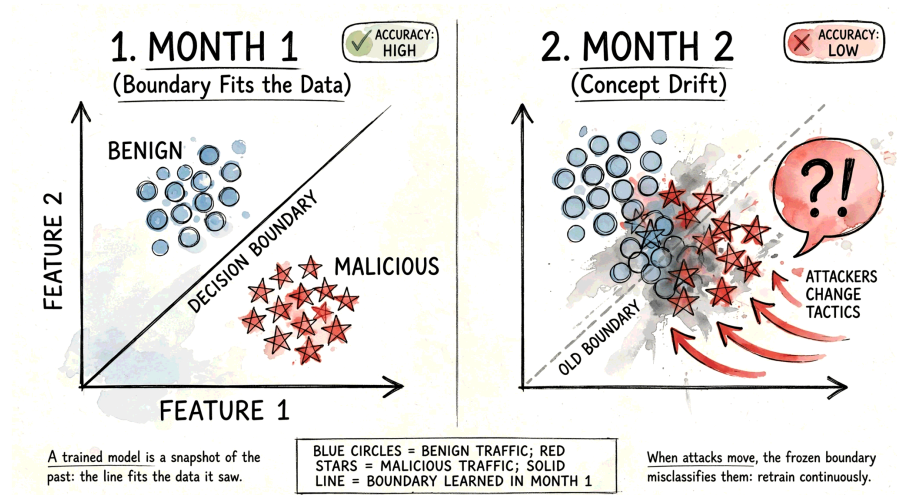
• أوقات الوصول البيني (التوقف المؤقت بين الحزم).

• أنماط المصافحة (مجموعات التشفير المعروضة).

تعد نماذج Deep Learning (مثل LSTMs) جيدة بشكل خاص في أخذ بصمات التطبيقات (مثلاً، تمييز حركة مرور Netflix عن سرقة البيانات) بناءً فقط على أنماط التوقيت هذه.

### 3.5.2 Concept Drift

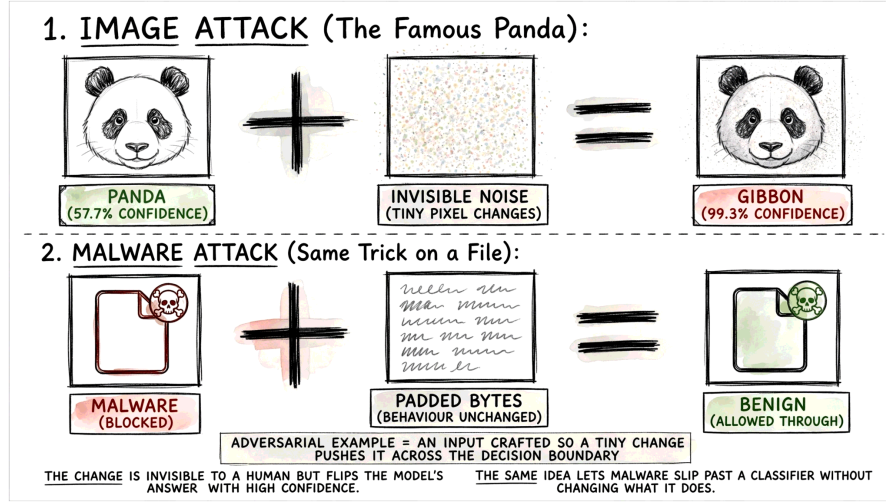
يحدث Concept drift عندما يتغير سلوك الهجمات بمرور الوقت (الشكل 23.2). نموذج ML هو لقطة من الماضي. إذا تم تدريبه على malware من عام 2020، فمن المحتمل أن يفشل ضد ransomware من عام 2026. الحل الممكن: خطوط أنابيب إعادة التدريب المستمرة مطلوبة. في الأمن، غالباً ما يعتبر النموذج "قديمًا" بعد بضعة أسابيع فقط.



شكل 23.2: Concept Drift: مع تغيير المهاجمين لتكتيكاتهم، تصبح حدود القرار القديمة بالية.

## 4.5.2 Adversarial Examples

Adversarial example هو مدخل تم صياغته بعناية لخداع الذكاء الاصطناعي. يمكن للمهاجم إضافة "ضوضاء" غير مرئية إلى ملف malware تغير توقعه الرياضي بما يكفي لعبور حدود القرار من "Malware" إلى "حميد".



شكل 24.2: Adversarial Attacks: إضافة ضوضاء غير مرئية لخداع النموذج.

## 5.5.2 قابلية التفسير (Explainable AI - XAI)

نماذج Deep Learning هي "صناديق سوداء". إذا قام جدار حماية بحظر كمبيوتر محمول للرئيس التنفيذي قائلاً "99% Malicious"، يحتاج محل الأمن إلى معرفة السبب للتحقق من التنبيه. الحل الممكن: نستخدم أدوات مثل SHAP (SHapley Additive exPlanations) لحساب الميزة المحددة التي رفعت الاحتمالية.

## 6.2 مشروع عملي: سير عمل ML أساسي لتصنيف التهديدات

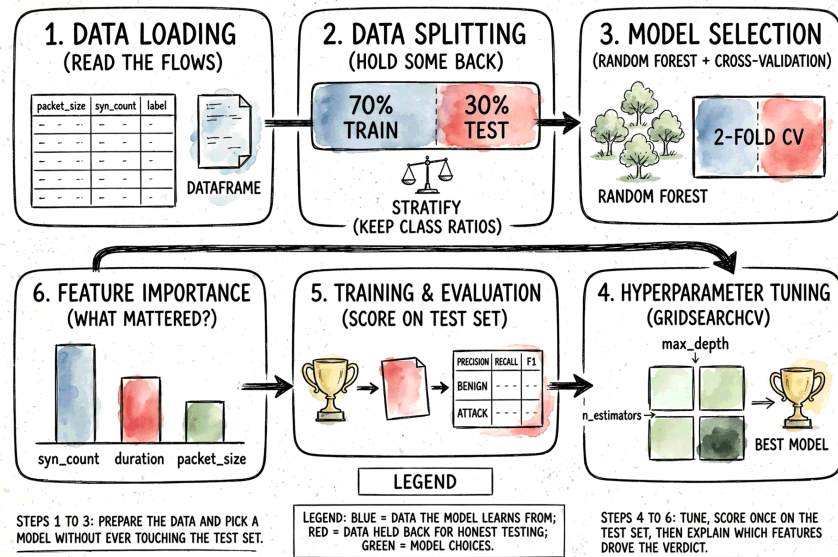
سيرشك هذا المشروع عبر سير عمل supervised machine learning كامل وأساسي. سنقوم ببناء نظام بسيط لكشف التسلسل للشبكة (NIDS) باستخدام مصنف Random Forest (الشكل 25.2).

```

1 import shap
2 import xgboost
3
4 # 1. Train a model (Black Box)
5 model = xgboost.XGBClassifier().fit(X_train, y_train)
6
7 # 2. Use SHAP to explain a single prediction
8 explainer = shap.Explainer(model)
9 shap_values = explainer(X_test)
10
11 # 3. Visualize
12 # This tells the analyst: "Blocked because 'Packet_Size' > 1500"
13 shap.plots.waterfall(shap_values[0])

```

Listing 33: استخدام SHAP لشرح سبب وضع علامة على حدث معين كهجوم.



شكل 25.2: سير عمل ML أساسي لتصنيف التهديدات

## 1.6.2 الخطوة 1: تحميل البيانات

أولاً، ننشئ مجموعة البيانات الخاصة بنا في سيناريو العالم الحقيقي، ستقوم بتحميل هذا من ملف CSV (مثلاً، `pd.read_csv('network_logs.csv')`). هنا، ننشئ مجموعة بيانات اصطناعية صغيرة تمثل تدفقات الشبكة مع ميزات مثل حجم الحزمة وعدد أعلام SYN.

---

```

import pandas as pd
import numpy as np

1. Create a dummy dataset mimicking network traffic
data = {
 # Feature A: Average size of packets in the flow (Bytes)
 'packet_size': [64, 1500, 64, 200, 64, 1500, 500, 64, 1500, 64],
 # Feature B: Number of SYN flags (High counts often indicate scanning/DoS)
 'syn_count': [0, 0, 1, 50, 0, 0, 20, 0, 0, 100],
 # Feature C: Duration of the connection (Seconds)
 'duration': [0.1, 5.2, 0.1, 1.5, 0.2, 10.0, 0.5, 0.1, 8.5, 2.0],
 # Target Label: 0 = Benign, 1 = Attack
 'label': [0, 0, 0, 1, 0, 0, 1, 0, 0, 1]
}

df_flows = pd.DataFrame(data)
X = df_flows.drop('label', axis=1) # Features
y = df_flows['label'] # Target labels

print("--- Data Loaded ---")
print(X.head())

```

---

Listing 34: الخطوة 1: تحميل وفحص مجموعة بيانات cybersecurity.

## 2.6.2 الخطوة 2: تقسيم البيانات

نقوم بتقسيم بياناتنا إلى مجموعة تدريب ومجموعة اختبار. نستخدم `stratify=y` لضمان أنه إذا كان 30% من إجمالي بياناتنا ضاراً، فإن 30% من مجموعة التدريب و 30% من مجموعة الاختبار ستكون ضارة أيضاً. هذا أمر بالغ الأهمية للحفاظ على "الإشارة" في بيانات الأمن غير المتوازنة.

---

```

from sklearn.model_selection import train_test_split

2. Split Data (70% Train, 30% Test)
X_train, X_test, y_train, y_test = train_test_split(
 X, y, test_size=0.3, random_state=42, stratify=y
)

print(f"Training shape: {X_train.shape}")
print(f"Testing shape: {X_test.shape}")

```

---

Listing 35: الخطوة 2: تقسيم البيانات مع الحفاظ على نسب الفئات.

## 3.6.2 الخطوة 3: اختيار النموذج و Cross-Validation

نختار مصنف *Random Forest* لأنه يتعامل مع البيانات العددية المختلطة بشكل جيد ويقاوم *overfitting*. قبل التدريب النهائي، نستخدم Cross-Validation للتحقق مما إذا كان النموذج مستقرًا أو إذا كان محظوظًا فقط بنقاط بيانات سهلة.

---

```

from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import cross_val_score

3. Initialize Model and Check Stability
model = RandomForestClassifier(random_state=42)

cv=2 splits the training data into 2 parts to validate performance
scores = cross_val_score(model, X_train, y_train, cv=2, scoring='accuracy')

print(f"Cross-Validation Scores: {scores}")
print(f"Average Stability: {scores.mean():.2f}")

```

---

Listing 36: الخطوة 3: استخدام Cross-Validation للتحقق من استقرار النموذج.

## 4.6.2 الخطوة 4: ضبط المعلمات الفائقة

نبحث عن أفضل تكوين باستخدام `GridSearchCV`. نجرب أعداداً مختلفة من الأشجار (`n\_estimators`) لنرى أيها يعطي أفضل F1-score.

---

```

from sklearn.model_selection import GridSearchCV

4. Hyperparameter Tuning
We test: 10 vs 50 trees, and varying depth
param_grid = {'n_estimators': [10, 50], 'max_depth': [None, 5]}

grid_search = GridSearchCV(
 RandomForestClassifier(random_state=42),
 param_grid,
 cv=2,
 scoring='f1' # Optimize for F1 because attacks are rare
)
grid_search.fit(X_train, y_train)

best_model = grid_search.best_estimator_
print(f"Best Parameters: {grid_search.best_params_}")

```

---

Listing 37: الخطوة 4: ضبط النموذج للعثور على الإعدادات المثلى.

## 5.6.2 الخطوة 5: تدريب النموذج وتقييمه

باستخدام أفضل الإعدادات التي تم العثور عليها أعلاه، يقوم النموذج بعمل تنبؤات على Testing Set (البيانات التي لم يرها من قبل). نستخدم تقرير تصنيف لنرى بالضبط مدى جودة التقاطه للهجمات.

---

```

from sklearn.metrics import classification_report

5. Final Evaluation
y_pred = best_model.predict(X_test)

print("--- Final Test Report ---")
print(classification_report(y_test, y_pred, target_names=['Benign', 'Attack']))

```

---

Listing 38: الخطوة 5: تقييم النموذج المضبوط على مجموعة الاختبار غير المرئية.

## 6.6.2 الخطوة 6: أهمية الميزة

أخيراً، نسأل النموذج: "أي ميزة كانت الأكثر أهمية؟" في cybersecurity، يشرح هذا \*سبب\* وضع النموذج علامة على حدث ما. هنا، نتوقع أن يكون `syn\_count` مهماً للغاية للكشف عن الهجمات.

```
import matplotlib.pyplot as plt

6. Visualize Explanations
importances = best_model.feature_importances_
indices = np.argsort(importances)[::-1]

plt.figure(figsize=(8, 4))
plt.title("What did the model learn?")
plt.bar(range(X.shape[1]), importances[indices], align="center")
plt.xticks(range(X.shape[1]), X.columns[indices], rotation=45)
plt.ylabel("Importance Score")
plt.tight_layout()
plt.show()
```

Listing 39: الخطوة 6: تصور أهمية الميزة.

## 7.6.2 كود المشروع الكامل

تجمع القائمة 40 جميع الخطوات المذكورة أعلاه في برنامج نصي واحد قابل للتنفيذ.

## خاتمة

لقد وضع هذا الفصل الأساس الجوهري لتطبيق machine learning على cybersecurity. لقد أزلنا الغموض عن المفاهيم الأساسية لـ AI و ML و DL واستكشفنا نماذج التعلم الأساسية التي تشكل أساس جميع الأنظمة الذكية. لقد تعلمت عن عائلات النماذج الرئيسية، من الخوارزميات الكلاسيكية المناسبة للبيانات المهيكلة إلى معماريات deep learning التي تتفوق مع البيانات النحاس المعقدة.

الأهم من ذلك، ركزنا على الحقائق العملية لبناء هذه الأنظمة للأمن. أنت تدرك الآن الدور الحاسم لـ feature engineering، وضرورة اختيار مقاييس التقييم مثل precision و recall بدلاً من الدقة البسيطة، والتحديات الكبيرة التي تفرضها imbalanced data و adversarial attacks. قدم المشروع العملي سير عمل ملموساً وتدريبياً يمكنك تكييفه لمهام تحليل بيانات الأمن الخاصة بك. مع هذا الأساس، أنت الآن مستعد للانتقال إلى تطبيقات أكثر تقدماً للذكاء الاصطناعي في بناء أنظمة دفاع قوية وذكية.

---

```

1 import pandas as pd
2 import numpy as np
3 import matplotlib.pyplot as plt
4 from sklearn.model_selection import train_test_split, cross_val_score, GridSearchCV
5 from sklearn.ensemble import RandomForestClassifier
6 from sklearn.metrics import classification_report
7
8 # --- 1. DATA LOADING ---
9 data = {
10 'packet_size': [64, 1500, 64, 200, 64, 1500, 500, 64, 1500, 64],
11 'syn_count': [0, 0, 1, 50, 0, 0, 20, 0, 0, 100],
12 'duration': [0.1, 5.2, 0.1, 1.5, 0.2, 10.0, 0.5, 0.1, 8.5, 2.0],
13 'label': [0, 0, 0, 1, 0, 0, 1, 0, 0, 1]
14 }
15 df_flows = pd.DataFrame(data)
16 X = df_flows.drop('label', axis=1)
17 y = df_flows['label']
18
19 print("--- Data Sample ---")
20 print(X.head())
21
22 # --- 2. DATA SPLITTING ---
23 X_train, X_test, y_train, y_test = train_test_split(
24 X, y, test_size=0.3, random_state=42, stratify=y
25)
26
27 # --- 3 & 4. TUNING AND TRAINING ---
28 # We combine cross-validation and grid search
29 param_grid = {'n_estimators': [10, 50], 'max_depth': [None, 5]}
30 grid_search = GridSearchCV(
31 RandomForestClassifier(random_state=42),
32 param_grid,
33 cv=2,
34 scoring='f1'
35)
36 grid_search.fit(X_train, y_train)
37 best_model = grid_search.best_estimator_
38
39 print(f"\nBest Parameters: {grid_search.best_params_}")
40
41 # --- 5. EVALUATION ---
42 y_pred = best_model.predict(X_test)
43 print("\n--- Final Test Report ---")
44 print(classification_report(y_test, y_pred, target_names=['Benign', 'Attack']))
45
46 # --- 6. VISUALIZATION ---
47 importances = best_model.feature_importances_
48 indices = np.argsort(importances)[-1:-1]
49
50 plt.figure(figsize=(8, 4))
51 plt.title("Feature Importance: What drives the alerts?")
52 plt.bar(range(X.shape[1]), importances[indices], align="center")
53 plt.xticks(range(X.shape[1]), X.columns[indices], rotation=45)
54 plt.ylabel("Importance")
55 plt.tight_layout()
56 plt.show()

```

---

Listing 40: الكود المصدري الكامل لمشروع تصنيف التهديدات.

## المختصرات والمصطلحات



**AI** الذكاء الاصطناعي Artificial Intelligence: محاكاة الذكاء البشري في الآلات، المبرمجة للتعلم والاستدلال وأتمتة المهام، خاصة لتحليل البيانات واتخاذ القرارات.

**API** واجهة برمجة التطبيقات Application Programming Interface: مجموعة من القواعد والأدوات التي تسمح لتطبيقات برمجية مختلفة بالتواصل مع بعضها البعض. يمكن أن تكون واجهات برمجة التطبيقات APIs غير الآمنة ثغرة أمنية vulnerability رئيسية.

**ATT&CK** التكتيكات والتقنيات والمعرفة العامة للخصوم Adversarial Tactics, Tech-niques, and Common Knowledge: قاعدة معارف knowledge base وإطار عمل framework يمكن الوصول إليه عالمياً من MITRE يصنف إجراءات الخصم بناءً على ملاحظات من العالم الحقيقي.

**Availability** التوافر Availability: مبدأ أساسي في ثلاث السي أي إيه CIA Triad يضمن أن الأنظمة والبيانات متاحة للمستخدمين المصرح لهم عند الحاجة إليها.

**C2** القيادة والتحكم Command and Control: قناة الاتصال التي يستخدمها المهاجم at-tacker للحفاظ على السيطرة على نظام مخترق، وإصدار الأوامر، واستخراج البيانات.

**Checksum** المجموع الاختباري Checksum: كلمة صغيرة من البيانات مشتقة من كلمة أكبر، تستخدم للكشف عن الأخطاء أو التغييرات غير المصرح بها في البيانات الأصلية، وبالتالي ضمان السلامة integrity.

**Triad CIA** ثلاث السي أي إيه CIA Triad: نموذج أمني أساسي يمثل الأهداف الأساسية الثلاثة لأمن المعلومات: السرية Confidentiality، والسلامة Integrity، والتوافر Availability.

**Confidentiality** السرية Confidentiality: مبدأ أساسي في ثلاث السي أي إيه CIA Triad يضمن أن المعلومات الحساسة متاحة فقط للأفراد المصرح لهم.

**Chain Kill Cyber** سلسلة القتل السيبراني Cyber Kill Chain: نموذج يصف التسلسل النموذجي للمراحل التي يجب على الخصم إكمالها لتنفيذ هجوم سيبراني.

**Denial-of-Service (DoS)** هجوم حجب الخدمة Denial-of-Service (DoS): هجوم مصمم لجعل نظام أو خدمة غير متاحة للمستخدمين الشرعيين عن طريق إغراقها بحركة مرور أو طلبات.

**Signature Digital** التوقيع الرقمي Digital Signature: تقنية تشفير cryptographic تستخدم للتحقق من أصالة وسلامة رسالة أو ملف رقمي، مما يؤكد أنه لم يتم تغييره.

**EDR** كشف واستجابة النقاط النهائية Endpoint Detection and Response: فئة من الأدوات الأمنية التي تراقب باستمرار أجهزة المستخدمين النهائيين (مثل أجهزة الكمبيوتر المحمولة والخوادم) للكشف عن التهديدات المتقدمة والاستجابة لها.

**ELF** تنسيق قابل للتنفيذ والربط Executable and Linkable Format: تنسيق الملفات القياسي للبرامج والمكتبات القابلة للتنفيذ على أنظمة لينكس ومعظم أنظمة التشغيل الشبيهة بيونكس.

**Encryption** التشفير Encryption: عملية تحويل البيانات إلى تنسيق غير قابل للقراءة (نص مشفر ciphertext) لحماية سريتها confidentiality. يتطلب الأمر مفتاحاً لفك تشفيرها وإعادةتها إلى بيانات قابلة للقراءة.

**False Positive** نتيجة إيجابية خاطئة False Positive: تنبيه من نظام أمني يحدد بشكل غير صحيح نشاطاً طبيعياً حميداً على أنه ضار.

**IAM** إدارة الهوية والوصول Identity and Access Management: فرع أمني يضمن أن الأفراد المناسبين لديهم المستوى المناسب من الوصول إلى موارد التكنولوجيا.

**IDS** نظام كشف التسلل Intrusion Detection System: أداة أمنية تراقب أنشطة الشبكة أو النظام بحثاً عن أنماط ضارة أو انتهاكات للسياسات وتقوم بالإبلاغ عنها.

**Integrity** السلامة Integrity: مبدأ أساسي في ثالوث السي أي إيه CIA Triad يركز على الحفاظ على دقة وموثوقية البيانات، مما يضمن عدم تغييرها دون إذن.

**IOCs** مؤشرات الاختراق Indicators of Compromise: أدلة جنائية أو أجزاء من البيانات (مثل تجزئات الملفات أو عناوين IP أو أسماء النطاقات) تشير إلى حدوث خرق أمني.

**MFA** المصادقة متعددة العوامل Multi-Factor Authentication: آلية أمنية تتطلب من المستخدم تقديم نوعين أو أكثر من عوامل التحقق المختلفة للوصول إلى حساب أو نظام.

**MITRE ATT&CK** انظر ATT&CK.

**NIST** المعهد الوطني للمعايير والتكنولوجيا National Institute of Standards and Technology: وكالة حكومية أمريكية تضع معايير وأطر عمل مؤثرة للأمن السيبراني.

**PCAP** التقاط الحزم Packet Capture: ملف يحتوي على البيانات الأولية للحزم الملتقطة من شبكة، مما يوفر سجلاً مفصلاً لحركة مرور الشبكة للتحليل.

**PE** قابل للتنفيذ المحمول Portable Executable: تنسيق الملفات القياسي للبرامج والمكتبات القابلة للتنفيذ المستخدمة بواسطة نظام تشغيل ويندوز.

**SAST** اختبار أمان التطبيقات الثابت Static Application Security Testing: عملية تحليل الكود المصدري للتطبيق بحثاً عن ثغرات أمنية دون تشغيل الكود فعلياً.

**Control Security** ضابط أمني Security Control: إجراء وقائي أو مضاد، مثل جدار الحماية firewall أو التشفير encryption، يتم وضعه لتجنب أو اكتشاف أو تقليل المخاطر الأمنية.

**Telemetry Security** بيانات الأمان عن بعد Security Telemetry: مصطلح عام لمجموعة واسعة من بيانات الأمان (مثل السجلات، تدفقات الشبكة) التي يتم جمعها من جميع أنحاء البيئة الرقمية للمراقبة والتحليل.

**SOC** مركز عمليات الأمن Security Operations Center: فريق مركزي، إلى جانب التكنولوجيا والعمليات المرتبطة به، مسؤول عن المراقبة المستمرة والدفاع عن منظمة ضد التهديدات السيبرانية.

**Spoofing** الانتحال Spoofing: هجوم ينجح فيه شخص أو برنامج في انتحال شخصية كيان آخر عن طريق تزيف البيانات، مثل مرسل بريد إلكتروني أو عنوان IP.

**Injection SQL** حقن إس كيو إل SQL Injection: ثغرة أمنية في الويب تسمح للمهاجم بالتدخل في الاستعلامات التي يرسلها التطبيق إلى قاعدة بياناته، غالباً لاستخراج البيانات.

**STRIDE** ستريد STRIDE: إطار عمل لنمذجة التهديدات يصنف التهديدات المحتملة إلى ستة أنواع: الانتحال Spoofing، والعبث Tampering، والتنصل Repudiation، وكشف المعلومات Information Disclosure، وحجب الخدمة Denial of Service، ورفع الامتيازات Elevation of Privilege.

**Sysmon** سيسمون Sysmon: أداة من مايكروسوفت توفر تسجيلاً مفصلاً وعالي الجودة لنشاط النظام على ويندوز، مثل إنشاء العمليات واتصالات الشبكة.

**Tampering** العبث Tampering: التعديل غير المصرح به للبيانات، وهو تهديد مباشر لسلامة البيانات data integrity.

**Threat Intelligence** استخبارات التهديدات Threat Intelligence: معلومات وبيانات حول التهديدات السيبرانية الحالية أو الناشئة، بما في ذلك تفاصيل عن المهاجمين وأسلابهم، تستخدم لإرشاد القرارات الأمنية.

**Positive True** نتيجة إيجابية حقيقية True Positive: تنبيه من نظام أمني يحدد بشكل صحيح نشاطاً ضاراً حقيقياً أو تهديداً أمنياً.

**TTPs** التكتيكات والتقنيات والإجراءات Tactics, Techniques, and Procedures: مصطلح يستخدم في استخبارات التهديدات لوصف سلوك وأساليب الخصم أثناء الهجوم.

**XSS** البرمجة النصية عبر المواقع Cross-Site Scripting: نوع من الثغرات الأمنية في الويب يسمح للمهاجم بحقن نصوص برمجية ضارة في صفحات الويب التي يشاهدها المستخدمون الآخرون.

```latex

القسم 2

الذكاء الاصطناعي التطبيقي للأمن
السيبراني

يتعمق هذا الجزء في مجالات محددة في الأمن السيبراني Cybersecurity، ويوضح كيفية تطبيق تقنيات الذكاء الاصطناعي AI المختلفة لحل المشكلات الملحة في العالم الحقيقي. سيقدم كل فصل مشروعاً مخصصاً ومثيراً لبناء مهاراتك. ...

باب 3

مشروع متكامل: مكتشف رسائل التصيد الاحتيالي (Phishing) (Email Detector)

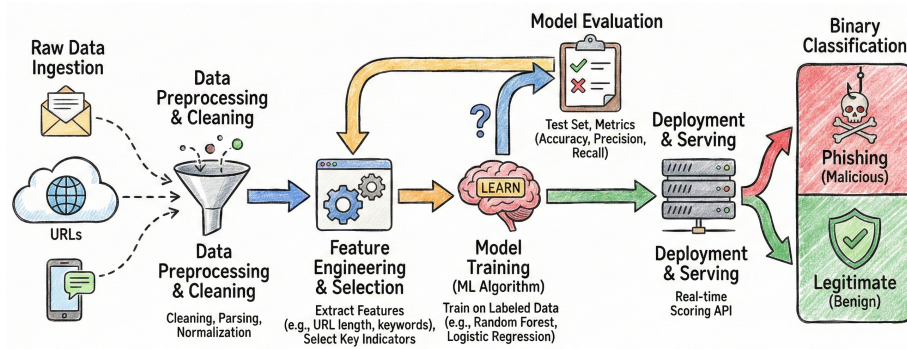
المقدمة (Introduction)

نظراً لأن التهديدات السيبرانية (cyber threats) تمثل مشكلة بارزة، فإن تطوير بنية ذكاء اصطناعي (artificial intelligence) مرنة يعد مطلباً بالغ الأهمية. بناءً على ذلك، نوضح في هذا الفصل إطار عمل شامل للتعليم الآلي (machine learning) لتمييز اتصالات التصيد الاحتيالي (phishing). بالإضافة إلى ذلك، ينسق هذا المسعى مجموعة متنوعة من النماذج النظرية التي تم تأسيسها في الأقسام السابقة من خلال معالجة وتقييم وتطبيق تنفيذ دقيق. علاوة على ذلك، فإن الطبيعة المدمرة لنواقل التصيد الاحتيالي (phishing vectors) تجعل آليات الكشف الآلي إطار عمل مهماً للغاية ضمن أي استراتيجية دفاع مرنة. في النهاية، نؤكد على صياغة التحدي التحليلي، واكتساب البيانات، ومعالجة المعلومات، ووضع بصمات لمجموعات البريد الإلكتروني، مع الاستفادة من نماذج مميزة، واستيعاب عدم توازن الفئات (class imbalance)، وتقييم إطار العمل. لذلك، سيزود هذا العرض الممارسين بالقدرة على الانتقال من التجريدات النظرية إلى تنسيق مخطط دفاعي فعال.

1.3 صياغة المشكلة واكتساب البيانات (Problem Fram- ing and Data Acquisition)

نظراً لأن الأهداف غير الواضحة تخلق مشاكل، فإن تحديد هدف دقيق يشكل المتطلب الأساسي في أي مسعى خوارزمي. لهذا السبب، نقترح نموذجاً مرناً مصمماً خصيصاً لتمييز نواقل التصيد الاحتيالي (phishing vectors) الخبيثة عن الاتصالات السليمة. بالإضافة إلى ذلك، يستوعب إطار العمل هذا الضرورة القصوى لحماية المستخدمين النهائيين ضد المخططات المعقدة المنسقة لاستخراج البيانات الحساسة.

نظراً لأن سرقة البيانات تمثل مشكلة بارزة، فإن اكتشاف التصيد الاحتيالي (Phish- ing Detection) يمثل العملية التحليلية الدقيقة لتحديد الآليات الخادعة المصممة لاستخراج المعلومات السرية بشكل غير قانوني من خلال التظاهر ككيان تم التحقق منه داخل الاتصالات الإلكترونية [78]. بناءً على ذلك، نستفيد من نموذج التصنيف الثنائي (binary classifica- tion) من خلال معالجة وتحليل وتمييز كل اتصال إما كشذوذ تصيد احتيالي أو إرسال مشروع، كما هو موضح في الشكل 1.3.



شكل 1.3: تصور عياني لخط أنابيب التعلم الآلي (machine learning pipeline) الشامل المنشور لاكتشاف التصيد الاحتيالي، مع التركيز على التقدم المنهجي من استيعاب البيانات الأولية إلى نموذج التصنيف الثنائي (binary classification) النهائي.

1.1.3 مصادر البيانات (Data Sources)

نظراً لأن النماذج تحتاج إلى بيانات جيدة، يظل الحصول على مجموعات بيانات تجريبية عالية الدقة مطلباً بالغ الأهمية لتدريب إطار عمل تحليلي فعال. تحقيقاً لهذه الغاية، نستفيد من مجموعة من

مستودعات البيانات لتحليل التصيد الاحتيالي، حيث يميز كل نموذج مزايا واضحة ولكنه يقدم في الوقت نفسه تعقيدات تشغيلية فريدة.

• **مجموعات البيانات المتاحة للجمهور (Publicly Available Datasets):** بالنسبة لمجموعات البيانات المتاحة للجمهور، تنشر المجتمعات الأكاديمية مجموعة من مجموعات البيانات المهيكلة. في هذه الحالة، نستفيد من أرشيفات بارزة مثل مجموعة إنرون (Enron Corpus) [40] لتلبية متطلبات الاتصالات السليمة. على العكس من ذلك، نستفيد من المستودعات المركزية، بما في ذلك مستودع التعلم الآلي في جامعة كاليفورنيا في إيرفين (UCI Machine Learning Repository) ومنصات التجميع مثل كاغل (Kaggle)، للحصول على النواقل الخبيثة.

• **التحديات (Challenges):** بالنسبة للتحديات، فإن استخدام أرشيفات الاتصالات الأصلية يسهل الصلاحية التجريبية العالية، ولكنه يجلب أيضاً فرصة لحدوث انتهاكات أخلاقية وخصوصية خطيرة. بناءً على ذلك، يجب علينا تنسيق حماية صارمة للخصوصية من خلال إخفاء الهوية بدقة، وتأمين، ومعالجة معلومات التعريف الشخصية إلى جانب الحصول على موافقة صريحة من المستخدم. علاوة على ذلك، فإن إهمال هذه البروتوكولات الصارمة يهدد بالتعجيل بتداعيات قانونية وأخلاقية عميقة.

• **البيانات المحاكاة (Simulated Data):** بالنسبة للبيانات المحاكاة، نقوم بتجميع نواقل تصيد احتيالي (phishing vectors) محاكاة في السياقات التي تكون فيها مجموعات البيانات التجريبية غير قابلة للوصول أو حساسة بشكل يمنع استخدامها. يؤسس توليد المجموعات الاصطناعية بيئة تطوير آمنة، ولكنه يجلب أيضاً فرصة لقابلية تعميم مقيدة.

2.3 جمع البيانات والمعالجة المسبقة (Data Collection and Preprocessing)

نظراً لأن البيانات الأولية فوضوية، فإن مجموعات الاتصالات الإلكترونية تكون غير منظمة بطبيعتها، حيث تغلف مزيجاً معقداً من رؤوس التوجيه، والنص العادي، ورميزات (HTML encodings)، والحمولات الثنائية. لذلك، نقوم بتنسيق تعقيم هذا الطيف المعلوماتي الخام من خلال المعالجة الدقيقة، والتعقيم، وهيكلية البيانات قبل دمجها داخل أي إطار عمل تنبؤي للتعلم الآلي (machine learning).

1.2.3 تنظيف النص (Text Cleaning)

نظراً لأن الضوضاء تعطل النماذج، فإن تعقيم البيانات النصية يشكل الآلية الأساسية لتكييف التسلسلات الأولية للتحليل الدقيق عبر التخفيف المنهجي للمخالفات الهيكلية. لذلك، تتطلب المراحل المنهجية القصوى ما يلي:

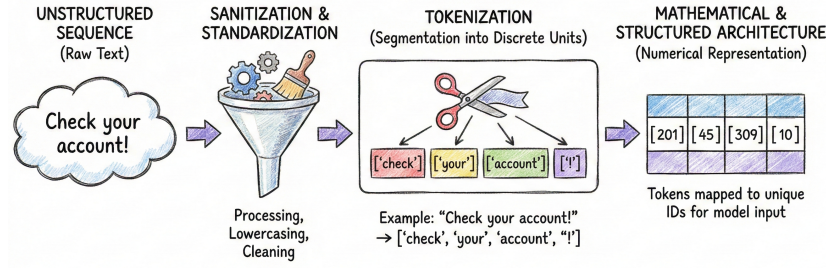
- أولاً، نقوم بالتخفيف من علامات (HTML markup)، وبني (JavaScript) المضمنة، وهندسة (CSS)، والتي تظل بارزة في الاتصالات ولكنها تساهم عادةً بحجم دلالي ضئيل لنموذج التصنيف.
- ثانياً، نحبط علامات الترقيم، والأحرف غير القياسية، والتسلسلات الرقمية، باستثناء الحالات التي يتم فيها الاحتفاظ بهذه الرموز (tokens) عمداً لتشكيل ميزة تحليلية متميزة.
- ثالثاً، نستوعب توحيد المجموعة بأكملها إلى تنسيق أحرف صغيرة متجانس لضمان إسناد الرموز (tokens) المتطابقة دلاليًا بشكل منهجي إلى نفس التمثيل البعدي الدقيق.
- أخيراً، نستفيد من حل شذوذ ترميز الأحرف الدقيق لإحباط إخفاقات التحليل النظامي أثناء المعالجة الحسابية اللاحقة.

2.2.3 الترميز والتوحيد القياسي (Tokenization and Standardization)

نظراً لأن النماذج تتطلب مدخلات رياضية، يجب ترجمة التسلسلات غير المهيكلة إلى بنية رياضية ومهيكلية باتباع تعقيم دقيق. وبالتالي، نبدأ هذا التحول من خلال معالجة، وتجزئة، واستخراج الرموز (tokens). في هذا السياق، يمثل الترميز (tokenization) الآلية الأساسية لتقسيم تدفق نصي مستمر إلى وحدات منفصلة تسمى رموزاً (tokens). علاوة على ذلك، فإن هذه المكونات الأساسية هي في الغالب مصطلحات كاملة أو كلمات فرعية متميزة صرفياً. لتوضيح هذه الآلية، سيتم تقسيم التسلسل التوجيهي "Check your account!" بشكل حتمي إلى المصفوفة: `['check', 'your', 'account', '!']`.

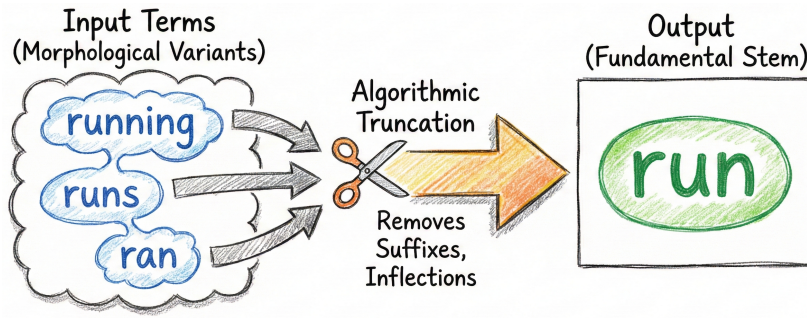
لاحقاً، نقوم بتوحيد هذه الرموز (tokens) المستخرجة بدقة من خلال تصفية، وتعيين، وتوحيد المصطلحات المتقاربة دلاليًا لتقليل الحجم الإجمالي للمفردات بشكل صريح.

- التجذير (Stemming): بالنسبة للتجذير (stemming) (الشكل 3.3)، يقتطع هذا المخطط الخوارزمي المصطلحات إلى بنيتها الجذرية الصرفية. كعرض توضيحي، نقوم بتعيين



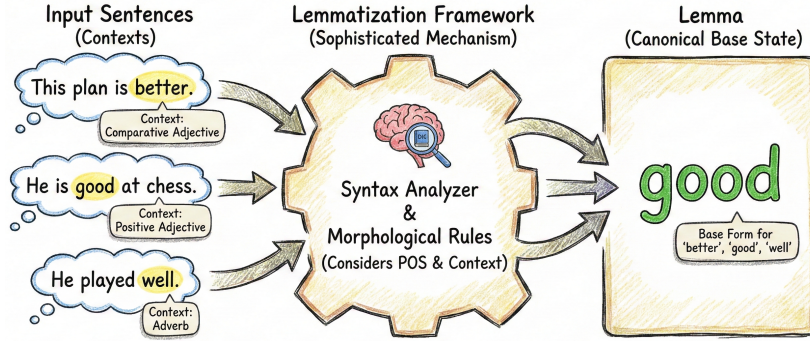
شكل 2.3: تصور لآلية الترميز (tokenization) التي تقسم تدفقاً نصياً مستمراً إلى وحدات منفصلة.

الأفعال المتميزة "running" و "runs" و "ran" بشكل منهجي إلى الجذر الأساسي "run".

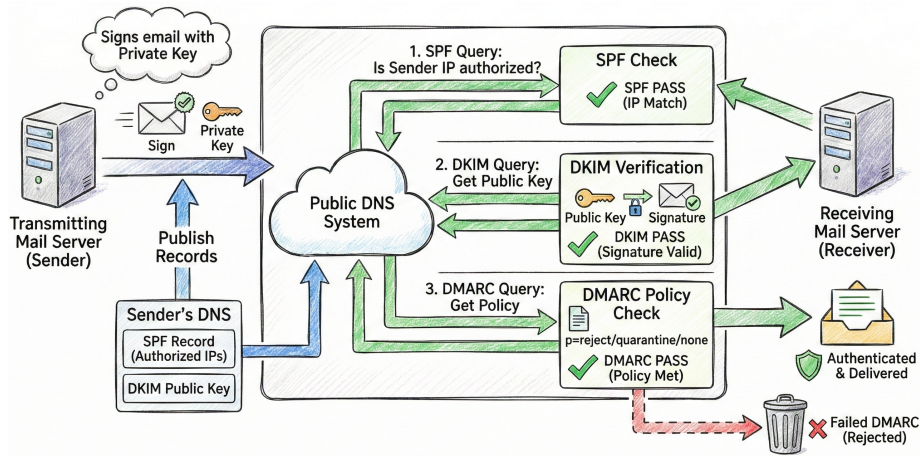


شكل 3.3: مثال على عملية التجذير (stemming) التي تقتطع المتغيرات الصرفية إلى بنية جذر أساسية.

• الرد إلى الكلمة الأصلية (Lemmatization): بالنسبة للرد إلى الكلمة الأصلية (lemma-tization) (الشكل 4.3)، يشكل إطار العمل هذا آلية متطورة للغاية تعيد الكلمات بشكل منهجي إلى حالتها الأساسية المتعارف عليها. على سبيل المثال، نوضح صفة المقارنة "better" إلى الأصل "good". نحن نؤكد على الرد إلى الكلمة الأصلية (lemmatization) كنهج بالغ الأهمية لأنه يستوعب بدقة بناء الجملة السياقي للرمز (token)، مما يؤدي إلى تمثيلات متميزة.



شكل 4.3: توضيح للرد إلى الكلمة الأصلية (lemmatization) الذي يحل المصطلحات إلى حالتها الأساسية المتعارف عليها من خلال استيعاب بناء الجملة السياقي.



شكل 5.3: تسلسل المصادقة التشفيرية المنسق بين بنى البريد المرسل والمستقبل، مع التركيز على النماذج الطبقة للتحقق لبروتوكولات (SPF) و (DKIM) و (DMARC).

3.2.3 استخراج الإشارات من رؤوس البريد الإلكتروني (Extracting Signals from Email Headers)

نظراً لأن البيانات الوصفية (metadata) المخفية تكشف عن الهجمات، فإن رؤوس الإرسال للاتصالات الإلكترونية تغلف بيانات وصفية مهمة، وبالتالي تضيء مؤشرات دقيقة للنية العدائية. علاوة على ذلك، تم تفصيل تسلسل المصادقة التشفيرية هذا في الشكل 5.3. وبالتالي، نستفيد من ثلاثة معايير بارزة للتحقق بدقة من شرعية المرسل:

- إطار سياسة المرسل ((SPF (Sender Policy Framework): أولاً، (SPF) هي آلية مصادقة صريحة تحدد خوادم البريد المحددة المصرح لها رسمياً بنقل الاتصالات نيابة عن نطاق متميز. بناءً على ذلك، ننسب اكتشاف فشل التحقق من صحة (SPF) إلى أصل مخادع بشكل أساسي (spoofed origin) [39].

- بريد معرف بمفاتيح النطاق ((DKIM (DomainKeys Identified Mail): ثانياً، يمكن هذا المخطط التشفيري بنية الاستقبال من التحقق بشكل قاطع من أن الاتصال نشأ من نطاق معتمد، ولكنه يجلب أيضاً فرصة لمتطلبات إدارة مفاتيح معقدة [17].

- مصادقة الرسائل المستندة إلى النطاق والإبلاغ عنها والمطابقة (DMARC (Domain-based Message Authentication, Reporting, and Conformance): ثالثاً، (DMARC) هو إطار عمل يدمج ويوضح النماذج التأسيسية لـ (SPF) و (DKIM). على وجه التحديد، نستفيد منه لإنشاء توجيه سياسة دقيق يوجه بصرامة بنى البريد المستقبلية بشأن آليات التخفيف المناسبة لعمليات الإرسال الواردة التي تفشل في التقييمات التشفيرية الأولية [43].

لذلك، فإن حالة التحقق الثنائية المشتقة من هذه البروتوكولات المعنية تشكل ميزات (features) فعالة للغاية للتكامل داخل إطار عمل اكتشاف التصيد الاحتيالي (phishing detection). بالإضافة إلى ذلك، نوضح المنهجية المحددة لتنفيذ التعقيم الأساسي للنص واستخراج ميزة نحوية (syntactic feature) متميزة بنجاح من المجموعة الأولية من خلال الاستفادة من بنية (pandas) في القائمة 41.

3.3 هندسة الميزات لرسائل البريد الإلكتروني (Feature Engineering for Emails)

نظراً لأن انخوارزميات لا يمكنها قراءة النص الخام، فإن تحويل مجموعات البيانات غير المهيكلة إلى تنسيق رياضي قابل للقياس الكمي بدرجة عالية يعد ضرورة أساسية. لذلك، تمثل هندسة الميزات

```

1 import pandas as pd
2 import re
3
4 # Sample DataFrame of emails
5 df_emails = pd.DataFrame({
6     'text': [
7         '<html><body><p>Hi, click <a href="http://malicious.com">here</a> for a gift!</p></body></html>',
8         'Subject: Your password needs reset\nFrom: support@evil.com\nReceived-SPF: fail (domain of evil.com does not designate 1.2.3.',
9         'Legitimate email regarding your order #123. Thank you for your business.',
10        '<html><p>Important update: <script>alert("xss")</script> Login now!</p></html>'
11    ],
12    'label': [1, 1, 0, 1] # 1: phishing, 0: legitimate
13 })
14 print("Original DataFrame:\n", df_emails)
15
16 def clean_email_text(text):
17     # Remove HTML tags
18     text = re.sub(r'<[>]+>', '', text)
19     # Remove extra whitespace and convert to lowercase
20     text = re.sub(r'\s+', ' ', text).strip().lower()
21     return text
22
23 df_emails['clean_text'] = df_emails['text'].apply(clean_email_text)
24 print("\nDataFrame with 'clean_text':\n", df_emails[['text', 'clean_text']])
25
26 # Simple feature: check for SPF failure in the text
27 df_emails['spf_failed_indicator'] = df_emails['text'].apply(lambda x: 1 if 'spf: fail' in x.lower() else 0)
28 print("\nDataFrame with 'spf_failed_indicator':\n", df_emails[['text', 'spf_failed_indicator']])
29
30 # Another simple feature: check for common phishing keywords
31 phishing_keywords = ['password reset', 'account locked', 'urgent action', 'verify your account']
32 df_emails['has_phishing_keywords'] = df_emails['clean_text'].apply(
33     lambda x: 1 if any(kw in x for kw in phishing_keywords) else 0
34 )
35 print("\nDataFrame with 'has_phishing_keywords':\n", df_emails[['clean_text', 'has_phishing_keywords']])

```

Listing 41: استيعاب وتعقيم تدفقات بيانات الاتصالات الإلكترونية عبر إطار عمل (Pandas).

(feature engineering) العملية الدقيقة لتحويل الإشارات التجريبية الأولية إلى مصفوفات عددية متميزة يمكن لنموذج خوارزمي معالجتها بفعالية. في مجال الاتصالات الإلكترونية، نستخرج بشكل منهجي ميزات متعددة الأبعاد (multidimensional features) من خلال تحليل، وتقييم، وتحديد الحمل اللغوي، والارتباطات التشعبية المضمنة، والبيانات الوصفية الهيكلية.

1.3.3 الميزات النصية (Textual Features)

في هذا السياق، يظل تحويل تسلسلات السلاسل الأولية إلى تنسيق رقمي قابل للقياس الكمي مطلباً بالغ الأهمية. بناءً على ذلك، نوضح إطار عمل قوي مخصص صراحة لمعالجة، واستخراج، ووضع بصمات للإشارات الرياضية من الحمل النصي المعقم.

- **حقبة الكلمات (Bag-of-Words (BoW)):** بالنسبة لحقيبة الكلمات، يشكل هذا النموذج تمثيلاً أساسياً حيث يتم تحديد المعلمات للبيانات النصية حصرياً من خلال التردد

المطلق للمصطلحات المكونة. لذلك، نستفيد من المفردات الإجمالية عبر المجموعة بأكملها لإنشاء مساحة متجه (vector space) شاملة للميزات المستخرجة.

- **تردد المصطلح - تردد المستند العكسي (TF-IDF)** (Term Frequency-Inverse Document Frequency): بالنسبة لـ (TF-IDF)، نستفيد من آلية إحصائية أكثر تطوراً مصممة هندسياً لتحديد الأهمية المتميزة لمصطلح معين بدقة داخل مستند محلي بالنسبة للمجموعة العالمية. ينسب هذا المخطط بنجاح وزناً رياضياً مكبراً للمصطلحات التي تظهر تردداً عالياً داخل اتصال دقيق، ولكنه يجلب أيضاً فرصة لتجاهل ترتيب الكلمات [68]. رسمياً، نحدد وزن (TF-IDF) للمصطلح t في المستند d ضمن المجموعة D كالتالي:

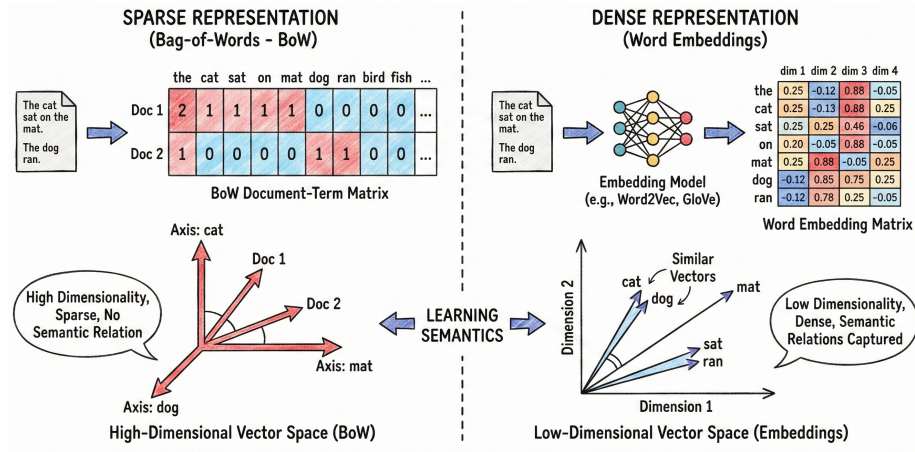
$$\text{TF-IDF}(t, d, D) = \text{tf}(t, d) \times \log \left(\frac{|D|}{1 + |\{d \in D : t \in d\}|} \right) \quad (1)$$

- **تضمين الكلمات (Word Embeddings)**: بالنسبة لتضمين الكلمات (Word Em-beddings)، نستفيد من هذه الآليات المتقدمة لإنشاء تمثيلات متجهة كثيفة (dense vector representations) للرموز المنفصلة بشكل منهجي. في تناقض صارخ مع المصفوفات المتناثرة (sparse matrices) المميزة لمنهجيات (BoW) أو (TF-IDF)، تغلف التضمينات بنجاح الحجم الدلالي والسياق النحوي. علاوة على ذلك، تشمل البنى البارزة في هذا المجال (Word2Vec) [53] و (GloVe) والأطر السياقية المرنة مثل (BERT) [21]. ونتيجة لذلك، يتم توضيح هذا التعيين الهندسي في الشكل 6.3.

2.3.3 ميزات محدد موقع الموارد الموحد (URL Features)

نظراً لأن المهاجمين يستخدمون الروابط، فإن التضمين المتعمد للارتباطات التشعبية العدائية يمثل ناقلاً هجوم أساسياً في الاتصالات الخادعة. تحقيقاً لهذه الغاية، نستوعب تحليل محددات المواقع المضمنة هذه من خلال تحليل، واستخراج، والتحقق من مؤشرات رياضية مميزة للنية الخبيثة.

- **سمعة النطاق (Domain Reputation)**: بالنسبة لسمعة النطاق، نقوم بتنسيق التحقق المنهجي من النطاق المطلق لمحدد الموقع مقابل القوائم السوداء المعمول بها بشكل بارز.
- **خصائص محدد الموقع (URL Characteristics)**: بالنسبة لخصائص محدد الموقع، نؤكد على استخراج ميزات هيكلية دقيقة، بما في ذلك الطول الكمي الإجمالي للسلسلة، والوجود الشاذ للأحرف الخاصة، وتنسيق النطاقات الفرعية الخادعة.



شكل 6.3: تصور هندسي يحدد مساحة الميزة (feature space) المتميزة، ويميز المتجهات المتعامدة والمتناثرة للنماذج التقليدية لحقية الكلمات (Bag-of-Words) عن المجموعات الدلالية الكثيفة التي تنتجها التضمينات المستمرة للكلمات (continuous word embeddings).

- **الكلمات الرئيسية المخادعة (Deceptive Keywords):** بالنسبة للكلمات الرئيسية المخادعة، نؤكد على تحليل سلسلة محدد الموقع لتحديد معجم مميز مصمم صراحة للهندسة الاجتماعية (social engineering)، مثل "تسجيل الدخول" أو "آمن" أو "تحقق".

3.3.3 ميزات البيانات الوصفية للبريد الإلكتروني (Email Metadata Features)

نظراً لأن النص وحده غير كاف، فإن البيانات الوصفية (metadata) الهيكلية المصاحبة بطبيعتها للإرسال توفر مجموعة من الإشارات الفعالة للغاية.

- **موثوقية المرسل (Sender Authenticity):** بالنسبة لموثوقية المرسل، نستفيد من نتائج التحقق الثنائية الصريحة المشتقة من التقييمات التشفيرية لـ (SPF) و (DKIM) و (DMARC).
- **أنواع المرفقات (Attachment Types):** بالنسبة لأنواع المرفقات، نستوعب الاكتشاف الموضوعي لامتدادات الملفات عالية المخاطر، بما في ذلك صراحة التنسيقات مثل .exe أو .docm. بنى المستندات الممكنة بوحدات الماكرو مثل .docm.
- **عدد المستلمين (Number of Recipients):** بالنسبة لعدد المستلمين، نميز الحجم الهائل

الشاذ للأهداف المحددة داخل رؤوس التوجيه، والذي غالباً ما يعمل كمؤشر بارز على تنسيق تصيد احتيالي آلي (automated phishing).

- عدم تطابق عنوان الرد (Reply-To Mismatch): بالنسبة لعدم تطابق عنوان الرد، نخفض من الانحراف الهيكلية المكتشف صراحة بين الأصل وعنوان التوجيه المحدد، مما يشكل تكتيكاً بارزاً يستفيد منه الفاعلون الخبيثون.

لإثبات هذا النموذج تجريبياً، نقدم تنفيذاً حسابياً صارماً في القائمة 42 من خلال الاستفادة من إطار عمل (scikit-learn) وحسابه وتطبيقه لتحويل مجموعة نصية معقدة بشكل منهجي إلى مصفوفة ميزات (TF-IDF) عالية الأبعاد.

```
1 from sklearn.feature_extraction.text import TfidfVectorizer
2 import pandas as pd
3
4 corpus = [
5     "click this malicious link now",
6     "this is a legitimate email regarding your account",
7     "urgent action required login to verify your details",
8     "please find attached invoice for payment"
9 ]
10
11 # Initialize TF-IDF Vectorizer
12 vectorizer = TfidfVectorizer(max_features=10) # Limit features for simpler demo
13
14 # Fit the vectorizer and transform the corpus
15 X_tfidf = vectorizer.fit_transform(corpus)
16
17 print("TF-IDF features (sparse matrix converted to array for viewing):\n", X_tfidf.toarray())
18 print("\nFeatures learned by TF-IDF Vectorizer (vocabulary mapping to column index):")
19 print(vectorizer.get_feature_names_out())
20
21 # You can also put this into a DataFrame for easier inspection
22 df_tfidf = pd.DataFrame(X_tfidf.toarray(), columns=vectorizer.get_feature_names_out())
23 print("\nTF-IDF DataFrame:\n", df_tfidf)
```

Listing 42: تنسيق هندسة الميزات عبر المخطط الإحصائي (TF-IDF) باستخدام إطار عمل (scikit-learn).

4.3 اختيار النموذج والتدريب (Model Selection and Training)

نظراً لأن الميزات الأولية تحتاج إلى التعيين إلى النتائج، يجب علينا تحديد بنية تصنيف مناسبة وتحسينها بشكل قاطع بعد الاستخراج الدقيق لمساحة الميزة وتحديد معالمها. ضمن هذا الطيف التحليلي، تشكل كل من نماذج التعلم الآلي (machine learning) الكلاسيكية والأطر العصبية العميقة (deep neural frameworks) المتقدمة خيارات فعالة للغاية.

1.4.3 مصنفات التعلم الآلي الكلاسيكية (Classical ML Classifiers)

تُظهر هذه الخوارزميات التقليدية فعالية هائلة عندما نطبقها صراحةً على مصنفات الميزات عالية التنظيم التي صممناها بشكل منهجي.

- الانحدار اللوجستي (Logistic Regression): أولاً، الانحدار اللوجستي هو مخطط تصنيف مبسط رياضياً يعمل بشكل مثالي نَخط أساس بارز لتقييم مهام التصنيف الثنائي.

- آلة المتجه الداعم (Support Vector Machine (SVM)): ثانياً، آلة المتجه الداعم هي إطار عمل فعال للغاية مصمم بشكل مثالي لطوبولوجيا البيانات عالية الأبعاد، ومكلف صراحةً باكتشاف المستوى الفائق ذي الهامش الأقصى (maximum-margin hyperplane) الذي يميز بصرامة التصنيفات المتنافسة.

- الغابة العشوائية (Random Forest): ثالثاً، الغابة العشوائية هي منهجية تجميعية بارزة تنسق بناء أشجار قرار عشوائية متعددة ثم تجمع مخرجاتها التنبؤية. يخفف استخدام مجموعات الأشجار من التباين (variance)، ولكنه يجلب أيضاً فرصة لقابلية تفسير مقيدة مقارنة بالنماذج الخطية البسيطة [9].

- تعزيز التدرج (Gradient Boosting): أخيراً، يتضمن تعزيز التدرج نماذج تجميعية متطورة، مثل (LightGBM) [37] أو (XGBoost) [16]، التي تبني أشجار الانحدار بشكل تسلسلي. وبالتالي، نستفيد من هذه البنى لتحقيق مقادير أداء حديثة (state-of-the-art) بشكل متكرر عبر مجموعات البيانات الجدولة المعقدة.

لذلك، يوضح التسلسل الخوارزمي المقدم في القائمة 43 صراحةً المنهجية الصارمة المطلوبة لتدريب مصنف من خلال تهيئة وملاءمة والتنبؤ بمصفوفة مركبة من الميزات الإحصائية المصممة بعناية.

2.4.3 نماذج التعلم العميق البسيطة (Simple Deep Learning Models)

نظراً لأن الهندسة اليدوية بطيئة، فإن الحاجة إلى استخراج الميزات الآلي تشكل مطلباً حاسماً في خطوط الأنابيب الحديثة. تحقيقاً لهذه الغاية، نستفيد من بنى التعلم العميق (deep learning) القادرة على استخراج الميزات المكانية الهرمية بشكل مستقل مباشرة من التسلسلات النصية الخام. علاوة على ذلك، يتم عرض المخطط الهيكلي لمثل هذه البنية في الشكل 7.3.

```

1 from sklearn.ensemble import RandomForestClassifier
2 from sklearn.model_selection import train_test_split
3 import pandas as pd
4 import numpy as np
5
6 # Dummy data: features could be TF-IDF, URL length, SPF check etc.
7 X_dummy = np.random.rand(100, 20) # 100 samples, 20 features
8 y_dummy = np.random.randint(0, 2, 100) # 100 labels (0 or 1)
9
10 # Ensure some imbalance for illustration of handling imbalance later
11 y_dummy[y_dummy == 1] = np.random.choice([0, 1], size=y_dummy[y_dummy == 1].shape[0], p=[0.8, 0.2])
12 y_dummy[0:5] = [0,1,0,0,1] # Ensure a few 1s exist
13
14 # Split data
15 X_train, X_test, y_train, y_test = train_test_split(
16     X_dummy, y_dummy, test_size=0.3, random_state=42, stratify=y_dummy
17 )
18 print(f"Training set shape: {X_train.shape}, Label distribution: {pd.Series(y_train).value_counts(normalize=True)}")
19 print(f"Test set shape: {X_test.shape}, Label distribution: {pd.Series(y_test).value_counts(normalize=True)}")
20
21 # Train a RandomForestClassifier
22 model_rf = RandomForestClassifier(n_estimators=100, random_state=42, class_weight='balanced')
23 model_rf.fit(X_train, y_train)
24 print("\nRandomForestClassifier training complete.")
25
26 # Basic prediction check
27 y_pred_train = model_rf.predict(X_train)
28 train_accuracy = (y_pred_train == y_train).mean()
29 print(f"Training Accuracy: {train_accuracy:.2f}")

```

Listing 43: تدريب نموذج تصنيف مرن (scikit-learn) للكشف الآلي عن التصيد الاحتيالي.

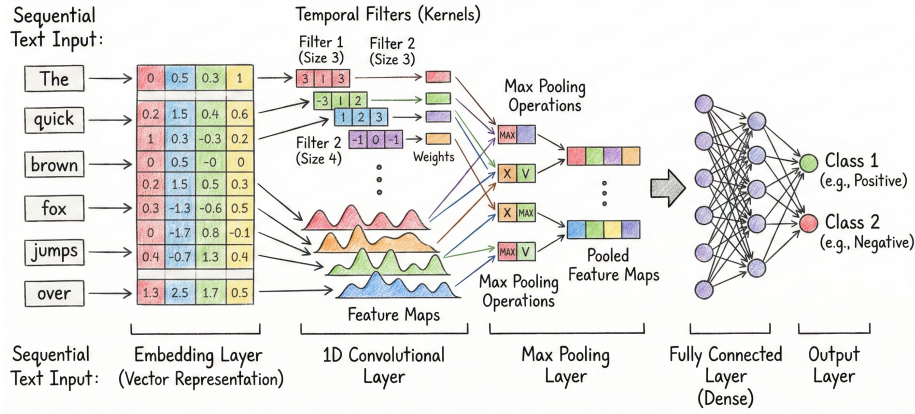
• الشبكات العصبية التلافيفية (Convolutional Neural Networks (CNNs))

[38]: بالنسبة للشبكات العصبية التلافيفية، تمثل شبكات (CNN) أحادية البعد بنى فعالة للغاية لتصنيف النص المتسلسل. على وجه التحديد، تعمل ككواشف (n-gram) مترجمة عن طريق قراءة وتعلم واستخراج أنماط نحوية بارزة عبر الطيف اللغوي.

• شبكات الذاكرة طويلة قصيرة المدى (Long Short-Term Memory (LSTM) Networks)

[33]: بالنسبة لشبكات الذاكرة طويلة قصيرة المدى، فهي متغير مميز للشبكات العصبية المتكررة (recurrent neural networks) المحسنة بشكل استثنائي لمعالجة مصفوفات البيانات المتسلسلة زمنياً. نستفيد من شبكات (LSTM) لتغليف التبعيات الدلالية طويلة المدى بنجاح والحفاظ بدقة على الحجم السياقي لتسلسل معين.

بالإضافة إلى ذلك، تحدد القائمة 44 بدقة الصياغة البرمجية الدقيقة لتحديد بنية أساسية مصممة خصيصاً لتصنيف النص عبر مكتبة موتر (tensor) لـ (PyTorch). علاوة على ذلك، نستفيد من إطار عمل هيكلي شامل مثل (PyTorch Lightning) لضمان إمكانية استخراج منطق التدريب الحسابي المتأصل وتبسيطه بكفاءة.



شكل 7.3: المخطط الهيكلي لشبكة عصبية تلافيفية أحادية البعد (1D Convolutional Neural Network) تعالج بصرامة المصفوفات النصية المتسلسلة، مع إبراز مرشحات الالتفاف الزمنية وطبقات التجميع (pooling layers) البعدية اللاحقة لها بشكل صريح.

5.3 التعامل مع مجموعات البيانات غير المتوازنة (Handling Imbalanced Datasets)

نظراً لأن معظم رسائل البريد الإلكتروني آمنة، تشكل التوزيعات المستهدفة غير المتوازنة تحدياً منتشرًا في بيئات العالم الحقيقي. ضمن أي تدفق اتصال حقيقي، يفوق الحجم الهائل لحركة المرور السليمة بكثير وجود حالات التصيد الاحتيالي (phishing) الخبيثة. ونتيجة لذلك، يولد هذا التناقض الأساسي بشكل صريح نموذج عدم توازن فئات (class imbalance) شديد. بناءً على ذلك، فإن البنية الساذجة المدربة على مثل هذه البيانات غير المعدلة تخلق مقياس دقة عالمياً بارزاً، ولكنها تجلب أيضاً فرصة للفشل التام في اكتشاف تدخلات التصيد الاحتيالي بنجاح. لذلك، نؤكد على مجموعة من الاستراتيجيات المنهجية القوية للتخفيف من هذا النقص الإحصائي المتميز بدقة:

- أخذ العينات الزائدة وأخذ العينات الناقص (Oversampling and Undersampling): بالنسبة لأخذ العينات الزائدة والناقص، ينسق تطبيق أخذ العينات الزائدة (over-sampling) التكرار الصريح للحالات المقابلة لتصنيف الأقلية، في حين يقضي أخذ العينات الناقص (undersampling) بشكل منهجي على العينات التجريبية الناشئة عن مجموعة الأغلبية الساحقة.

- تقنية أخذ العينات الزائدة للأقلية الاصطناعية (SMOTE (Synthetic Minor-

```

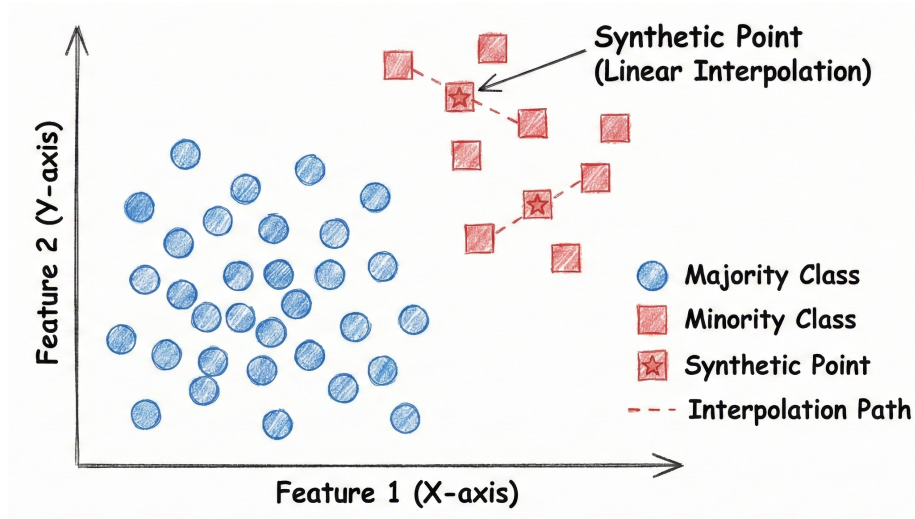
1 import torch
2 import torch.nn as nn
3 import torch.nn.functional as F
4
5 VOCAB_SIZE = 1000
6 EMBED_DIM = 64
7 SEQ_LEN = 50
8 NUM_CLASSES = 2 # Phishing or Legitimate
9
10 class SimpleTextCNN(nn.Module):
11     def __init__(self, vocab_size, embed_dim, num_classes, num_filters=128, kernel_size=3):
12         super(SimpleTextCNN, self).__init__()
13         self.embedding = nn.Embedding(vocab_size, embed_dim)
14         self.conv1d = nn.Conv1d(in_channels=embed_dim, out_channels=num_filters, kernel_size=kernel_size)
15         self.pool = nn.AdaptiveMaxPool1d(1)
16         self.fc = nn.Linear(num_filters, num_classes)
17
18     def forward(self, text_indices):
19         embedded = self.embedding(text_indices)
20         embedded = embedded.permute(0, 2, 1)
21         conv1d = F.relu(self.conv1d(embedded))
22         pooled = self.pool(conv1d).squeeze(-1)
23         return self.fc(pooled)
24
25 model = SimpleTextCNN(VOCAB_SIZE, EMBED_DIM, NUM_CLASSES)
26 print(model)
27
28 # Create dummy input: batch of 4 emails, each 50 tokens long
29 dummy_input = torch.randint(0, VOCAB_SIZE, (4, SEQ_LEN))
30 output = model(dummy_input)
31 print("\nOutput from model for dummy input (logits for 2 classes):\n", output)
32 print(f"Output shape: {output.shape}")
33
34 # Conceptual PyTorch Lightning outline:
35 # import pytorch_lightning as pl
36 # class PhishingDetector(pl.LightningModule):
37 #     def __init__(self, model):
38 #         super().__init__()
39 #         self.model = model
40 #         self.loss_fn = nn.CrossEntropyLoss()
41 #     def training_step(self, batch, batch_idx):
42 #         x, y = batch; logits = self.model(x); loss = self.loss_fn(logits, y)
43 #         self.log('train_loss', loss); return loss
44 #     def configure_optimizers(self):
45 #         return torch.optim.Adam(self.parameters(), lr=1e-3)

```

Listing 44: تحديد مصنف نصي عصبي عميق أساسي عبر (PyTorch).

(ity Over-sampling Technique): بالنسبة لـ (SMOTE)، نستفيد من آلية إحصائية بارزة تولد بشكل منهجي عينات اصطناعية جديدة تقابل فئة الأقلية المتناثرة. على وجه التحديد، نبني نقاط البيانات الاصطناعية هذه عن طريق حساب واستيفاء وتعيين المتجهات مباشرة بين حالات الأقلية المعزولة وجيرانها المحليين الأقرب هندسياً، كما هو ممثل هندسياً في الشكل 8.3 [15].

• أوزان الفئات (Class Weights): بالنسبة لأوزان الفئات، يخصص هذا التعديل الخوارزمي بشكل منهجي عقوبة خسارة مكبرة بشكل أساسي لسوء تصنيف حالات الأقلية



شكل 8.3: تمثيل هندسي لنموذج (SMOTE)، يوضح التوليد الاصطناعي الدقيق لنقاط بيانات فئة الأقلية الذي يتحقق عبر الاستيفاء الخطي (linear interpolation) ضمن مساحة ميزات ثنائية الأبعاد مستمرة.

المحددة أثناء مرحلة التحسين النشطة. علاوة على ذلك، يفرض هذا القيد رياضياً على النموذج الأساسي إيلاء اهتمام بالغ للتحديد الدقيق لهجمات التصيد الاحتيالي المراوغة للغاية. من أجل التنفيذ التجريبي، نستخدم من الملعبة الصريحة `class_weight='balanced'` عبر العديد من المقدرات لتنفيذ إعادة المعايرة الإحصائية هذه تلقائياً. لذلك، يوضح المنطق التسلسلي المقدم في القائمة 45 بصراحة المنهجية البرمجية للاستفادة من نظام (imblearn) البيئي لتطبيق خوارزمية (SMOTE) مباشرة.

6.3 تقييم النموذج والتكرار (Model Evaluation and Iteration)

نظراً لأن الدقة القياسية تخفي أخطاء الأقلية، فإن التحقق من صحة أنظمة الدفاع الآلية يتطلب تقييماً يمتد إلى ما هو أبعد من قيود مقاييس الدقة العامة. تحقيقاً لهذه الغاية، يجب أن نستفيد حصرياً من المؤشرات الإحصائية القوية التي توفر صراحةً تمثيلاً دقيقاً للغاية للفعالية التنبؤية المتركة تحديداً على فئة الأقلية الضعيفة.

```

1 from collections import Counter
2 from sklearn.datasets import make_classification
3 from imblearn.over_sampling import SMOTE
4 from sklearn.model_selection import train_test_split
5
6 # 1. Create a synthetic imbalanced dataset
7 X_imb, y_imb = make_classification(
8     n_samples=1000, n_features=2, n_informative=2,
9     n_redundant=0, n_repeated=0, n_classes=2,
10    n_clusters_per_class=1, weights=[0.95, 0.05], # 95% majority, 5% minority
11    flip_y=0, random_state=42
12 )
13 print(f"Original dataset shape: {X_imb.shape}")
14 print(f"Original class distribution: {Counter(y_imb)}")
15
16 # Split into training and testing sets (SMOTE only on training data)
17 X_train_imb, X_test_imb, y_train_imb, y_test_imb = train_test_split(
18     X_imb, y_imb, test_size=0.3, random_state=42, stratify=y_imb
19 )
20 print(f"\nTraining set class distribution before SMOTE: {Counter(y_train_imb)}")
21
22 # 2. Apply SMOTE to the training data
23 smote = SMOTE(random_state=42)
24 X_resampled, y_resampled = smote.fit_resample(X_train_imb, y_train_imb)
25
26 # 3. Print the class distribution after resampling
27 print(f"\nTraining set class distribution after SMOTE: {Counter(y_resampled)}")
28 print(f"Resampled training set shape: {X_resampled.shape}")

```

Listing 45: تطبيق المخطط الخوارزمي (SMOTE) لموازنة مجموعة بيانات تجريبية غير متوازنة بشكل أساسي بدقة.

1.6.3 المقاييس الخاصة بالأمن (Security-Specific Metrics)

- **الدقة (Precision):** أولاً، الدقة هي مقياس محدد يحدد بصرامة النسبة الصريحة للتأكدات الخوارزمية الإيجابية التي كانت قابلة للتحقق التجريبي وصحيحة. ضمن هذا المجال، يؤدي الحفاظ على حجم دقة مرتفع بشكل استثنائي إلى خلق أمان من الإيجابيات الخاطئة (false positives)، ولكنه يجلب أيضاً فرصة لفقدان التهديدات الحدودية. نحدد هذه الديناميكية الرياضية رسمياً على النحو التالي:

$$\text{Precision} = \frac{\text{Positives True}}{\text{Positives True} + \text{Positives False}} \quad (2)$$

- **الاستدعاء أو الحساسية ((Recall (Sensitivity)):** ثانياً، يقيم الاستدعاء بصرامة النسبة التجريبية المطلقة للتدخلات الخبيثة الحقيقية التي نجح المخطط الآلي في اكتشافها. بالإضافة إلى ذلك، يظل تحقيق استدعاء عالٍ مطلباً بالغ الأهمية لتقليل السلبات الخاطئة (false

(negatives) الخطيرة بشكل كبير. يتم حسابه رياضياً كالتالي:

$$\text{Recall} = \frac{\text{Positives True}}{\text{Positives True} + \text{Negatives False}} \quad (3)$$

• **درجة F1 (F1-score):** ثالثاً، درجة F1 تعني أننا نستفيد بشكل صريح من المتوسط التوافقي (harmonic mean) المشتق من مقاييس كل من الدقة (precision) والاستدعاء (recall) لإنشاء معيار تقييم موحد بصرامة.

$$F_1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (4)$$

• **المساحة تحت منحنى الدقة والاستدعاء (Area Under the Precision-Recall Curve (AUPRC):** أخيراً، تنسق المساحة تحت منحنى الدقة والاستدعاء ملخصاً شاملاً لمرونة النموذج الإجمالية المقاسة عبر الطيف المستمر بأكمله لعتبات التصنيف. علاوة على ذلك، نستفيد من هذه الديناميكية لتقييم مجموعات البيانات غير المتوازنة بشكل كبير، على وجه التحديد لأن تركيزها التحليلي الأساسي يظل مثبتاً بشكل صارم على ديناميكيات الأداء الإحصائي التي تنتمي صراحةً إلى توزيع الأقلية.

2.6.3 تفسير النتائج والتكرار (Interpreting Results and Iteration)

نظراً لأن النماذج تتدهور بمرور الوقت، فإن التقييم الدقيق للنموذج لا يمثل المرحلة النهائية للنشر؛ بل إنه يشكل مرحلة محلية ضمن دورة تحسين مستمرة. بناءً على ذلك، يجب علينا التحقيق بشكل منهجي، وتحليل، وتعيين النواقل الدقيقة المحددة حيث تفشل البنية التنبؤية بشكل قاطع.

• **تحليل الإيجابيات الخاطئة (Analyzing False Positives):** بالنسبة لتحليل الإيجابيات الخاطئة، يمثل هذا التحديد المنهجي للاتصالات السليمة تماماً والتي تم تعيينها خطأً كمحاولات تصيد احتيالي عدائية. يكشف هذا التحليل الدقيق في كثير من الأحيان أن بنية النموذج الأساسية حساسة بشكل غير متناسب لمجموعة محددة من الكلمات الرئيسية البارزة للغاية والمضمنة في كل مكان ضمن توزيعات التسويق الترويجي القياسية.

• **تحليل السلبيات الخاطئة (Analyzing False Negatives):** بالنسبة لتحليل السلبيات الخاطئة، يمثل هذا الفحص الشامل لتدخلات التصيد الاحتياطي التي تم التحقق منها والتي فشل المخطط التشغيلي تماماً في اكتشافها. بالإضافة إلى ذلك، تضيء هذه الحالات الشاذة المحددة بشكل روتيني نواقل هجوم جديدة تماماً تستلزم على الفور هندسة سريعة لميزات جديدة و متميزة للتخفيف منها بنجاح.

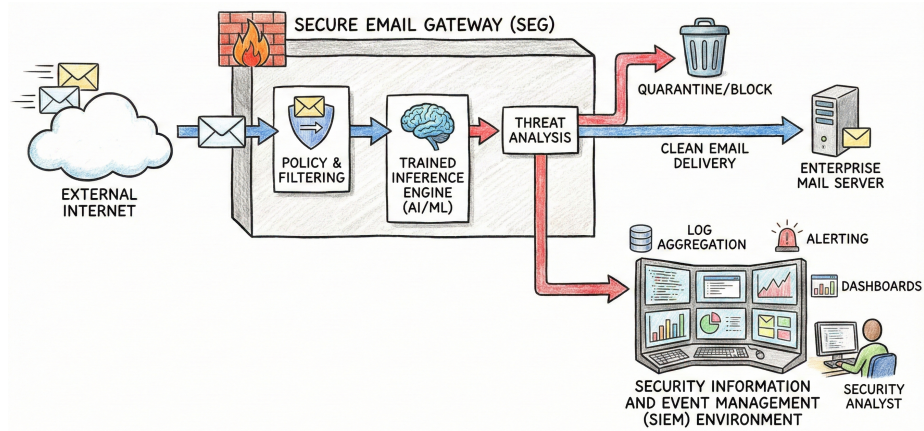
بناءً على هذا التحليل التجريبي المفصل، يمكننا التكرار بشكل منهجي من خلال التحسين المستمر واستخراج وتوليف ميزات جديدة. لذلك، فإن المنطق القابل للتنفيذ المضمن في القائمة 46 يوضح صراحة الإجراء الحسابي الضروري لتجميع تقرير تصنيف شامل ورسم منحنى دقة واستدعاء مرن هندسياً.

```

1 from sklearn.metrics import classification_report, precision_recall_curve, auc
2 import matplotlib.pyplot as plt
3 from sklearn.ensemble import RandomForestClassifier
4 from sklearn.model_selection import train_test_split
5 from sklearn.datasets import make_classification
6 from imblearn.over_sampling import SMOTE
7
8 # Generate synthetic data for the demo
9 X_imb, y_imb = make_classification(n_samples=500, n_features=10, n_informative=5,
10 n_redundant=0, n_classes=2, weights=[0.9, 0.1], random_state=42)
11 X_train_imb, X_test_imb, y_train_imb, y_test_imb = train_test_split(
12     X_imb, y_imb, test_size=0.3, random_state=42, stratify=y_imb
13 )
14
15 # Apply SMOTE to training data for better training
16 smote = SMOTE(random_state=42)
17 X_train_resampled, y_train_resampled = smote.fit_resample(X_train_imb, y_train_imb)
18
19 model_rf = RandomForestClassifier(n_estimators=100, random_state=42)
20 model_rf.fit(X_train_resampled, y_train_resampled)
21
22 y_pred = model_rf.predict(X_test_imb)
23 y_pred_proba = model_rf.predict_proba(X_test_imb)[: , 1] # Probability for the positive class
24
25 print("---- Classification Report for Phishing Detector ----")
26 print(classification_report(y_test_imb, y_pred, target_names=['Legitimate', 'Phishing']))
27
28 # Calculate and plot Precision-Recall Curve for the Phishing class
29 precision, recall, _ = precision_recall_curve(y_test_imb, y_pred_proba)
30 pr_auc = auc(recall, precision)
31
32 plt.figure(figsize=(7, 6))
33 plt.plot(recall, precision, color='purple', lw=2, label=f'PR curve (Area = {pr_auc:.2f})')
34 plt.xlabel('Recall (True Positive Rate)')
35 plt.ylabel('Precision (Positive Predictive Value)')
36 plt.title('Precision-Recall Curve for Phishing Detection')
37 plt.legend(loc="lower left")
38 plt.grid(True)
39 plt.show() # In a real environment, this will display the plot.

```

Listing 46: منطق تقييم النموذج الشامل جنباً إلى جنب مع التصور الهندسي لمنحنى الدقة والاستدعاء (Precision-Recall curve) الدقيق.



شكل 9.3: بنية نشر مؤسسية شاملة، توضح صراحة التكامل الهيكلي لمحرك الاستدلال التنبؤي المدرب حديثاً مباشرة داخل بوابة بريد إلكتروني رقمية آمنة جنباً إلى جنب مع نظام بيئي متكامل للغاية لإدارة المعلومات والأحداث الأمنية (SIEM) في المراحل النهائية.

7.3 اعتبارات النشر الأساسية (Basic Deployment Considerations)

نظراً لأن النماذج النظرية ليس لها تأثير حقيقي، فإن الفائدة التشغيلية لنموذج مدرب بدقة تظل نظرية في الأساس ما لم يتم نشرها بنجاح مباشرة في بيئة إنتاج نشطة، كما هو مصمم في الشكل 9.3.

1.7.3 نقاط التكامل (Integration Points)

- تكامل بوابة البريد الإلكتروني (Email Gateway Integration): بالنسبة لتكامل بوابة البريد الإلكتروني، نشر إطار العمل المحسن ك مكون أساسي ضمن بوابة بريد إلكتروني آمنة شاملة عن طريق فحص وتصفية وحظر الطيف الكامل للاتصالات الواردة.
- خدمة واجهة برمجة تطبيقات مخصصة (Custom API Service): بالنسبة لخدمة واجهة برمجة تطبيقات مخصصة، تغلف بنية الاستدلال (inference architecture) داخل طبقة خدمة (REST API) قوية. بعد ذلك، يمكن لأدوات الأمان المؤسسية المساعدة الاستعلام بنشاط عن هذه الواجهة لتسجيل الاتصالات المستوعبة بشكل منهجي بحثاً عن التهديدات المضمنة المحتملة.
- مكون إضافي لعميل البريد الإلكتروني (Email Client Plugin): بالنسبة لمكون

إضافي لعمل البريد الإلكتروني، يمكن أن يعمل النموذج التنبؤي مباشرة كمكون إضافي مدمج محلياً داخل واجهة عميل بريد إلكتروني محلية.

2.7.3 الأداء وقابلية التوسع (Performance and Scalability)

• الكمون أو وقت الاستجابة (Latency): بالنسبة للكمون، فإن متطلب التسليم السريع للرسائل يمثل متطلباً وظيفياً مطلقاً. بناءً على ذلك، يجب أن ينفذ النموذج المنشور بشكل قاطع الاستدلال الرياضي المعقد مع كمون تشغيلي منخفض بشكل استثنائي لتجنب فرض أي تأخيرات ملحوظة في خط الأنابيب بصرامة. لاستيعاب هذا القيد بنجاح، نستفيد من تقنيات تحسين متقدمة للغاية، بما في ذلك مخططات التكميم الصارمة للنموذج (model quantization schemas).

• قابلية التوسع (Scalability): بالنسبة لقابلية التوسع، يجب أن تحافظ البنية الشاملة على القدرة المرنة المطلقة لمعالجة حجم يومي هائل من الاتصالات الإلكترونية المستوعبة في وقت واحد. عادة ما يتطلب تنفيذ هذا المقدار من الإنتاجية تنسيق طوبولوجيا قابلة للتوسع بدرجة كبيرة عن طريق نشر الأجهزة الافتراضية المعيارية القائمة على السحابة وتوسيع نطاقها وإدارتها.

• الصيانة (Maintenance): نظراً لأن تكتيكات التصيد الاحتيالي (phishing) الخسومية تخضع لتطور مستمر، فإن الحفاظ على الأداء الأساسي يمثل تحدياً. لذلك، نقوم بتنسيق مراقبة النموذج النشط وإخضاعه بشكل منهجي لأنظمة إعادة تدريب منتظمة عبر الاستيعاب المستمر لبيانات التهديد التجريبية الجديدة.

الخاتمة (Conclusion)

نظراً لأن التهديدات السيبرانية (cyber threats) شديدة، فإن تجميع أنظمة دفاع موثوقة يتطلب منهجيات شاملة. بناءً على ذلك، قمنا بتوضيح إطار العمل المنهجي الصارم المطلوب لبناء كاشف شذوذ آلي مرّن للتصيد الاحتيالي (phishing). لقد بدأنا هذا المسعى صراحةً من خلال صياغة وتحديد مصادر وتقييم التحدي التحليلي والبيانات التجريبية. علاوة على ذلك، استفدنا من نموذج (SMOTE) لمعالجة التحدي التخريبي المتمثل في عدم توازن الفئات (class imbalance) الشديد، مع التأكيد مراراً وتكراراً على الضرورة المطلقة لتقييم إطار العمل عبر المقاييس المناسبة. في النهاية، تناولنا اعتبارات البنية التحتية الأساسية المطلوبة بصرامة لنشر مثل هذا المخطط التشغيلي

المعقد بنجاح في بيئة حية. لذلك، يعمل هذا المشروع الهيكلي المحدد بقوة كقالب قابل للتطوير بدرجة كبيرة يمكننا تكييفه بثقة لتنسيق مجموعة واسعة من بنى الدفاع الرقمي المتقدمة بشكل منهجي. الكفاءات التحليلية المحددة التي يتم ممارستها بصرامة داخل هذا المجال تخلق دفاعات قوية، ولكنها تجلب أيضاً فرصة لارتفاع التكاليف الحسابية.

المختصرات والمصطلحات

الذكاء الاصطناعي (Artificial Intelligence) AI المجال الواسع لعلوم الحاسوب -com-puter science الذي يركز على إنشاء آلات أو أنظمة يمكنها أداء المهام التي تتطلب عادةً ذكاءً بشرياً.

المساحة تحت منحنى ROC (Area Under the ROC Curve) AUC-ROC قيمة واحدة تتراوح من 0 إلى 1 تلخص أداء النموذج model عبر جميع عتبات التصنيف -classification thresholds. تشير القيمة الأقرب إلى 1 إلى نموذج أفضل.

المثال الخشبي Adversarial Example مُدخل إلى نموذج ذكاء اصطناعي AI model تم تعديله عمداً من قبل مهاجم attacker ليتسبب في ارتكاب النموذج خطأ.

كشف الشذوذ Anomaly Detection تقنية، عادةً من التعلم غير الخاضع للإشراف -unsupervised learning، تُستخدم لتحديد نقاط البيانات أو الأحداث التي تختلف بشكل كبير عن المعتاد. وهي حاسمة للكشف عن التهديدات الجديدة مثل هجمات اليوم-صفر -zero-day attacks.

الصندوق الأسود Black Box مصطلح يطلق على نموذج ذكاء اصطناعي AI model معقد، مثل شبكة عصبونية عميقة deep neural network، لا يمكن للبشر فهم عملية اتخاذ القرار الداخلية فيه بسهولة.

الشبكة العصبونية الالتفافية (Convolutional Neural Network) CNN نوع من نماذج التعلم العميق deep learning model يتفوق في إيجاد الأنماط المحلية في البيانات شبكية الشبكات، مثل الصور. في الأمن السيبراني cybersecurity، يمكنها تحليل الملفات التنفيذية للبرمجيات الخبيثة malware binaries الممثلة كصور.

التصنيف Classification مهمة من مهام التعلم الخاضع للإشراف supervised learning يكون الهدف منها هو التنبؤ بفئة منفصلة، مثل تصنيف حركة مرور الشبكة network traffic إلى "سليمة" أو "هجوم".

العنقدة Clustering تقنية تعلم غير خاضع للإشراف unsupervised learning تقوم بتجميع نقاط البيانات المتشابهة معاً، والتي يمكن استخدامها لتحديد عائلات جديدة من البرمجيات الخبيثة malware أو الأحداث الأمنية ذات الصلة.

انحراف المفهوم Concept Drift التغيير في الخصائص الإحصائية للبيانات مع مرور الوقت، مما قد يتسبب في تدهور أداء نموذج التعلم الآلي machine learning model إذا لم يتم إعادة تدريبه.

التحقق المتقاطع Cross-Validation طريقة لتقييم كيفية تعميم نتائج تحليل إحصائي على مجموعة بيانات مستقلة. تساعد في الحصول على تقدير أكثر موثوقية لأداء النموذج model performance.

التعلم العميق (DL (Deep Learning) حقل فرعي من التعلم الآلي machine learning يستخدم شبكات عصبونية اصطناعية متعددة الطبقات artificial neural networks (شبكات عصبونية عميقة deep neural networks) لتعلم الأنماط المعقدة من كميات كبيرة من البيانات.

الأساليب التجميعية Ensemble Methods تقنيات تجمع التنبؤات من نماذج تعلم آلي machine learning models متعددة لإنتاج تنبؤ أكثر دقة ومثانة من أي نموذج منفرد. الذكاء الاصطناعي القابل للتفسير Explainable AI (XAI) مجال بحثي في الذكاء الاصطناعي AI يهدف إلى إنشاء أنظمة يمكن للبشر فهم قراراتها والثقة بها بسهولة.

مقياس F1-score F1 مقياس أداء performance metric يوفر درجة واحدة من خلال الموازنة بين كل من الدقة precision والاستدعاء recall، مما يجعله مفيداً جداً لتقييم النماذج على مجموعات البيانات غير المتوازنة imbalanced datasets.

سلي كاذب FN (False Negative) خطأ حرج حيث يفشل النموذج model في كشف هجوم حقيقي، مصنفاً إياه على أنه سليم. يمكن أن يؤدي هذا إلى خرق أمني security breach.

إيجابي كاذب FP (False Positive) خطأ حيث يقوم النموذج model بالإبلاغ عن حدث سليم بشكل غير صحيح على أنه هجوم. يمكن أن يؤدي العدد الكبير من الإيجابيات الكاذبة إلى إرهاق التنبيهات alert fatigue للحللي الأمن security analysts.

هندسة السمات Feature Engineering عملية استخدام معرفة المجال لاختيار وإنشاء وتحويل السمات features الرقمية من البيانات الأولية raw data لتحسين أداء نموذج التعلم الآلي machine learning model.

المعلمات الفائقة Hyperparameters إعدادات تكوين النموذج model التي لا يتم تعلمها من البيانات ولكن يتم تحديدها قبل بدء عملية التدريب، مثل عدد الأشجار في غابة عشوائية Random Forest.

البيانات غير المتوازنة Imbalanced Data سيناريو شائع في مجموعات بيانات الأمن السيبراني cybersecurity حيث تكون حالات فئة واحدة (مثل الأحداث الخبيثة) أقل تكراراً بكثير من فئة أخرى (مثل الأحداث السليمة).

نموذج اللغة الكبير LLM (Large Language Model) نموذج تعلم عميق deep learn- ing model كبير جداً، يعتمد عادةً على بنية المحولات Transformer architecture، مصمم لفهم وتوليد نص يشبه النص البشري.

الذاكرة طويلة قصيرة المدى LSTM (Long Short-Term Memory) نوع متقدم من الشبكات العصبونية المتكررة Recurrent Neural Network (RNN) فعال بشكل خاص في التعلم من تسلسلات طويلة من البيانات، مثل السجلات logs أو تسلسلات استدعاء واجهة برمجة التطبيقات API call sequences.

التعلم الآلي ML (Machine Learning) حقل فرعي من الذكاء الاصطناعي AI يمنح أجهزة الكمبيوتر القدرة على التعلم من البيانات لعمل تنبؤات أو قرارات، دون أن تكون مبرمجة بشكل صريح لكل سيناريو.

معالجة اللغات الطبيعية NLP (Natural Language Processing) مجال من مجالات الذكاء الاصطناعي AI يركز على تمكين أجهزة الكمبيوتر من فهم وتفسير ومعالجة اللغة البشرية، ويستخدم لمهام مثل تحليل رسائل التصيد الاحتيالي phishing emails.

التخصيص المفرط Overfitting مشكلة تحدث عندما يتعلم النموذج model بيانات التدريب training data بشكل مفرط، بما في ذلك الضوضاء noise الموجودة فيها، مما يؤدي إلى أداء ضعيف على البيانات الجديدة غير المرئية.

منحنى الدقة-الاستدعاء PR (Precision-Recall) Curve رسم بياني يوضح المقايضة بين دقة precision النموذج واستدعائه recall عند عتبات مختلفة، وهو مفيد بشكل خاص لتقييم النماذج على مجموعات البيانات غير المتوازنة imbalanced datasets.

الدقة Precision مقياس تقييم evaluation metric يجب على السؤال: "من بين جميع الأحداث التي توقع النموذج model أنها خبيثة، كم منها كان خبيثاً بالفعل؟" الدقة العالية تقلل من الإنذارات الكاذبة false alarms.

التعلم المعزز RL (Reinforcement Learning) نموذج من نماذج التعلم الآلي machine learning حيث يتعلم وكيل agent اتخاذ القرارات المثلى عن طريق أداء إجراءات في بيئة وتلقي مكافآت أو عقوبات.

الشبكة العصبونية المتكررة (RNN (Recurrent Neural Network نوع من الشبكات العصبونية neural network مصمم للتعامل مع البيانات المتسلسلة sequential data، مثل سلسلة من أحداث السجلات log events، لأنه يحتوي على ذاكرة داخلية.

الغابة العشوائية Random Forest نموذج تعلم آلي تجميعي ensemble machine learning model قوي وشائع يبني عدداً كبيراً من أشجار القرار decision trees ويخرج الفئة التي تمثل المنوال لفئات الأشجار الفردية.

الاستدعاء Recall يُعرف أيضاً بالحساسية Sensitivity أو معدل الإيجابيات الحقيقية True Positive Rate. مقياس تقييم evaluation metric يجيب على السؤال: "من بين جميع الأحداث الخبيثة الفعلية، كم عدد الأحداث التي تعرف عليها النموذج model بشكل صحيح؟" الاستدعاء العالي أمر حاسم لعدم تفويت التهديدات.

الانحدار Regression مهمة تعلم خاضع للإشراف supervised learning يكون الهدف منها التنبؤ بقيمة عددية مستمرة، مثل درجة خطورة ثغرة برمجية - software vulnerability.

تعلم التمثيل Representation Learning نهج، غالباً ما يرتبط بالتعلم العميق - deep learning، حيث يكتشف النموذج تلقائياً أكثر السمات features أو التمثيلات فائدة من البيانات الأولية raw data.

منحنى خاصية تشغيل المستقبل ROC (Receiver Operating Characteristic) Curve رسم بياني يوضح أداء نموذج تصنيف classification model عن طريق رسم معدل الإيجابيات الحقيقية true positive rate مقابل معدل الإيجابيات الكاذبة false positive rate عند عتبات مختلفة.

آلة المتجهات الداعمة (SVM (Support Vector Machine نموذج تعلم خاضع للإشراف supervised learning model يجد الحد الأمثل (المستوى الفائق hyperplane) لفصل نقاط البيانات إلى فئات مختلفة.

التعلم شبه الخاضع للإشراف Semi-supervised Learning نهج تدريب هجين يستخدم مزيجاً من كمية صغيرة من البيانات المصنفة labeled data وكمية كبيرة من البيانات غير المصنفة unlabeled data.

التعلم الخاضع للإشراف Supervised Learning نموذج التعلم الآلي machine learning الأكثر شيوعاً، حيث تتعلم الخوارزمية algorithm من مجموعة بيانات تكون فيها كل نقطة بيانات مصنفة labeled بالإخراج الصحيح.

سلي حقيقي (True Negative) TN تنبؤ صحيح حيث يحدد النموذج model حدثاً سليماً على أنه سليم.

إيجابي حقيقي (True Positive) TP تنبؤ صحيح حيث يحدد النموذج model حدثاً خبيثاً على أنه خبيث.

مجموعة الاختبار Testing Set الجزء من مجموعة البيانات الذي يتم حجزه من عملية التدريب ويستخدم فقط في النهاية لتقييم أداء النموذج model النهائي على البيانات غير المرئية.

مجموعة التدريب Training Set الجزء من مجموعة البيانات المستخدم لتدريب نموذج التعلم الآلي machine learning model. يتعلم النموذج model الأنماط والعلاقات من هذه البيانات.

التعلم غير الخاضع للإشراف Unsupervised Learning نموذج تعلم آلي machine learning حيث تعمل الخوارزمية algorithm مع بيانات غير مصنفة unlabeled data للعثور على أنماط أو هياكل مخفية بمفردها.

باب 4

اكتشاف اختراق الشبكات (Network Intrusion Detection): هندسة الميزات (Feature Engineering) والتعلم الآلي الكلاسيكي (Classical ML)

مقدمة

يُعد نظام اكتشاف اختراق الشبكات (Network Intrusion Detection System - NIDS) إطار عمل بارز لدفاعات الأمن السيبراني (cybersecurity) الحديثة. تؤكد أن وظيفته الأساسية هي مراقبة حركة مرور الشبكة (network traffic) وتحديد الأنشطة الضارة (malicious activities) قبل أن تتسبب في حجم كبير من الضرر. مع تزايد تعقيد الشبكات وحجم حركة المرور، يصبح التحليل اليدوي لهذه البيانات أمراً مستحيلاً. لهذا السبب، يوفر الذكاء الاصطناعي (artificial intelligence)، وتحديدًا التعلم الآلي (machine learning - ML)، مخططاً فعالاً وقابلاً للتطوير. من خلال معالجة وتحليل ووضع بصمات (finger-printing) لكميات هائلة من بيانات الشبكة، نستفيد من نماذج التعلم الآلي (ML models) لاكتشاف أنماط الهجوم (attack patterns) المعروفة تلقائياً وتحديد التهديدات الجديدة

(novel threats) التي قد تفوتها الأنظمة التقليدية القائمة على التوقيع (signature based systems). يخلق الاكتشاف الآلي دفاعاً قابلاً للتطوير، ولكنه يجلب أيضاً فرصة لتفويت الهجمات شديدة التشويش (highly obfuscated attacks). في هذا الفصل، نوضح إطار عمل بارز لبناء نظام اكتشاف اختراق الشبكات (NIDS) مدعوم بالتعلم الآلي (ML) باستخدام الخوارزميات الكلاسيكية (classical algorithms). نتيجة لذلك، نبدأ بتعريف مشكلة اختراق الشبكة (network intrusion) وفحص نماذج الهجوم (attack paradigms) الشائعة. بالإضافة إلى ذلك، نناقش المصادر الأساسية لبيانات الشبكة، مثل التقاط الحزم الخام (raw packet captures) وسجلات التدفق الملخصة (summarized flow records)، إلى جانب أدوات بايثون (Python tools) التي نستفيد منها لمعالجتها. يتطلب التركيز الأسمى دراسة متأنية لهندسة الميزات (feature engineering)، وهي الآلية الحاسمة لتحويل بيانات الشبكة الخام إلى ميزات عددية (numerical features) ذات معنى يمكن لنماذج التعلم الآلي (ML models) تقييمها. أخيراً، نقترح مشروعاً عملياً لبناء وتقييم مُصنّف الغابة العشوائية (Random Forest classifier) لاكتشاف التطفل (in-trusion detection)، مما يوفر رؤية دقيقة لكيفية تنسيق هذه الأنظمة لتحديد التهديدات (threat identification).

1.4 المشكلة: اكتشاف اختراقات الشبكة (The Problem: Detecting Network Intrusions)

الهدف الأساسي لنظام NIDS هو تحديد الأنشطة غير المصرح بها أو الضارة أو الشاذة (anomalies) التي تحدث على شبكة الكمبيوتر. لذلك، يعمل نظام NIDS كمحرك تحليلي مرن، يراقب باستمرار تدفق البيانات وينسق إنذاراً عندما يكتشف شذوذاً مشبوهاً (suspicious activities). نؤكد أن هذه الأنظمة مصممة لاكتشاف طيف متميز من التهديدات، بدءاً من تحقيقات الاستطلاع (reconnaissance probes) البسيطة وحتى الهجمات المنسقة المعقدة (complex coordinated attacks).

1.1.4 أنواع الاختراق الشائعة (Common Intrusion Types)

لبناء نظام NIDS فعال، من المهم جداً التمييز بين أنواع الأنشطة التي نحتاج إلى إحباطها. لهذا السبب، بينما يتطور مشهد التهديدات السيبرانية (cyber threats) باستمرار، فإننا نعزو العديد من الهجمات إلى عدة فئات متميزة.

- عمليات فحص المنافذ (Port Scans): هذه آلية استطلاع (reconnaissance mech- anism) بارزة حيث يرسل المهاجم طلبات بشكل منهجي إلى طيف من المنافذ (ports) على المضيف (host). نتيجة لذلك، الهدف هو اكتشاف المنافذ المفتوحة وما هي الخدمات (services) التي تعمل عليها، وتحديد نقاط الضعف (vulnerabilities) المحتملة لهجوم مستقبلي. يخلق اكتشاف المنافذ المفتوحة ميزة معلوماتية، ولكنه يجلب أيضاً فرصة للاكتشاف الفوري بواسطة آليات الأمان (security mechanisms).
- هجمات حجب الخدمة (Denial of Service (DoS) Attacks): الهدف من هجوم DoS هو جعل الجهاز أو مورد الشبكة غير متاح لمستخدميه المقصودين. في هذه الحالة، يحقق المهاجمون ذلك عن طريق إغراق النظام المستهدف بسيل من حركة المرور أو الطلبات المشوهة (malformed requests)، واستهلاك موارده حتى لا يعد قادراً على استيعاب المستخدمين الشرعيين.
- محاولات الاستغلال (Exploit Attempts): يتضمن ذلك استفادة المهاجم من ثغرة أمنية (vulnerability) معروفة في إطار عمل برمجي أو نظام تشغيل. لذلك، تشمل الأمثلة الشائعة تجاوز سعة الخزن المؤقت (buffer overflows) أو هجمات حقن قواعد البيانات (SQL injection attacks)، والتي يمكن أن تسمح للمهاجم بالحصول على وصول غير مصرح به (unauthorized access)، أو تنفيذ تعليمات برمجية ضارة (malicious code)، أو تصعيد الامتيازات (escalate privileges) على النظام المستهدف.
- الاستطلاع (Reconnaissance): هذه هي مرحلة جمع المعلومات الأولية للهجوم. قبل شن هجوم، يجمع المهاجم بيانات حول الشبكة المستهدفة والأنظمة ونقاط الضعف المحتملة. بالإضافة إلى ذلك، يمكن أن يتضمن ذلك تقنيات سلبية (passive techniques) أو تقنيات نشطة (active techniques)، مثل رسم الخرائط (mapping)، والمسح (scanning)، والسبر (probing)، والتي نستفيد من نظام NIDS لاكتشافها.

2.4 مصادر البيانات والأدوات: ملفات التقاط الحزم وبيانات التدفق (PCAPs & Flow Data)

أساس أي نظام NIDS مدعوم بالتعلم الآلي (ML) هو البيانات الدقيقة التي نقوم بتحليلها. لذلك، تأتي هذه البيانات من مراقبة حركة مرور الشبكة (network traffic)، ويمكننا التقاطها وتخزينها

في عدة تنسيقات، ولكل منها مقايضاتها الخاصة بين التفاصيل والحجم.

1.2.4 التقاط حركة مرور الشبكة الخام (Raw Network Traffic Cap- (tures - PCAP files

ملف التقاط الحزم (Packet Capture file) هو ملف ثنائي يحتوي على الحجم الكامل لحزم الشبكة (network packets) التي نعرضها من الشبكة. نتيجة لذلك، يتضمن هذا جميع الرؤوس (headers) وحمولة البيانات (data payload) الفعلية. بالإضافة إلى ذلك، توفر ملفات التقاط الحزم العرض الأكثر دقة واكتمالاً لنشاط الشبكة، مما يجعلها فعالة لفحص الحزم العميق (deep packet inspection) والتحليل الجنائي المفصل (forensic analysis).
تخلق بيانات الحزم الخام إطاراً تحليلياً كاملاً، ولكنها تجلب أيضاً فرصة لمعالجة التخزين المكلفة حسابياً.

2.2.4 بيانات التدفق المهيكلية (Structured Flow Data - NetFlow,) (IPFIX, Zeek logs

للتخفيف من تحديات معالجة الالتقاطات الخام، نقوم غالباً بتلخيص حركة مرور الشبكة في نموذج تدفق مهيكل (structured flow paradigm). لذلك، تدفق الشبكة (network flow) هو سلسلة من الحزم التي تشترك في خصائص مشتركة. عادةً ما نحدد التدفق بواسطة مجموعة من 5 عناصر (5-tuple)، والتي تتكون من خمس قطع متميزة من المعلومات. في هذه الحالة، توفر بيانات التدفق إحصائيات مجمعة حول المحادثة، مثل إجمالي البايتات (bytes) والحزم المتبادلة، ومدة الاتصال (connection duration)، وعلامات بروتوكول التحكم في الإرسال (TCP) (flags) المرصودة، دون تخزين حمولات الحزم.

• نت فلو / آي بي فيكس (NetFlow/IPFIX): NetFlow هو بروتوكول شبكة بارز لجمع معلومات حركة مرور بروتوكول الإنترنت (IP traffic)، والذي نؤكد أنه تم توحيدده من قبل قوة مهمة هندسة الإنترنت (IETF) باسم تصدير معلومات تدفق بروتوكول الإنترنت (Internet Protocol Flow Information Export - IPFIX).
نتيجة لذلك، يمكن لأجهزة التوجيه (routers) والمحولات (switches) التي تستوعب هذه البروتوكولات إنشاء سجلات تدفق (flow records) تلخص جميع المحادثات التي تمر عبرها، مما يوفر نظرة عامة مرنة على نشاط الشبكة.

• زيك (Zeek): Zeek هو إطار عمل مفتوح المصدر وفعال لتحليل الشبكة. يعمل عن طريق معالجة وتفسير وتلخيص بيانات الحزم الخام لإنشاء ملفات سجل مهيكلية (struc-

(tured log files). بدلاً من سجلات التدفق العامة، يُنتج Zeek سجلات متميزة لبروتوكولات محددة، مما يسهل استخراج الميزات ذات المعنى لنماذج التعلم الآلي (ma-chine learning models) [63].

3.2.4 مكتبات بايثون لبيانات الشبكة (Python Libraries for Net-work Data)

العديد من مكتبات بايثون (Python libraries) مهمة جداً للعمل مع بيانات الشبكة برمجياً.

- في شارك (tshark): على الرغم من أنها ليست مكتبة بايثون أصلية، إلا أن tshark هو واجهة سطر الأوامر لبرنامج Wireshark، وهو أبرز محلل لبروتوكولات الشبكة. يمكننا الاستفادة منه من بايثون باستخدام وحدة subprocess لتحليل الملفات واستخراج حقول معينة في تنسيقات نصية، والتي يمكننا بعد ذلك تحميلها في بنية بيانات (data structure).

- سكايب (Scapy): Scapy هي مكتبة بايثون مرنة وقابلة للتطوير لمعالجة الحزم (packet manipulation) [8]. نستخدم Scapy لتزوير (forging) واستنشاق (sniffing) وتشريح (dissecting) حزم الشبكة لطيف واسع من البروتوكولات. لهذا السبب، فهي أداة فعالة لتحليل الملفات برمجياً أو صياغة حزم شبكة مخصصة لآليات الاختبار.

- دي بي كيه تي (dpkt): وحدة بايثون قابلة للتطوير مصممة لإنشاء وتحليل الحزم بشكل سريع وبسيط. لذلك، فهي عموماً أسرع من Scapy للتحليل للقراءة فقط للملفات، مما يجعلها خياراً فعالاً عندما يكون الأداء مصدر قلق بالغ لمعالجة حجم كبير من البيانات.

4.2.4 عرض عملي: تحليل ملف التقاط حزم باستخدام سكايب (Practical Demo: Parsing a PCAP File with Scapy)

لتوضيح كيفية الوصول إلى بيانات الشبكة في بايثون (Python)، يوضح الكود في القائمة 47 سير عمل بسيط باستخدام مكتبة Scapy. أولاً، نقوم برمجياً بإنشاء ملف صغير ومؤقت يحتوي على ثلاثة أنواع متميزة من الحزم. ثم، نستخدم دوال المكتبة لتحميل هذا الملف والتكرار عبر كل حزمة، واستخراج وطباعة المعلومات الأساسية مثل العناوين والمنافذ والبروتوكولات. بالإضافة إلى ذلك، يوضح هذا المثال مدى سهولة البدء في تشريح حركة مرور الشبكة لمزيد من التحليل. يخلق تنفيذ بايثون إطار معالجة مرن، ولكنه يجلب أيضاً فرصة لزيادة كبيرة في الذكاء لمجموعات البيانات الكبيرة.

```

1 from scapy.all import IP, TCP, UDP, Raw, Ether, wrpcap, rdpcap, ICMP
2 import os
3
4 # Create a dummy PCAP file for demonstration
5 # Packet 1: HTTP SYN request
6 http_request = (
7     Ether()/IP(src="192.168.1.10", dst="192.168.1.1")
8     /TCP(sport=12345, dport=80, flags="S")
9 )
10 # Packet 2: DNS query
11 dns_query = (
12     Ether()/IP(src="192.168.1.11", dst="8.8.8.8")
13     /UDP(sport=54321, dport=53)
14 )
15 # Packet 3: ICMP Echo Request (Ping)
16 icmp_request = (
17     Ether()/IP(src="192.168.1.12", dst="192.168.1.1")
18     /ICMP()
19 )
20
21 dummy_pcap_path = 'sample_network_traffic.pcap'
22 wrpcap(dummy_pcap_path, [http_request, dns_query, icmp_request])
23 print(f"Dummy PCAP file created at: {dummy_pcap_path}")
24
25 # Load the PCAP file
26 try:
27     packets = rdpcap(dummy_pcap_path)
28     print(f"Loaded {len(packets)} packets from {dummy_pcap_path}")
29
30     # Iterate through packets and extract info
31     for i, pkt in enumerate(packets):
32         print(f"\n--- Packet {i+1} ---")
33         if pkt.haslayer(IP):
34             ip_layer = pkt[IP]
35             print(f" Source IP: {ip_layer.src}")
36             print(f" Dest IP: {ip_layer.dst}")
37             # Common protocol numbers: TCP=6, UDP=17, ICMP=1
38             print(f" Protocol: {ip_layer.proto}")
39
40             if pkt.haslayer(TCP):
41                 tcp_layer = pkt[TCP]
42                 print(f" Source Port: {tcp_layer.sport}")
43                 print(f" Dest Port: {tcp_layer.dport}")
44                 print(f" TCP Flags: {tcp_layer.flags}")
45             elif pkt.haslayer(UDP):
46                 udp_layer = pkt[UDP]
47                 print(f" Source Port: {udp_layer.sport}")
48                 print(f" Dest Port: {udp_layer.dport}")
49             elif pkt.haslayer(ICMP):
50                 print(" Layer: ICMP Packet Detected")
51         else:
52             print(" Non-IP packet (e.g., ARP)")
53
54 except FileNotFoundError:
55     print(f"Error: PCAP file not found at {dummy_pcap_path}")
56 except Exception as e:
57     print(f"An error occurred: {e}")
58 finally:
59     # Clean up dummy file
60     if os.path.exists(dummy_pcap_path):
61         os.remove(dummy_pcap_path)
62     print(f"Cleaned up dummy PCAP file: {dummy_pcap_path}")

```

Listing 47: تحليل نموذج ملف التقاط حزم باستخدام مكتبة سكابي (a) Parsing
 .(sample PCAP file using the Scapy library)

3.4 هندسة الميزات لبيانات الشبكة (Feature Engineering) (for Network Data)

بيانات الشبكة الخام ليست مناسبة للإدخال المباشر في معظم نماذج التعلم الآلي (machine learning models). لذلك، فإن الهدف من هندسة الميزات (feature engineering) هو تحويل هذه البيانات الخام إلى مخطط رقمي مهيكل يلتقط السلوكيات ذات الصلة باكتشاف الاختراقات (detecting intrusions). تؤكد أن هذه ربما تكون الخطوة الأكثر أهمية في

بناء نظام NIDS قائم على التعلم الآلي، حيث إن جودة الميزات تحدد بشكل مباشر أداء النموذج. نتيجة لذلك، تتطلب هذه العملية دراسة متأنية ومعرفة كبيرة بالمجال لبروتوكولات الشبكة ونماذج الهجوم (attack paradigms).

1.3.4 ميزات على مستوى الحزمة (Packet Level Features)

نستخرج هذه الميزات من الحزم الفردية ونصف خصائصها المميزة.

- حجم الحزمة (Packet Size): الطول الإجمالي للحزمة بالبايت. في هذه الحالة، يمكن أن تكون الحزم الكبيرة أو الصغيرة بشكل شاذ مؤشراً على هجمات معينة، مثل هجمات التجزئة (fragmentation attacks) أو الأنفاق (tunneling).
- وقت وصول الحزم البيني (Inter Packet Arrival Time - IPAT): الفارق الزمني بين وصول الحزم المتتالية في التدفق. نتيجة لذلك، يمكن أن تشير الحزم السريعة والموقوتة باستمرار إلى نشاط آلي مثل المسح (scanning) أو هجوم حجب الخدمة (DoS attack).
- علامات بروتوكول التحكم في الإرسال (TCP Flags): تشير علامات التحكم في الرأس إلى حالة الاتصال. لذلك، تعد مجموعات أو أنماط العلامات غير المعتادة مؤشراً قوياً على هجوم حجب الخدمة الفيضاني أو فحص المنافذ.
- أنواع البروتوكولات (Protocol Types): بروتوكولات الشبكة والنقل المستخدمة. بالإضافة إلى ذلك، يمكن أن يشير استخدام بروتوكولات غير متوقعة على منافذ قياسية إلى نشاط ضار.
- طول الحمولة (Payload Length): حجم جزء البيانات من الحزمة. لهذا السبب، قد تكون الحمولات الفارغة أو ذات الحجم غير المعتاد مشبوهة.
- وقت البقاء (Time to Live - TTL): قيمة في الرأس تحد من عمر الحزمة. غالباً ما تستخدم أنظمة التشغيل المختلفة قيمةً افتراضية متميزة، لذا قد تلحح القيمة غير المعتادة إلى الانتحال (spoofing).

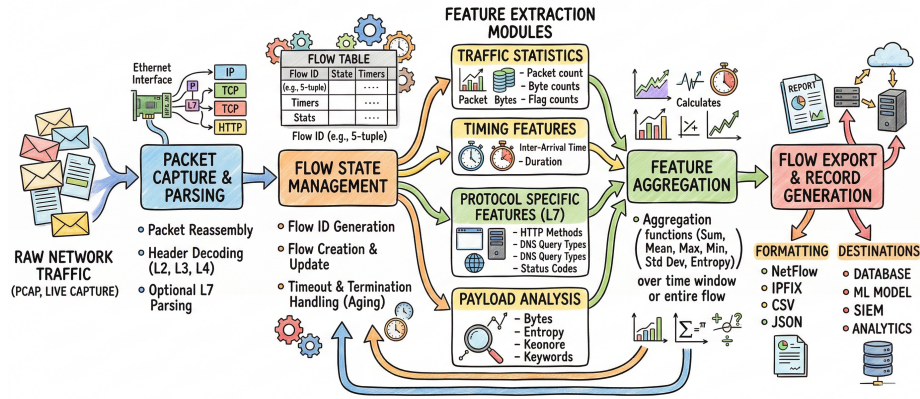
2.3.4 ميزات على مستوى التدفق (Flow Level Features)

نوضح أن هذه الميزات هي إحصائيات مجمعة (aggregated statistics) محسوبة عبر مجموعة من الحزم التي تشكل تدفقاً واحداً أو تتم ملاحظتها ضمن نافذة زمنية محددة. لذلك، فإنها توفر رؤية أعلى مستوى لسلوك الشبكة.

- البايتات والحزم لكل تدفق (Bytes and Packets per Flow): الحجم الإجمالي للبايتات والحزم المرسل من المصدر إلى الوجهة والعكس ضمن التدفق. نتيجة لذلك، قد تشير عمليات نقل البيانات الكبيرة إلى استخراج البيانات (data exfiltration).
- مدة الاتصال (Connection Duration): إجمالي الوقت الذي يكون فيه التدفق نشطاً. يمكن أن تكون الاتصالات القصيرة جداً علامة على المسح، في حين أن الاتصالات الطويلة بشكل غير معتاد قد تستخدم لقنوات القيادة والتحكم (command and control channels).
- منافذ المصدر/الوجهة (Source/Destination Ports): أرقام المنافذ المستخدمة للاتصال. في هذه الحالة، غالباً ما يكون استخدام المنافذ غير القياسية لخدمات معروفة أمراً مشبوهاً.
- عدد المنافذ الفريدة المستخدمة (Number of Unique Ports Used): حساب المنافذ الوجهة المتميزة التي يتصل بها عنوان بروتوكول إنترنت مصدر واحد خلال نافذة زمنية. لذلك، فإن العدد الكبير هو علامة كلاسيكية على فحص المنافذ (port scan).
- الملخصات الإحصائية (Statistical Summaries): حساب المتوسط، والوسيط، والانحراف المعياري، والحد الأدنى أو الأقصى لميزات مستوى الحزمة ضمن تدفق واحد. بالإضافة إلى ذلك، يمكن لهذه الإحصائيات أن تكشف عن استقرار وطبيعة الاتصال.
- الميزات الزمنية (Temporal Features): المقاييس المتعلقة بالوقت، مثل معدل الحزم أو البايتات في الثانية للتدفق. لهذا السبب، يمكن أن تشير الزيادات المفاجئة في حركة المرور إلى حدوث هجوم.
- عدد عناوين بروتوكول الإنترنت الفريدة (Count of Unique IP Addresses): لمضيف واحد، يتم حساب عدد عناوين الوجهات المتميزة التي يتواصل معها في نافذة زمنية. قد يشير الارتفاع المفاجئ إلى أن المضيف قد تم اختراقه وأنه يقوم بمسح الشبكة.

3.3.4 التحليل الخوارزمي: توليد ميزات على مستوى التدفق (Algorithmic) (Breakdown: Generating Flow Level Features)

- يعد إنشاء متجه الميزات (feature vector) من البيانات الخام عملية مرنة ومنهجية. لذلك، تحدد الخطوات التالية نهجاً عاماً لتوليد ميزات على مستوى التدفق (flow level features).
- (1) تحديد التدفق (Define a Flow): تحديد الحزم التي تنتمي إلى نفس المحادثة. عادة ما نقوم بذلك باستخدام آلية العناصر الخمسة (5-tuple mechanism).



شكل 1.4: خط أنابيب معماري لاستخراج وتجميع الميزات على مستوى التدفق من حركة مرور الشبكة الخام (Architectural pipeline for extracting and aggregating flow-level features from raw network traffic).

- (2) تحديد نافذة زمنية (Define a Time Window): تجميع الحزم في فترات زمنية ثابتة أو تحديد حدود التدفق بواسطة إشارات موجهة للاتصال.
- (3) معالجة البيانات الخام (Process Raw Data): قراءة البيانات من مصدر، مثل تنصت شبكة مباشر، أو ملف، أو سجلات.
- (4) تجميع الحزم (Group Packets): لكل نافذة زمنية أو جلسة، قم بتجميع كل الحزم في تدفقاتها الخاصة بناءً على مجموعاتها.
- (5) استخراج التعدادات الأساسية (Extract Basic Counts): لكل تدفق، احسب التعدادات البسيطة، مثل الحجم الإجمالي للحزم، وإجمالي البايتات المنقولة، وعدد كل نوع من العلامات المرصودة.
- (6) حساب الإحصائيات (Calculate Statistics): للسمات الرقمية مثل طول الحزمة ووقت الوصول، احسب إحصائيات الملخص مثل المتوسط والانحراف المعياري والحد الأدنى والحد الأقصى داخل كل تدفق.
- (7) اشتقاق الميزات الزمنية (Derive Temporal Features): حساب المقاييس المستندة إلى الوقت، مثل مدة التدفق، والحزم في الثانية، والبايتات في الثانية.
- (8) تحديد مقاييس التنوع (Identify Diversity Metrics): احسب حجم القيم الفريدة لسمات معينة. على سبيل المثال، لمصدر واحد، احسب العناوين الوجهة المتميزة أو المنافذ التي اتصل بها ضمن النافذة الزمنية.

(9) إنشاء متجه الميزات (Feature Vector Creation): ادج كل المقاييس المحسوبة من الخطوات المذكورة أعلاه في متجه رقمي واحد لكل تدفق. نتيجة لذلك، هذا المتجه هو المخرجات النهائية، الجاهزة لاستخدامها كعينة واحدة لتدريب أو تقييم نموذج التعلم الآلي.

4.4 التقنيات: الاكتشاف الخاضع للإشراف وغير الخاضع للإشراف (Techniques: Supervised & Unsupervised Detection)

بمجرد هندسة الميزات، نطبق خوارزميات التعلم الآلي لتصنيف حركة مرور الشبكة. في هذه الحالة، تقع هذه الخوارزميات عموماً في فئتين متميزتين من المخططات الخاضعة للإشراف وغير الخاضعة للإشراف، وكل منها مناسبة لأهداف اكتشاف مختلفة.

1.4.4 التصنيف الخاضع للإشراف (Supervised Classification)

نستفيد من التعلم الخاضع للإشراف عندما يكون لدينا مجموعة بيانات تحتوي على أمثلة مصنفة مسبقاً لحركة المرور العادية والضارة. لذلك، يتعلم النموذج تعييناً من ميزات الإدخال إلى التسميات المعروفة.

- الخوارزميات (Algorithms): تعد خوارزميات التعلم الآلي الكلاسيكية مثل الغابة العشوائية (Random Forest)، وتعزيز التدرج (Gradient Boosting)، وآلات المتجهات الداعمة (Support Vector Machines - SVM) فعالة للغاية لهذه المهمة. نتيجة لذلك، تتفوق في تعلم الأنماط المعقدة من متجهات الميزات للتمييز بين فئات مختلفة من حركة المرور.

- حالة الاستخدام (Use Case): هذه النماذج بارزة لاكتشاف الأنواع المعروفة من الهجمات الممثلة بشكل جيد في بيانات التدريب. بالإضافة إلى ذلك، فهي تعمل كتعزيز فعال للأنظمة التقليدية القائمة على التوقيع، مما يوفر طريقة أكثر مرونة لتحديد الاختلافات في التهديدات الشائعة.

2.4.4 اكتشاف الشذوذ غير الخاضع للإشراف (Unsupervised Anomaly Detection)

نستخدم التعلم غير الخاضع للإشراف عندما لا يكون لدينا بيانات مصنفة، أو عندما يكون الهدف الأسمى هو اكتشاف هجمات جديدة لم يتم رؤيتها من قبل. لهذا السبب، تتعلم هذه الخوارزميات نموذجاً لما يبدو عليه سلوك الشبكة الطبيعي ثم تضع علامة على أي نقاط بيانات تنحرف بشكل كبير عن هذا الأساس على أنها شذوذ [4].

- غابة العزل (Isolation Forest - `sklearn.ensemble.IsolationForest`): تعمل هذه الخوارزمية عن طريق بناء أشجار قرار عشوائية. نؤكد أن الفكرة الأساسية هي أن الحالات الشاذة قليلة ومختلفة، مما يسهل عزلها عن بقية البيانات. لذلك، ستطلب نقاط البيانات الشاذة عمليات تقسيم عشوائية أقل لعزلها في بنية شجرة. إنها فعالة وتؤدي بشكل جيد على البيانات عالية الأبعاد.

- آلة المتجهات الداعمة للفئة الواحدة (One Class SVM - `sklearn.svm.OneClassSVM`): نقوم بتدريب هذه الخوارزمية على بيانات من فئة واحدة فقط، وعادة ما تكون الفئة العادية. نتيجة لذلك، تتعلم حداً أو كرة فائقة تغلف أغلبية نقاط البيانات العادية في مساحة الميزة. تُصنف أي نقطة جديدة تقع خارج هذا الحد المكتسب على أنها حالة شاذة.

- حالة الاستخدام (Use Case): تعد الطرق غير الخاضعة للإشراف مهمة جداً لاكتشاف الهجمات الجديدة أو هجمات يوم الصفر (zero day attacks). نظراً لأن هذه الهجمات ليس لها توقعات أو تسميات موجودة مسبقاً، فإن النظام الذي يمكنه تحديد السلوك غير الطبيعي العام هو الطريقة الوحيدة لاكتشافها. بالإضافة إلى ذلك، تتضمن الأمثلة اكتشاف قنوات قيادة وتحكم جديدة للبرامج الضارة أو أنماط استخراج بيانات غير معتادة.

5.4 مجموعات البيانات المفتوحة لاكتشاف الشبكات (Open Datasets for Network ID)

مجموعات البيانات عالية الجودة المتاحة للجمهور هي مطلب لتدريب ووضع معايير لنماذج NIDS. نتيجة لذلك، تُستخدم عدة مجموعات بيانات قياسية على نطاق واسع في الأبحاث الأكاديمية والصناعية.

- NSL-KDD: نسخة منقحة ومحسنة من مجموعة البيانات الأصلية [73]. لطالما كانت

معياراً لأبحاث NIDS، حيث تحتوي على اتصالات مصنفة مع أنواع هجوم مختلفة مثل حجب الخدمة، وعن بُعد إلى محلي، ومستخدم إلى جذر، والسبر.

• **CIC-IDS2017/2018:** هذه مجموعات بيانات حديثة وأكثر واقعية تم تطويرها من قبل المعهد الكندي للأمن السيبراني [71]. تحتوي على هجمات معاصرة، مثل القوة العاشمة، وهجمات الويب، والشبكات الروبوتية، وتوفر كلاً من الملفات الخام الكاملة والميزات المولدة مسبقاً والمصنفة على مستوى التدفق.

• **UNSW-NB15:** مجموعة بيانات شاملة تم إنشاؤها في جامعة نيو ساوث ويلز، والتي تتضمن مزيماً من نشاط الشبكة الطبيعي وتسع عائلات من الهجمات الحديثة [57]. توفر كلاً من بيانات الحزم الخام ومجموعة من 49 ميزة مستخرجة مسبقاً، مما يجعلها مناسبة لمجموعة متنوعة من تجارب التعلم الآلي.

6.4 مشروع عملي: التعلم الآلي الكلاسيكي لنظام اكتشاف اختراق الشبكات (Hands on Project: Classical ML for NIDS)

في هذا المشروع، نوضح نظام NIDS بسيط ولكنه فعال باستخدام مُصنّف الغابة العشوائية (Random Forest classifier). نستفيد من مجموعة بيانات محاكاة لميزات مستخرجة مسبقاً على مستوى التدفق لتدريب نموذج يمكنه التمييز بين حركة المرور الحميدة وحركة الهجوم. العملية بأكملها، من تحميل البيانات إلى تقييم النموذج، موضحة في كود بايثون في القائمة 48.

1.6.4 تحميل البيانات (Data Loading)

تتطلب الخطوة الأولى دراسة متأنية لتحميل بياناتنا. من أجل هذا العرض التوضيحي، نقوم برمجياً بإنشاء إطار بيانات pandas يحاكي مجموعة بيانات حقيقية لتدفقات الشبكة. في سيناريو حقيقي، سيتم تحميل هذه البيانات من ملف CSV تم إنشاؤه بواسطة أداة مثل Zeek أو برنامج نصي مخصص يعالج الملفات. بالإضافة إلى ذلك، تتضمن مجموعة بياناتنا ميزات مثل مدة التدفق، والبايتات المنقولة، وعلامات البروتوكول، وتسمية تشير إلى ما إذا كان التدفق حميداً أم هجوماً. تخلق البيانات التركيبية بيئة تعليمية يمكن الوصول إليها، ولكنها تجلب أيضاً فرصة لعدم تمثيل التوزيعات المعقدة في العالم الحقيقي.

2.6.4 توليد الميزات (مفاهيمي) (Feature Generation - Conceptual) (tural)

من المهم جداً أن نتذكر أن البيانات التي نعملها معالجة بالفعل. في نظام حقيقي، يسبق هذه الخطوة جهد هندسي كبير. نلتقط البيانات الخام من الملفات أو حركة المرور المباشرة ونعالجها من خلال خط أنابيب، مثل ذلك الموصوف في قسم التحليل الخوارزمي، لحساب وتجميع الميزات لكل تدفق. لذلك، يسمح لنا هذا التجريد بالتركيز على جانب التعلم الآلي من المشكلة.

3.6.4 تقسيم البيانات وقياسها (Data Splitting & Scaling)

في هذا السياق، نفصل مصفوفة بياناتنا المجمعة إلى مساحة ميزات مستقلة X ومنتجه هدف تابع y . لتقييم قدرات تعميم النموذج بشكل فعال، نقسم مجموعة البيانات إلى مجموعة تدريب متميزة لتحسين المعلومات ومجموعة تحقق مقطوعة للتقييم الصارم للأداء. بينما تكون المجموعات القائمة على الأشجار ثابتة بطبيعتها للتحويلات الرتيبة ولا تتطلب بشكل صارم تسوية الميزات، فإننا ننسق قياس الميزات بالاستفادة من StandardScaler لإنشاء خط أنابيب أساسي موحد للتحليل المقارن ضد الخوارزميات القائمة على المسافة. نتيجة لذلك، توحد هذه الآلية توزيع كل ميزة متميزة للحفاظ على متوسط يساوي صفراً وتبايناً موحداً، وهو مطلب ذو أهمية قصوى عند دمج خوارزميات المصنوع بسلسلة مثل آلات المتجهات الداعمة أو الهياكل العصبية العميقة.

4.6.4 تدريب النموذج (Model Training)

علاوة على ذلك، وبعد إنشاء خط أنابيب معالجة مسبقة للبيانات مرنة، ننتقل إلى تنسيق مرحلة التدريب لإطار العمل المقترح. تحقيقاً لهذه الغاية، نستفيد من تنفيذ RandomForestClassifier المقدم من مكتبة scikit-learn لبناء نموذج مجمع فعال وقابل للتطوير. يساعد تعقيد توزيعات حركة المرور المنحرفة في بناء نماذج واقعية، ولكنه يجلب أيضاً فرصة لتحيز فئة الأغلبية. وبالتالي، نحدد class_weight='balanced' لمعالجة التصنيفات الخاطئة لفئة الأقلية ديناميكياً واستيعاب الاختلافات الفئات الأساسية. في النهاية، نقوم بتحسين حدود قرار المجموعة من خلال تنفيذ آلية fit() حصرياً على قسم التدريب الموحد الخاص بنا.

5.6.4 التقييم (Evaluation)

بعد التدريب، يجب علينا تقييم أداء النموذج على بيانات الاختبار غير المرئية. نستخدم classification_report لعرض المقاييس الرئيسية مثل الدقة والاستدعاء ودرجة F1 لكل فئة. في الأمن السيبراني، نهتم بشكل خاص بمقايضتين.

- السليبيات الخاطئة (استدعاء منخفض) (False Negatives - Low Recall): يحدث هذا عندما يصف النموذج هجوماً حقيقياً بأنه حميد. هذا هو أخطر أنواع الأخطاء، لأنه يعني أن التهديد يمر دون اكتشاف. نهدف إلى استدعاء عالٍ جداً لفئة الهجوم.

- الإيجابيات الخاطئة (دقة منخفضة) (False Positives - Low Precision): يحدث هذا عندما يصف النموذج حركة المرور الحميدة بأنها هجوم. على الرغم من كونها أقل خطورة، إلا أن كثرة الإيجابيات الخاطئة تولد إنذارات كاذبة، مما يؤدي إلى إرهاق التنبيهات (alert fatigue) حيث قد يبدأ محللو الأمن في تجاهل التحذيرات. نقوم بحساب معدل الإيجابيات الخاطئة (False Positive Rate)، والذي يقيس نسبة العينات الحميدة التي تم تصنيفها بشكل غير صحيح كهجمات.

يحقق نموذج NIDS المرن توازناً مميزاً لتعظيم اكتشاف الهجوم مع تقليل الإنذارات الكاذبة إلى مستوى يمكن إدارته. في هذا السياق، نؤكد على حالة الحافة الحاسمة للتهرب العدائي (ad-ersarial evasion). كثيراً ما يصوغ المهاجمون حملات ضارة لتقليد التوزيعات الإحصائية للفئة الحميدة. وبالتالي، فإن تعظيم الاستدعاء على مجموعات البيانات القياسية لا يضمن المرونة ضد الأمثلة المضادة التكيفية والعدائية. لهذا السبب، يتطلب هذا المطلب تقييماً مستمراً ودقيقاً.

6.6.4 تحليل أهمية الميزات (Feature Importance Analysis)

أخيراً، إحدى فوائد النماذج القائمة على الأشجار مثل الغابة العشوائية هي أنها تستطيع أن تخبرنا عن الميزات التي كانت أكثر تأثيراً في اتخاذ قراراتها. نصل إلى السمة feature_importances_ لنموذجنا المدرب. من خلال رسم هذه القيم، يمكننا رؤية خصائص الشبكة التي تعد أقوى المؤشرات على الهجوم. بالإضافة إلى ذلك، هذه المعلومات قيمة للغاية، حيث يمكن أن تتحقق من صحة معرفة المجال الخاصة بنا وتوفر رؤى قابلة للتنفيذ لمحللي الأمن البشريين.

خاتمة (Conclusion)

قدم هذا الفصل مقدمة شاملة لبناء نظام اكتشاف اختراق الشبكات باستخدام تقنيات التعلم الآلي الكلاسيكية. تعلمنا أن العملية تبدأ بفهم مشهد التهديدات وأنواع البيانات المتاحة للتحليل، مثل الملفات الخام وسجلات التدفق المخصصة. تتطلب الخطوة الأكثر أهمية في هذه العملية دراسة متأنية لهندسة الميزات (feature engineering)، حيث نطبق خبرة المجال لتحويل البيانات الخام إلى تمثيلات رقمية ذات معنى تلتقط سلوك حركة مرور الشبكة. استكشفنا الأدوار المتميزة للتعلم الخاضع للإشراف لاكتشاف الهجمات المعروفة والتعلم غير الخاضع للإشراف لتحديد الحالات الشاذة الجديدة. أظهر المشروع العملي سير عمل عملياً للتخصيص والتدريب وتقييم الأداء مع التركيز على المقاييس ذات الصلة بالأمان، مثل معدل الإيجابيات الخاطئة. أخيراً، نؤكد على كيف يمكن لتحليل أهمية الميزات أن يوفر رؤى قيمة لمختبري الأمان. مع هذا الأساس، أنت الآن مجهز لفهم وبناء أنظمة ذكية تشكل خط دفاع أول مرن في الأمن السيبراني الحديث.

```

1 import pandas as pd
2 import numpy as np
3 import matplotlib.pyplot as plt
4 from sklearn.model_selection import train_test_split
5 from sklearn.preprocessing import StandardScaler
6 from sklearn.ensemble import RandomForestClassifier
7 from sklearn.metrics import classification_report, confusion_matrix
8
9 # --- 1. Data Loading and Preparation ---
10 # Create a dummy dataset resembling network flow data for demonstration.
11 # In a real scenario, you would load this from a CSV file.
12 np.random.seed(42) # for reproducibility
13 data_size = 200
14 data_flows = {
15     'duration': np.random.rand(data_size) * 10,
16     'src_bytes': np.random.randint(20, 1500, data_size),
17     'dst_bytes': np.random.randint(50, 3000, data_size),
18     'protocol_tcp': np.random.randint(0, 2, data_size),
19     'flag_syn': np.random.randint(0, 2, data_size),
20     'flag_ack': np.random.randint(0, 2, data_size),
21     'num_connections_min': np.random.randint(1, 50, data_size),
22     'label': np.random.choice([0, 1], data_size, p=[0.7, 0.3]) # 0=benign, 1=attack
23 }
24 df_flows = pd.DataFrame(data_flows)
25 df_flows['protocol_udp'] = 1 - df_flows['protocol_tcp']
26
27 # Make attack data more distinct to ensure the model can learn patterns
28 attack_indices = df_flows[df_flows['label'] == 1].index
29 df_flows.loc[attack_indices, 'duration'] = np.random.rand(len(attack_indices)) * 0.1
30 df_flows.loc[attack_indices, 'num_connections_min'] = np.random.randint(100, 500, len(attack_indices))
31
32 X = df_flows.drop('label', axis=1)
33 y = df_flows['label']
34 print("--- Data Preparation ---")
35 print("Network Flow Features (X) Head:\n", X.head())
36 print("\nLabels (y) Value Counts:\n", y.value_counts())
37
38 # --- 2. Data Splitting & Scaling ---
39 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42, stratify=y)
40 scaler = StandardScaler()
41 X_train_scaled = scaler.fit_transform(X_train)
42 X_test_scaled = scaler.transform(X_test)
43 print("\n--- Data Splitting and Scaling ---")
44 print(f"Training set shape: {X_train_scaled.shape}, Test set shape: {X_test_scaled.shape}")
45
46 # --- 3. Model Training ---
47 model_nids = RandomForestClassifier(n_estimators=100, random_state=42, class_weight='balanced')
48 model_nids.fit(X_train_scaled, y_train)
49 print("\n--- Model Training ---")
50 print("RandomForestClassifier for NIDS training complete.")
51
52 # --- 4. Evaluation ---
53 y_pred_nids = model_nids.predict(X_test_scaled)
54 print("\n--- NIDS Model Evaluation on Test Set ---")
55 print(classification_report(y_test, y_pred_nids, target_names=['Benign', 'Attack']))
56
57 # Calculate and print False Positive Rate
58 tn, fp, fn, tp = confusion_matrix(y_test, y_pred_nids).ravel()
59 fpr = fp / (fp + tn) if (fp + tn) > 0 else 0
60 print(f"False Positive Rate (FPR): {fpr:.2f}")
61 print("\nNote: Low False Negative Rate (high Recall for 'Attack') is crucial.")
62 print("However, a high False Positive Rate leads to 'alert fatigue'.")
63
64 # --- 5. Feature Importance Analysis ---
65 importances = model_nids.feature_importances_
66 feature_names = X.columns
67 feature_importance_df = pd.DataFrame({'feature': feature_names, 'importance': importances})
68 feature_importance_df = feature_importance_df.sort_values(by='importance', ascending=False)
69 print("\n--- Top Feature Importances ---")
70 print(feature_importance_df.head(10))
71
72 # Visualize feature importances
73 plt.figure(figsize=(10, 6))
74 plt.barh(feature_importance_df['feature'], feature_importance_df['importance'], color='skyblue')
75 plt.xlabel('Importance')
76 plt.ylabel('Feature')
77 plt.title('Feature Importances for Network Intrusion Detection')
78 plt.gca().invert_yaxis()
79 plt.tight_layout()
80 # In a script, you would use plt.show() here. In a book, we might save the figure.
81 plt.show()

```

Listing 48: خط أنابيب كامل لتدريب وتقييم نموذج غابة عشوائية لنظام اكتشاف اختراق الشبكات (Forest NIDS model).
A complete pipeline for training and evaluating a Random

المختصرات والمصطلحات

واجهة برمجة التطبيقات (API) مجموعة من القواعد التي تسمح لتطبيقات برمجية مختلفة بالتواصل وتبادل البيانات مع بعضها البعض. في هذا السياق، تسمح لنظام أمني بأن يطلب من نموذج الذكاء الاصطناعي إجراء تنبؤ.

المساحة تحت منحنى الدقة والاستدعاء (AUPRC) (Area Under the Precision-Recall Curve) مقياس أداء لنموذج تصنيف، وهو مفيد بشكل خاص في مجموعات البيانات غير المتوازنة. يلخص قدرة النموذج على الموازنة بين الدقة precision والاستدعاء recall عبر جميع عتبات القرار.

حقيبة الكلمات (Bag-of-Words (BoW طريقة بسيطة لتمثيل البيانات النصية عددياً عن طريق حساب تكرار كل كلمة، مع تجاهل القواعد النحوية وترتيب الكلمات.

تشفير ثنائية الاتجاه من المحولات (BERT) (Bidirectional Encoder Representations from Transformers) نموذج لغوي قوي يعتمد على الذكاء الاصطناعي، يفهم سياق الكلمة من خلال فحص الكلمات التي تأتي قبلها وبعدها.

المصنف الثنائي Binary Classifier نموذج ذكاء اصطناعي يصنف المدخلات إلى فئتين متميزتين، مثل 'تصيد احتيالي' أو 'شرعي'.

عدم توازن الفئات Class Imbalance مشكلة شائعة في مجموعات البيانات حيث تحتوي فئة (مثل رسائل البريد الإلكتروني الشرعية) على أمثلة أكثر بكثير من فئة أخرى (مثل رسائل التصيد الاحتيالي)، مما قد يؤدي إلى تحيز النموذج.

أوزان الفئات Class Weights تقنية تُستخدم أثناء تدريب النموذج، تقوم بتعيين عقوبة أعلى لتصنيف الفئة الأقل شيوعاً (فئة الأقلية) بشكل خاطئ، مما يجبر النموذج على إيلاء المزيد من الاهتمام لها.

الشبكة العصبية الالتفافية (CNN) (Convolutional Neural Network) نوع من نماذج التعلم العميق، يُستخدم غالباً للتعرف على الصور، ويمكن تكييفه للعثور على الأنماط والتسلسلات الرئيسية للكلمات في النص لأغراض التصنيف.

بريد مُعرّف بمفاتيح النطاق DKIM (DomainKeys Identified Mail) طريقة لمصادقة البريد الإلكتروني تضيف توقيعاً رقمياً إلى رسائل البريد الإلكتروني للتحقق من أن محتوى الرسالة لم يتم تغييره وأنه تم إرسالها من نطاق مصرح به.

القائمة على النطاق (DMARC (Domain-based Message Authentication, Reporting, and Conformance)

بروتوكول لأمن البريد الإلكتروني يعتمد على SPF و DKIM ويوفر سياسة لكيفية التعامل مع رسائل البريد الإلكتروني التي تفشل في فحوصات المصادقة.

الطريقة التجميعية Ensemble Method تقنية في تعلم الآلة machine learning تجمع بين تنبؤات نماذج متعددة لتحقيق أداء ودقة أفضل مما يمكن لأي نموذج واحد تحقيقه بمفرده.

مقياس F1-score F1 مقياس أداء وحيد يمثل التوازن بين الدقة precision والاستدعاء recall، ويُحسب كمتوسط توافقي لهما. وهو مفيد بشكل خاص لتقييم النماذج على البيانات غير المتوازنة.

السلبي الخاطئ False Negative خطأ يحدث عندما يفشل النموذج في اكتشاف تهديد. في هذا السياق، يتم تصنيف بريد إلكتروني للتصيد الاحتمالي بشكل غير صحيح على أنه شرعي.

الإيجابي الخاطئ False Positive خطأ يحدث عندما يحدد النموذج تهديداً بشكل غير صحيح. في هذا السياق، يتم الإبلاغ عن بريد إلكتروني شرعي بشكل خاطئ على أنه تصيد احتمالي، مما قد يؤدي إلى حظر الاتصالات المهمة.

هندسة الميزات Feature Engineering عملية اختيار وتحويل وإنشاء متغيرات الإدخال (الميزات) من البيانات الأولية لتحسين أداء نموذج تعلم الآلة.

الاستدلال Inference عملية استخدام نموذج ذكاء اصطناعي مدرب لإجراء تنبؤ على بيانات جديدة لم يسبق رؤيتها.

الإرجاع إلى الأصل المعجمي Lemmatization تقنية لمعالجة النصوص تقوم بإرجاع الكلمات إلى صيغتها الأساسية أو شكلها في القاموس (lemma)، مع مراعاة سياق الكلمة (على سبيل المثال، "better" تصبح "good").

الذاكرة طويلة قصيرة المدى LSTM (Long Short-Term Memory) نوع متخصص من الشبكات العصبية المتكررة Recurrent Neural Network (RNN) قادر على تعلم الاعتماديات طويلة الأمد في البيانات التسلسلية مثل النصوص، مما يجعله فعالاً في فهم السياق.

زيادة عينات الفئة الأقلية Oversampling استراتيجية للتعامل مع مجموعات البيانات غير المتوازنة عن طريق زيادة عدد الأمثلة في فئة الأقلية، على سبيل المثال، عن طريق تكرارها.

التصيد الاحتيالي Phishing هجوم سبراني cyberattack يقوم فيه المجرمون بإرسال رسائل بريد إلكتروني احتيالية متخفية كمصدر شرعي لخداع الضحايا للكشف عن معلومات حساسة مثل كلمات المرور أو التفاصيل المالية.

المعلومات الشخصية المحددة للهوية PII (Personally Identifiable Information) أي بيانات يمكن استخدامها لتحديد هوية شخص معين، مثل الاسم أو عنوان البريد الإلكتروني أو رقم الضمان الاجتماعي، والتي يجب حمايتها لضمان الخصوصية.

الدقة Precision مقياس أداء يقيس نسبة التنبؤات الإيجابية التي كانت صحيحة بالفعل. الدقة العالية تقلل من الإيجابيات الخاطئة false positives.

الاستدعاء Recall مقياس أداء، يسمى أيضاً الحساسية، يقيس نسبة الحالات الإيجابية الفعلية التي تم تحديدها بشكل صحيح. الاستدعاء العالي يقلل من السلبات الخاطئة false negatives.

الشبكة العصبية المتكررة RNN (Recurrent Neural Network) نوع من الشبكات العصبية neural network مصمم لمعالجة تسلسلات البيانات، مثل النصوص، حيث يمكن للمعلومات أن تستمر من خطوة إلى أخرى.

إدارة معلومات وأحداث الأمان SIEM (Security Information and Event Management) نظام يُستخدم في الأمن السبراني يجمع ويحلل البيانات المتعلقة بالأمان من جميع أنحاء المؤسسة لاكتشاف التهديدات وإدارة الحوادث.

تقنية زيادة عينات الأقلية الاصطناعية SMOTE (Synthetic Minority Over-sampling Technique) طريقة متقدمة للتعامل مع البيانات غير المتوازنة تقوم بإنشاء أمثلة اصطناعية جديدة لفئة الأقلية بدلاً من مجرد تكرار الأمثلة الموجودة.

إطار سياسة المرسل SPF (Sender Policy Framework) طريقة لمصادقة البريد الإلكتروني تسمح للنطاق بتحديد خوادم البريد المصرح لها بإرسال البريد الإلكتروني نيابة عنه، مما يساعد على منع تزوير عنوان المرسل.

التجذير Stemming تقنية لمعالجة النصوص تقوم بإرجاع الكلمات إلى جذرها عن طريق إزالة السوابق أو اللواحق (على سبيل المثال، "running" تصبح "run") وهي أبسط وأسرع من الإرجاع إلى الأصل المعجمي lemmatization.

آلة المتجهات الداعمة SVM (Support Vector Machine) خوارزمية تعلم آلة خاضعة للإشراف supervised machine learning algorithm تصنف البيانات عن طريق إيجاد الحد الأمثل (أو المستوى الفائق) الذي يفصل بشكل أفضل بين الفئات المختلفة.

تردد المصطلح-معكوس تردد الوثيقة (TF-IDF (Term Frequency-Inverse Document Frequency
إحصائية عددية تعكس مدى أهمية كلمة ما لوثيقة في مجموعة من الوثائق. تسلط الضوء على
الكلمات التي تتكرر في بريد إلكتروني واحد ولكنها نادرة بشكل عام.

الترميز Tokenization الخطوة الأولى في معالجة النصوص حيث يتم تقسيم سلسلة من النصوص
إلى وحدات أصغر تسمى الرموز (tokens) (مثل الكلمات أو علامات الترقيم أو الكلمات
الفرعية).

تقليل عينات الفئة الأكثرية Undersampling استراتيجية للتعامل مع مجموعات البيانات غير
المتوازنة عن طريق تقليل عدد الأمثلة في فئة الأكثرية لإنشاء توزيع أكثر توازناً.

تضمين الكلمات Word Embeddings تقنية في معالجة اللغات الطبيعية -Natural Lan-
guage Processing حيث يتم تمثيل الكلمات كمتجهات عددية كثيفة. الكلمات ذات
المعاني المتشابهة لها تمثيلات متجهة متشابهة، مما يسمح لنماذج الذكاء الاصطناعي بفهم
العلاقات الدلالية.

باب 5

كشف اختراق الشبكات: التعلم العميق للبيانات الأولية والزمنية

مقدمة

غالباً ما تواجه أنظمة كشف اختراق الشبكات التقليدية (Traditional Network Intrusion Detection Systems - NIDS) صعوبة في تحديد الهجمات المعقدة التي تتكشف بمرور الوقت. تم تصميم هذه التهديدات الخفية، مثل نشاط شبكات الروبوت botnet المنسق، أو تسريب البيانات البطيء والمنخفض low-and-slow data exfiltration، أو التهديدات المستمرة المتقدمة متعددة المراحل (Advanced Persistent Threats - APTs)، لتجنب الكشف البسيط القائم على التوقيع signature-based detection. فطبيعتها الخبيثة لا تظهر في حزمة شبكة أو تدفق واحد، بل يتم الكشف عنها في تسلسل وسياق أحداث متعددة. ويمثل إنشاء القواعد أو السمات يدوياً لاكتشاف هذه الأنماط تحدياً كبيراً، حيث يغير المهاجمون تكتيكاتهم باستمرار.

يستكشف هذا الفصل كيف يمكن لنماذج التعلم العميق deep learning التغلب على هذه القيود. من خلال تحليل بيانات الشبكة الأولية raw network data أو التسلسلات الطويلة لأحداث الشبكة، يمكن لتقنيات التعلم العميق أن تتعلم تلقائياً الأنماط المعقدة والهرمية والزمنية التي تشير إلى هجوم منسق. تتيح لهم هذه القدرة اكتشاف التهديدات التي قد تفوتها الطرق التقليدية. تتمثل أهداف التعلم الرئيسية لهذا الفصل في فهم تطبيق معماريات التعلم العميق المختلفة لبيانات الشبكة، وإعداد بيانات السلاسل الزمنية لهذه النماذج، وبناء نظام كشف اختراق شبكات عملي باستخدام PyTorch لتصنيف تسلسلات حركة مرور الشبكة.

1.5 التقنيات: معماريات التعلم العميق

يقدم التعلم العميق Deep learning مجموعة أدوات قوية لتحليل البيانات المعقدة. بالنسبة لكشف اختراق الشبكات، تعد بعض المعماريات المحددة فعالة للغاية في تعلم الأنماط من البيانات التسلسلية والزمنية دون الحاجة إلى هندسة سمات يدوية مكثفة.

1.1.5 الشبكات العصبونية الالتفافية للبيانات الأولية

الشبكة العصبونية الالتفافية (Convolutional Neural Network - CNN) هي نوع من نماذج التعلم العميق التي تتفوق في إيجاد الأنماط المحلية في البيانات، مما جعلها مشهورة في التعرف على الصور [?]. يمكننا تطبيق هذا المبدأ نفسه على حركة مرور الشبكة. على سبيل المثال، يمكن التعامل مع بيانات الحزمة الأولية كتسلسل أحادي البعد من البايتات، على غرار صورة أحادية البعد. يمكن لشبكة CNN تمرير مرشح عبر تسلسل البايت هذا لتعلم الأنماط المفيدة تلقائياً، مثل هياكل ترويسات معينة أو توقعات حمولات خبيثة.

وبالمثل، يمكن تطبيق شبكة CNN على تسلسل من تدفقات الشبكة، حيث يتم تمثيل كل تدفق بمجمعه من السمات الرقمية. يؤدي هذا إلى إنشاء مصفوفة ثنائية الأبعاد (تدفقات مرتبة زمنياً مقابل السمات)، ويمكن للشبكة العصبونية الالتفافية تحديد الارتباطات المحلية بين التدفقات المتتالية التي قد تشير إلى هجوم. تمثل الفائدة الأساسية لاستخدام شبكة CNN في قدرتها على تعلم هذه الأنماط المكانية الهرمية تلقائياً. في PyTorch، يتم تنفيذ ذلك بشكل شائع باستخدام الوحدة `torch.nn.Conv1d` للبيانات التسلسلية.

2.1.5 RNNs و LSTMs و GRUs للبيانات الزمنية

عندما يكون ترتيب وتوقيت الأحداث حاسماً، تكون الشبكات العصبونية المتكررة (Recurrent Neural Networks - RNNs) خياراً طبيعياً. تم تصميم شبكات RNNs لمعالجة التسلسلات عن طريق الحفاظ على حالة داخلية أو ذاكرة تلتقط المعلومات من الخطوات الزمنية السابقة. ومع ذلك، تواجه شبكات RNNs القياسية صعوبة في تذكر المعلومات عبر التسلسلات الطويلة بسبب مشكلة تُعرف باسم التدرج المتلاشي vanishing gradient.

لحل هذه المشكلة، تم تطوير متغيرات أكثر تقدماً مثل الذاكرة طويلة المدى قصيرة الأجل (Long Short-Term Memory - LSTM) [33] ووحدات البوابات المتكررة (Gated Recurrent Units - GRUs). تستخدم LSTMs و GRUs آليات بوابات gating mechanisms خاصة تسمح للنموذج بتذكر المعلومات أو نسيانها بشكل انتقائي على مدى فترات

طويلة. هذا يجعلها مثالية للكشف عن الهجمات ذات الحالة، مثل اتصالات القيادة والتحكم المستمرة C2، أو بروتوكولات الأنفاق، أو تسريب البيانات البطيء والمنخفض حيث ينتشر النشاط الخبيث عبر العديد من الأحداث التي تبدو حميدة. توفر مكتبة PyTorch تطبيقات مُحسَّنة مع `torch.nn.LSTM` و`torch.nn.GRU`.

3.1.5 نماذج المحولات للتبعيات طويلة المدى

أثبتت معمارية المحول Transformer، التي صُممت في الأصل لمعالجة اللغة الطبيعية، فعاليتها الاستثنائية في التقاط التبعيات طويلة المدى جداً في البيانات التسلسلية [2]. ابتكارها الأساسي هو آلية الانتباه (attention mechanism)، التي تسمح للنموذج بتقدير أهمية الأحداث المختلفة في تسلسل عند إجراء تنبؤ، بغض النظر عن بعدها عن بعضها البعض. بالنسبة لأمن الشبكات، يعني هذا أن المحول Transformer يمكنه تحليل سلسلة طويلة من سجلات NetFlow أو سجلات النظام وتحديد الأنشطة الدقيقة والمنسقة التي تفصل بينها فترات زمنية طويلة. على سبيل المثال، يمكنه ربط فحص استطلاع أولي بمحاولة تسريب بيانات تحدث بعد ساعات أو أيام. هذا يجعل المحولات قوية لفهم اتصالات القيادة والتحكم المعقدة C2 communications أو تحديد الهجمات الموزعة متعددة النواقل. يمكن بناء نماذج محولات مخصصة في PyTorch باستخدام وحدات مثل `torch.nn.TransformerEncoderLayer`.

4.1.5 إعداد بيانات الشبكة كسلاسل زمنية

قبل إدخال بيانات الشبكة في نموذج تعلم عميق، يجب تحويلها إلى تنسيق رقمي منظم. تتضمن هذه العملية عدة خطوات رئيسية [47].

- (1) تحديد جلسة أو نافذة زمنية: يجب تجميع أحداث الشبكة في تسلسلات ذات معنى. يمكن القيام بذلك عن طريق تحديد فاصل زمني ثابت (على سبيل المثال، جميع التدفقات في نافذة مدتها 5 ثوانٍ)، أو جلسة مستخدم (على سبيل المثال، كل نشاط من تسجيل دخول مستخدم واحد)، أو جلسة اتصال (على سبيل المثال، جميع الحزم لاتصال TCP واحد).
- (2) إنشاء التسلسل: داخل كل نافذة، قم بإنشاء تسلسل مرتب من البيانات الرقمية. بالنسبة للحزم الأولية، يتضمن ذلك تحويل كل بايت إلى عدد صحيح (0-255) وتعبئة المحولات أو اقتطاعها إلى طول ثابت. بالنسبة لتدفقات الشبكة أو السجلات، يعني هذا تكديس متجهات السمات الخاصة بها في مصفوفة (على سبيل المثال، يصبح تسلسل من 10 تدفقات مع 20 سمّة لكل منها مصفوفة 10×20).

(3) **التطبيع/التحجيم:** غالباً ما تحتوي السمات الرقمية على مقاييس مختلفة (على سبيل المثال، عدد الحزم مقابل عدد البايتات). يجب عليك تحجيمها إلى نطاق مشترك، مثل 0 إلى 1، باستخدام أدوات مثل `MinMaxScaler` من `scikit-learn`. هذا يمنع السمات ذات القيم الكبيرة من السيطرة على عملية التعلم.

(4) **التعبئة/الاقتطاع:** تتطلب نماذج التعلم العميق عادةً أن يكون لجميع التسلسلات في دفعة تدريب نفس الطول. يتم تعبئة التسلسلات الأقصر بالأصفار، بينما يتم اقتطاع التسلسلات الأطول إلى أقصى طول.

(5) **التقسيم إلى دفعات وتحويل الموترات:** أخيراً، يتم تجميع التسلسلات المعدة في دفعات وتحويلها إلى موترات PyTorch. الموتر (tensor) هو مصفوفة متعددة الأبعاد يمكن معالجتها بكفاءة على وحدات معالجة الرسومات. تُستخدم فئة `torch.utils.data.DataLoader` لإدارة هذه العملية.

2.5 مجموعات البيانات المفتوحة لأنظمة كشف الاختراق العميقة

تعد مجموعات البيانات عالية الجودة ضرورية لتدريب وتقييم نماذج التعلم العميق لأنظمة كشف اختراق الشبكات. يجب أن تحتوي مجموعات البيانات هذه على أمثلة مصنفة لكل من حركة المرور العادية والخبيثة، ويفضل أن تحتوي على التقاطات كاملة للحزم أو سجلات مفصلة ومختومة زمنياً.

1.2.5 مجموعة بيانات CTU-13

تعد مجموعة بيانات CTU-13 مصدراً معروفاً للبحث في تحليل حركة مرور شبكات الروبوت botnet [28]. تم التقاطها في جامعة CTU بجمهورية التشيك، وتتكون من ثلاثة عشر سيناريو مختلفاً لإصابات حقيقية بشبكات الروبوت. تتضمن مجموعة البيانات التقاطات كاملة للحزم (PCAPs)، وتدفعات شبكة NetFlows، وتصنيفات تم إنشاؤها بخبرة تحدد أي حركة مرور حميدة، أو خلفية، أو من شبكة روبوت معينة. معلوماتها الزمنية الغنية وسيناريوهات الهجوم الواقعية تجعلها خياراً ممتازاً لتطوير واختبار النماذج المصممة لاكتشاف اتصالات القيادة والتحكم المعقدة C2.

2.2.5 مجموعات بيانات أخرى

هناك العديد من مجموعات البيانات العامة الأخرى المناسبة لأنظمة كشف الاختراق القائمة على التعلم العميق. غالباً ما توفر مجموعات البيانات الأحداث مثل سلسلة CIC-IDS سجلات تدفق مفصلة مع طابع زمنية يمكن استخدامها لنمذجة التسلسل. بالإضافة إلى ذلك، تصدر مسابقات أمن الشبكات وأحداث التقاط العلم (capture-the-flag - CTF) أحياناً مجموعات بيانات مع التقاطات حزم أولية لتقنيات الهجوم الحديثة. غالباً ما تتطلب مجموعات البيانات هذه برامج نصية مخصصة للتحليل والتصنيف لإعدادها للتعلم العميق، لكنها توفر أمثلة قيمة للتهديدات الحالية.

3.5 مشروع عملي: التعلم العميق لأنظمة كشف اختراق الشبكات

يوضح هذا المشروع كيفية بناء نظام كشف اختراق شبكات بسيط قائم على LSTM باستخدام PyTorch. الهدف هو تصنيف تسلسلات تدفقات الشبكة على أنها إما حميدة أو شاذة. سنستخدم pandas لإعداد البيانات وPyTorch لبناء النموذج وتدريبه.

1.3.5 إعداد البيانات

الخطوة الأولى هي تحميل ومعالجة بيانات الشبكة في تسلسلات ذات طول ثابت. يتضمن ذلك إنشاء بيانات تدفق اصطناعية، وتجميعها في نوافذ زمنية، واستخراج السمات الرقمية، وتعبئة التسلسلات إلى طول موحد. يحاكي الكود في القائمة 49 هذه العملية. نقوم بإنشاء DataFrame لتدفقات الشبكة، وندخل بعض التدفقات الشاذة ذات أعداد بايتات عالية بشكل غير عادي، ثم نجمع هذه التدفقات في نوافذ مدتها 10 ثوان. تصبح كل نافذة تسلسلاً، يتم تعبئته لضمان أن جميع التسلسلات لها نفس الطول لتدريب النموذج.

2.3.5 تعريف نموذج PyTorch

بعد ذلك، نحدد نموذج LSTM الخاص بنا باستخدام فئة nn.Module في PyTorch. كما هو موضح في القائمة 50، تحتوي فئة LSTMNet الخاصة بنا على طبقة LSTM لمعالجة تسلسلات الإدخال وطبقة نهائية متصلة بالكامل (nn.Linear) لإنتاج مخرجات التصنيف. نحدد الوسيطة batch_first=True في LSTM لأن تنسيق موتر الإدخال سيكون (الدفع، طول التسلسل، السمات)، وهو ما يتطابق مع بياناتنا المعدة. تأخذ التمريرة الأمامية تسلسل إدخال، وتعالجه من خلال LSTM، وتستخدم الإخراج من الخطوة الزمنية النهائية للتصنيف.

3.3.5 التدريب باستخدام PyTorch

مع إعداد البيانات وتعريف النموذج، يمكننا الآن تنفيذ عملية التدريب. يوضح الكود في القائمة 51 حلقة تدريب قياسية في PyTorch. أولاً، نقوم بتحويل مصفوفات NumPy الخاصة بنا إلى موترات PyTorch ونستخدم `TensorDataset` و `DataLoader` لإنشاء محملات بيانات قابلة للتكرار للتدريب والاختبار. نحدد دالة الخسارة الخاصة بنا، `nn.CrossEntropyLoss` وهي مناسبة للتصنيف متعدد الفئات، ومحسن `Adam` لتحديث أوزان النموذج. تتكرر حلقة التدريب لعدد محدد من الحقب، وتقوم بقريرات أمامية وخلفية لتدريب الشبكة.

4.3.5 التقييم والتفسير

أخيراً، نقوم بتقييم النموذج المدرب على بيانات الاختبار غير المرئية. كما هو موضح في القائمة 52، نضبط النموذج على وضع التقييم باستخدام `model.eval()` ونعطل حسابات التدرج باستخدام `torch.no\_grad()` لتحقيق الكفاءة. تتكرر عبر بيانات الاختبار، ونقوم بالتنبؤات، ونقارنها بالتصنيفات الحقيقية. نستخدم مقاييس قياسية مثل الدقة `accuracy`، والدقة `precision`، والاستدعاء `recall`، ودرجة F1-score، التي تم إنشاؤها بواسطة `classification\_report` من `scikit-learn` لتقييم الأداء. بينما يركز هذا العرض التوضيحي على نموذج LSTM، يمكن أن يكون تفسير نماذج التعلم العميق معقداً. بالنسبة لنماذج المحولات Transformer، يمكن أن تساعد تقنيات مثل تصور أوزان الانتباه attention weights في الكشف عن الأحداث في التسلسل التي وجدها النموذج أكثر أهمية لقراره.

الخلاصة

أظهر هذا الفصل قوة التعلم العميق لكشف اختراق الشبكات المتقدم. نماذج مثل CNNs و LSTMs و Transformers مناسبة بشكل فريد لتحليل بيانات الشبكة الأولية والزمنية، مما يسمح لها بتعلم الأنماط المعقدة والتسلسلية التي تشير إلى التهديدات السيبرانية الحديثة تلقائياً. على عكس الأنظمة التقليدية التي تعتمد على السمات المهندسة يدوياً، يمكن لهذه النماذج الكشف عن الارتباطات الدقيقة والتبعيات طويلة المدى المخفية داخل تسلسلات أحداث الشبكة. أظهر المشروع العملي سير العمل الكامل، من إعداد بيانات السلاسل الزمنية إلى بناء وتدريب وتقييم نموذج LSTM في PyTorch. من خلال إتقان هذه التقنيات، يمكن لمختري الأمن السيبراني بناء أنظمة دفاع أكثر ذكاءً وتكيفاً قادرة على تحديد الهجمات المعقدة التي لم يتم اكتشافها سابقاً.

```

1 import pandas as pd
2 import numpy as np
3 from collections import Counter
4 from sklearn.model_selection import train_test_split
5
6 # Dummy data: Simulate network flows with timestamps and simple features
7 np.random.seed(42)
8 num_flows = 200
9 timestamps = pd.to_datetime('2025-08-16 10:00:00') + pd.to_timedelta(np.arange(num_flows), unit='s')
10 df_flows_raw = pd.DataFrame({
11     'timestamp': timestamps,
12     'flow_id': np.arange(num_flows),
13     'src_ip': ['192.168.1.' + str(np.random.randint(1, 20)) for _ in range(num_flows)],
14     'dst_ip': ['10.0.0.' + str(np.random.randint(1, 20)) for _ in range(num_flows)],
15     'bytes_in': np.random.randint(50, 5000, num_flows),
16     'bytes_out': np.random.randint(50, 5000, num_flows),
17     'packets_in': np.random.randint(1, 50, num_flows),
18     'packets_out': np.random.randint(1, 50, num_flows),
19     'protocol': np.random.choice(['TCP', 'UDP', 'ICMP'], num_flows, p=[0.6, 0.3, 0.1])
20 })
21
22 # Introduce a few "anomalous" flows (e.g., very high bytes in/out)
23 df_flows_raw.loc[50:55, ['bytes_in', 'bytes_out']] = np.array([] * 6)
24 df_flows_raw.loc[120:125, ['bytes_in', 'bytes_out']] = np.array([] * 6)
25
26 # Prepare features for sequence modeling
27 df_flows_raw = pd.get_dummies(df_flows_raw, columns=['protocol'], prefix='protocol')
28 feature_cols = ['bytes_in', 'bytes_out', 'packets_in', 'packets_out', 'protocol_TCP', 'protocol_UDP', 'protocol_ICMP']
29 df_flows_raw['label'] = 0 # Assume all benign for now
30 df_flows_raw.loc[50:55, 'label'] = 1 # Mark our 'anomalies'
31 df_flows_raw.loc[120:125, 'label'] = 1
32
33 df_flows_raw = df_flows_raw.sort_values(by='timestamp')
34
35 # Group flows into time windows
36 window_size_seconds = 10
37 df_flows_raw['window_start'] = df_flows_raw['timestamp'].dt.floor(f'{window_size_seconds}S')
38
39 grouped_sequences = []
40 grouped_labels = []
41
42 for window_start, group in df_flows_raw.groupby('window_start'):
43     sequence_features = group[feature_cols].values
44     grouped_sequences.append(sequence_features)
45     # A window is anomalous if any flow within it is anomalous
46     grouped_labels.append(1 if group['label'].sum() > 0 else 0)
47
48 max_seq_len = max(len(s) for s in grouped_sequences)
49 print(f"Max sequence length in windows: {max_seq_len}")
50
51 # Pad sequences to max_seq_len
52 padded_sequences = []
53 for seq in grouped_sequences:
54     if len(seq) < max_seq_len:
55         padding_needed = max_seq_len - len(seq)
56         padded_seq = np.vstack([seq, np.zeros((padding_needed, seq.shape))])
57     else:
58         padded_seq = seq[:max_seq_len]
59     padded_sequences.append(padded_seq)
60
61 X_sequences = np.array(padded_sequences)
62 y_sequences = np.array(grouped_labels)
63
64 print(f"\nShape of X_sequences (batch, seq_len, num_features): {X_sequences.shape}")
65 print(f"Shape of y_sequences: {y_sequences.shape}")
66 print(f"Sequence Label distribution: {Counter(y_sequences)}")
67
68 # Split sequences for training/testing
69 X_train_seq, X_test_seq, y_train_seq, y_test_seq = train_test_split(
70     X_sequences, y_sequences, test_size=0.3, random_state=42, stratify=y_sequences
71 )

```

Listing 49: إنشاء تسلسلات سلاسل زمنية من تدفقات الشبكة باستخدام Pandas.

```

1  import torch
2  import torch.nn as nn
3  from torch.utils.data import TensorDataset, DataLoader
4
5  # Define model parameters based on our dummy data
6  input_dim = X_train_seq.shape # Number of features per time step
7  hidden_dim = 128
8  output_dim = 2 # Binary classification: Benign (0) / Anomaly (1)
9  num_layers = 2
10 batch_size = 32
11
12 class LSTMNet(nn.Module):
13     def __init__(self, input_dim, hidden_dim, output_dim, num_layers):
14         super(LSTMNet, self).__init__()
15         self.hidden_dim = hidden_dim
16         self.num_layers = num_layers
17         # LSTM layer: batch_first=True means input has batch dim first
18         self.lstm = nn.LSTM(input_dim, hidden_dim, num_layers, batch_first=True)
19         # Fully connected layer after LSTM
20         self.fc = nn.Linear(hidden_dim, output_dim)
21
22     def forward(self, x):
23         # Initialize hidden and cell states
24         h_0 = torch.zeros(self.num_layers, x.size(0), self.hidden_dim).to(x.device)
25         c_0 = torch.zeros(self.num_layers, x.size(0), self.hidden_dim).to(x.device)
26
27         # Pass input and initial states through LSTM
28         out, _ = self.lstm(x, (h_0, c_0))
29
30         # Take the output from the last time step for classification
31         out = self.fc(out[:, -1, :])
32         return out
33
34 # Instantiate the model
35 model_dl_nids = LSTMNet(input_dim, hidden_dim, output_dim, num_layers)
36 print(model_dl_nids)

```

Listing 50: بناء نموذج LSTM PyTorch بسيط لتصنيف التسلسل.

```

1  # Convert numpy arrays to PyTorch tensors
2  X_train_tensor = torch.tensor(X_train_seq, dtype=torch.float32)
3  y_train_tensor = torch.tensor(y_train_seq, dtype=torch.long)
4  X_test_tensor = torch.tensor(X_test_seq, dtype=torch.float32)
5  y_test_tensor = torch.tensor(y_test_seq, dtype=torch.long)
6
7  # Create TensorDatasets and DataLoaders
8  train_dataset = TensorDataset(X_train_tensor, y_train_tensor)
9  train_loader = DataLoader(train_dataset, batch_size=batch_size, shuffle=True)
10 test_dataset = TensorDataset(X_test_tensor, y_test_tensor)
11 test_loader = DataLoader(test_dataset, batch_size=batch_size, shuffle=False)
12
13 # Define loss function and optimizer
14 criterion = nn.CrossEntropyLoss()
15 optimizer = torch.optim.Adam(model_dl_nids.parameters(), lr=0.001)
16
17 # Move model to GPU if available
18 device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
19 model_dl_nids.to(device)
20 print(f"\nUsing device: {device}")
21
22 num_epochs = 10
23 print("\n--- Starting PyTorch LSTM Training ---")
24 for epoch in range(num_epochs):
25     model_dl_nids.train() # Set model to training mode
26     running_loss = 0.0
27     for batch_idx, (inputs, labels) in enumerate(train_loader):
28         inputs, labels = inputs.to(device), labels.to(device)
29
30         optimizer.zero_grad() # Clear gradients
31         outputs = model_dl_nids(inputs) # Forward pass
32         loss = criterion(outputs, labels) # Calculate loss
33         loss.backward() # Backward pass
34         optimizer.step() # Update weights
35         running_loss += loss.item() * inputs.size(0)
36
37     epoch_loss = running_loss / len(train_dataset)
38     print(f"Epoch [{epoch+1}/{num_epochs}], Loss: {epoch_loss:.4f}")
39
40 print("\nPyTorch LSTM Training complete.")

```

Listing 51: حلقة تدريب قياسية لنموذج LSTM. PyTorch

```

1  from sklearn.metrics import classification_report
2
3  model_dl_nids.eval() # Set model to evaluation mode
4  correct = 0
5  total = 0
6  all_preds = []
7  all_labels = []
8
9  with torch.no_grad(): # Disable gradient calculation during inference
10     for inputs, labels in test_loader:
11         inputs, labels = inputs.to(device), labels.to(device)
12         outputs = model_dl_nids(inputs)
13         _, predicted = torch.max(outputs.data, 1) # Get the class with max probability
14
15         total += labels.size(0)
16         correct += (predicted == labels).sum().item()
17
18         all_preds.extend(predicted.cpu().numpy())
19         all_labels.extend(labels.cpu().numpy())
20
21 accuracy = correct / total
22 print(f"\n--- Deep Learning NIDS Model Evaluation on Test Set ---")
23 print(f"Test Accuracy: {accuracy:.4f}")
24 print("\nClassification Report:\n", classification_report(all_labels, all_preds, target_names=['Benign', 'Anomaly']))

```

Listing 52: تقييم نموذج LSTM PyTorch المدرب على مجموعة الاختبار.

المختصرات والمصطلحات

APT التهديد المتقدم المستمر Advanced Persistent Threat: نوع من الهجمات السيبرانية cyberattack المتطورة والمطولة حيث ينشئ المتسلل وجوداً غير مكتشف داخل شبكة لسرقة البيانات الحساسة على مدى فترة طويلة.

Mechanism Attention آلية الانتباه Attention Mechanism: المكون الأساسي لنموذج المحولات Transformer model الذي يسمح له بتقدير أهمية نقاط البيانات المختلفة في تسلسل بشكل ديناميكي، مما يمكنه من التقاط العلاقات بين الأحداث بغض النظر عن المسافة بينها.

C2 القيادة والتحكم Command and Control: يشير إلى البنية التحتية وطرق الاتصال التي يستخدمها المهاجمون لإدارة الأنظمة المخترقة والتحكم فيها، مثل شبكات الروبوت botnets، ولسرقة البيانات.

CNN الشبكة العصبية الالتفافية Convolutional Neural Network: نوع من نماذج التعلم العميق deep learning يتفوق في تحديد الأنماط المحلية في البيانات. في مجال الأمن السيبراني cybersecurity، يمكن تطبيقه على بايتات الحزم الأولية raw packet bytes أو تسلسلات تدفقات الشبكة network flows لاكتشاف التوقعات الخبيثة.

CTU-13 مجموعة بيانات عامة من جامعة CTU تحتوي على حركة مرور شبكية مصنفة من إصابات حقيقية لشبكات الروبوت botnet، وتستخدم على نطاق واسع لتدريب وتقييم أنظمة كشف التسلل الشبكي Network Intrusion Detection Systems.

GRU وحدة البوابات المتكررة Gated Recurrent Unit: نوع من الشبكات العصبية المتكررة Recurrent Neural Network ببنية مبسطة مقارنةً بذاكرة المدى القصير والطويل LSTM. تستخدم آليات البوابات gating mechanisms للتعلم بفعالية من تسلسلات البيانات الطويلة.

LSTM ذاكرة المدى القصير والطويل Long Short-Term Memory: نوع متقدم من الشبكات العصبية المتكررة Recurrent Neural Network مصمم للتغلب على قيود الشبكات العصبية المتكررة القياسية standard RNNs باستخدام بوابات خاصة لتذكر أو نسيان المعلومات على مدى فترات طويلة، مما يجعله مثالياً لاكتشاف الهجمات ذات الحالة stateful attacks.

NetFlow بروتوكول شبكة network protocol يجمع ويلخص بيانات حركة مرور بروتوكول الإنترنت IP traffic أثناء تدفقها عبر جهاز توجيه router أو محول switch. يوفر بيانات وصفية metadata عالية المستوى حول الاتصالات بدلاً من المحتوى الكامل للحزم.

NIDS نظام كشف التسلل الشبكي **Network Intrusion Detection System**: نظام مصمم لمراقبة وتحليل حركة مرور الشبكة لاكتشاف الأنشطة المشبوهة أو انتهاكات السياسات أو الهجمات الصريحة.

Padding/Truncation الحشو/البتر **Padding/Truncation**: تقنية للمعالجة المسبقة للبيانات **data preprocessing** يتم فيها توحيد طول التسلسلات. يتم حشو التسلسلات الأقصر (عادةً بالأصفار)، ويتم بتر التسلسلات الأطول لتناسب حجم الإدخال الثابت الذي تتطلبه العديد من نماذج التعلم العميق **deep learning models**.

PCAP التقاط الحزم **Packet Capture**: تنسيق ملف يخزن البيانات الأولية للحزم الملتقطة من واجهة شبكة **network interface**. يوفر رؤية مفصلة ومنخفضة المستوى لحركة مرور الشبكة.

PyTorch إطار عمل مفتوح المصدر **open-source** لتعلم الآلة **machine learning** يستخدم لبناء وتدريب ونشر نماذج التعلم العميق **deep learning models**.

RNN الشبكة العصبية المتكررة **Recurrent Neural Network**: فئة من الشبكات العصبية **neural networks** مصممة لمعالجة البيانات المتسلسلة عن طريق الحفاظ على حالة داخلية أو ذاكرة تلتقط المعلومات من المدخلات السابقة في التسلسل.

Tensor الموتر **Tensor**: مصفوفة متعددة الأبعاد تعمل كهيكل بيانات أساسي لمكتبات التعلم العميق **deep learning** مثل باي تورش **PyTorch**. يتم استخدامه لتمثيل البيانات ومعلمات النموذج، مما يتيح إجراء حسابات فعالة على أجهزة مثل وحدات معالجة الرسومات **GPUs**.

Transformer المحول **Transformer**: بنية تعلم عميق **deep learning** قوية، نشأت في مجال معالجة اللغات الطبيعية **natural language processing**، تستخدم آلية الانتباه **attention mechanism** لالتقاط التبعية بعيدة المدى في البيانات المتسلسلة، مما يجعلها فعالة في تحديد الهجمات المنسقة المنتشرة عبر الزمن.

Gradient Vanishing تلاشي التدرج **Vanishing Gradient**: مشكلة شائعة في تدريب الشبكات العصبية العميقة **deep neural networks** حيث تصبح الإشارة المستخدمة لتحديث أوزان النموذج صغيرة جداً، مما يمنع النموذج من تعلم التبعية في تسلسلات البيانات الطويلة.

باب 6

التعلم العميق لتحليل البرمجيات الخبيثة: رؤى ثابتة وديناميكية

مقدمة

يواجه الأمن السيبراني الحديث تحدياً مستمراً: الظهور المتواصل لبرمجيات خبيثة جديدة وغير معروفة. تفشل أنظمة الكشف التقليدية القائمة على التوقعات في مواجهة هذه التهديدات الجديدة لأنها تعتمد على التعرف على الأنماط المعروفة. يستكشف هذا الفصل كيف يقدم التعلم العميق `deep learning` حلاً قوياً لهذه المشكلة. سنتناول نهجين أساسيين لتحليل البرمجيات الخبيثة: التحليل الثابت `static analysis`، الذي يفحص بنية الملف دون تنفيذه، والتحليل الديناميكي `dynamic analysis`، الذي يراقب سلوك الملف في بيئة خاضعة للرقابة. أهداف التعلم الرئيسية لهذا الفصل هي:

- فهم التحدي الأساسي المتمثل في تصنيف البرمجيات الخبيثة الجديدة وغير المعروفة.
- التمييز بين تقنيات التحليل الثابت والديناميكي، بما في ذلك نقاط القوة والضعف لكل منهما.
- تعلم كيفية تمثيل الملفات الثنائية للبرمجيات الخبيثة كصور، وتسلسلات، ورسوم بيانية لنماذج التعلم العميق.
- استكشاف كيفية تطبيق نماذج التعلم العميق مثل الشبكات العصبونية الالتفافية `Convolutional Neural Networks (CNNs)`، والشبكات العصبونية المتكررة `Recurrent Neural Networks (RNNs)`، والشبكات العصبونية البيانية `Graph Neural Networks (GNNs)` على تصنيف البرمجيات الخبيثة.

- بناء مُصنّف برمجيات خبيثة قائم على الصور بشكل عملي باستخدام `PyTorch` .
- بنهاية هذا الفصل، سيكون لديك فهم راسخ لكيفية قيام الذكاء الاصطناعي بتحليل البرمجيات الخبيثة وتحديد تلقائياً، مما يشكل مكوناً حاسماً في أي نظام دفاع ذكي.

1.6 المشكلة: تصنيف البرمجيات الخبيثة الجديدة وغير المعروفة

الهدف الأساسي لتحليل البرمجيات الخبيثة المعتمد على الذكاء الاصطناعي `AI-driven malware analysis` هو التصنيف التلقائي لمتغيرات البرمجيات الخبيثة الجديدة وغير المعروفة من خلال فحص بنيتها الداخلية وسلوكها وقت التشغيل. تتيح هذه القدرة لأنظمة الأمان تحديد التهديدات حتى في حالة عدم توفر توقيع محدد، وهو سيناريو شائع مع هجمات اليوم-صفر `zero-day attacks` أو البرمجيات الخبيثة متعددة الأشكال `polymorphic malware` التي تغير شفرتها باستمرار. الهدف النهائي هو تحديد عائلة البرمجية الخبيثة، مثل برامج الفدية `ransomware`، أو برامج التجسس `spyware`، أو برامج الجذور الخفية `rootkit`، أو تحديد قدراتها المحددة، مثل تسجيل ضغوطات المفاتيح `keylogging` أو تسريب البيانات `data exfiltration`. يعتمد هذا التصنيف على طريقتين متكاملتين: التحليل الثابت والديناميكي.

التحليل الثابت `Static analysis` يتضمن تحليل الشفرة القابلة للتنفيذ دون تشغيلها. يقوم المحقق بفحص بنية الملف، مثل ترويساته، والمكتبات المستوردة، والسلاسل النصية المضمنة، وتسلسل تعليمات لغة التجميع. هذه الطريقة آمنة لأن الشفرة التي يُحتمل أن تكون خبيثة لا يتم تنفيذها أبداً. ومع ذلك، يمكن للمهاجمين استخدام تقنيات التشويش `obfuscation`، مثل التغليف `packing` أو التشفير `encryption`، لإخفاء الحمولة الخبيثة، مما قد يجعل التحليل الثابت أقل فعالية [13].

التحليل الديناميكي `Dynamic analysis`، في المقابل، يتضمن مراقبة سلوك ملف مشبوه أثناء تنفيذه. تتم هذه العملية في بيئة خاضعة للرقابة ومعزولة تسمى بيئة الاختبار المعزولة `sandbox`، والتي عادة ما تكون جهازاً افتراضياً يمنع البرمجية الخبيثة من التأثير على النظام المضيف. يلتقط التحليل الديناميكي الإجراءات وقت التشغيل التي يصعب إخفاؤها، مثل تغييرات نظام الملفات، والاتصالات الشبكية بخوادم القيادة والتحكم، وإنشاء عمليات جديدة، واستدعاءات واجهة برمجة تطبيقات نظام التشغيل المحددة. وعلى الرغم من قوته، يمكن أن تكون هذه الطريقة أبطأ وقد تتجنبها البرمجيات الخبيثة التي تكتشف أنها تعمل داخل بيئة معزولة `sandbox` [?].

2.6 تقنيات التحليل الثابت

يستخدم التحليل الثابت الميزات الكامنة في الملف لاتخاذ قرار التصنيف. أثبتت نماذج التعلم العميق فعاليتها الاستثنائية في تعلم الأنماط المعقدة مباشرة من المحتوى الثنائي الخام أو المعالج.

1.2.6 التصنيف القائم على الصور باستخدام الشبكات العصبونية الالتفافية

أحد أكثر الأساليب ابتكاراً في التحليل الثابت هو تصور الملفات الثنائية للبرمجيات الخبيثة كصور. تحول هذه الطريقة ملفاً تنفيذياً خاماً، مثل ملف `Portable Executable (PE)` لنظام ويندوز أو `Executable and Linkable Format (ELF)` لنظام لينكس، إلى صورة بتدرج الرمادي. الفكرة هي أن ترتيب البايتات ورموز العمليات وأقسام البيانات داخل الملف الثنائي يخلق ملامح بصرية فريدة. غالباً ما تشترك البرمجيات الخبيثة من نفس العائلة في أنماط هيكلية متشابهة، مما ينتج عنه صور متشابهة بصرياً [58].

يمكننا استخدام شبكة عصبونية الالتفافية `Convolutional Neural Network (CNN)` لتعلم هذه الميزات البصرية تلقائياً. تعد الشبكات العصبونية الالتفافية فئة من الشبكات العصبونية العميقة عالية الفعالية في تحديد التسلسلات الهرمية المكانية والأنماط المحلية في البيانات الشبيهة بالشبكات، مما يجعلها مناسبة تماماً لمهام تصنيف الصور. يمكن تقسيم عملية تحويل ملف ثنائي إلى صورة بتدرج الرمادي إلى بضع خطوات:

(1) قراءة البايتات الخام: يُقرأ محتوى الملف التنفيذي بالكامل كتيار أحادي البعد من البايتات الخام. كل بايت هو قيمة عددية صحيحة بين 0 و 255.

(2) إعادة تشكيل التيار أحادي البعد إلى مصفوفة ثنائية الأبعاد: يتم إعادة تشكيل تسلسل البايتات أحادي البعد إلى مصفوفة ثنائية الأبعاد. غالباً ما تكون هذه المصفوفة مربعة (على سبيل المثال، 256x256 بكسل) أو بعرض ثابت. إذا كان تيار البايتات أصغر من حجم الصورة المستهدف، يتم ملء المساحة المتبقية بالأصفار (حشو). إذا كان أكبر، يمكن اقتطاع التيار.

(3) تطبيع قيم البايتات: تعمل قيم البايتات (0-255) كقيم شدة البكسل. عادةً ما يتم تطبيعها إلى نطاق النقطة العائمة من 0.0 إلى 0.1 عن طريق القسمة على 0.255، وهي ممارسة قياسية لإعداد المدخلات للشبكات العصبونية.

(4) إنشاء الصورة: المصفوفة ثنائية الأبعاد هي الآن صورة بتدرج الرمادي أحادية القناة.

(5) الإدخال إلى الشبكة العصبونية الالتفافية: يتم تغذية هذه الصورة إلى شبكة عصبونية التلافية 'CNN'، والتي تتعلم استخلاص الميزات المميزة لتصنيف الملف على أنه حميد أو خبيث [?].

2.2.6 التحليل القائم على التسلسل

بدلاً من تصور الملفات الثنائية، يمكننا التعامل مع محتوياتها كتسلسلات من المعلومات. يحافظ هذا النهج على الطبيعة المرتبة للشفرة والبيانات، وهو أمر حاسم غالباً لفهم وظيفتها. يمكننا تطبيق نماذج تعلم عميق متنوعة قائمة على التسلسل لتحليل هذه التيارات. تشمل الأنواع الشائعة من التسلسلات المستخرجة من الملفات الثنائية ما يلي:

- رموز العمليات 'Opcodes': هي تسلسلات من تعليمات الآلة مثل 'MOV'، 'JMP' أو 'CALL'. يمكن أن تشير أنماط معينة من رموز العمليات إلى أنشطة خبيثة، مثل فك حزمة خفية أو اعتراض وظائف النظام.

- استدعاءات واجهة برمجة التطبيقات 'API Calls': هي تسلسلات من الاستدعاءات لواجهة برمجية تطبيقات نظام التشغيل 'Application Programming Interface' (API)، مثل 'CreateRemoteThread' أو 'WriteProcessMemory' في ويندوز. تكشف هذه الاستدعاءات عن النوايا عالية المستوى للبرنامج وتفاعلاته مع نظام التشغيل.

- إن-جرامات البايت 'Byte N-grams': هي تسلسلات متداخلة من N بايتات متتالية (على سبيل المثال، 2-جرام أو 3-جرام) من الملف الثنائي الخام. تلتقط هذه الطريقة أنماط البايتات منخفضة المستوى دون الحاجة إلى تفكيك الشفرة.

بالنسبة لهذه التسلسلات، يمكننا استخدام نماذج مثل الشبكات العصبونية الالتفافية أحادية البعد '1D CNNs' للعثور على الأنماط المحلية، أو شبكات الذاكرة طويلة المدى القصيرة 'Long Short-Term Memory (LSTM)' لفهم التبعيات طويلة المدى، أو نماذج المحولات 'Transformer models' لالتقاط العلاقات السياقية المعقدة عبر التسلسل بأكمله.

3.2.6 التحليل القائم على الرسوم البيانية (مقدمة موجزة عن الشبكات العصبونية البيانية)

النهج الثالث هو تمثيل البنية الداخلية للبرنامج كرسوم بيانية، حيث تمثل العقد عناصر الشفرة وتمثل الحواف العلاقات بينها. اثنان من التمثيلات البيانية الشائعة هما:

• رسوم استدعاء الدوال البيانية `Function Call Graphs`: في هذا الرسم البياني، تكون العقد هي الدوال الفردية داخل الملف الثنائي، والحافة الموجهة من الدالة أ إلى الدالة ب تعني أن أ تستدعي ب. يكشف هذا الرسم البياني عن تدفق التحكم عالي المستوى للبرنامج والتبعيات بين الدوال.

• رسوم تدفق التحكم البيانية `Control Flow Graphs (CFGs)`: يمثل رسم تدفق التحكم البياني تدفق التنفيذ داخل دالة واحدة. العقد هي كُمل شفرة أساسية (تسلسلات من التعليمات بنقطة دخول واحدة ونقطة خروج واحدة)، وتُظهر الحواف مسارات التنفيذ الممكنة بين هذه الكُمل.

لتحليل هذه الرسوم البيانية، نستخدم الشبكات العصبونية البيانية `Graph Neural Net-works (GNNs)`. الشبكة العصبونية البيانية هي نوع من الشبكات العصبية مصممة للعمل مباشرة على البيانات الهيكلية بِنائياً. تتعلم ميزات لكل عقدة عن طريق تجميع المعلومات من جيرانها، مما يسمح للنموذج بفهم الخصائص الهيكلية والعلائقية للرسم البياني [80]. في تحليل البرمجيات الخبيثة، يمكن للشبكات العصبونية البيانية اكتشاف الأنماط الهيكلية المشبوهة، مثل تسلسلات استدعاء الدوال غير العادية أو تدفقات التحكم المعقدة والمشوشة التي تشير إلى نية خبيثة.

3.6 تقنيات التحليل الديناميكي

يلتقط التحليل الديناميكي السلوك الحقيقي للبرنامج عن طريق تنفيذه. هذا أمر ضروري لأن العديد من عائلات البرمجيات الخبيثة الحديثة تستخدم تقنيات مراوغة لا يمكن الكشف عنها إلا في وقت التشغيل.

1.3.6 استخراج الميزات السلوكية

لإجراء التحليل الديناميكي، يتم تنفيذ ملف مشبوه في بيئة معزولة `sandbox` (مثل `Cuckoo`، `Any.Run`، `Sandbox`). تقوم البيئة المعزولة بمراقبة وتسجيل جميع تفاعلات البرنامج مع نظام التشغيل والشبكة. تشمل الميزات الرئيسية التي يتم جمعها خلال هذه العملية ما يلي:

• إنشاء/إنهاء العمليات: ما هي العمليات الجديدة التي يبدأها البرنامج؟ يمكن أن يكشف فهم علاقات العمليات الأم-الابن عن حقن العمليات `process injection` أو تقنيات خبيثة أخرى.

- تغييرات نظام الملفات: أي ملفات يتم قراءتها أو كتابتها أو حذفها. على سبيل المثال، ستقوم برامج الفدية بتشغيل ملفات المستخدم، وسيقوم برنامج التنزيل `dropper` بإنشاء ملفات تنفيذية خبيثة جديدة.
- تعديلات السجل: التغييرات التي تطرأ على سجل ويندوز `Windows Registry`، والتي غالباً ما تستخدمها البرمجيات الخبيثة لضمان استمراريتها `persistence` (أي، ضمان تشغيلها مرة أخرى بعد إعادة تشغيل النظام).
- الاتصالات الشبكية: أي محاولة للاتصال بالإنترنت، بما في ذلك طلبات `DNS` حل أسماء النطاقات وحركة مرور `HTTP` التي يمكن أن تكون اتصالاً بخادم القيادة والتحكم `command-and-control (C2) server`.
- تسلسلات استدعاء واجهة برمجة التطبيقات: توفر القائمة المرتبة لاستدعاءات واجهة برمجة التطبيقات التي تم إجراؤها أثناء التنفيذ تتبعاً تفصيلياً لسلوك البرنامج.
- إنشاء كائن المزامنة `Mutex` : كائن المزامنة `Mutex` (اختصار لـ `mutual exclusion`) هو كائن تستخدمه البرامج لمنع تشغيل مثيلات متعددة في نفس الوقت. غالباً ما تنشئ البرمجيات الخبيثة كائنات مزامنة بأسماء محددة، والتي يمكن أن تكون مؤشراً قوياً على عائلة برمجيات خبيثة معينة.

2.3.6 نمذجة السلاسل الزمنية/التسلسل

يمكن التعامل مع الأحداث السلوكية التي تسجلها بيئة معزولة كسلسلة زمنية `time-series` أو تسلسل `sequence`. الترتيب الذي تحدث به الأحداث له أهمية حاسمة. على سبيل المثال، تسلسل استدعاءات واجهة برمجة التطبيقات مثل `LoadLibrary`، `CreateRemoteThread`، `WriteProcessMemory` و `ResumeThread` هو نمط كلاسيكي لتقنية خبيثة تسمى حقن العمليات `process injection`.

يمكننا استخدام نماذج التعلم العميق مثل `LSTMs`، ووحدات التكرار المسورة `Gated Recurrent Units (GRUs)`، أو المحولات `Transformers` لتحليل هذه التسلسلات. تم تصميم هذه النماذج للتعلم من البيانات المرتبة ويمكنها اكتشاف الأنماط السلوكية الشاذة أو تصنيف تدفقات العمل الخبيثة المعروفة بناءً على العلاقات الزمنية بين الأحداث.

3.3.6 التحليل الديناميكي القائم على الرسوم البيانية

تماماً كما هو الحال مع التحليل الثابت، يمكننا أيضاً استخدام الرسوم البيانية لتمثيل السلوك الديناميكي. يمكن بناء "رسم بياني سلوكي" `behavioral graph` من تتبع التنفيذ حيث تمثل العقد كيانات مثل العمليات والملفات ومقابس الشبكة ومفاتيح السجل. تمثل الحواف في هذا الرسم البياني التفاعلات بين هذه الكيانات (على سبيل المثال، عملية تكتب في ملف، والذي يقوم بعد ذلك بإجراء اتصال شبكي).

يساعد هذا التمثيل البياني في اكتشاف الأنماط الخبيثة المعقدة التي تتضمن مكونات متفاعلة متعددة. على سبيل المثال، يمكن أن يكشف تحليل تدفق التنفيذ والوصول إلى الموارد في الرسم البياني عن تهديدات متقدمة مثل تصعيد الامتيازات `privilege escalation`، حيث يكتسب المهاجم حقوق وصول ذات مستوى أعلى على النظام.

4.6 مجموعات البيانات المفتوحة لتحليل البرمجيات الخبيثة

لتدريب وتقييم نماذج الكشف عن البرمجيات الخبيثة، يعتمد الباحثون والممارسون على مجموعات بيانات كبيرة ومتاحة للجمهور. ومن أبرزها:

- EMBER (Elastic Malware Benchmark for Empowering Research):

مجموعة بيانات مستخدمة على نطاق واسع تحتوي على ميزات من أكثر من 2 مليون ملف `Windows Portable Executable (PE)` . توفر مجموعة موحدة من 2,351 ميزة ثابتة، مما يسهل مقارنة نماذج تعلم الآلة المختلفة دون الحاجة إلى معالجة الملفات الثنائية الخام [5].

- MalNet: مجموعة كبيرة من البرمجيات الخبيثة التي تمت معالجتها مسبقاً وتحويلها إلى صور بتدرج الرمادي، مصممة خصيصاً لتدريب واختبار نماذج التصنيف القائمة على الصور مثل الشبكات العصبونية الالتفافية `CNNs` [?].

- SOREL-20M: مجموعة بيانات ضخمة تحتوي على 20 مليون ملف `PE`، مع تسميات للعينات الحميدة والخبيثة وغير المسماة. تتضمن أيضاً مجموعة غنية من ميزات التحليل الثابت وهي مخصصة لأبحاث الكشف عن البرمجيات الخبيثة على نطاق واسع [31].

- مستودعات عينات البرمجيات الخبيثة العامة: للباحثين الذين يحتاجون إلى عينات برمجيات خبيثة خام، فإن مستودعات مثل VirusShare [1]، و MalwareBazaar [3]، والقسم المجتمعي من Hybrid Analysis [18] لا تقدر بثمن. يتطلب العمل مع هذه العينات بيئة معزولة آمنة لإنشاء سجلات تحليل ديناميكي أو إجراء استخراج ميزات مخصص.

5.6 مشروع عملي: التعلم العميق لتصنيف البرمجيات الخبيثة

في هذا المشروع، سنقوم بتنفيذ نموذج تعلم عميق باستخدام `PyTorch` لتصنيف الملفات إما كحميدة أو خبيثة. سنركز على نهج التحليل الثابت حيث يتم تحويل الملفات الثنائية للبرمجيات الخبيثة إلى صور وتصنيفها باستخدام شبكة عصبونية التلافيفية `CNN`.

1.5.6 الخطوة 1: المعالجة المسبقة للبيانات

خطوتنا الأولى هي تحويل الملفات الثنائية إلى صور بتدرج الرمادي. لهذا العرض التوضيحي، سنقوم بإنشاء ملفات ثنائية وهمية لتمثيل العينات الحميدة والخبيثة. في سيناريو واقعي، ستستخدم ملفات تنفيذية فعلية. سنقرأ كل ملف ثنائي، ونعيد تشكيل تيار البايتات الخاص به إلى مصفوفة ثنائية الأبعاد، ونخزن مسارات الملفات والتسميات (0 للحميدة، 1 للخبيثة) في إطار بيانات `pandas` `DataFrame`. يقوم الكود في القائمة 53 بهذه المعالجة المسبقة.

2.5.6 الخطوة 2: مجموعة بيانات PyTorch ومحمل البيانات

بعد ذلك، نحتاج إلى طريقة فعالة لتحميل بياناتنا للنموذج. نقوم بإنشاء فئة `Dataset` مخصصة من `PyTorch`، وهي طريقة قياسية للتعامل مع تحميل البيانات. ستدير هذه الفئة مصفوفات الصور الخاصة بنا وتسمياتها المقابلة. ثم نغلف مجموعة البيانات الخاصة بنا في مثيلات `DataLoader`، والتي ستغذي البيانات إلى نموذجنا في دفعات عشوائية أثناء التدريب. توضح القائمة 54 هذه العملية.

3.5.6 الخطوة 3: تعريف نموذج الشبكة العصبونية الالتفافية

الآن نحدد بنية الشبكة العصبونية الالتفافية `CNN`. نقوم بإنشاء نموذج بسيط باستخدام `nn.Module` من `PyTorch`. يحتوي على طبقة التلافيفية `Conv2d` لاستخراج الميزات، ودالة تنشيط `ReLU`، وطبقة تجميع قصوى `MaxPool2d` لتقليل الأبعاد المكانية، وطبقة خطية `Linear` نهائية لإنتاج ناتج التصنيف. الكود الخاص بهذا النموذج موجود في القائمة 55.

4.5.6 الخطوة 4: التدريب والتقييم

أخيراً، نقوم بتدريب نموذج الشبكة العصبونية الالتفافية `CNN`. نحدد دالة خسارة (`CrossEntropyLoss`) ومُحسِّن (`Adam`). ثم نقوم بالمرور عبر بيانات التدريب لعدد محدد من

العصور، محدثين أوزان النموذج في كل خطوة لتقليل الخسارة. بعد التدريب، نقوم بتقييم أداء النموذج على بيانات الاختبار غير المرئية باستخدام مقاييس التصنيف القياسية مثل الدقة `pre-cision`، والاستدعاء `recall`، ودرجة إف-1 `F1-score`، تحتوي القائمة 56 على حلقة التدريب والتقييم الكاملة.

يوضح هذا المشروع العملي العملية الكاملة لبناء نموذج تعلم عميق لتحليل البرمجيات الخبيثة الثابت. على الرغم من أننا استخدمنا بيانات وهمية، فإن نفس سير العمل ينطبق على مجموعات البيانات الواقعية، مما يوضح قوة وسهولة استخدام الذكاء الاصطناعي للدفاع السيبراني.

الخاتمة

أظهر هذا الفصل الإمكانيات التحويلية للتعلم العميق `deep learning` في تحليل وتصنيف البرمجيات الخبيثة. لقد استكشفنا الاختلافات الأساسية بين التحليل الثابت `static analysis` والتحليل الديناميكي `dynamic analysis`، مع إدراك أن كلاهما يقدم رؤى فريدة حول احتمالية كون البرنامج خبيثاً. من خلال تمثيل البرمجيات الخبيثة كصور أو تسلسلات أو رسوم بيانية، يمكننا استخدام بنى التعلم العميق القوية مثل الشبكات العصبونية الالتفافية `CNNs`، و شبكات الذاكرة طويلة المدى القصيرة `LSTMs`، والشبكات العصبونية البيانية `GNNs` لتحديد التهديدات تلقائياً، حتى تلك التي لم يسبق رؤيتها من قبل.

النقاط الرئيسية المستفادة هي أن الذكاء الاصطناعي `AI` يتيح استخراج الميزات تلقائياً من البيانات الثنائية المعقدة ويمكنه نمذجة كل من الخصائص الهيكلية والسلوكيات وقت التشغيل للبرنامج. أظهر المشروع العملي أن بناء مُصنّف برمجيات خبيثة قادر أمر ممكن باستخدام أطر التعلم العميق الحديثة. مع استمرار تطور البرمجيات الخبيثة في التعقيد، ستصبح أنظمة الدفاع الذكية هذه ضرورية بشكل متزايد لحماية البيئات الرقمية.

```

1  from PIL import Image
2  import numpy as np
3  import os
4  import pandas as pd
5  import random
6  from collections import Counter
7
8  # Function to convert binary data to a grayscale image (simplified for demo)
9  def bin_to_img_array(filepath, width=64):
10     try:
11         with open(filepath, 'rb') as f:
12             binary_data = f.read()
13             # Convert byte string to numpy array of uint8
14             img_array = np.frombuffer(binary_data, dtype=np.uint8)
15
16             # Pad/truncate to fit a width x width square image
17             target_size = width * width
18             if len(img_array) < target_size:
19                 padding_needed = target_size - len(img_array)
20                 img_array = np.pad(img_array, (0, padding_needed), 'constant', constant_values=0)
21             elif len(img_array) > target_size:
22                 img_array = img_array[:target_size]
23
24             # Reshape to 2D image
25             img_array = img_array.reshape((width, width))
26             return img_array # Return numpy array, not PIL Image directly
27
28     except Exception as e:
29         print(f"Error processing {filepath}: {e}")
30         return None
31
32 # Create dummy binary files for demonstration
33 # In a real project, you would use actual benign/malware samples.
34 os.makedirs('dummy_binaries', exist_ok=True)
35 benign_content = b"This is a legitimate program file content." * 10
36 malware_content = b"EVIL" * 100 + b"MALWARE_SIGNATURE_1234" + b"EVIL" * 50
37 benign_files = []
38 malware_files = []
39
40 for i in range(5): # 5 benign files
41     path = f'dummy_binaries/benign_{i}.bin'
42     with open(path, 'wb') as f:
43         f.write(benign_content[:random.randint(50, len(benign_content))]) # Vary size
44     benign_files.append((path, 0)) # label 0 for benign
45 for i in range(5): # 5 malware files
46     path = f'dummy_binaries/malware_{i}.bin'
47     with open(path, 'wb') as f:
48         f.write(malware_content[:random.randint(100, len(malware_content))]) # Vary size
49     malware_files.append((path, 1)) # label 1 for malware
50
51 all_files_info = benign_files + malware_files
52 random.shuffle(all_files_info) # Shuffle for train/test split
53
54 # Create a DataFrame with file paths and labels
55 df_malware_data = pd.DataFrame(all_files_info, columns=['filepath', 'label'])
56 print("DataFrame of dummy files:\n", df_malware_data)
57
58 # Convert binaries to image arrays
59 image_data = []
60 labels = []
61 for index, row in df_malware_data.iterrows():
62     img_arr = bin_to_img_array(row['filepath'], width=64)
63     if img_arr is not None:
64         image_data.append(img_arr)
65         labels.append(row['label'])
66
67 X_images = np.array(image_data)
68 y_labels = np.array(labels)
69
70 print(f"\nShape of image data (N, width, height): {X_images.shape}")
71 print(f"Labels shape: {y_labels.shape}")
72 print(f"Label distribution: {Counter(y_labels)}")

```

```

1  import torch
2  from torch.utils.data import Dataset, DataLoader
3  from sklearn.model_selection import train_test_split
4
5  # Convert numpy arrays to PyTorch tensors
6  X_tensor = torch.tensor(X_images, dtype=torch.float32).unsqueeze(1) / 255.0 # Add channel dim, normalize
7  y_tensor = torch.tensor(y_labels, dtype=torch.long)
8
9  # Split data into training and testing sets
10 X_train_img, X_test_img, y_train_img, y_test_img = train_test_split(
11     X_tensor, y_tensor, test_size=0.3, random_state=42, stratify=y_tensor
12 )
13
14 # Define a custom PyTorch Dataset
15 class MalwareImageDataset(Dataset):
16     def __init__(self, images, labels):
17         self.images = images
18         self.labels = labels
19
20     def __len__(self):
21         return len(self.images)
22
23     def __getitem__(self, idx):
24         return self.images[idx], self.labels[idx]
25
26 # Create dataset instances
27 train_dataset_img = MalwareImageDataset(X_train_img, y_train_img)
28 test_dataset_img = MalwareImageDataset(X_test_img, y_test_img)
29
30 # Create DataLoader instances
31 batch_size = 4 # Small batch size for demo
32 train_loader_img = DataLoader(train_dataset_img, batch_size=batch_size, shuffle=True)
33 test_loader_img = DataLoader(test_dataset_img, batch_size=batch_size, shuffle=False)
34
35 print(f"\nTraining images shape: {X_train_img.shape}")
36 print(f"Test images shape: {X_test_img.shape}")
37 print(f"Number of training batches: {len(train_loader_img)}")
38 print(f"Number of test batches: {len(test_loader_img)}")

```

Listing 54: بناء مجموعة بيانات `PyTorch Dataset` ومجلات بيانات
`DataLoaders` مخصصة لصور البرمجيات الخبيثة.

```

1 import torch.nn as nn
2 import torch.nn.functional as F
3
4 class MalwareCNN(nn.Module):
5     def __init__(self, img_width=64, num_classes=2):
6         super(MalwareCNN, self).__init__()
7         # Input channel is 1 for grayscale image
8         # (batch_size, 1, img_width, img_width)
9         self.conv1 = nn.Conv2d(in_channels=1, out_channels=16, kernel_size=3, padding=1) # Output: (batch_size, 16, img_width, img_width)
10        self.pool = nn.MaxPool2d(kernel_size=2, stride=2) # Output: (batch_size, 16, img_width/2, img_width/2)
11
12        # Calculate input features for the first fully connected layer
13        # After first conv and pool, image size becomes img_width/2 x img_width/2
14        # So, 16 * (img_width/2) * (img_width/2)
15        # For img_width=64, it's 16 * 32 * 32 = 16384
16        self.fc1_input_features = 16 * (img_width // 2) * (img_width // 2)
17        self.fc1 = nn.Linear(self.fc1_input_features, num_classes)
18
19    def forward(self, x):
20        # x shape: (batch_size, 1, img_width, img_width)
21        x = self.pool(F.relu(self.conv1(x)))
22        # Flatten the output for the fully connected layer
23        x = x.view(-1, self.fc1_input_features) # -1 infers batch size
24        x = self.fc1(x)
25        return x
26
27    # Instantiate the model with our image width
28    model_malware_cnn = MalwareCNN(img_width=X_train_img.shape, num_classes=len(torch.unique(y_train_img)))
29    print("\nMalware CNN Model Architecture:\n", model_malware_cnn)

```

Listing 55: تعريف شبكة عصبونية التلافية بسيطة لتصنيف صور البرمجيات الخبيثة.

```

1 import torch.optim as optim
2 from sklearn.metrics import classification_report
3 import shutil # For cleanup
4
5 # Move model to GPU if available
6 device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
7 model_malware_cnn.to(device)
8 print(f"\nUsing device for training: {device}")
9
10 # Define loss function and optimizer
11 criterion_malware = nn.CrossEntropyLoss()
12 optimizer_malware = optim.Adam(model_malware_cnn.parameters(), lr=0.001)
13
14 num_epochs_malware = 20 # More epochs needed for real data, but for demo
15 print("\n--- Starting PyTorch CNN Malware Classification Training ---")
16 for epoch in range(num_epochs_malware):
17     model_malware_cnn.train()
18     running_loss = 0.0
19     for batch_idx, (inputs, labels) in enumerate(train_loader_img):
20         inputs, labels = inputs.to(device), labels.to(device)
21         optimizer_malware.zero_grad()
22         outputs = model_malware_cnn(inputs)
23         loss = criterion_malware(outputs, labels)
24         loss.backward()
25         optimizer_malware.step()
26         running_loss += loss.item()
27
28     print(f"Epoch [{epoch+1}/{num_epochs_malware}], Loss: {running_loss / len(train_loader_img):.4f}")
29
30 print("\nPyTorch CNN Malware Classification Training complete.")
31
32 # Evaluation
33 model_malware_cnn.eval()
34 all_preds_malware = []
35 all_labels_malware = []
36
37 with torch.no_grad():
38     for inputs, labels in test_loader_img:
39         inputs, labels = inputs.to(device), labels.to(device)
40         outputs = model_malware_cnn(inputs)
41         _, predicted = torch.max(outputs.data, 1)
42         all_preds_malware.extend(predicted.cpu().numpy())
43         all_labels_malware.extend(labels.cpu().numpy())
44
45 print("\n--- Malware CNN Model Evaluation on Test Set ---")
46 print(classification_report(all_labels_malware, all_preds_malware, target_names=['Benign', 'Malware']))
47
48 # Clean up dummy binaries
49 if os.path.exists('dummy_binaries'):
50     shutil.rmtree('dummy_binaries')
51 print("\nCleaned up dummy_binaries folder.")

```

Listing 56: حلقة التدريب والتقييم لنموذج تصنيف البرمجيات الخبيثة باستخدام الشبكة العصبونية الالتفافية.

المختصرات والمصطلحات

AI الذكاء الاصطناعي Artificial Intelligence: مجال علوم الحاسوب الذي يركز على إنشاء آلات يمكنها أداء المهام التي تتطلب عادةً ذكاءاً بشرياً، مثل التعلم وحل المشكلات.

API واجهة برمجة التطبيقات Application Programming Interface: مجموعة من القواعد التي تسمح للبرامج بالتواصل مع بعضها البعض. غالباً ما يقوم تحليل البرمجيات الخبيثة بفحص استدعاءات واجهة برمجة التطبيقات API calls لفهم تفاعل البرنامج مع نظام التشغيل.

C2 خادم القيادة والتحكم Command-and-Control Server: خادم يتحكم فيه المهاجم ويستخدم لإرسال الأوامر إلى أجهزة الكمبيوتر المصابة بالبرمجيات الخبيثة واستقبال البيانات منها.

CFG الرسم البياني لتدفق التحكم Control Flow Graph: رسم بياني يمثل بصرياً جميع المسارات المحتملة التي يمكن أن يسلكها تنفيذ البرنامج داخل دالة واحدة.

CNN الشبكة العصبونية الالتفافية Convolutional Neural Network: نوع من نماذج التعلم العميق deep learning فعال للغاية في تحليل البيانات شبيهة بالشبكات، مثل الصور، عن طريق تعلم الأنماط المكانية.

Analysis Dynamic التحليل الديناميكي Dynamic Analysis: عملية تحليل برنامج عن طريق تنفيذه في بيئة آمنة وخاضعة للرقابة (صندوق رمل sandbox) لمراقبة سلوكه الفعلي.

ELF الصيغة التنفيذية وال قابلة للربط Executable and Linkable Format: صيغة ملفات شائعة للملفات التنفيذية والمكتبات المشتركة على أنظمة لينكس وأنظمة التشغيل الأخرى الشبيهة بيونكس.

EMBER معيار مرن للبرمجيات الخبيثة لتمكين البحث Elastic Malware Benchmark for Empowering Research: مجموعة بيانات عامة مستخدمة على نطاق واسع تحتوي على ميزات من ملايين ملفات ويندوز، وتستخدم لتدريب واختبار نماذج كشف البرمجيات الخبيثة malware detection.

Epoch العصر التدريبي Epoch: في تعلم الآلة machine learning، هو مرور كامل واحد عبر مجموعة بيانات التدريب بأكملها.

F1-score مقياس F1 F1-score: مقياس لتقييم دقة النموذج، يُحسب كتوسط توافقي للدقة precision والاستدعاء recall.

GNN الشبكة العصبونية للرسم البيانية Graph Neural Network: شبكة عصبونية neu-ral network متخصصة مصممة للعمل مباشرة على البيانات الهيكلية كرسوم بياني graph، وتتعلم من العلاقات بين العقد.

GRU وحدة التكرار المسورة Gated Recurrent Unit: نوع من الشبكات العصبونية المتكررة recurrent neural network، مشابه لـ LSTM، يستخدم لمعالجة البيانات المتسلسلة.

LSTM الذاكرة الطويلة قصيرة المدى Long Short-Term Memory: نوع خاص من الشبكات العصبونية المتكررة (RNN) Recurrent Neural Network قادر على تعلم الاعتماديات طويلة المدى في البيانات المتسلسلة، مما يجعله مناسباً لتحليل تسلسلات استدعاءات واجهة برمجة التطبيقات API calls أو رموز التشغيل opcodes.

Malware البرمجيات الخبيثة Malware: مصطلح عام لأي برنامج مصمم عمداً لإلحاق الضرر بجهاز كمبيوتر أو خادم أو عميل أو شبكة كمبيوتر.

Mutex الاستبعاد المتبادل Mutual Exclusion: كائن برمجي يستخدم لمنع تشغيل أكثر من نسخة واحدة من البرنامج في نفس الوقت. يمكن استخدام أسماء mutex محددة لتحديد عائلات البرمجيات الخبيثة.

Obfuscation التعمية أو التشويش Obfuscation: تقنية جعل الكود البرمجي صعب الفهم عمداً على البشر أو أدوات الأمان من أجل إخفاء غرضه الخبيث.

Opcode رمز التشغيل Operation Code: تعليمة في لغة الآلة machine language تحدد العملية التي يجب أن تنفذها وحدة المعالجة المركزية central processing unit (CPU) للكمبيوتر.

Packing التغليف أو الحزم Packing: طريقة لضغط أو تشفير ملف تنفيذي. غالباً ما تستخدم البرمجيات الخبيثة التغليف لإخفاء شفرتها الخبيثة عن أدوات التحليل الساكن static analysis.

PE الصيغة التنفيذية المحمولة Portable Executable: صيغة الملفات القياسية للملفات التنفيذية والمكتبات على أنظمة تشغيل ويندوز.

Persistence الاستمرارية أو الثبات Persistence: تقنية تستخدمها البرمجيات الخبيثة لضمان تشغيلها تلقائياً مرة أخرى بعد إعادة تشغيل نظام الكمبيوتر.

Polymorphic Malware البرمجيات الخبيثة متعددة الأشكال -Polymorphic Malware: برمجية خبيثة يمكنها تغيير شفرتها وهيكلها لتجنب الكشف من قبل برامج مكافحة الفيروسات القائمة على التوقيع signature-based antivirus software.

Precision الدقة Precision: مقياس أداء يقيس نسبة التحديدات الإيجابية التي كانت صحيحة بالفعل.

Injection Process حقن العمليات Process Injection: تقنية خبيثة تقوم فيها البرمجية الخبيثة بتشغيل شفرتها في مساحة الذاكرة لعملية شرعية لإخفاء أنشطتها.

PyTorch باي تورش PyTorch: إطار عمل مفتوح المصدر لتعلم الآلة-machine learning، يستخدم على نطاق واسع لتطوير وتدريب نماذج التعلم العميق-deep learning.

Ransomware برمجيات الفدية Ransomware: نوع من البرمجيات الخبيثة يقوم بتشفير ملفات الضحية ويطلب بدفع فدية مقابل مفتاح فك التشفير.

Recall الاستدعاء Recall: مقياس أداء يقيس نسبة الإيجابيات الفعلية التي تم تحديدها بشكل صحيح.

RNN الشبكة العصبونية المتكررة Recurrent Neural Network: نوع من الشبكات العصبونية neural network مصمم للتعامل مع البيانات المتسلسلة، حيث يتم تغذية المخرجات من الخطوة السابقة كمدخلات للخطوة الحالية.

Sandbox صندوق الرمل Sandbox: بيئة معزولة وخاضعة للرقابة (غالباً ما تكون آلة افتراضية virtual machine) تستخدم لتنفيذ وتحليل الملفات المشبوهة أو الخبيثة بأمان دون التأثير على النظام المضيف.

Spyware برمجيات التجسس Spyware: برمجية خبيثة تراقب سراً نشاط كمبيوتر المستخدم وتجمع معلومات شخصية دون موافقته.

Static Analysis التحليل الساكن Static Analysis: عملية تحليل شفرة الملف وهيكله لتحديد المحتوى الخبيث دون تشغيل البرنامج فعلياً.

Transformer المحول Transformer: بنية حديثة للتعلم العميق-deep learning تتفوق في فهم السياق والعلاقات في البيانات المتسلسلة، مثل تسلسلات استدعاءات واجهة برمجية التطبيقات API call sequences.

Attack Zero-Day هجوم يوم الصفر Zero-Day Attack: هجوم إلكتروني-cyber attack يحدث في نفس اليوم الذي يتم فيه اكتشاف نقطة ضعف في برنامج ما، قبل توفر تصحيح أو إصلاح للجمهور.

باب 7

الذكاء الاصطناعي لأمن شبكات المركبات: ناقل CAN وإيثرنت السيارات

مقدمة

تعتبر المركبات الحديثة شبكات معقدة على عجالات، تحتوي على العشرات من أجهزة الكمبيوتر المترابطة التي تدير كل شيء بدءًا من أداء المحرك إلى ميزات السلامة المتقدمة. يستكشف هذا الفصل المجال الحيوي لأمن الشبكات داخل المركبات، مع التركيز على كيف يوفر الذكاء الاصطناعي (AI) طبقة دفاع جديدة وقوية. سندرس معياري الشبكات السائد في المركبات، وهما ناقل منطقة التحكم Bus (CAN) Controller Area Network وإيثرنت السيارات Automotive Ethernet، وسنفهم نقاط ضعفهما الفريدة. الهدف التعليمي الأساسي هو فهم وتطبيق تقنيات تعلم الآلة machine learning لاكتشاف الهجمات السيبرانية والسلوك الشاذ داخل هذه الشبكات المتخصصة. سنغطي طرق استخراج الميزات feature extraction من بيانات قياس المركبات عن بعد، وتطبيق نماذج التعلم الخاضع للإشراف supervised learning وغير الخاضع للإشراف unsupervised learning، ونختتم بمشروع عملي لبناء نظام كامل للكشف عن التسلل القائم على الشذوذ anomaly-based intrusion detection system لحركة مرور ناقل CAN Bus.

1.7 المشكلة: تأمين الشبكات داخل المركبات

الهدف الرئيسي لأمن الشبكات داخل المركبات هو اكتشاف ومنع الهجمات السيبرانية والسلوكيات غير العادية التي يمكن أن تعرض سلامة المركبة ووظائفها للخطر. مع اعتماد المركبات بشكل متزايد على أنظمة مساعدة السائق المتقدمة Advanced Driver-Assistance Systems (ADAS) وميزات القيادة الذاتية، يزداد التأثير المحتمل لأي هجوم سيبراني ناجح بشكل كبير. يمكن للهجوم أن يتسبب في فقدان السيطرة على المركبة، أو سرقة البيانات الخاصة، أو إلحاق ضرر جسدي بالركاب [54]. لذلك، فإن حماية أنظمة الاتصالات الداخلية أمر ضروري.

1.1.7 ناقل (CAN Controller Area Network)

يعتبر ناقل منطقة التحكم Bus (CAN) معيار اتصالات قوي وموثوق للمركبات. يسمح لوحدة التحكم الدقيقة والأجهزة، المعروفة باسم وحدات التحكم الإلكترونية Electronic Control Units (ECUs)، بالتواصل مع بعضها البعض دون الحاجة إلى حاسوب مضيف مركزي [?]. في السيارة النموذجية، تتحكم وحدات ECUs في المكونات الحيوية للسلامة مثل المحرك والمكابح والوسائد الهوائية ونظام التوجيه المعزز. يعمل ناقل CAN Bus بسرعات منخفضة نسبياً وقد صُمم للموثوقية وليس للأمان. طبيعته القائمة على البث وعدم وجود مصادقة يجعله عرضة للهجمات حيث يمكن حقن الرسائل الخبيثة بسهولة في الشبكة.

2.1.7 إيثرنت السيارات (Automotive Ethernet)

إيثرنت السيارات Automotive Ethernet هي تقنية شبكات أحدث وعالية السرعة مصممة لتلبية متطلبات البيانات المكثفة للمركبات الحديثة. توفر نطاقاً ترددياً أعلى بكثير (يصل إلى 1 جيجابت في الثانية وما بعدها) مقارنة بناقل CAN Bus. هذا يجعلها مثالية للتطبيقات مثل بث الفيديو عالي الدقة من الكاميرات لأنظمة ADAS، وأنظمة المعلومات والترفيه، وتدفقات بيانات أجهزة الاستشعار الضخمة اللازمة لميزات القيادة الذاتية [?]. على الرغم من أنها تتضمن مبادئ شبكات أكثر حداثة، إلا أنها تقدم أيضاً سطح هجوم أكبر وأكثر تعقيداً يشبه شبكات IP التقليدية.

2.7 مصادر البيانات: بيانات القياس عن بعد داخل المركبة

لبناء نظام دفاع مدعوم بالذكاء الاصطناعي، يجب أن نفهم أولاً البيانات التي سيحللها. توفر بيانات القياس عن بعد داخل المركبة المعلومات الأولية اللازمة لنمذجة السلوك الطبيعي واكتشاف الانحرافات.

- رسائل ناقل CAN الأولية: هي حزم بيانات قصيرة ذات تنسيق ثابت يتم تبادلها بين وحدات ECUs. يعتبر التدفق المستمر لهذه الرسائل مصدر البيانات الأساسي لتحليل أمان CAN. تحتوي كل رسالة على معرف CAN ID (معرف لنوع الرسالة)، ورمز طول البيانات (Data Length Code (DLC)، وحمولة بيانات من 0 إلى 8 بايت.
- النقاط حزم إيثرنت السيارات: تشبه هذه الالتقاطات التقاطات حزم الشبكة القياسية (PCAPs) ولكنها خاصة بحركة المرور داخل المركبة. تلتقط الاتصالات ذات النطاق الترددي الأعلى وقد تحتوي على بروتوكولات مثل SOME/IP (البرمجيات الوسيطة القابلة للتطوير الموجهة نحو الخدمة عبر IP) أو DoIP (التشخيص عبر IP).
- سجلات من وحدات التحكم الإلكترونية (ECUs) وحدة التحكم الإلكترونية - Elec-tronic Control Unit (ECU) هي حاسوب صغير مدمج يتحكم في نظام واحد أو أكثر في المركبة، مثل وحدة التحكم في المحرك أو نظام الفرامل المانعة للانغلاق. يمكن أن تقدم السجلات من هذه الوحدات رؤى قيمة حول حالتها التشغيلية والأخطاء والعلامات المحتملة للاختراق.

3.7 التقنيات: تعلم الآلة والتعلم العميق لأمن المركبات

يقدم تعلم الآلة (ML) والتعلم العميق (DL) طرقاً قوية لتحليل الكم الهائل من البيانات الناتجة عن شبكات المركبات لاكتشاف التهديدات [?].

1.3.7 التعلم الخاضع للإشراف (Supervised Learning)

يمكن تدريب نماذج التعلم الخاضع للإشراف supervised learning للتعرف على أنماط هجوم محددة ومعروفة. يقوم المحلل أولاً بتصنيف مجموعة بيانات من حركة مرور الشبكة، ووضع علامات على الأجزاء إما "سليمة" أو مرتبطة بنوع هجوم معروف (مثل "فيضان DoS" أو "هجوم انتحال"). بعد ذلك، يتعلم مصنف، مثل Random Forest أو شبكة عصبونية مبنية باستخدام PyTorch،

التمييز بين هذه الفئات. هذا النهج فعال للغاية في اكتشاف التهديدات المعروفة، مثل قيام مهاجم بحقن إشارات سرعة خاطئة أو إعادة إرسال رسائل مشروعة لتعطيل وظائف المركبة.

2.3.7 الكشف عن الشذوذ غير الخاضع للإشراف (Anomaly Detection)

الكشف عن الشذوذ غير الخاضع للإشراف unsupervised anomaly detection ضروري لتحديد الهجمات الجديدة أو هجمات اليوم-صفر zero-day attacks التي ليس لها توقيع محدد مسبقاً. يتم تدريب خوارزميات مثل Isolation Forest [46] أو One-Class SVM حصرياً على بيانات تمثل التشغيل الطبيعي للمركبة. يتعلم النموذج خصائص هذا السلوك الطبيعي. أثناء التشغيل، يتم الإبلاغ عن أي حركة مرور تخرف بشكل كبير عن هذا الخط الأساسي المكتسب على أنها شذوذ. هذا أمر بالغ الأهمية لاكتشاف نواقل الهجوم الجديدة، مثل تسلسل غير عادي من الأوامر المرسلة إلى نظام الكبح.

3.3.7 نماذج التسلسل العميق (Deep Sequence Models)

غالباً ما يكون ترتيب وتوقيت الرسائل في شبكة المركبة بنفس أهمية محتوى الرسائل الفردية. تم تصميم نماذج التعلم العميق مثل شبكات الذاكرة طويلة قصيرة المدى Long Short-Term Memory (LSTM)، ووحدات البوابات المتكررة Gated Recurrent Units (GRUs)، أو المحولات Transformers لتحليل الأنماط الزمنية في تسلسل البيانات. باستخدام إطار عمل مثل PyTorch، يمكن تدريب هذه النماذج على تسلسلات من رسائل CAN أو تدفقات إيثرنت السيارات. تتفوق هذه النماذج في تحديد الهجمات المعقدة متعددة المراحل التي تتكشف بمرور الوقت، مثل قيام مهاجم أولاً باستكشاف الشبكة ثم شن هجوم حقن أوامر منسق عبر عدة وحدات ECUs.

4.3.7 اعتبارات النماذج خفيفة الوزن (TinyML)

يمثل التحدي الرئيسي في نشر نماذج الذكاء الاصطناعي هذه في مركبة حقيقية. غالباً ما تكون وحدات ECUs أجهزة محدودة الموارد ذات قدرة معالجة وذاكرة محدودة. يركز مجال TinyML على تطوير نماذج تعلم آلة عالية الكفاءة وخفيفة الوزن يمكن تشغيلها مباشرة على هذه الأجهزة. يتيح نشر نموذج صغير للكشف عن الشذوذ على وحدة ECU الكشف عن التهديدات في الوقت الفعلي وعلى الجهاز نفسه. يقلل هذا النهج من زمن الوصول عن طريق معالجة البيانات عند المصدر

ويقلل من الحاجة إلى نقل كميات كبيرة من بيانات القياس عن بعد إلى وحدة معالجة مركزية.

5.3.7 استخراج الميزات من رسائل CAN

قبل تطبيق أي نموذج تعلم آلة، يجب تحويل رسائل CAN الأولية إلى مجموعة ذات معنى من الميزات الرقمية. هذه العملية، المعروفة باسم هندسة الميزات feature engineering، هي خطوة حاسمة.

(1) تحليل رسالة CAN: لكل رسالة، استخرج مكوناتها الأساسية: معرف CAN ID (معرف رقمي لنوع الرسالة)، ورمز طول البيانات Data Length Code (DLC) (من 0 إلى 8)، وبيانات حمولة البيانات، وطابع زمني دقيق.

(2) تحليل التردد: احسب تردد الإرسال لكل معرف CAN ID فريد خلال نوافذ زمنية قصيرة (على سبيل المثال، 100 ميلي ثانية). قد يحاول المهاجم الذي يشن هجوم حجب الخدمة Denial-of-Service (DoS) إغراق الناقل بالرسائل، مما يتسبب في ارتفاع حاد وغير طبيعي في تردد معرف CAN ID معين.

(3) إنتروبيا حمولة البيانات: احسب إنتروبيا شانون Shannon entropy لحمولة البيانات. تقيس الإنتروبيا عشوائية أو عدم القدرة على التنبؤ بالبيانات. تحتوي معظم حمولات CAN على بيانات مستشعرات منظمة ولها إنتروبيا منخفضة. قد يشير الارتفاع المفاجئ في الإنتروبيا إلى بيانات مشفرة أو حقن بيانات عشوائية من هجوم.

(4) الميزات الزمنية: احسب الوقت بين الوصول (IAT) inter-arrival time، وهو الفاصل الزمني بين الرسائل المتتالية التي تشترك في نفس معرف CAN ID. غالباً ما تؤدي عمليات المركبة العادية إلى رسائل دورية ذات أوقات IATs متسقة. يمكن أن تشير الانحرافات الكبيرة إلى هجمات حقن الرسائل أو التشويش.

(5) ترميز One-Hot لمعرفة CAN: معرف CAN ID هو ميزة فئوية. لاستخدامه في معظم نماذج تعلم الآلة، يجب تحويله إلى تنسيق رقمي. يقوم ترميز One-hot encoding بإنشاء متجه ثنائي حيث يتوافق كل موضع مع معرف CAN ID فريد، مع وجود '1' للإشارة إلى وجود هذا المعرف.

(6) ملخصات إحصائية للحمولة: لكل معرف CAN ID، احسب المقاييس الإحصائية مثل المتوسط والانحراف المعياري والحد الأدنى والحد الأقصى لقيم حمولة بياناته خلال نافذة زمنية. يساعد هذا في التقاط التغيرات غير العادية في قراءات أجهزة الاستشعار أو قيم التحكم التي قد تشير إلى هجوم انتحال.

4.7 مشروع عملي: نظام كشف التسلل لناقل CAN

في هذا المشروع، سنقوم ببناء نظام كشف التسلل (Intrusion Detection System (IDS القائم على الشذوذ لحركة مرور ناقل CAN Bus باستخدام `pandas` لمعالجة البيانات و `scikit-learn` لتعلم الآلة. سيتعلم النظام أنماط حركة مرور CAN العادية ثم يستخدم تلك المعرفة لتحديد الرسائل التي تتخرف عن هذا السلوك الطبيعي.

1.4.7 إعداد البيانات

أولاً، نقوم بتحميل ومعالجة بيانات ناقل CAN الأولية. هدفنا هو تحويل رسائل السجل الأولية إلى مجموعة ميزات منظمة مناسبة لنموذج الكشف عن الشذوذ. يوضح كود Python في القائمة 57 هذه العملية عن طريق إنشاء مجموعة بيانات وهمية تحتوي على حركة مرور سليمة وخبيثة، ثم يستخرج الميزات الرئيسية مثل الوقت بين الوصول وقيم حمولة البيانات. كما يقوم أيضاً بترميز one-hot encoding لمعرفة CAN ID.

2.4.7 اختيار النموذج وتدريبه (الكشف عن الشذوذ)

بالنسبة لنظام كشف التسلل الخاص بنا، سنستخدم خوارزمية كشف الشذوذ غير الخاضعة للإشراف. يتيح هذا الاختيار للنظام اكتشاف الهجمات الجديدة دون معرفة مسبقة بها. نختار `IsolationForest` من `scikit-learn` وهو نموذج مناسب تماماً لتحديد القيم المتطرفة في البيانات. الخطوة الحاسمة هنا هي أننا ندرّب النموذج فقط على حركة مرور CAN العادية (السليمة). هذا يعلم النموذج تعريف "السلوك الطبيعي"، حتى يتمكن لاحقاً من تحديد أي نقاط بيانات لا تتوافق مع هذا التعريف. يوضح الكود في القائمة 58 كيفية بناء خط أنابيب يقوم أولاً بتوسيع نطاق البيانات ثم تدريب نموذج `IsolationForest`.

3.4.7 تحديد العتبة والتنبيه

بعد التدريب، يمكن للنموذج تعيين درجة شذوذ لكل رسالة CAN واردة. تشير الدرجة المنخفضة إلى احتمالية أكبر لكونها شاذة. يجب علينا تحديد عتبة لهذه الدرجة لنقرر متى نطلق تنبيهاً. نقوم بطريقة `predict` في `IsolationForest` بذلك تلقائياً، حيث تصنف النقاط على أنها طبيعية (1) أو شاذة (-1). يعد ضبط هذه العتبة توازناً دقيقاً. فالعتبة الحساسة جداً ستلتقط المزيد من الهجمات الحقيقية (استدعاء عالٍ high recall) ولكنها قد تولد أيضاً المزيد من الإنذارات الكاذبة (معدل إيجابيات كاذبة عالٍ high false positive rate)، مما قد

```

1 import pandas as pd
2 import numpy as np
3 from collections import Counter
4 import os
5
6 # Create dummy CAN log data (simulate benign and some attack traffic)
7 num_benign = 1000
8 num_attack = 50
9 can_ids_benign = np.random.choice([0x100, 0x101, 0x200, 0x201, 0x300], num_benign)
10 timestamps_benign = pd.to_datetime('2025-08-16 00:00:00') + pd.to_timedelta(np.cumsum(np.random.rand(num_benign) * 0.01), unit='s')
11 # Simulate normal data payloads (e.g., sensor readings)
12 data_benign = [format(np.random.randint(0, 256), '02x') * 8 for _ in range(num_benign)]
13
14 # Simulate an attack: high frequency for a specific ID, or unusual data payload
15 can_ids_attack = np.full(num_attack, 0x100) # Flood with ID 0x100
16 timestamps_attack = pd.to_datetime('2025-08-16 00:00:10') + pd.to_timedelta(np.cumsum(np.random.rand(num_attack) * 0.001), unit='s')
17 data_attack = [format(np.random.randint(0, 256), '02x') * 8 for _ in range(num_attack)] # Could be random or specific malicious payloads
18
19 # Combine benign and attack data
20 df_can = pd.DataFrame({
21     'timestamp': np.concatenate([timestamps_benign, timestamps_attack]),
22     'can_id': np.concatenate([can_ids_benign, can_ids_attack]),
23     'data': np.concatenate([data_benign, data_attack]),
24     'label': np.concatenate([np.zeros(num_benign), np.ones(num_attack)]) # 0: benign, 1: attack
25 })
26 df_can = df_can.sort_values(by='timestamp').reset_index(drop=True) # Sort by time
27
28 # Feature Engineering
29 df_can['can_id_hex'] = df_can['can_id'].apply(lambda x: hex(x))
30
31 # Calculate inter-arrival time for each CAN ID
32 df_can['prev_timestamp'] = df_can.groupby('can_id')['timestamp'].shift(1)
33 df_can['inter_arrival_time'] = (df_can['timestamp'] - df_can['prev_timestamp']).dt.total_seconds().fillna(0)
34
35 # Convert hex data payload to numerical features (e.g., integer representation of first 2 bytes)
36 df_can['data_value_int'] = df_can['data'].apply(lambda x: int(x[:2], 16) if isinstance(x, str) and len(x) >= 2 else 0)
37
38 # One-hot encode CAN ID (for a limited number of common IDs)
39 # For many IDs, frequency features might be better or an embedding
40 df_can = pd.get_dummies(df_can, columns=['can_id_hex'], prefix='id')
41
42 # Select features for the model
43 feature_cols = ['inter_arrival_time', 'data_value_int'] + [col for col in df_can.columns if col.startswith('id_')]
44 X_can = df_can[feature_cols]
45 y_can = df_can['label']
46
47 print("CAN Data Features Head:\n", X_can.head())
48 print("\nCAN Data Labels Distribution:\n", Counter(y_can))
49
50 # Handle potential NaN from `shift(1)` for inter_arrival_time
51 X_can['inter_arrival_time'] = X_can['inter_arrival_time'].fillna(X_can['inter_arrival_time'].mean())
52
53 # Split data - important: train only on benign data for unsupervised anomaly detection
54 X_train_benign = X_can[y_can == 0]
55 X_test = X_can # The test set can contain both benign and anomalous data
56 y_test = y_can
57
58 print(f"\nShape of training data (benign only): {X_train_benign.shape}")
59 print(f"Shape of test data (mixed): {X_test.shape}")

```

Listing 57: معالجة سجلات ناقل CAN باستخدام Pandas وهندسة الميزات. يتضمن الناتج المتوقع رأس إطار بيانات الميزات، وتوزيع التسميات (معظمها 0 مع القليل من 1)، وأشكال مجموعات بيانات التدريب والاختبار.

```

1 from sklearn.ensemble import IsolationForest
2 from sklearn.preprocessing import StandardScaler
3 from sklearn.pipeline import Pipeline
4
5 # It's good practice to scale features for IsolationForest too, though less critical than for SVMs
6 # Create a pipeline to first scale, then apply IsolationForest
7 model_iso_can = Pipeline([
8     ('scaler', StandardScaler()),
9     ('isolationforest', IsolationForest(random_state=42, contamination=0.05)) # contamination is an estimate of the proportion of
10 ])
11
12 # Train the IsolationForest model only on benign data
13 model_iso_can.fit(X_train_benign)
14 print("\nIsolationForest model for CAN Bus training complete on benign data.")
15
16 # Predict anomaly scores and labels on the entire test set (which may contain anomalies)
17 # decision_function gives a score, predict gives -1 for anomaly, 1 for normal
18 df_test_can = X_test.copy()
19 df_test_can['anomaly_score'] = model_iso_can.decision_function(X_test)
20 df_test_can['is_anomaly_pred'] = model_iso_can.predict(X_test) # -1 for anomaly, 1 for normal
21 df_test_can['true_label'] = y_test
22
23 print("\nTest data with anomaly scores and predictions:\n", df_test_can.head())
24 print("\nAnomaly prediction counts (-1 = anomaly, 1 = normal):\n", df_test_can['is_anomaly_pred'].value_counts())

```

Listing 58: تدريب نموذج IsolationForest للكشف عن الشذوذ. يتضمن الناتج المتوقع تأكيداً للتدريب، ورأس إطار بيانات الاختبار مع أعمدة جديدة لدرجات الشذوذ والتنبؤات، وعدد الحالات المتوقعة العادية مقابل الشاذة.

يكون معطلاً في نظام حرج للسلامة. على العكس من ذلك، فإن العتبة الأقل حساسية تقلل من الإنذارات الكاذبة ولكنها قد تفوت الهجمات الدقيقة. يقوم الكود في القائمة 59 بتقييم أداء النموذج باستخدام مقاييس التصنيف القياسية ويعرض توزيع درجات الشذوذ، مما يساعد المحلل على فهم قرارات النموذج.

خاتمة

يعد تأمين الشبكات داخل المركبات جانباً حيوياً من الأمن السيبراني للسيارات الحديثة. أظهر هذا الفصل كيف يوفر الذكاء الاصطناعي وتعلم الآلة أدوات متطورة لهذه المهمة. لقد رأينا أنه من خلال تحليل بيانات القياس عن بعد من ناقل CAN Bus وإثرت السيارات، يمكننا بناء أنظمة ذكية قادرة على اكتشاف التهديدات المعروفة وغير المعروفة. أوضح المشروع العملي سير العمل الكامل لإنشاء نظام كشف تسلسل قائم على الشذوذ، بدءاً من معالجة البيانات الأولية وهندسة الميزات إلى تدريب النموذج وتقييمه. النقطة الأساسية المستفادة هي أن طرق التعلم غير الخاضع للإشراف، مثل Isolation Forest، قوية بشكل خاص في هذا المجال لأنها يمكنها تحديد الهجمات الجديدة وهجمات اليوم-صفر zero-day attacks عن طريق تعلم نموذج للسلوك

```

1 from sklearn.metrics import classification_report, confusion_matrix
2 import matplotlib.pyplot as plt
3
4 # Convert IsolationForest's -1/1 prediction to 1/0 for standard metric calculation
5 # -1 (anomaly) -> 1, 1 (normal) -> 0
6 y_pred_binary = np.where(df_test_can['is_anomaly_pred'] == -1, 1, 0)
7 # True labels (1 for anomaly, 0 for normal) are already y_test
8
9 print("\n--- Anomaly Detection Performance on Test Set ---")
10 print(classification_report(y_test, y_pred_binary, target_names=['Normal', 'Anomaly']))
11
12 cm = confusion_matrix(y_test, y_pred_binary)
13 print("\nConfusion Matrix:\n", cm)
14 print(f"True Negatives (TN): {cm} (Normal correctly classified as Normal)")
15 print(f"False Positives (FP): {cm} (Normal incorrectly classified as Anomaly)")
16 print(f"False Negatives (FN): {cm} (Anomaly incorrectly classified as Normal)")
17 print(f"True Positives (TP): {cm} (Anomaly correctly classified as Anomaly)")
18
19 # Visualize anomaly scores
20 plt.figure(figsize=(10, 6))
21 plt.hist(df_test_can[df_test_can['true_label'] == 0]['anomaly_score'], bins=50, alpha=0.7, label='Normal Scores', color='green')
22 plt.hist(df_test_can[df_test_can['true_label'] == 1]['anomaly_score'], bins=50, alpha=0.7, label='Anomaly Scores', color='red')
23 plt.axvline(x=model_iso_can.named_steps['isolationforest'].threshold_, color='blue', linestyle='--', label=f'Model Threshold ({mo
24 plt.xlabel('Anomaly Score (lower is more anomalous)'); plt.ylabel('Frequency')
25 plt.title('Distribution of Anomaly Scores for CAN Bus Traffic')
26 plt.legend(); plt.grid(True);
27 # In a real application, you would save the figure, e.g., plt.savefig('anomaly_scores.png')
28 plt.show() # Displays the plot

```

Listing 59: تقييم أداء نموذج الكشف عن الشذوذ وتصور النتائج. يوضح الرسم البياني الناتج توزيعين لدرجات الشذوذ - أحدهما لحركة المرور العادية والآخر للشذوذ - مع خط يشير إلى عتبة قرار النموذج.

الطبيعي. مع ازدياد اتصال المركبات واستقلاليتها، ستكون آليات الدفاع هذه التي تعتمد على الذكاء الاصطناعي لا غنى عنها لضمان سلامتها وأمنها.

المختصرات والمصطلحات

أنظمة مساعدة السائق المتقدمة ADAS أنظمة في المركبة تستخدم الأتمتة automation والمستشعرات sensors لمساعدة السائق وتحسين السلامة، مثل الكبح التلقائي في حالات الطوارئ أو المساعدة في الحفاظ على المسار.

الذكاء الاصطناعي AI مجال في علوم الحاسوب computer science يركز على إنشاء أنظمة يمكنها أداء مهام تتطلب ذكاءً شبيهاً بالذكاء البشري، مثل التعلم من البيانات واتخاذ القرارات.

إيثرنت السيارات Automotive Ethernet تقنية شبكات عالية السرعة للمركبات مصممة للتطبيقات كثيفة البيانات مثل الكاميرات عالية الدقة، وأنظمة المعلومات والترفيه in-fotainment، وتدفقات بيانات المستشعرات sensor data الضخمة اللازمة للقيادة الذاتية autonomous driving.

ناقل شبكة منطقة التحكم CAN Bus شبكة اتصالات communication network قياسية تُستخدم في المركبات، وتسمح للمكونات الإلكترونية بتبادل المعلومات مع بعضها البعض دون الحاجة إلى حاسوب مركزي.

مُعرِّف CAN ID مُعرِّف رقمي ضمن رسالة CAN يحدد نوع البيانات التي يتم إرسالها، مثل قراءة السرعة أو تحديث حالة المحرك.

التعلم العميق DL أحد الحقول الفرعية للتعلم الآلي machine learning الذي يستخدم شبكات عصبونية neural networks معقدة ومتعددة الطبقات لتعلم الأنماط الدقيقة من كميات كبيرة من البيانات، وهو مفيد بشكل خاص لتحليل تسلسل رسائل الشبكة.

رمز طول البيانات DLC حقل في رسالة CAN يشير إلى حجم حمولة البيانات data pay-load، محددًا بالبايت (من 0 إلى 8).

التشخيص عبر بروتوكول الإنترنت DoIP بروتوكول protocol يمكن اتصالات تشخيص المركبة من أن تتم عبر شبكة قائمة على بروتوكول الإنترنت IP-based network مثل إيثرنت السيارات Automotive Ethernet.

هجوم حجب الخدمة DoS نوع من الهجمات السيبرانية cyberattack يهدف إلى جعل شبكة أو نظام غير متاح لمستخدميه المستهدفين عن طريق إغراقه بحركة مرور خبيثة malicious traffic.

وحدة التحكم الإلكترونية ECU حاسوب مدمج embedded computer صغير داخل المركبة يتحكم في نظام معين، مثل المحرك أو المكابح أو الوسائد الهوائية.

الإنتروبي Entropy في هذا السياق، هو مقياس للعشوائية أو عدم القدرة على التنبؤ ضمن حمولة البيانات data payload لرسالة شبكة. التغير المفاجئ في الإنتروبي يمكن أن يشير إلى هجوم محتمل.

هندسة الميزات Feature Engineering عملية اختيار وتحويل وإنشاء مدخلات رقمية ذات صلة (ميزات features) من البيانات الخام raw data ليتم استخدامها بواسطة نموذج تعلم آلي machine learning model.

وحدة البوابات المتكررة GRU نوع من بنية الشبكات العصبونية -neural network architecture مصمم لتحليل تسلسل البيانات، وغالباً ما يُستخدم كبديل أكثر كفاءة من الناحية الحسابية لشبكات الذاكرة طويلة قصيرة المدى LSTM.

الزمن الفاصل بين الوصول IAT التأخير الزمني المقاس بين رسالتين متتاليتين تشتركان في نفس المُعرِّف (على سبيل المثال، نفس مُعرِّف CAN ID). التغييرات الكبيرة في هذا الزمن يمكن أن تشير إلى هجمات حقن الرسائل message injection أو التشويش jamming.

نظام كشف التسلل IDS نظام أمني مصمم لمراقبة حركة مرور الشبكة network traffic بحثاً عن أنشطة خبيثة malicious activities أو انتهاكات للسياسات، ولإصدار تنبيهات عند اكتشاف مثل هذه الأنشطة.

بروتوكول الإنترنت IP المجموعة الأساسية من القواعد لتوجيه حزم البيانات data packets عبر الشبكات، ويُستخدم في كل من أنظمة تكنولوجيا المعلومات التقليدية وإيثرنت السيارات Automotive Ethernet الحديث.

غابة العزل Isolation Forest خوارزمية تعلم آلي غير خاضعة للإشراف unsupervised machine learning فعالة بشكل خاص في تحديد الحالات الشاذة anomalies (القيم المتطرفة outliers) في البيانات عن طريق تقسيم البيانات عشوائياً لعزل النقاط الشاذة.

الذاكرة طويلة قصيرة المدى LSTM نوع متقدم من الشبكات العصبونية المتكررة recurrent neural network (RNN) قادر على تعلم الأنماط في تسلسلات طويلة من البيانات، مما يجعله مثالياً لتحديد الهجمات المعقدة ومتعددة المراحل التي تتكشف بمرور الوقت.

التعلم الآلي ML فرع من الذكاء الاصطناعي AI حيث يتم تدريب الخوارزميات algorithms على البيانات لإيجاد الأنماط أو عمل تنبؤات دون أن تكون مبرمجة بشكل صريح لمهمة معينة.

الترميز الأحادي الساخن One-hot encoding تقنية تُستخدم لتحويل البيانات الفئوية cat-egorical data، مثل مُعرّفات CAN IDs، إلى صيغة متجه ثنائي binary vector رقمي مناسب للاستخدام في نماذج التعلم الآلي machine learning models.

التقاط الحزم PCAP ملف يحتوي على حزم البيانات data packets التي تم التقاطها من شبكة حاسوبية. يُستخدم بشكل شائع لتحليل الشبكات network analysis والتحقيقات الأمنية.

بروتوكول SOME/IP بروتوكول اتصالات communication protocol يُستخدم مع إيثرنت السيارات Automotive Ethernet ويسمح للخدمات والتطبيقات المختلفة داخل المركبة بالتواصل مع بعضها البعض.

التعلم الخاضع للإشراف Supervised Learning نوع من التعلم الآلي machine learning حيث يتم تدريب نموذج على مجموعة بيانات dataset تحتوي على كل من بيانات الإدخال input data ومسميات المخرجات output labels الصحيحة المقابلة لها، بهدف تعلم التنبؤ بالمخرجات لبيانات جديدة لم يسبق رؤيتها.

التعلم الآلي المصغر TinyML مجال من مجالات التعلم الآلي machine learning يركز على تطوير ونشر نماذج ذكاء اصطناعي AI models عالية الكفاءة وخفيفة الوزن يمكن تشغيلها على أجهزة محدودة الموارد resource-constrained hardware، مثل وحدة التحكم الإلكترونية ECU في المركبة.

كشف الحالات الشاذة غير الخاضع للإشراف Unsupervised Anomaly Detection نهج تعلم آلي machine learning حيث يتم تدريب النموذج فقط على بيانات تمثل السلوك الطبيعي. يتعلم النموذج خصائص هذه الحالة الطبيعية ثم يحدد أي بيانات تنحرف عنها كحالة شاذة anomaly أو تهديد محتمل.

هجوم اليوم صفر Zero-day attack هجوم سيبراني cyberattack يستغل ثغرة أمنية security vulnerability غير معروفة سابقاً، مما يعني عدم وجود دفاع أو توقيع signature محدد للهجوم وقت اكتشافه.

باب 8

الذكاء الاصطناعي لأمن المستشعرات وسلامة البيانات في الأنظمة الفيزيائية السيبرانية

مقدمة

تُعد الأنظمة الفيزيائية السيبرانية (CPS) Cyber-Physical Systems جوهر التكنولوجيا الحديثة، من السيارات ذاتية القيادة إلى أنظمة التحكم الصناعية. تعتمد هذه الأنظمة على المستشعرات لفهم العالم المادي والتفاعل معه. لذا، فإن سلامة بيانات المستشعرات ضرورية لتشغيلها الآمن والموثوق. إذا تمكن مهاجم من التلاعب بهذه البيانات، فقد تكون العواقب كارثية، مما يؤدي إلى قرارات غير صحيحة، أو فشل النظام، أو ضرر مادي. يستكشف هذا الفصل كيف يمكننا استخدام الذكاء الاصطناعي (AI) Artificial Intelligence لحماية هذه المستشعرات الحيوية. سندرس الهجمات الشائعة، ونناقش مصادر البيانات والتقنيات ذات الصلة، ونبني أداة عملية لمراقبة سلامة المستشعرات. تتمثل أهداف التعلم الرئيسية في فهم التحديات الأمنية الفريدة في الأنظمة الفيزيائية السيبرانية، وتعلم كيف يمكن للذكاء الاصطناعي اكتشاف شذوذ المستشعرات، وتطبيق هذه المفاهيم على مشكلة عملية للكشف عن انتحال إشارات نظام تحديد المواقع العالمي GPS spoofing.

1.8 المشكلة: حماية المستشعرات في الأنظمة الفيزيائية السيبرانية

الهدف الرئيسي هو كشف ومواجهة الهجمات ضد المستشعرات في الأنظمة المستقلة مثل الطائرات بدون طيار والسيارات ذاتية القيادة، بالإضافة إلى الأنظمة الفيزيائية السيبرانية -Cyber Physical Systems (CPS) الأخرى. في هذه البيئات، تعد بيانات المستشعرات الموثوقة حاسمة للسلامة. يمكن أن يؤدي الهجوم الناجح الذي يعرض بيانات المستشعرات للخطر إلى نتائج خطيرة.

1.1.8 الأنظمة الفيزيائية السيبرانية

النظام الفيزيائي السيبراني (Cyber-Physical System (CPS)) هو نظام يجمع بين الخوارزميات الحاسوبية والمكونات المادية [45]. تراقب هذه الأنظمة العمليات المادية وتتحكم فيها. وهي تستخدم حلقات التغذية الراجعة feedback loops حيث تؤثر الإجراءات المادية على العمليات الحاسوبية، وتؤثر العمليات الحاسوبية بدورها على الإجراءات المادية. تشمل الأمثلة الشائعة للأنظمة الفيزيائية السيبرانية الشبكات الذكية، والأنظمة الروبوتية، وأنظمة التحكم الصناعية (ICS)، والأجهزة الطبية، والمركبات المستقلة. إن الترابط الوثيق بين العالمين الرقمي والمادي يجعلها هدفاً فريداً للمهاجمين.

2.1.8 الهجمات الشائعة

يمكن للمهاجمين استهداف مستشعرات الأنظمة الفيزيائية السيبرانية بعدة طرق لتعطيل تشغيلها. اثنتان من أكثر الطرق شيوعاً هما انتحال إشارات المستشعرات وحقن البيانات.

انتحال إشارات المستشعرات انتحال إشارات المستشعرات (Sensor spoofing) هو هجوم يقوم فيه الخادم بتغذية بيانات خاطئة أو مُعدّلة إلى مستشعر لتضليل النظام. يمكن القيام بذلك عن طريق توليد إشارات مزيفة مادياً أو عن طريق تغيير تدفق البيانات رقمياً. على سبيل المثال، يمكن لمهاجم إرسال إشارات GPS مزيفة لجعل سيارة ذاتية القيادة تعتقد أنها في موقع مختلف [?]. مثال آخر هو التلاعب ببيانات LIDAR إما لإنشاء عقبات "وهمية" أو إخفاء عقبات حقيقية عن نظام الإدراك في المركبة.

حقن البيانات يتضمن حقن البيانات (Data injection) إدخال بيانات ضارة في تدفق بيانات النظام. غالباً ما يستغل هذا الهجوم الواجهات المشروعة أو المكونات التي تم اختراقها مسبقاً للتسبب في سلوك غير صحيح أو فشل النظام. قد يكون هذا بسيطاً مثل حقن قراءات

درجة حرارة خاطئة في وحدة تحكم صناعية للتسبب في ارتفاع درجة الحرارة، أو معقداً مثل حقن الأوامر command injection للسيطرة على جهاز.

2.8 مصادر البيانات: بيانات المستشعرات

تستخدم الأنظمة الفيزيائية السيبرانية مجموعة واسعة من المستشعرات لإدراك بيئتها. تعمل البيانات من هذه المستشعرات، سواء في شكلها الخام أو المعالج، كأساس للكشف عن الهجمات.

1.2.8 بيانات المستشعرات المختلفة (خام ومعالجة)

توفر المستشعرات المختلفة تدفقات بيانات فريدة تعتبر حاسمة لتشغيل النظام.

- مقياس التسارع والجيروسكوب (من وحدة القياس بالقصور الذاتي): توفر هذه المستشعرات بيانات عن التسارع الخطي والسرعة الزاوية، وهما أمران ضروريان لتقدير حركة النظام واتجاهه.
- ليدار (الكشف عن الضوء وتحديد المدى): تولد مستشعرات LIDAR سحابات نقطية ثلاثية الأبعاد للبيئة المحيطة. تُستخدم هذه البيانات في رسم الخرائط، واكتشاف الأجسام، والملاحة. غالباً ما تكون بيانات LIDAR الخام كبيرة جداً.
- الرادار: يكتشف الرادار Radar الأجسام وسرعتها باستخدام موجات الراديو. وهو أقل تأثراً بالظروف الجوية مثل المطر أو الضباب مقارنة بالليدار والكاميرات، مما يجعله مستشعراً موثوقاً لوظائف مثل نظام تثبيت السرعة التكيفي.
- الكاميرا: توفر الكاميرات بيانات صور وفيديو غنية. تُستخدم هذه البيانات للتعرف على الأشياء، واكتشاف المسارات، وقراءة إشارات المرور.
- مستشعرات درجة الحرارة والضغط والتدفق: هذه شائعة في الأنظمة الفيزيائية السيبرانية الصناعية والبيئية لمراقبة الظروف المادية وضمان معايير التشغيل الآمنة.

2.2.8 سجلات NMEA GPS

تنسيق NMEA (National Marine Electronics Association) هو معيار للاتصال بين الأجهزة الإلكترونية البحرية، ولكنه تم اعتماده على نطاق واسع من قبل أجهزة استقبال GPS. تحتوي سجلات NMEA GPS على جمل تُخرج معلومات الموقع (خط العرض وخط

الطول)، والسرعة، والوقت. تهدف هجمات انتحال إشارات GPS إلى تغيير هذه السجلات للإبلاغ عن موقع أو وقت خاطئ.

3.2.8 قراءات وحدة القياس بالقصور الذاتي (IMU)

وحدة القياس بالقصور الذاتي (Inertial Measurement Unit (IMU)) هي جهاز إلكتروني يقيس ويبلغ عن القوة النوعية للجسم، والمعدل الزاوي، وأحياناً مجاله المغناطيسي. يستخدم مقاييس التسارع، والجيروسكوبات، ومقاييس المغناطيسية للقيام بذلك. تعتبر وحدات IMU حاسمة لتقنية تسمى الملاحة التقديرية dead reckoning، والتي تحسب الموقع الحالي للنظام باستخدام موقع محدد مسبقاً وتقديمه بناءً على السرعات المعروفة أو المقدرة بمرور الوقت. هذا يجعل وحدات IMU ضرورية للملاحة القوية، خاصة عندما تكون إشارات GPS غير متوفرة أو غير موثوقة.

4.2.8 آثار الهجوم المحاكاة

نظراً لأن بيانات الهجوم في العالم الحقيقي على الأنظمة الفيزيائية السيبرانية غالباً ما تكون نادرة ويصعب الحصول عليها، يستخدم الباحثون بشكل متكرر البيانات المولدة اصطناعياً. تمثل آثار الهجوم المحاكاة (simulated attack traces) هذه هجمات انتحال إشارات المستشعرات أو حقن البيانات الشائعة. وهي ذات قيمة كبيرة لتدريب واختبار نماذج كشف الشذوذ القائمة على الذكاء الاصطناعي في بيئة خاضعة للرقابة.

3.8 مجموعات البيانات المفتوحة لأمن الأنظمة الفيزيائية السيبرانية

تتوفر العديد من مجموعات البيانات العامة واسعة النطاق للبحث في الأنظمة المستقلة وأمن الأنظمة الفيزيائية السيبرانية. توفر مجموعات البيانات هذه بيانات مستشعرات متزامنة ومتعددة الوسائط ومثالية لتطوير واختبار خوارزميات الأمان.

- مجموعة بيانات RobotCar: Oxford مجموعة بيانات كبيرة من سيارة تم قيادتها في أكسفورد، المملكة المتحدة، على مدار عام كامل في العديد من الظروف الجوية المختلفة. وهي تتضمن بيانات متزامنة من كاميرات ستيريو، و LIDAR، و GPS، و IMU، مما يجعلها ممتازة لأبحاث دمج المستشعرات [49].

- مجموعة بيانات KITTI مجموعة شائعة من بيانات المقارنة المعيارية لمهام الرؤية الحاسوبية في القيادة الذاتية. وهي تتضمن صور ستيريو، وتدفق بصري، واكتشافات كائنات ثلاثية

الأبعاد، والمزيد. على الرغم من تركيزها على الإدراك، إلا أن بيانات المستشعرات الغنية بها مفيدة للتحليل الأمني [30].

- **مجموعة بيانات NuScenes:** مجموعة بيانات قيادة ذاتية واسعة النطاق من بوسطن وسنغافورة. تتميز بصناديق محيطة ثلاثية الأبعاد للكائنات وتوفر بيانات من مجموعة كاملة من المستشعرات: 6 كاميرات، 5 رادارات، LIDAR 1، GPS، و IMU [11].

- **مجموعة بيانات Spoofing: GPS A&M Texas:** مجموعة بيانات تم إنشاؤها خصيصاً للبحث في الكشف عن انتحال إشارات GPS. تحتوي على بيانات GPS حقيقية تعرضت لهجمات انتحال محقونة، مما يوفر حقيقة ميدانية واضحة لتقييم خوارزميات الكشف [?].

- **مجموعة بيانات Water (Secure SWaT):** مجموعة بيانات من منصة اختبار لنظام تحكم صناعي في العالم الحقيقي. تتضمن حركة مرور الشبكة وبيانات العمليات المادية من كل من التشغيل العادي وسيناريوهات الهجوم المختلفة، مما يجعلها مورداً قيماً لأبحاث أمن الأنظمة الفيزيائية السيبرانية الصناعية [51].

4.8 التقنيات: التعلم الآلي والتعلم العميق لسلامة بيانات المستشعرات

يقدم التعلم الآلي ((Machine learning (ML)) والتعلم العميق ((deep learning (DL)) أدوات قوية لضمان سلامة بيانات المستشعرات في الأنظمة الفيزيائية السيبرانية. يمكن لهذه التقنيات تعلم العلاقات المعقدة بين المستشعرات المختلفة وتحديد الانحرافات الدقيقة التي تشير إلى وجود هجوم.

1.4.8 كشف الشذوذ بدمج المستشعرات

تجمع هذه الطريقة البيانات من مستشعرات متعددة ومتنوعة (على سبيل المثال، GPS، و IMU، والكاميرا) لإنشاء فهم أكثر قوة واكتمالاً لحالة النظام. يتم الكشف عن الهجمات من خلال إيجاد التناقضات بين قراءات المستشعرات التي يجب أن ترتبط عادةً. على سبيل المثال، إذا أبلغ نظام GPS للركبة عن سرعة عالية مفاجئة ولكن لم تظهر وحدة IMU الخاصة بها أي تسارع مقابل، فإن هذا الاختلاف يشير بقوة إلى هجوم على نظام GPS أو عطل فيه.

مرشحات كالمان **مرشح كالمان (Kalman Filter)** هو خوارزمية رياضية قوية للتقدير [36]. يأخذ سلسلة من القياسات التي يتم ملاحظتها بمرور الوقت، والتي تحتوي على ضوضاء وأخطاء

أخرى، وينتج تقديرات للمتغيرات غير المعروفة تكون أكثر دقة من تلك الناتجة عن قياس واحد بمفرده. يحقق ذلك من خلال الجمع بين التنبؤات من نموذج النظام والقياسات الفعلية، مع تصحيح نفسه باستمرار.

معزز بأجهزة التشفير التلقائي أو التعلم العميق بينما يمكن لمرشح كالمان التعامل مع الضوضاء وتقديم تقدير أساسي، يمكن لنماذج الذكاء الاصطناعي اكتشاف الحالات الشاذة الأكثر تعقيداً. بعد الدمج الأولي، يمكن لنموذج ذكاء اصطناعي مثل جهاز التشفير التلقائي Autoencoder أو LSTM تعلم الحالة "الطبيعية" للمستشعر المدمج. يمكنه بعد ذلك اكتشاف الانحرافات الدقيقة التي قد لا يتم اكتشافها عن طريق الفحوصات البسيطة القائمة على القواعد.

2.4.8 الدمج العميق متعدد الوسائط

يستخدم هذا النهج بنية تعلم عميق deep learning واحدة لمعالجة ودمج البيانات مباشرة من أنواع مختلفة من المستشعرات، مثل الصور، والسحب النقطية، والبيانات الرقمية للسلاسل الزمنية. وهذا يسمح للنموذج بتعلم الارتباطات المعقدة وغير الخطية بين تدفقات البيانات المتنوعة [?]. غالباً ما تستخدم هذه التقنية طبقات متخصصة لكل نوع من أنواع البيانات، مثل الشبكات العصبية التلافيفية (CNNs) للصور وشبكات الذاكرة طويلة قصيرة المدى (LSTM) لبيانات السلاسل الزمنية من وحدة IMU. ثم يتم دمج المخرجات من هذه الطبقات المتخصصة (دمجها)، غالباً باستخدام آليات التسلسل concatenation أو الانتباه attention mechanisms نماذج Transformer. ثم يتم تغذية هذا التمثيل الفردي والشامل إلى طبقة نهائية للتصنيف أو كشف الشذوذ.

3.4.8 كشف القيم المتطرفة (الأساليب الإحصائية)

تركز طرق كشف القيم المتطرفة Outlier detection على تحديد قراءات المستشعرات الفردية المختلفة إحصائياً عن بقية البيانات. هذه مفيدة لالتقاط الأحداث المفاجئة وغير الطبيعية.

مسافة ماها لانوبيس مسافة ماها لانوبيس (Mahalanobis distance) هي مقياس إحصائي للمسافة بين نقطة وتوزيع نقاط [50]. على عكس مسافة إقليدس البسيطة، فإنها تأخذ في الاعتبار الارتباطات بين الميزات في البيانات. هذا يجعلها فعالة جداً في العثور على القيم المتطرفة في البيانات متعددة الأبعاد. تعتبر نقطة البيانات ذات مسافة ماها لانوبيس العالية من مركز توزيع البيانات العادي قيمة متطرفة.

تحليل المكونات الرئيسية القوي (PCA) تحليل المكونات الرئيسية القوي (Robust PCA) هو نوع من خوارزمية تحليل المكونات الرئيسية القياسية (PCA) المصممة لتكون مقاومة للقيم المتطرفة. يعمل عن طريق تحليل مصفوفة البيانات إلى مكونين منفصلين: مصفوفة منخفضة الرتبة تمثل البيانات العادية والمتراكبة بشكل كبير، ومصفوفة متفرقة تمثل البيانات الشاذة وغير المتراكبة. يتم عزل الحالات الشاذة بشكل فعال في المكون المتفرق، مما يسهل التعرف عليها.

4.4.8 نماذج التحقق المتبادل

تمثل إحدى الاستراتيجيات القوية لأمن المستشعرات في بناء نماذج التحقق المتبادل cross-check models. تتضمن هذه الطريقة تطوير نماذج تعلم آلي ML منفصلة للتنبؤ بالقيمة المتوقعة لمستشعر واحد بناءً على قراءات مستشعرات أخرى مستقلة. يمكن أن يشير وجود فرق كبير بين القيمة المتوقعة والقيمة الفعلية المرصودة إلى وجود هجوم أو عطل في المستشعر. على سبيل المثال، يمكن للهرء تدريب نموذج للتنبؤ بسرعة مركبة من نظام GPS الخاص بها باستخدام بيانات تسارع IMU الخاصة بها فقط. إذا انحرفت السرعة الفعلية التي أبلغ عنها نظام GPS بشكل كبير عن السرعة التي تنبأ بها النموذج القائم على IMU، فهذا مؤشر قوي على هجوم محتمل لانتحال إشارات GPS. تستخدم هذه التقنية تكرار المستشعرات sensor redundancy لبناء الثقة.

التفصيل الخوارزمي: منطق مرشح كالمان المبسط

يعمل مرشح كالمان في دورة من خطوتين: التنبؤ والتحديث. يسمح له هذا المنطق بتحسين تقديره لحالة النظام مع كل قياس جديد.

(1) نموذج النظام: أولاً، نحدد نموذجاً رياضياً لكيفية توقع تغير حالة النظام بمرور الوقت. على سبيل المثال، يمكننا نمذجة كيفية تغير موضع السيارة بناءً على سرعتها وتسارعها. يتضمن هذا النموذج أيضاً افتراضات حول ضوضاء العملية، والتي تأخذ في الاعتبار الاضطرابات غير المتوقعة.

(2) خطوة التنبؤ (تحديث الوقت): باستخدام تقدير الحالة السابق ونموذج النظام، يتنبأ المرشح بالحالة الحالية للنظام. كما يتنبأ بعدم اليقين (التباين المشترك) المرتبط بهذه الحالة الجديدة. تسقط هذه الخطوة الحالة إلى الأمام في الوقت.

(3) خطوة تحديث القياس (التصحيح): عند وصول قياس مستشعر جديد، يقارنه المرشح بالحالة المتوقعة. ثم يحسب كسب كالمان (Kalman Gain)، وهي قيمة تحدد مقدار الوزن الذي يجب إعطاؤه للقياس الجديد مقابل التنبؤ. يوازن الكسب الثقة بين تنبؤ النموذج

وقراءة المستشعر المزججة. ثم يتم تحديث (تصحيح) تقدير الحالة من خلال دمج القياس،
مرجحاً بكسب كالمان.

(4) الناتج: النتيجة هي تقدير أكثر دقة وقوة للحالة الحقيقية للنظام (على سبيل المثال، الموضع،
السرعة) مما يمكن الحصول عليه من التنبؤ أو من قياس مستشعر واحد بمفرده. يمكن أيضاً
مراقبة الفرق بين القياسات المتوقعة والمرصودة للكشف عن الحالات الشاذة.

5.8 مشروع عملي: مراقب سلامة بيانات المستشعرات

في هذا المشروع، سنقوم ببناء نظام للكشف عن هجمات انتحال إشارات GPS من خلال مقارنة
قراءات GPS ببيانات من وحدة القياس بالقصور الذاتي (IMU). يوضح هذا المشروع مفاهيم
دمج المستشعرات sensor fusion باستخدام pandas لمعالجة البيانات ونموذج PyTorch بسيط
للكشف عن الشذوذ.

1.5.8 الخطوة 1: إعداد البيانات

أولاً، نقوم بتحميل مجموعة بيانات تحتوي على بيانات GPS و IMU متزامنة في إطار بيانات
pandas DataFrame. ثم نقوم بتنفيذ هندسة الميزات feature engineering لاستخراج
معلومات مفيدة. لهذا العرض التوضيحي، نقوم بإنشاء مجموعة بيانات اصطناعية تتضمن هجوم
انتحال إشارات GPS محاكى. يوضح الكود في القائمة 60 هذه العملية.

2.5.8 الخطوة 2: اختيار وبناء النموذج

بعد ذلك، نقوم ببناء نموذج PyTorch يأخذ كلاً من ميزات GPS و IMU كمدخلات. تحتوي
البنية على فروع منفصلة لكل نوع من أنواع المستشعرات. ثم يتم تسلسل التمثيلات المكتسبة من
هذه الفروع (دمجها) وتمثيلها إلى طبقة تصنيف نهائية. يسمح هذا للنموذج بتعلم العلاقة بين
طريقتي المستشعرين. يظهر تعريف النموذج في القائمة 61.

3.5.8 الخطوة 3: تدريب النموذج

نقوم بتدريب النموذج بشكل أساسي على البيانات العادية. هذا يعلم النموذج الارتباطات المتوقعة
بين قراءات GPS و IMU أثناء التشغيل العادي. من خلال التدريب على البيانات المصنفة
على أنها "عادية"، يتعلم النموذج تحديد الأنماط التي تنحرف عن هذا الخط الأساسي. تظهر حلقة
التدريب في القائمة 62.

```

1 import pandas as pd
2 import numpy as np
3
4 # Create dummy synchronized GPS and IMU data
5 np.random.seed(42)
6 num_points = 100
7 timestamps = pd.to_datetime('2025-08-16 12:00:00') + pd.to_timedelta(np.arange(num_points), unit='s')
8
9 # Simulate normal GPS data (gradual movement)
10 gps_lat = 34.05 + np.cumsum(np.random.normal(0, 0.0001, num_points))
11 gps_lon = -118.25 + np.cumsum(np.random.normal(0, 0.0001, num_points))
12 gps_vel = 10 + np.random.normal(0, 0.5, num_points) # meters/sec
13
14 # Simulate normal IMU data (accelerations consistent with velocity changes)
15 imu_accel_x = np.random.normal(0, 0.1, num_points)
16 imu_accel_y = np.random.normal(0, 0.1, num_points)
17 imu_accel_z = np.random.normal(9.8, 0.05, num_points) # Gravity
18 imu_gyro_z = np.random.normal(0, 0.01, num_points) # Small turning
19
20 df_sensor = pd.DataFrame({
21     'timestamp': timestamps,
22     'gps_lat': gps_lat, 'gps_lon': gps_lon, 'gps_vel': gps_vel,
23     'imu_accel_x': imu_accel_x, 'imu_accel_y': imu_accel_y, 'imu_accel_z': imu_accel_z,
24     'imu_gyro_z': imu_gyro_z
25 })
26
27 # Introduce GPS spoofing (e.g., sudden jump in location or velocity)
28 spoof_start_idx = 40
29 spoof_end_idx = 60
30 df_sensor.loc[spoof_start_idx:spoof_end_idx, 'gps_lat'] += 0.05 # Sudden shift
31 df_sensor.loc[spoof_start_idx:spoof_end_idx, 'gps_lon'] -= 0.05
32 df_sensor.loc[spoof_start_idx:spoof_end_idx, 'gps_vel'] = 50 + np.random.normal(0, 2, spoof_end_idx - spoof_start_idx + 1) # Unre
33 df_sensor['is_spoofed'] = 0
34 df_sensor.loc[spoof_start_idx:spoof_end_idx, 'is_spoofed'] = 1
35
36 # Feature Engineering: Derive IMU-based velocity change (simple integration for concept)
37 # This is a very simplified integration for demo purposes; real IMU integration is complex.
38 time_diff = df_sensor['timestamp'].diff().dt.total_seconds().fillna(0)
39 # Estimated velocity change from IMU: delta_v = accel * delta_t
40 df_sensor['imu_delta_v_x'] = df_sensor['imu_accel_x'] * time_diff
41 df_sensor['imu_delta_v_y'] = df_sensor['imu_accel_y'] * time_diff
42
43 # Consider the current IMU readings as features
44 imu_features = ['imu_accel_x', 'imu_accel_y', 'imu_accel_z', 'imu_gyro_z', 'imu_delta_v_x', 'imu_delta_v_y']
45 gps_features = ['gps_lat', 'gps_lon', 'gps_vel']
46
47 X_imu = df_sensor[imu_features]
48 X_gps = df_sensor[gps_features]
49 y_spoofed = df_sensor['is_spoofed']
50
51 print("Sample Sensor Data Head:\n", df_sensor.head())
52 print(f"\nIMU Features Shape: {X_imu.shape}")
53 print(f"GPS Features Shape: {X_gps.shape}")
54 print(f"Spoofing Label Distribution: {y_spoofed.value_counts()}")

```

Listing 60: تحميل وإعداد بيانات المستشعرات المتزامنة مع هجوم محاكي.

4.5.8 الخطوة 4: كشف الشذوذ والتقييم

بعد التدريب، نستخدم النموذج للكشف عن الحالات الشاذة. عند وصول بيانات جديدة، نقوم بتقرير كل من ميزات GPS و IMU عبر النموذج. إذا قام النموذج بتصنيف المدخلات على أنها

```

1 import torch
2 import torch.nn as nn
3 import torch.nn.functional as F
4 from torch.utils.data import TensorDataset, DataLoader
5 from sklearn.model_selection import train_test_split
6 from sklearn.preprocessing import StandardScaler
7
8 # Scale features before converting to tensor
9 scaler_imu = StandardScaler()
10 scaler_gps = StandardScaler()
11
12 X_imu_scaled = scaler_imu.fit_transform(X_imu)
13 X_gps_scaled = scaler_gps.fit_transform(X_gps)
14
15 # Convert to tensors
16 X_imu_tensor = torch.tensor(X_imu_scaled, dtype=torch.float32)
17 X_gps_tensor = torch.tensor(X_gps_scaled, dtype=torch.float32)
18 y_spoofed_tensor = torch.tensor(y_spoofed.values, dtype=torch.long)
19
20 # Split data
21 X_imu_train, X_imu_test, X_gps_train, X_gps_test, y_train_spoofed, y_test_spoofed = train_test_split(
22     X_imu_tensor, X_gps_tensor, y_spoofed_tensor, test_size=0.01, random_state=42, stratify=y_spoofed
23 )
24 # We create a training set with mostly normal data
25 X_imu_train_normal = X_imu_tensor[y_spoofed_tensor == 0]
26 X_gps_train_normal = X_gps_tensor[y_spoofed_tensor == 0]
27 y_train_normal_labels = y_spoofed_tensor[y_spoofed_tensor == 0]
28
29 # Create datasets and loaders
30 train_dataset_fusion = TensorDataset(X_imu_train_normal, X_gps_train_normal, y_train_normal_labels)
31 train_loader_fusion = DataLoader(train_dataset_fusion, batch_size=32, shuffle=True)
32 test_dataset_fusion = TensorDataset(X_imu_test, X_gps_test, y_test_spoofed)
33 test_loader_fusion = DataLoader(test_dataset_fusion, batch_size=len(test_dataset_fusion), shuffle=False)
34
35 imu_dim = X_imu_tensor.shape
36 gps_dim = X_gps_tensor.shape
37 hidden_dim = 64
38 num_classes = 2 # 0: Normal, 1: Spoofed/Anomaly
39
40 class SensorFusionNet(nn.Module):
41     def __init__(self, imu_dim, gps_dim, hidden_dim, num_classes):
42         super(SensorFusionNet, self).__init__()
43         self.imu_fc = nn.Linear(imu_dim, hidden_dim)
44         self.gps_fc = nn.Linear(gps_dim, hidden_dim)
45         self.fusion_fc = nn.Linear(hidden_dim * 2, hidden_dim)
46         self.output_fc = nn.Linear(hidden_dim, num_classes)
47
48     def forward(self, imu_data, gps_data):
49         imu_rep = F.relu(self.imu_fc(imu_data))
50         gps_rep = F.relu(self.gps_fc(gps_data))
51         fused_rep = torch.cat((imu_rep, gps_rep), dim=1)
52         fused_rep = F.relu(self.fusion_fc(fused_rep))
53         output = self.output_fc(fused_rep)
54         return output
55
56 model_sensor_fusion = SensorFusionNet(imu_dim, gps_dim, hidden_dim, num_classes)
57 print("\nSensor Fusion Model Architecture:\n", model_sensor_fusion)

```

PyTorch. 61 Listing: بناء نموذج دمج المستشعرات باستخدام

شاذة (في حالتنا، "تم انتحالها")، فإننا نضع عليها علامة. نقوم بتقييم أداء النموذج على مجموعة اختبار صغيرة تحتوي على كل من نقاط البيانات العادية والمنتحلة، كما هو موضح في القائمة 63.

```

1 import torch.optim as optim
2
3 device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
4 model_sensor_fusion.to(device)
5 print(f"\nUsing device for fusion model training: {device}")
6
7 criterion_fusion = nn.CrossEntropyLoss()
8 optimizer_fusion = optim.Adam(model_sensor_fusion.parameters(), lr=0.001)
9
10 num_epochs_fusion = 50
11 print("\n--- Starting PyTorch Sensor Fusion Training ---")
12 for epoch in range(num_epochs_fusion):
13     model_sensor_fusion.train()
14     running_loss = 0.0
15     for batch_idx, (imu_data, gps_data, labels) in enumerate(train_loader_fusion):
16         imu_data, gps_data, labels = imu_data.to(device), gps_data.to(device), labels.to(device)
17         optimizer_fusion.zero_grad()
18         outputs = model_sensor_fusion(imu_data, gps_data)
19         loss = criterion_fusion(outputs, labels)
20         loss.backward()
21         optimizer_fusion.step()
22         running_loss += loss.item()
23
24     if (epoch + 1) % 10 == 0:
25         print(f"Epoch [{epoch+1}/{num_epochs_fusion}], Loss: {running_loss / len(train_loader_fusion):.4f}")
26
27 print("\nPyTorch Sensor Fusion Training complete.")

```

PyTorch. 62 Listing: تدريب نموذج دمج المستشعرات باستخدام

```

1 from sklearn.metrics import classification_report, accuracy_score
2
3 model_sensor_fusion.eval()
4 all_preds_fusion = []
5 all_labels_fusion = []
6
7 with torch.no_grad():
8     for imu_data, gps_data, labels in test_loader_fusion:
9         imu_data, gps_data, labels = imu_data.to(device), gps_data.to(device), labels.to(device)
10        outputs = model_sensor_fusion(imu_data, gps_data)
11        _, predicted = torch.max(outputs.data, 1)
12        all_preds_fusion.extend(predicted.cpu().numpy())
13        all_labels_fusion.extend(labels.cpu().numpy())
14
15 print("\n--- Sensor Fusion Model Evaluation on Test Set (Spoofing Detection) ---")
16 print(f"Test Accuracy: {accuracy_score(all_labels_fusion, all_preds_fusion):.4f}")
17 print("\nClassification Report:\n", classification_report(all_labels_fusion, all_labels_fusion, target_names=['Normal', 'Spoofed']

```

63 Listing: تقييم نموذج الدمج للكشف عن الانتحال.

5.5.8 الخطوة 5: التصور

أخيراً، نستخدم matplotlib لتصوير النتائج. نرسم مسار GPS ونسلط الضوء على نقاط البيانات التي كانت جزءاً من هجوم الانتحال الحقيقي، بالإضافة إلى النقاط التي حددها نموذجنا بشكل صحيح على أنها شاذة. يوفر هذا طريقة واضحة وبديهية لرؤية أداء النموذج. يوجد كود التصور في القائمة 64.

```

1 import matplotlib.pyplot as plt
2
3 # Get the actual data points for the test set
4 test_spoof_indices = y_test_spoofed.cpu().numpy() == 1
5 predicted_spoof_indices = np.array(all_preds_fusion) == 1
6
7 plt.figure(figsize=(10, 8))
8 plt.plot(X_gps_tensor[:, 1].cpu().numpy(), X_gps_tensor[:, 0].cpu().numpy(), 'b-o', markersize=3, label='Full Trajectory (GPS Data)')
9 plt.plot(X_gps_tensor[y_spoofed_tensor == 1, 1].cpu().numpy(), X_gps_tensor[y_spoofed_tensor == 1, 0].cpu().numpy(), 'ro', marker='o', label='Spoofed Trajectory (GPS Data)')
10 plt.plot(X_gps_test[predicted_spoof_indices, 1].cpu().numpy(), X_gps_test[predicted_spoof_indices, 0].cpu().numpy(), 'kx', marker='x', label='Predicted Spoofing Events')
11
12 plt.xlabel('Longitude'); plt.ylabel('Latitude')
13 plt.title('GPS Trajectory with True and Predicted Spoofing Events')
14 plt.grid(True)
15 plt.legend()
16 plt.show()

```

Listing 64: تصور مسار GPS مع الحالات الشاذة المكتشفة.

الخلاصة

تعد حماية المستشعرات في الأنظمة الفيزيائية السيبرانية Cyber-Physical Systems تحدياً أمنياً حاسماً. يمكن أن يكون للتلاعب بالخبيث بيانات المستشعرات عواقب وخيمة في العالم الحقيقي. أظهر هذا الفصل أن الذكاء الاصطناعي AI يوفر حلاً قوياً من خلال تعلم العلاقات المعقدة بين المستشعرات المختلفة. تسمح لنا تقنيات مثل دمج المستشعرات sensor fusion، والتعلم العميق متعدد الوسائط deep multimodal learning، والتحقق المتبادل cross-checking ببناء أجهزة مراقبة ذكية يمكنها اكتشاف الحالات الشاذة الدقيقة التي تشير إلى وجود هجوم. أظهر المشروع العملي كيف يمكن لنموذج تعلم عميق deep learning بسيط تحديد انتقال إشارات GPS بشكل فعال من خلال التحقق المتبادل من بيانات GPS مع قراءات IMU. مع ازدياد شيوع الأنظمة الفيزيائية السيبرانية، ستكون هذه التقنيات الأمنية القائمة على الذكاء الاصطناعي ضرورية لضمان سلامتها وموثوقيتها.

المختصرات والمصطلحات

الذكاء الاصطناعي (Artificial Intelligence) AI هو مجال علوم الحاسوب المخصص لإنشاء أنظمة يمكنها أداء المهام التي تتطلب عادةً ذكاءً بشرياً، مثل التعلم والاستدلال وحل المشكلات.

المُرْمِزُ التلقائي Autoencoder هو نوع من الشبكات العصبية neural network يُستخدم في التعلم غير الخاضع للإشراف unsupervised learning، وغالباً ما يهدف إلى تعلم ترميزات فعالة للبيانات أو كشف الشذوذ عبر تحديد البيانات التي لا يستطيع إعادة بنائها بدقة.

الشبكة العصبية الالتفافية (Convolutional Neural Network) CNN هي نوع متخصص من نماذج التعلم العميق deep learning مصممة لمعالجة البيانات ذات البنية الشبكية، مثل الصور. تُستخدم بشكل شائع في مهام التعرف على الكائنات وتصنيف الصور.

النظام السيبراني الفيزيائي (Cyber-Physical System) CPS هو نظام يدمج الخوارزميات الحاسوبية مع العمليات الفيزيائية. تستخدم هذه الأنظمة أجهزة استشعار لمراقبة العالم المادي ومُشغلات للتحكم فيه.

نموذج التحقق المتقاطع Cross-Check Model هو أسلوب أمني يتم فيه تدريب نموذج تعلم الآلة machine learning للتنبؤ بقيمة مستشعر واحد بناءً على بيانات من مستشعرات أخرى مستقلة. وجود تباين كبير بين القيم المتوقعة والفعلية يمكن أن يشير إلى وجود هجوم. حقن البيانات Data Injection هو هجوم سيبراني يقوم فيه المهاجم بإدخال بيانات ضارة في مجرى بيانات النظام لتعطيل تشغيله أو دفعه لاتخاذ قرارات خاطئة.

الحساب التقديري للموقع Dead Reckoning هي طريقة ملاحية تحسب الموقع الحالي للنظام باستخدام موقع معروف سابقاً وتقديره بناءً على السرعات والاتجاهات المقدرة بمرور الوقت، وغالباً ما تستخدم بيانات من وحدة القياس بالقصور الذاتي IMU.

التعلم العميق (Deep Learning) DL هو فرع من تعلم الآلة machine learning يستخدم الشبكات العصبية neural networks متعددة الطبقات (البنى العميقة) لتعلم الأنماط المعقدة من كميات كبيرة من البيانات.

نظام تحديد المواقع العالمي (Global Positioning System) GPS هو نظام ملاحية يعتمد على الأقمار الصناعية ويوفر معلومات عن الموقع والسرعة والوقت لجهاز استقبال على الأرض أو بالقرب منها.

نظام التحكم الصناعي ICS (Industrial Control System) هو مصطلح عام للأنظمة المستخدمة لمراقبة العمليات الصناعية والتحكم فيها، مثل تلك الموجودة في منشآت التصنيع والطاقة ومعالجة المياه.

وحدة القياس بالقصور الذاتي IMU (Inertial Measurement Unit) هو جهاز إلكتروني يقيس ويبلغ عن القوة النوعية لجسم ما، ومعدل زاويته، واتجاهه باستخدام مجموعة من مقاييس التسارع، والجيروسكوبات، وأحياناً مقاييس المغناطيسية.

مرشح كالمان Kalman Filter هو خوارزمية رياضية تقدر حالة نظام ديناميكي من سلسلة من القياسات المشوشة. يُستخدم على نطاق واسع في أنظمة الملاحة والتحكم لتنقية بيانات المستشعرات وتقديم تنبؤات أكثر دقة.

كسب كالمان Kalman Gain هي قيمة حاسمة يتم حسابها داخل مرشح كالمان Kalman Filter توازن بين الوزن المعطى لتنبؤ المرشح الخاص به مقابل قياس المستشعر المستلم حديثاً.

ليدار (كشف الضوء وتحديد المدى) LIDAR (Light Detection and Ranging) هي طريقة استشعار عن بعد تستخدم ضوء الليزر لقياس المسافات إلى الأجسام، مما ينشئ سحابة نقطية ثلاثية الأبعاد ودقيقة للبيئة المحيطة.

الذاكرة طويلة قصيرة المدى LSTM (Long Short-Term Memory) هي نوع من الشبكات العصبية المتكررة RNN مناسبة جداً لمعالجة وتقديم تنبؤات من تسلسلات البيانات، مثل قراءات المستشعرات المتسلسلة زمنياً.

مسافة ماهالانوبيس Mahalanobis Distance هي مقياس إحصائي يُستخدم لتحديد القيم الشاذة في البيانات متعددة الأبعاد. تحسب المسافة بين نقطة بيانات ومركز التوزيع، مع مراعاة الارتباطات بين المتغيرات.

تعلم الآلة ML (Machine Learning) هو فرع من فروع الذكاء الاصطناعي AI يمكن الأنظمة من التعلم والتحسين تلقائياً من التجربة دون أن تكون مبرمجة بشكل صريح.

الجمعية الوطنية للإلكترونيات البحرية NMEA (National Marine Electronics Association) هو تنسيق بيانات قياسي للاتصال بين الأجهزة الإلكترونية، معتمد على نطاق واسع من قبل أجهزة استقبال نظام تحديد المواقع العالمي GPS لإخراج بيانات الموقع والسرعة وغيرها من البيانات في "جمل" نصية.

كشف القيم الشاذة Outlier Detection هي عملية تحديد نقاط البيانات أو الأحداث التي تختلف بشكل كبير عن غالبية البيانات، والتي يمكن أن تكون مؤشراً على عطل في المستشعر أو هجوم سبيرياني.

تحليل المكونات الرئيسية (PCA) (Principal Component Analysis) هو أسلوب إحصائي لتقليل أبعاد مجموعة البيانات مع الحفاظ على أكبر قدر ممكن من تباينها.

تحليل المكونات الرئيسية المتين Robust PCA هو نسخة معدلة من تحليل المكونات الرئيسية PCA مصممة لتكون مقاومة للقيم الشاذة، مما يمكنها من فصل البيانات العادية المترابطة عن البيانات الشاذة المتناثرة.

دمج المستشعرات Sensor Fusion هي عملية دمج البيانات من عدة مستشعرات مختلفة لإنتاج فهم أكثر دقة واكتمالاً وموثوقية لحالة النظام أو بيئته.

انتحال بيانات المستشعر Sensor Spoofing هو هجوم سبيرياني يقوم فيه المهاجم بتزويد المستشعر ببيانات زائفة أو معدلة لخداع النظام وجعله يدرك واقعاً مادياً غير صحيح.

منصة اختبار معالجة المياه الآمنة (SWaT) (Secure Water Treatment) هي منصة اختبار ومجموعة بيانات معروفة تمثل نظام تحكم صناعي industrial control system واقعي (محطة معالجة مياه)، يستخدمها الباحثون لدراسة واختبار حلول الأمن السبيرياني للبنية التحتية الحيوية.

باب 9

المحولات والبرمجة اللغوية العصبية للتصيد المتقدم والهندسة الاجتماعية واستخبارات التهديدات

مقدمة

لقد تطورت تهديدات الأمن السيبراني الحديثة إلى ما هو أبعد من رسائل التصيد الاحتيالي البسيطة التي تحتوي على أخطاء إملائية. اليوم، يواجه المدافعون هجمات مستهدفة للغاية، بما في ذلك اختراق البريد الإلكتروني للأعمال (BEC) Business Email Compromise، والنصوص المزيفة بتقنية التزييف العميق deepfake text، وحملات الهندسة الاجتماعية المتطورة sophisticated social engineering. تستخدم هذه الهجمات لغة دقيقة ومناسبة للسياق، مما يجعل من شبه المستحيل اكتشافها باستخدام الأنظمة التقليدية القائمة على الكلمات المفتاحية أو القواعد البسيطة. علاوة على ذلك، فإن محلي الأمن غارقون في كميات هائلة من البيانات غير المهيكلة من تقارير استخبارات التهديدات، ومدونات الأمن، والتنبيهات الداخلية. يعد استخراج المعلومات القابلة للتنفيذ من هذه النصوص عملية يدوية وبطيئة.

يقدم هذا الفصل حلاً قوياً: صقل fine-tuning نماذج اللغة الكبيرة القائمة على المحولات Transformer-based Large Language Models (LLMs) لمهام معالجة اللغات الطبيعية (NLP) Natural Language Processing الخاصة بالأمن. سوف نستكشف كيف يمكن لنماذج الذكاء الاصطناعي المتقدمة هذه فهم السياق والقصد من وراء النص، مما يمكنها من مكافحة الهجمات المتطورة وأتمتة استخراج معلومات استخبارات التهديدات. بنهاية هذا الفصل، ستفهم المفاهيم الأساسية للمحولات Transformers والصقل fine-tuning

وستكون قادراً على بناء نموذج لاكتشاف تهديد معقد مثل اختراق البريد الإلكتروني للأعمال
Business Email Compromise.

1.9 المشكلة، التقنية، والمفاهيم الأساسية

المشكلة الأساسية ذات شقين: يستخدم الخصوم لغة متطورة لشن هجمات تتجاوز الدفاعات البسيطة، ويجب على فرق الأمن معالجة كميات هائلة من النصوص غير المهيكلة يدوياً للعثور على بيانات التهديدات الخفية. تفشل الطرق التقليدية لأنها تفتقر إلى فهم السياق، والسخرية، والقصد. توفر نماذج المحولات Transformer models وسيلة للتغلب على هذا القيد من خلال تعلم العلاقات السياقية العميقة في اللغة.

1.1.9 معالجة اللغات الطبيعية ونماذج اللغة الكبيرة

معالجة اللغات الطبيعية (NLP) هي أحد مجالات الذكاء الاصطناعي -Artificial Intelligence (AI) التي تمكن أجهزة الكمبيوتر من فهم اللغة البشرية وتفسيرها وتوليدها [35]. إنها بمثابة الجسر بين التواصل البشري والمعالجة الحاسوبية، مما يسمح لنا ببناء أنظمة يمكنها التعامل مع البيانات النصية والكلامية. تعد نماذج اللغة الكبيرة (LLMs) تطوراً حديثاً وقوياً في مجال معالجة اللغات الطبيعية. وهي نماذج تعلم عميق deep learning متقدمة، غالباً ما تحتوي على مليارات المعلمات parameters، وقد تم تدريبها مسبقاً على كميات هائلة من النصوص من الإنترنت. تتيح لهم عملية التدريب المسبق pre-training هذه تطوير فهم عام وقوي للقواعد النحوية، والحقائق، وقدرات التفكير، وتوليد اللغة. هذه المعرفة التأسيسية تجعلها قادرة على أداء مجموعة واسعة من المهام، مثل الترجمة والتلخيص والإجابة على الأسئلة، غالباً مع القليل من التدريب الإضافي.

2.1.9 تضمين النصوص: من الثابت إلى المدرك للسياق

إن تضمين النص text embedding هو متجه رقمي يمثل جزءاً من النص، مثل كلمة أو جملة. الفكرة الرئيسية هي أن هذه المتجهات تلتقط المعنى الدلالي للنص. الكلمات ذات المعاني المتشابهة تكون أقرب إلى بعضها البعض في الفضاء المتجهي عالي الأبعاد. أنتجت تقنيات التضمين المبكرة، مثل Word2Vec و GloVe، متجهاً واحداً ثابتاً لكل كلمة. على سبيل المثال، كلمة "bank" سيكون لها نفس التضمين في "river" "bank" (ضفة النهر) و

investment "bank" (بنك استثماري). هذا القيد يفشل في التقاط المعنى الحقيقي، الذي يعتمد بشكل كبير على السياق. نماذج المحولات Transformer models، مع ذلك، تنتج تضمينات مدركة للسياق context-aware embeddings. يتم إنشاء تضمين كلمة "bank" ديناميكياً بناءً على الكلمات الأخرى في الجملة. وهذا يسمح للنموذج بالتمييز بين المعنيين، مما يؤدي إلى فهم أكثر ثراءً ودقة للنص. هذه القدرة على فهم السياق هي ما يجعل المحولات Transformers قوية جداً للمهام الأمنية الدقيقة.

3.1.9 صقل نماذج المحولات

الصقل Fine-tuning هو عملية أخذ نموذج لغة كبير مدرب مسبقاً pre-trained LLM وتدريبه بشكل إضافي على مجموعة بيانات أصغر ومخصصة لمجال معين. لقد تعلم النموذج بالفعل الأنماط اللغوية العامة من تدريبه الأولي الضخم. يقوم الصقل بتكييف هذه المعرفة العامة لمهمة متخصصة، مثل تحديد رسائل البريد الإلكتروني الخادعة أو استخراج التهديدات الأمنية. هذا النهج فعال للغاية لأنه ينقل معرفة واسعة إلى مشكلة معينة، مما يحقق أداءً عالياً ببيانات أقل بكثير من تدريب نموذج من الصفر.

تشمل النماذج الشهيرة المدربة مسبقاً لهذا الغرض Encoder (Bidirectional BERT from Representations، Transformers)، DistilBERT و (نسخة أصغر وأسرع من BERT و RoBERTa [21]). تتوفر هذه النماذج بسهولة من خلال مكتبات مثل Hugging Face Transformers، والتي تم بناؤها فوق أطر التعلم العميق مثل PyTorch. العملية العامة لصقل نموذج محول مهمة تصنيف هي كما يلي:

- (1) اختر نموذجاً مدرباً مسبقاً: حدد نموذجاً يناسب مهمتك وميزانيتك الحاسوبية. على سبيل المثال، يعد `bert-base-uncased` نقطة انطلاق رائعة لتحليل النصوص الإنجليزية.
- (2) إعداد بيانات خاصة بالمهمة: اجمع وصنف البيانات لمهمتك المحددة. بالنسبة لاكتشاف التصيد الاحتيالي، ستكون هذه مجموعة بيانات من رسائل البريد الإلكتروني، كل منها مصنفة على أنها "تصيد" أو "شرعية".
- (3) الترميز Tokenization: استخدم المرمز tokenizer الخاص بالنموذج لتحويل النص إلى معرفات رقمية. يقوم المرمز بتقسيم النص إلى كلمات أو كلمات فرعية (رموز tokens) ويربطها بأعداد صحيحة. كما يضيف رموزاً خاصة يتطلبها النموذج، مثل `[CLS]` لمهام التصنيف و `[SEP]` لفصل الجمل. يقوم بإنشاء قناع attention mask، الذي

يخبر النموذج بالانتباه إلى الرموز الحقيقية وتجاهل رموز الحشو padding tokens المضافة لجعل جميع التسلسلات بنفس الطول.

(4) تحميل النموذج: قم بتحميل النموذج المدرب مسبقاً وأضف "رأساً" خاصاً بالمهمة فوقه. بالنسبة للتصنيف، هذا عبارة عن طبقة شبكة عصبية بسيطة تنتج تنبؤات لفئاتك المستهدفة.

(5) التدريب (الصقل): قم بتدريب النموذج على بياناتك المصنفة. عادةً ما تستخدم معدل تعلم صغير لأن أوزان النموذج مهيأة جيداً بالفعل. تضبط هذه الخطوة معلمات النموذج للتخصص في مهمتك الأمنية.

(6) التقييم: قم بقياس أداء النموذج على مجموعة اختبار منفصلة باستخدام مقاييس مناسبة. بالنسبة للمهام الأمنية، غالباً ما تكون مقاييس مثل الدقة precision، والاستدعاء re-call، ودرجة F1-score F1 أكثر إفادة من الدقة البسيطة accuracy.

2.9 التطبيقات في الأمن السيبراني

يمكن تطبيق المحولات المصقولة على مجموعة واسعة من تحديات الأمن السيبراني التي تتضمن فهم النصوص.

1.2.9 كشف التصيد والهندسة الاجتماعية

- تصنيف رسائل التصيد الاحتيالي المتقدمة: يمكن للمحولات تحليل أسلوب ونبرة وسياق رسالة بريد إلكتروني لاكتشاف محاولات التصيد المتطورة التي تفوتها الأنظمة التقليدية. يمكنها تحديد الإشارات الدقيقة للإلحاح أو انتحال الشخصية التي لا ترتبط بكلمات مفتاحية محددة.

- كشف النصوص المزيفة بتقنية التزييف العميق: بينما يستخدم المهاجمون الذكاء الاصطناعي لتوليد نصوص خبيثة مقنعة (مثل مراجعات المنتجات المزيفة والأخبار الملفةقة)، يمكن تدريب نماذج الذكاء الاصطناعي الأخرى مثل المحولات لتحديد الأنماط الإحصائية الدقيقة التي تميز النص الذي تم إنشاؤه آلياً عن الكتابة البشرية.

- كشف عناوين URL/مواقع الويب الخبيثة: بدلاً من الاعتماد فقط على القوائم السوداء، يمكن للنموذج تحليل النص على صفحة ويب أو داخل عنوان URL لتحديد ما إذا كان جزءاً من حملة تصيد. على سبيل المثال، يمكنه قراءة سياسة خصوصية موقع ويب لمعرفة ما إذا كانت غير منطقية أو منسوخة من مصدر آخر.

- تحليل نية الرسائل المشبوهة: يمكن للنموذج تصنيف النية الأساسية للرسالة، مثل طلب قسري لإعادة تعيين كلمة المرور، أو طلب عاجل لإجراء معاملة مالية، أو محاولة لإثارة الفضول لجعل المستخدم ينقر على رابط.

2.2.9 أتمتة استخبارات التهديدات

المحاولات فعالة للغاية في أتمتة معالجة استخبارات التهديدات غير المهيكلية.

- استخراج مؤشرات الاختراق (Indicators of Compromise (IOCs): مؤشر الاختراق (IOC) هو جزء من البيانات الجنائية الرقمية التي تشير إلى خرق أمني محتمل. تشمل الأمثلة عناوين IP الخبيثة، وأسماء النطاقات، وتجزئات الملفات (مثل MD5، SHA256) وعناوين البريد الإلكتروني. يمكن صقل الحول كنموذج للتعرف على الكيانات المسماة (Named Entity Recognition (NER لقراءة مدونات الأمن وتقارير التهديدات ومنشورات المنتديات تلقائياً لاستخراج هذه المؤشرات، مما يوفر على المحللين ساعات لا تحصى.
- تلخيص وإثراء التنبيهات: يمكن للنماذج إنشاء ملخصات قصيرة وسهلة القراءة للتنبيهات الأمنية المعقدة. يمكنها أيضاً إثراء التنبيهات عن طريق العثور تلقائياً على المعلومات ذات الصلة، مثل ربط النشاط الملحوظ بتقنيات MITRE ATT&CK المحددة.
- إنشاء قواعد أمنية آلية: يمكن للنموذج ترجمة وصف لغة طبيعية لتهديد ما إلى قاعدة كشف رسمية. على سبيل المثال، يمكنه تحويل منشور مدونة يصف متغيراً جديداً من البرامج الضارة إلى قاعدة YARA، وهي توقيع يستخدم لتحديد وتصنيف عينات البرامج الضارة [79]، أو قاعدة Sigma، وهي تنسيق توقيع عام لأحداث السجل [?].

3.2.9 تحليل وصف الثغرات الأمنية

يمكن للمحاولات تحليل النصوص من قاعدة بيانات الثغرات والمخاطر الشائعة -Common Vul-nerabilities and Exposures (CVE) وغيرها من التحذيرات الأمنية. يمكنها استخراج المعلومات الرئيسية تلقائياً مثل المنتجات المتأثرة والإصدارات وناقلات الهجوم. علاوة على ذلك، يمكن تدريبها على تحليل وصف الثغرة للتنبؤ باحتمالية استغلالها في الواقع، مما يساعد فرق الأمن على تحديد أولويات جهود التصحيح.

3.9 مجموعات البيانات المفتوحة لمعالجة اللغات الطبيعية في الأمن

لصقل النماذج، تحتاج إلى بيانات مصنفة. تتوفر العديد من مجموعات البيانات العامة للمهام المتعلقة بمعالجة اللغات الطبيعية في مجال الأمن:

- مجموعات رسائل التصيد الاحتيالي لنازاريو Nazario Phishing Email Corpora: هي مجموعات معروفة من رسائل البريد الإلكتروني الحقيقية للتصيد والرسائل الشرعية، وهي لا تقدر بثمن لتدريب نماذج الكشف [?].
- مجموعة بيانات مواقع التصيد من UCI: تحتوي مجموعة البيانات هذه على ميزات مستخرجة من آلاف مواقع الويب، والتي يمكن استكشافها عن طريق استخلاص النص الفعلي من المواقع لتحليل المحتوى [56].
- قاعدة بيانات CVE: توفر قاعدة البيانات الوطنية للثغرات (NVD) جميع سجلات CVE بتنسيق منظم. تعد الأوصاف النصية للثغرات مثالية لمهام معالجة اللغات الطبيعية مثل استخراج المعلومات والتنبؤ بالخطورة.
- صائدات MISP: تحتوي منصة مشاركة معلومات البرامج الضارة (MISP) على بيانات تهديدات مهيكلية. على الرغم من أنها شبه مهيكلية بالفعل، يمكن استخدامها حقولها الوصفية لإنشاء مجموعات بيانات نصية لمهام مثل تلخيص التهديدات.
- مجموعات البيانات المخصصة المستخلصة: غالباً ما يقوم الباحثون والمؤسسات بإنشاء مجموعات بيانات مخصصة عن طريق استخلاص البيانات من المصادر العامة مثل مدونات الأمن أو المواقع الإخبارية أو المنتديات مثل r/netsec على Reddit. عند القيام بذلك، من الأهمية بمكان مراعاة الآثار القانونية والأخلاقية لجمع البيانات واستخدامها.

4.9 مشروع عملي: محول للكشف عن اختراق البريد الإلكتروني للأعمال

في هذا المشروع، سنقوم بصقل نموذج محول مدرب مسبقاً لاكتشاف هجمات اختراق البريد الإلكتروني للأعمال (BEC).

هجوم اختراق البريد الإلكتروني للأعمال (BEC) هو عملية احتيال مستهدفة للغاية حيث ينتحل المهاجم شخصية مسؤول تنفيذي في شركة أو بائع موثوق به. الهدف هو خداع موظف لإجراء تحويل بنكي احتيالي أو الكشف عن معلومات حساسة. غالباً ما تكون هذه الرسائل

مكتوبة جيداً، ولا تحتوي على روابط أو مرفقات ضارة، وتعتمد فقط على الهندسة الاجتماعية، مما يجعل من الصعب جداً على المرشحات التقليدية اكتشافها [2].

1.4.9 الخطوة 1: إعداد البيانات والترميز

أولاً، سنقوم بإنشاء مجموعة بيانات صغيرة ومتوازنة من رسائل البريد الإلكتروني المحاكاة لـ BEC والرسائل الشرعية. سنستخدم إطار بيانات `pandas` للاحتفاظ بالبيانات. بعد ذلك، سنستخدم المرمز `tokenizer` من مكتبة `transformers` في Hugging Face لتحويل النص إلى تنسيق يمكن للنموذج فهمه. يوضح الكود في القائمة 65 هذه العملية. يقوم بإعداد فئة `Dataset` مخصصة لـ PyTorch، ويقسم البيانات، ويعرض مثالاً للدخول مرمز.

2.4.9 الخطوة 2: تحميل النموذج والصقل

بعد ذلك، نقوم بتحميل نموذج `bert-base-uncased` المدرب مسبقاً مع رأس لتصنيف التسلسل. الوسيطة `num\_labels=2` تخبر النموذج أن لدينا فئتين إخراج: "شرعي" و "BEC". ثم نستخدم واجهة برمجة تطبيقات `Trainer` من Hugging Face، والتي تبسط حلقة التدريب. نحدد `TrainingArguments` للتحكم في المعلمات مثل عدد الحقب وحجم الدفعة. يقوم الكود في القائمة 66 بإعداد وبدء عملية الصقل.

3.4.9 الخطوة 3: التقييم

بعد التدريب، يجب علينا تقييم أداء النموذج على مجموعة الاختبار المحجوزة. يخبرنا هذا بمدى قدرة النموذج على التعميم على بيانات جديدة غير مرئية. المهمة مثل الكشف عن اختراق البريد الإلكتروني للأعمال، تعتبر الدقة `precision` والاستدعاء `recall` أمرين حاسمين. الدقة العالية لفئة "BEC" تعني أنه عندما يصنف النموذج رسالة بريد إلكتروني على أنها BEC، فمن المرجح جداً أن تكون صحيحة (إيجابيات كاذبة منخفضة). الاستدعاء العالي يعني أن النموذج ينجح في تحديد معظم رسائل BEC الفعلية (سلبيات كاذبة منخفضة). يقوم الكود في القائمة 67 بإجراء هذا التقييم وطباعة تقرير تصنيف مفصل.

4.4.9 استكشاف مفاهيمي: هندسة التلقين

بينما ينشئ الصقل نماذج متخصصة، هناك طريقة أخرى قوية لاستخدام نماذج اللغة الكبيرة جداً وذات الأغراض العامة (مثل GPT-4) وهي من خلال هندسة التلقين prompt en-

gineering. لا تتضمن هذه التقنية إعادة تدريب النموذج. بدلاً من ذلك، يقوم المحلل بصياغة تعليمات نصية مفصلة (تلقين prompt) لتوجيه النموذج لأداء مهمة محددة. يُعرف هذا أيضاً باسم "التعلم في السياق in-context learning". بالنسبة لمحلل الأمن، قد يعني هذا استخدام تلقين مثل:

حلل البريد الإلكتروني التالي بحثاً عن علامات هجوم اختراق البريد الإلكتروني للأعمال. ركز على الإلحاح، والطلبات المالية غير العادية، وتكتيكات انتحال الشخصية. قدم ملخصاً لنتائجك ودرجة ثقة من 1 إلى 10.

البريد الإلكتروني: [الصق نص البريد الإلكتروني هنا]

هذا النهج يحول نموذج لغة كبير عام إلى مساعد قوي للتحليل دون الحاجة إلى تطوير نموذج مخصص. إنه يمثل اتجاهًا مستقبلياً حيث يصبح التفاعل مع الذكاء الاصطناعي للمهام الأمنية أكثر حوارية وسهولة في الوصول.

الخاتمة

تمثل النماذج القائمة على المحولات نقلة نوعية في كيفية تعاملنا مع تحديات الأمن السيبراني القائمة على النصوص. من خلال فهم سياق اللغة ودلالاتها، يمكننا اكتشاف هجمات الهندسة الاجتماعية والتصيد المتطورة التي تتجنب بسهولة الأنظمة التقليدية القائمة على التوقعات. تتيح لنا عملية الصقل fine-tuning تكييف هذه النماذج العامة القوية مع مجالات أمنية محددة للغاية، مثل تحديد رسائل البريد الإلكتروني لاختراق الأعمال BEC أو استخراج مؤشرات الاختراق IOCs من تقارير التهديدات غير المهيكلة. كما هو موضح في مشروعنا العملي، فإن مكتبات مثل Hugging Face Transformers و PyTorch تجعل هذه التكنولوجيا المتقدمة في متناول الممارسين. سواء من خلال صقل النماذج المتخصصة أو استخدام هندسة التلقين مع النماذج العامة الضخمة، أصبحت معالجة اللغات الطبيعية أداة لا غنى عنها في مجموعة أدوات محلل الأمن الحديث، مما يتيح أنظمة دفاع أسرع وأكثر دقة وأكثر آلية.

```

1 import pandas as pd
2 from transformers import AutoTokenizer
3 import torch
4 from torch.utils.data import Dataset, DataLoader
5 from sklearn.model_selection import train_test_split
6 from sklearn.metrics import classification_report, accuracy_score
7 from tqdm.notebook import tqdm # For progress bars
8 import os
9 import shutil
10 import numpy as np
11
12 # Create dummy BEC and legitimate email data
13 data = {
14     'text': [
15         "Urgent: CEO requesting wire transfer to new vendor account. Process immediately.", # BEC
16         "Subject: Q3 Financial Report - Please review and provide feedback.", # Legitimate
17         "Action Required: Process invoice #12345 to new account per attached. From: John Smith (CFO).", # BEC
18         "Meeting reminder: Project Alpha review at 2 PM today.", # Legitimate
19         "I am out of office. Please transfer $50,000 to this new account for urgent deal. - CEO", # BEC
20         "Regarding your request for new software licenses, please follow internal process.", # Legitimate
21         "Confirming new bank details for payment of supplier X. Transfer by end of day.", # BEC
22         "Monthly security awareness training is scheduled for next week. Don't miss it." # Legitimate
23     ],
24     'label': # 1: BEC, 0: Legitimate
25 }
26 df_emails_bec = pd.DataFrame(data)
27 print("Original BEC/Legitimate Email DataFrame:\n", df_emails_bec)
28
29 # Choose a pre-trained tokenizer (e.g., from BERT base uncased model)
30 tokenizer = AutoTokenizer.from_pretrained('bert-base-uncased')
31 print(f"\nLoaded tokenizer: {tokenizer.name_or_path}")
32
33 # Max sequence length for tokenizer (e.g., 128 or 256 tokens)
34 MAX_LEN = 128
35
36 class EmailDataset(Dataset):
37     def __init__(self, texts, labels, tokenizer, max_len):
38         self.texts = texts
39         self.labels = labels
40         self.tokenizer = tokenizer
41         self.max_len = max_len
42
43     def __len__(self):
44         return len(self.texts)
45
46     def __getitem__(self, item):
47         text = str(self.texts[item])
48         label = self.labels[item]
49
50         encoding = self.tokenizer.encode_plus(
51             text,
52             add_special_tokens=True, # Add '[CLS]' and '[SEP]'
53             max_length=self.max_len, # Pad/truncate to max_len
54             return_token_type_ids=False, # No token type IDs for single sentence classification
55             padding='max_length', # Pad to max_length
56             return_attention_mask=True, # Return attention mask
57             return_tensors='pt', # Return PyTorch tensors
58             truncation=True # Truncate if sequence is longer than max_len
59         )
60
61         return {
62             'text': text,
63             'input_ids': encoding['input_ids'].flatten(),
64             'attention_mask': encoding['attention_mask'].flatten(),
65             'labels': torch.tensor(label, dtype=torch.long)
66         }
67
68 # Create datasets
69 train_texts, test_texts, train_labels, test_labels = train_test_split(
70     df_emails_bec['text'].to_numpy(),
71     df_emails_bec['label'].to_numpy(),
72     test_size=0.25, # Small test size for this tiny dataset
73     random_state=42,
74     stratify=df_emails_bec['label'] # Stratify to maintain label distribution
75 )
76
77 train_dataset_nlp = EmailDataset(list(train_texts), list(train_labels), tokenizer, MAX_LEN)
78 test_dataset_nlp = EmailDataset(list(test_texts), list(test_labels), tokenizer, MAX_LEN)
79
80 # Example of a tokenized input
81 sample_item = train_dataset_nlp
82 print(f"\nSample tokenized input_ids: {sample_item['input_ids']}")
83 print(f"Sample attention_mask: {sample_item['attention_mask']}")
84 print(f"Sample label: {sample_item['labels']}")
85 print(f"Decoded text: {tokenizer.decode(sample_item['input_ids'], skip_special_tokens=True)}")

```

```

1 from transformers import AutoModelForSequenceClassification, TrainingArguments, Trainer
2 from sklearn.metrics import f1_score, precision_score, recall_score
3
4 # Note: This code assumes the variables from the previous listing (Listing 10.1)
5 # such as train_dataset_nlp and test_dataset_nlp are available in the same session.
6
7 # Load a pre-trained BERT model for sequence classification
8 # num_labels=2 for binary classification (BEC or Legitimate)
9 model_nlp = AutoModelForSequenceClassification.from_pretrained('bert-base-uncased', num_labels=2)
10 print("\nBERT Model for Sequence Classification Loaded.")
11
12 # Define training arguments (adjust as needed for real projects)
13 training_args = TrainingArguments(
14     output_dir='./results_bec',          # output directory
15     num_train_epochs=10,                 # total number of training epochs
16     per_device_train_batch_size=2,       # batch size per device during training
17     per_device_eval_batch_size=2,       # batch size per device during evaluation
18     warmup_steps=50,                    # number of warmup steps for learning rate scheduler
19     weight_decay=0.01,                  # strength of weight decay
20     logging_dir='./logs_bec',           # directory for storing logs
21     logging_steps=100,
22     evaluation_strategy="epoch",         # Evaluate every epoch
23     save_strategy="epoch",              # Save model every epoch
24     load_best_model_at_end=True,         # Load the best model when training ends
25     metric_for_best_model="f1",          # Metric to use for best model selection
26     greater_is_better=True,
27     report_to="none"                    # Disable integrations like Weights & Biases for simple demo
28 )
29
30 # Define a compute_metrics function for evaluation
31 def compute_metrics(p):
32     predictions, labels = p
33     predictions = np.argmax(predictions, axis=1)
34     f1 = f1_score(labels, predictions, average='binary') # 'binary' for 2 classes
35     precision = precision_score(labels, predictions, average='binary')
36     recall = recall_score(labels, predictions, average='binary')
37     return {"f1": f1, "precision": precision, "recall": recall}
38
39 # Create the Trainer instance
40 trainer = Trainer(
41     model=model_nlp,                    # the instantiate Transformers model to be trained
42     args=training_args,                 # training arguments, defined above
43     train_dataset=train_dataset_nlp,    # training dataset
44     eval_dataset=test_dataset_nlp,     # evaluation dataset
45     compute_metrics=compute_metrics,    # the callback for metrics computation
46 )
47
48 # Train the model
49 print("\n--- Starting BERT Fine-tuning for BEC Classification ---")
50 trainer.train()
51 print("\nBERT Fine-tuning complete.")

```

Listing 66: صقل نموذج BERT لتصنيف اختراق البريد الإلكتروني للأعمال.

```

1  # Note: This code assumes the variables from the previous listings
2  # (e.g., trainer, test_dataset_nlp, test_labels) are available.
3
4  print("\n--- Evaluating Fine-tuned BERT Model on Test Set ---")
5  eval_results = trainer.evaluate()
6  print(f"Evaluation results: {eval_results}")
7
8  # Manually predict and print classification report for clarity
9  predictions = trainer.predict(test_dataset_nlp)
10 y_pred_nlp = np.argmax(predictions.predictions, axis=1)
11 y_true_nlp = test_labels
12
13 print("\n--- Classification Report for BEC Detection ---")
14 print(classification_report(y_true_nlp, y_pred_nlp, target_names=['Legitimate', 'BEC']))
15
16 # Clean up results directory
17 if os.path.exists('./results_bec'):
18     shutil.rmtree('./results_bec')
19     print("\nCleaned up ./results_bec directory.")
20 if os.path.exists('./logs_bec'):
21     shutil.rmtree('./logs_bec')
22     print("Cleaned up ./logs_bec directory.")

```

Listing 67: تقييم نموذج BERT المصقول.

المختصرات والمصطلحات

AI الذكاء الاصطناعي Artificial Intelligence: مجال واسع من علوم الحاسوب يركز على إنشاء أنظمة يمكنها أداء مهام تتطلب عادةً ذكاءً بشرياً، مثل فهم اللغة والاستدلال.

Mask Attention قناع الانتباه Attention Mask: مكون في نماذج المحولات Trans-former models يُعلم النموذج بأي أجزاء من المدخلات (الرموز tokens) هي بيانات حقيقية وأيها مجرد حشو padding، مما يضمن تركيز النموذج فقط على المحتوى الفعلي.

BEC اختراق البريد الإلكتروني للأعمال Business Email Compromise: هجوم إلكتروني cyberattack موجه ينتحل فيه المحتال شخصية مسؤول تنفيذي في شركة أو مورد موثوق به لخداع موظف للقيام بتحويل مالي احتيالي أو الكشف عن معلومات حساسة.

BERT تمثيلات التشفير ثنائية الاتجاه من المحولات Bidirectional Encoder Representations from Transformers: نموذج محولات Transformer قوي ومشهور مُدرَّب مسبقاً تم تطويره بواسطة جوجل، ويستخدم على نطاق واسع لمجموعة متنوعة من مهام معالجة اللغات الطبيعية NLP.

Embedding Context-aware التضمين المدرك للسياق Context-aware Embedding: متجه رقمي vector يمثل كلمة ويتغير بناءً على الكلمات المحيطة بها في الجملة. وهذا يسمح للنموذج بفهم المعاني المختلفة لنفس الكلمة (على سبيل المثال، "ضفة" النهر مقابل "مصرف" مالي).

CVE الثغرات الأمنية الشائعة ومواطن الانكشاف Common Vulnerabilities and Exposures: نظام موحد لتحديد وفهرسة الثغرات الأمنية في مجال الأمن السيبراني cybersecurity vulnerabilities التي يتم الكشف عنها علناً.

Text Deepfake النص المزيف بتقنية التزييف العميق Deepfake Text: نص مزيف ولكنه مقنع للغاية تم إنشاؤه بواسطة نموذج ذكاء اصطناعي AI model، وهو مصمم للاستخدام في أنشطة خبيثة مثل الأخبار الكاذبة أو عمليات الاحتيال أو الهندسة الاجتماعية social engineering.

Fine-tuning الضبط الدقيق Fine-tuning: عملية أخذ نموذج ذكاء اصطناعي AI model مُدرَّب مسبقاً للأغراض العامة وتدريبه بشكل إضافي على مجموعة بيانات dataset أصغر ومحددة لتخصيصه لمهمة معينة، مثل اكتشاف رسائل التصيد الاحتيالي phishing emails.

IOC مؤشر الاختراق Indicator of Compromise: دليل رقمي، مثل عنوان IP ضار، أو بصمة ملف file hash، أو اسم نطاق، يشير إلى احتمال حدوث خرق أمني security breach.

LLM نموذج لغوي كبير Large Language Model: نموذج ذكاء اصطناعي AI model

متقدم، يحتوي عادةً على مليارات من المعلمات parameters، تم تدريبه على كميات هائلة من البيانات النصية لفهم وإنشاء لغة شبيهة باللغة البشرية.

MISP منصة مشاركة معلومات البرمجيات الخبيثة Malware Information Sharing Platform: حل برمجي مفتوح المصدر open-source لجمع ومشاركة وتخزين وربط مؤشرات الاختراق Indicators of Compromise وغيرها من معلومات التهديدات الاستخباراتية threat intelligence.

MITRE ATT&CK إطار MITRE ATT&CK: قاعدة معرفية knowledge base متاحة عالمياً لتكتيكات وتقنيات الخصوم بناءً على ملاحظات من العالم الحقيقي، وتستخدم لنمذجة التهديدات threat modeling والتحليل الأمني.

NER التعرف على الكيانات المسماة Named Entity Recognition: مهمة محددة في معالجة اللغات الطبيعية NLP حيث يتم تدريب نموذج على تحديد وتصنيف أجزاء رئيسية من المعلومات (الكيانات entities) في النص، مثل الأسماء أو التواريخ أو المواقع أو مؤشرات الاختراق IOCs.

NLP معالجة اللغات الطبيعية Natural Language Processing: أحد مجالات الذكاء الاصطناعي AI التي تمنح أجهزة الحاسوب القدرة على قراءة وفهم وتفسير وإنشاء لغة بشرية.

NVD قاعدة بيانات الثغرات الوطنية National Vulnerability Database: مستودع حكومة الولايات المتحدة لبيانات إدارة الثغرات vulnerability management، والذي يعتمد على قائمة CVE ويتزامن معها.

Engineering Prompt هندسة الأوامر النصية Prompt Engineering: تقنية تصميم تعليمات نصية (أمر نصي prompt) بعناية لتوجيه نموذج لغوي كبير large language model لأداء مهمة محددة دون الحاجة إلى إعادة تدريب النموذج نفسه أو ضبطه بدقة.

Sigma سيجما Sigma: تنسيق توقيعات signature عام ومفتوح لوصف أحداث السجل log events، مما يسمح لمحللي الأمن بكتابة قواعد كشف detection rules يمكن استخدامها عبر أنظمة مختلفة لإدارة السجلات وأنظمة SIEM.

Embedding Text تضمين النص Text Embedding: تمثيل رقمي لجزء من النص (مثل كلمة أو جملة) على شكل متجه vector. الكلمات ذات المعاني المتشابهة لها تمثيلات متجهة متشابهة.

Tokenizer المُرمِّز Tokenizer: أداة تقوم بتقسيم جزء من النص إلى وحدات أصغر تسمى الرموز tokens (كلمات أو أجزاء من الكلمات) وتحولها إلى معرفات رقمية يمكن لنموذج الذكاء الاصطناعي AI model معالجتها.

Transformer المحول Transformer: بنية نموذج ذكاء اصطناعي AI model architecture حديثة وقوية تتفوق في فهم السياق في البيانات المتسلسلة sequential data، مثل النصوص، عن طريق تقدير أهمية الكلمات المختلفة في الجملة.

YARA يارا YARA: أداة تستخدم لإنشاء قواعد (أو توقيعات signatures) لتحديد وتصنيف عينات البرمجيات الخبيثة malware من خلال البحث عن أنماط نصية أو ثنائية محددة.

باب 10

نماذج التسلسل Sequence Models لكشف الشذوذ الزمني Temporal Anomaly Detection وتحليلات سلوك المستخدم User Behavior Analytics (UBA)

مقدمة

في مجال الأمن السيبراني الحديث، نادراً ما يقوم المهاجمون بتنفيذ إجراء واحد وواضح. بدلاً من ذلك، ينفذون سلسلة من الخطوات الدقيقة على مر الزمن لتحقيق أهدافهم. قد يبدو حدث واحد، مثل الوصول إلى ملف أو اتصال بالشبكة، حميداً عند النظر إليه بشكل منفصل. ولكن، عند عرضه كجزء من تسلسل، يمكن أن يكشف عن نمط أكبر وخبيث. يستكشف هذا الفصل كيف يمكن لنماذج التسلسل sequence models، وهي فئة قوية من خوارزميات الذكاء الاصطناعي، تحليل البيانات الزمنية للكشف عن هذه التهديدات الخفية.

سنركز على تحليلات سلوك المستخدم (UBA) User Behavior Analytics، وهو مجال حيوي مخصص لفهم الأنشطة المشبوهة للمستخدمين والإبلاغ عنها. سوف نتعلم عن نماذج التسلسل المتقدمة مثل الذاكرة طويلة قصيرة المدى Long Short-Term Memory (LSTM)، ووحدات التكرار المسورة Gated Recurrent Units (GRUs)، والشبكات

العصبونية الالتفافية الزمنية (TCNs) Temporal Convolutional Networks، والمحولات Transformers. سنقدم شرحاً تفصيلياً لكيفية إعداد بيانات سجلات الأمان لهذه النماذج. أخيراً، سيرشدك هذا الفصل خلال مشروع عملي باستخدام PyTorch لبناء نموذج LSTM يكتشف "السفر المستحيل" impossible travel—وهو علامة كلاسيكية على حساب مستخدم مخترق.

1.10 المشكلة: تحديد الأنماط المشبوهة عبر الزمن

الهدف الأساسي للتحليل الزمني في الأمن السيرياني هو الكشف عن الأنشطة الخبيثة التي تتكشف كتسلسل من الأحداث. غالباً ما تكون هذه الأنشطة دقيقة وموزعة عبر مصادر سجلات مختلفة. ولا تظهر بوضوح إلا عند تحليل الأنماط على مدى فترة زمنية. غالباً ما تفشل أدوات الأمان التقليدية التي تفحص الأحداث بشكل فردي في اكتشاف هذه الهجمات متعددة المراحل. اثنان من أكثر التهديدات صعوبة التي تتطلب تحليلاً زمنياً هما التهديدات الداخلية insider threats والحسابات المخترقة compromised accounts.

1.1.10 Insider Threats التهديدات الداخلية

يشير التهديد الداخلي insider threat إلى الإجراءات الخبيثة التي يقوم بها أفراد لديهم وصول مصرح به إلى أنظمة وبيانات المؤسسة. يمكن أن يكون هؤلاء الأفراد موظفين حاليين أو سابقين، أو مقاولين، أو شركاء. نظراً لأن المطلعين لديهم بالفعل وصول شرعي، يمكنهم تجاوز العديد من التدابير الأمنية القياسية مثل جدران الحماية firewalls وضوابط الوصول access controls. قد تشمل أنشطتهم الخبيثة سرقة البيانات الحساسة، أو تخريب الأنظمة، أو ارتكاب الاحتيال. يتطلب الكشف عن هذه التهديدات إنشاء خط أساس baseline للسلوك الطبيعي لكل مستخدم ثم تحديد الانحرافات الكبيرة عن هذا الخط الأساسي بمرور الوقت [12]. على سبيل المثال، الموظف الذي يبدأ فجأة في الوصول إلى كميات كبيرة من البيانات وتنزيلها من مشروع غير مكلف به قد يكون بصدد تسريب أسرار الشركة.

2.1.10 Compromised Accounts الحسابات المخترقة

إن الحساب المخترق compromised account هو حساب مستخدم، مثل حساب تسجيل دخول إلى الشبكة أو حساب تطبيق، استولى عليه مهاجم. يستخدم المهاجمون أساليب مختلفة مثل التصيد الاحتيالي phishing، وحشو بيانات الاعتماد credential stuffing، أو هجمات القوة

الغاشمة brute-force attacks لسرقة بيانات اعتماد المستخدمين الشرعيين. بمجرد سيطرتهم على الحساب، يمكنهم العمل بنفس الامتيازات التي يتمتع بها المستخدم الشرعي، مما يجعل من الصعب تمييز أفعالهم عن النشاط الطبيعي. يساعد تحليل التسلسل في الكشف عن الحسابات المخترقة من خلال تحديد الأنماط غير المعتادة للمستخدم الحقيقي. مثال على ذلك هو "السفر المستحيل" impossible travel، حيث يتم تسجيل دخول حساب مستخدم من نيويورك، وبعد دقائق فقط، من طوكيو—وهو أمر مستحيل مادياً ويشير بقوة إلى أن بيانات اعتماد الحساب قد سُرقَت.

2.10 التقنية: نماذج التسلسل المتقدمة للسلاسل الزمنية Time-Series

لتحليل تسلسلات الأحداث بفعالية، نحتاج إلى نماذج يمكنها فهم السياق وتذكر الأحداث الماضية. غالباً ما تواجه نماذج التعلم الآلي machine learning التقليدية صعوبة في ذلك. توفر نماذج التعلم العميق deep learning المتقدمة، المصممة خصيصاً للبيانات التسلسلية، حلاً قوياً.

1.2.10 الشبكات العصبونية المتكررة (LSTMs و GRUs)

إن الشبكات العصبونية المتكررة (Recurrent Neural Networks, RNNs) هي نوع من الشبكات العصبونية المصممة للتعامل مع البيانات التسلسلية. تحافظ هذه الشبكات على حالة داخلية، أو "ذاكرة"، مما يسمح لها بالاحتفاظ بمعلومات حول العناصر السابقة في التسلسل أثناء معالجة العنصر الحالي. هناك نوعان متقدمان من الشبكات العصبونية المتكررة RNNs فعالان بشكل خاص:

- الذاكرة طويلة قصيرة المدى (Long Short-Term Memory, LSTM): تستخدم نماذج LSTM آلية بوابات خاصة—تتألف من بوابات الإدخال والإخراج والنسيان—للتحكم في المعلومات التي يتم تخزينها وتحديثها وإزالتها من خلية الذاكرة الخاصة بها. تسمح هذه البنية بالتقاط الاعتماديات طويلة المدى long-term dependencies، مما يعني أنها تستطيع ربط الأحداث المتباعدة في التسلسل [33]. وهذا مثالي لنمذجة سلوك المستخدم المعقد على مدى أيام أو أسابيع.

- وحدة التكرار المسورة (Gated Recurrent Unit, GRU): تعد نماذج GRU نسخة مبسطة من نماذج LSTM. تستخدم بوابات أقل (بوابة تحديث وبوابة إعادة تعيين) ولكنها تحقق أداءً مشابهاً في العديد من المهام. وغالباً ما تكون أكثر كفاءة من الناحية الحسابية من نماذج LSTM.

2.2.10 الشبكات العصبونية الالتفافية الزمنية (TCNs)

إن الشبكات العصبونية الالتفافية الزمنية (Temporal Convolutional Networks, TCNs) هي فئة من الشبكات العصبونية الالتفافية مهيأة للبيانات التسلسلية. على عكس الشبكات العصبونية المتكررة RNNs التي تعالج البيانات خطوة بخطوة، يمكن لشبكات TCNs معالجة عناصر التسلسل بالتوازي. تستخدم هذه الشبكات تقنية تسمى الالتفافات المتوسعة-dilated convolutions، حيث يتم تطبيق المرشح على مساحة أكبر من طوله عن طريق تخطي قيم الإدخال بخطوة معينة. يسمح هذا للشبكة بامتلاك مجال استقبال receptive field كبير جداً—وهو مقدار السياق التاريخي الذي يمكنها "رؤيته"—دون الحاجة إلى شبكة عميقة جداً. غالباً ما تقدم شبكات TCNs أداءً وكفاءة ذاكرة أفضل من الشبكات العصبونية المتكررة RNNs، خاصة للتسلسلات الطويلة جداً.

3.2.10 نماذج التسلسل القائمة على المحولات Transformer

لقد أحدثت بنية المحول Transformer ثورة في نمذجة التسلسل، خاصة في معالجة اللغات الطبيعية. ابتكارها الأساسي هو آلية الانتباه الذاتي self-attention mechanism. تسمح هذه الآلية للنموذج بتقييم أهمية جميع العناصر الأخرى في التسلسل عند معالجة عنصر واحد. في مجال الأمن السيبراني، يعني هذا أن نموذج Transformer يمكنه ربط محاولة تسجيل دخول مشبوهة في بداية الأسبوع بوصول غير عادي إلى ملف في نهاية الأسبوع، بغض النظر عن الأحداث التي وقعت بينهما. هذه القدرة على التقاط العلاقات المعقدة وغير المحلية تجعل نماذج Transformers قوية للغاية لتحليل تسلسلات الأحداث الأمنية الطويلة والمعقدة.

4.2.10 التفصيل الخوارزمي: تحويل سجلات الأمان إلى جلسات Sessionizing

قبل أن نتكّن من إدخال بيانات الأمان إلى نموذج تسلسل، يجب علينا تحويل إدخالات السجلات الخام وغير المهيكلة إلى تنسيق مهيكل من تسلسلات الأحداث. غالباً ما تسمى هذه العملية تحويل إلى جلسات sessionizing. تتضمن هذه العملية تجميع الأحداث ذات الصلة في جلسات ذات معنى ليكن معين (مثل مستخدم أو مضيف). فيما يلي دليل خطوة بخطوة لهذه العملية [?].

(1) تحديد الكيانات الرئيسية: أولاً، حدد الكيانات التي تريد مراقبة سلوكها. تشمل الكيانات الشائعة معرفات المستخدمين، وعناوين IP للمضيفين، وأسماء العمليات، أو معرفات

التطبيقات. سيتم ربط كل تسلسل تقوم بإنشائه بأحد هذه الكيانات.

(2) تحديد حدود الجلسة: بعد ذلك، قم بتجميع أحداث الأمان الخام في "جلسات". الجلسة هي مجموعة من الأحداث ذات الصلة لكيان واحد. يمكنك تحديد حدود الجلسة بعدة طرق:

- قائمة على الزمن: تجميع كل الأحداث من مستخدم معين التي تحدث ضمن نافذة زمنية ثابتة، مثل 30 دقيقة أو ساعة واحدة.
- فواصل منطقية: يمكن تعريف الجلسة على أنها جميع الأحداث التي تقع بين تسجيل دخول المستخدم وتسجيل خروجه، أو بين بدء عملية وانتهائها.
- قائمة على النشاط: تجميع الأحداث التي تشكل جزءاً من نشاط منطقي واحد، مثل جميع اتصالات الشبكة التي بدأت بواسطة عملية معينة.

(3) تحويل الميزات إلى متجهات (لكل حدث): لكل حدث داخل الجلسة، قم بتحويل سماته إلى متجه ميزات رقمي. متجه الميزات feature vector هو قائمة من الأرقام تمثل الحدث. يتضمن ذلك تقنيات مثل الترميز الأحادي one-hot encoding للميزات الفئوية (مثل، `process\_name` وتحويل القيم المنطقية إلى أعداد صحيحة للإشارات (مثل، `login\_success` واستخراج القيم الرقمية (مثل، `source\_port`).

(4) إنشاء التسلسلات: قم بتجميع متجهات الميزات لكل حدث في الجلسة في تسلسل مرتب زمنياً. تكون النتيجة لكل جلسة مصفوفة ثنائية الأبعاد حيث تمثل الصفوف الخطوات الزمنية (الأحداث) وتمثل الأعمدة الميزات.

(5) الحشو/القطع (Padding/Truncation): تتطلب نماذج التسلسل أن تكون المدخلات ذات طول ثابت. لتحقيق ذلك، يجب عليك حشو التسلسلات القصيرة بالأصفار أو قطع التسلسلات الطويلة إلى طول أقصى محدد مسبقاً.

(6) التطبيع (Normalization): أخيراً، قم بتطبيق قياس الميزات على الميزات الرقمية عبر جميع الأحداث. إحدى الطرق الشائعة هي، `StandardScaler` التي تحول الميزات ليكون لها متوسط يساوي 0 وانحراف معياري يساوي 1. هذا يمنع الميزات ذات النطاقات القيمية الكبيرة من السيطرة على عملية التعلم.

3.10 التطبيقات

تمتلك نماذج التسلسل مجموعة واسعة من التطبيقات في مجال الأمن السيبراني، مما يتيح نهجاً استباقياً وذكياً للكشف عن التهديدات.

1.3.10 تحليلات سلوك المستخدم (UBA)

إن تحليلات سلوك المستخدم (User Behavior Analytics, UBA) هي عملية مراقبة أنشطة المستخدمين للكشف عن الحالات الشاذة التي قد تشير إلى تهديدات داخلية أو حسابات مخترقة. تقوم أنظمة UBA أولاً ببناء خط أساس baseline للسلوك "الطبيعي" لكل مستخدم ثم تقوم بالإبلاغ عن الانحرافات الكبيرة عن هذا الخط الأساسي. تعتبر نماذج التسلسل في صميم أنظمة UBA الحديثة، وتشمل حالات الاستخدام الرئيسية ما يلي:

- أنماط تسجيل الدخول غير العادية: الكشف عن "السفر المستحيل" impossible travel، وتسجيلات الدخول في أوقات غريبة، أو الوصول من أجهزة جديدة أو مواقع جغرافية جديدة.
- أنماط الوصول غير الطبيعية للملفات: تحديد مستخدم يصل إلى ملفات أو أدلة أو قواعد بيانات نادراً ما يتفاعل معها، أو يقوم بتنزيل حجم كبير بشكل غير عادي من البيانات.
- عمليات تنفيذ غريبة: الإبلاغ عن مستخدم يقوم بتشغيل أدوات النظام أو البرامج النصية أو الملفات التنفيذية التي ليست جزءاً من سير عمله المعتاد، مما قد يشير إلى نشاط خبيث.
- محاولات تصعيد الامتيازات: التعرف على سلسلة من الإجراءات التي تهدف إلى الحصول على حقوق وصول أعلى على النظام، والتي تنحرف عن خط الأساس لأذونات المستخدم العادية.

2.3.10 الكشف عن نشاط التهديدات المتقدمة المستمرة (APT)

إن التهديد المتقدم المستمر (Advanced Persistent Threat, APT) هو هجوم سيبراني طويل الأمد ومستهدف حيث يحصل متسلل على وصول غير مصرح به إلى شبكة ويبقى غير مكتشف لفترة طويلة. غالباً ما تتضمن هجمات APT نهج "البطء والهدوء" low and slow، باستخدام سلسلة من الإجراءات الدقيقة لتجنب الكشف. يمكن لنماذج التسلسل ربط هذه الأحداث منخفضة المستوى التي تبدو غير مترابطة من مصادر سجلات مختلفة (مثل سجلات

الشبكة، وسجلات نقاط النهاية، وسجلات التطبيقات). من خلال تحديد الروابط الزمنية والسببية بين الأحداث، يمكن لهذه النماذج إعادة بناء سرد الهجوم والإبلاغ عن الحملة بأكملها.

3.3.10 الكشف عن حقن العمليات Process Injection

إن حقن العمليات Process injection هو أسلوب شائع للبرامج الضارة يُستخدم لتجنب الكشف. يقوم المهاجم بحقن شفرة خبيثة في مساحة الذاكرة لعملية شرعية قيد التشغيل. من منظور نظام التشغيل، يبدو أن النشاط الخبيث ينشأ من عملية موثوقة. يمكن لنماذج التسلسل اكتشاف ذلك عن طريق تحليل تسلسلات استدعاءات النظام system calls أو عمليات الذاكرة. العملية الشرعية لها تسلسل متوقع من العمليات، ويمكن تحديد الانحراف الناجم عن الشفرة المحقونة على أنه شذوذ.

4.3.10 الكشف عن القيادة والتحكم (C2)

بعد أن يخترق المهاجم نظاماً ما، غالباً ما ينشئ اتصالاً بخادم القيادة والتحكم (Command and Control, C2) لتلقي الأوامر وتسريب البيانات. غالباً ما يتم تصميم اتصال C2 هذا ليكون متخفياً، باستخدام إشارات "نبضات قلب" heartbeat دورية ومنخفضة الحجم. يمكن لنماذج التسلسل تحليل حركة مرور الشبكة أو سجلات استعلامات DNS للتعرف على هذه الأنماط المنتظمة والمشفرة beaconing التي تميز اتصالات C2، حتى عندما يكون حجم حركة المرور منخفضاً.

4.10 مجموعات البيانات المفتوحة لتحليلات سلوك المستخدم UBA/التحليل الزمني

يتطلب تدريب وتقييم نماذج التسلسل مجموعات بيانات ذات صلة. في حين أن بيانات الشركات الحقيقية غالباً ما تكون حساسة، تتوفر العديد من مجموعات البيانات العامة والمفتوحة للبحث والتطبيق العملي.

- مجموعة بيانات اختبار التهديدات الداخلية من CERT (CMU): أنتج قسم CERT بجامعة كارنيجي ميلون العديد من مجموعات البيانات الاصطناعية التي تُمثل سيناريوهات مختلفة للتهديدات الداخلية، مثل تسريب البيانات والتخريب [?]. تحتوي مجموعات البيانات

هذه على سجلات أنشطة المستخدمين مثل تسجيلات الدخول، والوصول إلى الملفات، وتصفح الويب، وهي مصممة خصيصاً لاختبار خوارزميات UBA.

- بيانات السجلات المحاكاة: من الشائع للباحثين إنشاء مجموعات بيانات مخصصة تحاكي سجلات النظام أو التطبيقات أو الشبكات في العالم الحقيقي. ميزة البيانات المحاكاة هي القدرة على تضمين تسلسلات خبيثة محددة ومصنفة لإجراء تجارب محكمة وتقييم النماذج.
- بيانات القياس عن بعد المتاحة للعموم من نقاط النهاية: توفر سجلات نقاط النهاية بيانات تفصيلية حول نشاط النظام. تشمل المصادر سجلات Sysmon من أنظمة ويندوز أو سجلات auditd من لينكس. يمكن أيضاً أن تكون السجلات المجهولة من منصات الكشف والاستجابة لنقاط النهاية (Endpoint Detection and Response, EDR) ذات قيمة لبناء تسلسلات مفصلة لسلوك المستخدم والعمليات.
- سجلات بروكسي الشبكة الموثقة: تسجل هذه السجلات نشاط تصفح الويب للمستخدمين، بما في ذلك عناوين URL التي تمت زيارتها، والطابع الزمنية، ووكلاء المستخدم. وهي مفيدة للكشف عن أنماط التصفح المشبوهة، والوصول إلى مواقع الويب الخبيثة، أو محاولات تسريب البيانات عبر بروتوكول HTTP/S.

5.10 مشروع عملي: نماذج التسلسل لتحليلات سلوك المستخدم UBA

في هذا المشروع، سنقوم ببناء وتقييم نموذج LSTM باستخدام PyTorch للكشف عن أنماط تسجيل دخول المستخدمين الشاذة، مع التركيز على سيناريو "السفر المستحيل" impossible travel.

1.5.10 إعداد البيانات

أولاً، نحتاج إلى إعداد بياناتنا. سنقوم بإنشاء مجموعة بيانات محاكاة لتسجيلات دخول المستخدمين باستخدام `pandas`. ستحتوي مجموعة البيانات هذه على طابع زمنية ومعرفات المستخدمين ومواقع تسجيل الدخول. سندخل يدوياً حدث "سفر مستحيل" impossible travel لمستخدم واحد. بعد ذلك، سنقوم بمعالجة هذه البيانات إلى تسلسلات مرتبة زمنياً من الميزات (الموقع والفرق الزمني بين تسجيلات الدخول) لكل مستخدم، لإعدادها كمدخل لنموذج LSTM الخاص بنا. يوضح الكود في القائمة 68 هذه العملية.

2.5.10 تعريف نموذج LSTM

بعد ذلك، نحدد نموذج LSTM الخاص بنا باستخدام `nn.Module` من PyTorch. سيتألف هذا النموذج من طبقة LSTM تليها طبقة خطية متصلة بالكامل. تعالج طبقة LSTM تسلسلات الإدخال وتتعلم الأنماط الزمنية لسلوك تسجيل الدخول الطبيعي. يتم تمرير الحالة المخفية النهائية لطبقة LSTM إلى الطبقة الخطية، التي تصدر قرار تصنيف: 0 للتسلسل الطبيعي و 1 للتسلسل الذي يحتوي على سفر مستحيل. توضح القائمة 69 تنفيذ النموذج.

3.5.10 التدريب والكشف عن الشذوذ

بعد إعداد بياناتنا وتعريف نموذجنا، يمكننا المتابعة إلى التدريب. سندرب نموذج LSTM كمصنف خاضع للإشراف. سيتعلم النموذج التمييز بين أنماط تسلسلات تسجيل الدخول العادية والتسلسل الشاذ الذي يحتوي على حدث السفر المستحيل. سنستخدم دالة `CrossEntropyLoss` ومحسن `Adam`. بعد التدريب، سنقوم بتقييم أداء النموذج على مجموعة اختبار منفصلة لثرى مدى قدرته على تحديد سلوك المستخدم الشاذ. يظهر كود التدريب والتقييم في القائمة 70.

4.5.10 إدارة انحراف المفهوم Concept Drift

أحد التحديات الرئيسية في تحليلات سلوك المستخدم UBA هو انحراف المفهوم concept drift، والذي يحدث عندما يتغير السلوك الطبيعي للمستخدم بمرور الوقت. على سبيل المثال، قد ينتقل المستخدم إلى مدينة جديدة، أو يحصل على جهاز عمل جديد، أو يغير أدواره داخل الشركة. إذا لم يتم تحديث نموذج UBA، فسيبدأ في توليد نتائج إيجابية كاذبة false positives عن طريق الإبلاغ عن هذا السلوك الجديد والشرعي على أنه شاذ. يجب أن تكون لدينا استراتيجيات لإدارة انحراف المفهوم:

- إعادة التدريب الدورية: النهج الأكثر مباشرة هو إعادة تدريب النموذج بانتظام (على سبيل المثال، أسبوعياً أو شهرياً) باستخدام البيانات الحديثة. يسمح هذا للنموذج بتعلم خط الأساس السلوكي الجديد للمستخدم.
- التعلم عبر الإنترنت (Online Learning): بدلاً من إعادة التدريب الكاملة، يمكننا استخدام طرق التعلم عبر الإنترنت لتحديث أوزان النموذج بشكل تدريجي عند وصول بيانات جديدة. هذا أكثر كفاءة من الناحية الحسابية ويسمح بالتكيف المستمر.

- مراقبة العمليات الإحصائية: يمكننا تطبيق الأساليب الإحصائية لمراقبة أداء النموذج أو توزيع البيانات الواردة. يمكن أن يشير تغيير كبير في معدل خطأ النموذج أو الخصائص الإحصائية للبيانات إلى انحراف المفهوم ويؤدي إلى بدء عملية إعادة تدريب النموذج.
- العتبات التكميلية: يمكن جعل العتبة المستخدمة لتصنيف السلوك على أنه شاذ ديناميكية. بدلاً من قيمة ثابتة، يمكن تعديل العتبة بناءً على متوسط متحرك لسلوك المستخدم الأخير أو درجات النموذج، مما يجعلها أكثر مرونة في مواجهة التغييرات التدريجية.

5.5.10 التصوير المرئي

أخيراً، يعد التصوير المرئي لمخرجات النموذج أمراً بالغ الأهمية لحلل الأمان. يمكن أن يوفر التمثيل المرئي سياقاً فورياً ويساعد في التحقق من التنبيه. يمكننا استخدام `matplotlib` لرسم مواقع تسجيل دخول المستخدم على خريطة بمرور الوقت، مع إبراز الأحداث التي أبلغ عنها النموذج بصرياً على أنها "سفر مستحيل" impossible travel. يتيح هذا للمحلل فهم طبيعة الشذوذ بسرعة. توفر القائمة 71 الكود لإنشاء مثل هذا الرسم البياني.

الخاتمة

أوضح هذا الفصل الدور الحاسم لنماذج التسلسل في الأمن السيبراني الحديث للكشف عن التهديدات التي تظهر بمرور الوقت. لقد استكشفنا كيف يمكن لنماذج مثل Trans- و LSTMs و LSTMers تعلم الأنماط الزمنية للسلوك الطبيعي لتحديد الهجمات المتطورة والتهديدات الداخلية والحسابات المخترقة. لقد تعلمت التقنية الأساسية لإعداد البيانات المتمثلة في تحويل سجلات الأمان إلى جلسات sessionizing لتحويل الأحداث الخام إلى تسلسلات مهيكلية مناسبة لهذه النماذج. قدم المشروع العملي مثالاً تطبيقياً لبناء نظام تحليلات سلوك المستخدم UBA للكشف عن "السفر المستحيل" impossible travel، وهو مؤشر شائع على وجود حساب مخترق. سلط هذا التمرين الضوء على سير العمل الكامل، من محاكاة البيانات وهندسة الميزات إلى تدريب النموذج وتقييمه وتصويره المرئي. من خلال إتقان هذه التقنيات، يمكن لمختصي الأمن السيبراني بناء أنظمة دفاع أكثر ذكاءً واستباقية قادرة على كشف التكتيكات الدقيقة والتسلسلية التي يستخدمها المهاجمون اليوم.

```

1  import pandas as pd
2  import numpy as np
3  from sklearn.preprocessing import LabelEncoder
4  from torch.utils.data import TensorDataset, DataLoader
5  import torch
6  from sklearn.model_selection import train_test_split
7  from collections import Counter
8
9  # Create dummy user login data
10 np.random.seed(42)
11 num_logins = 500
12 users = ['userA', 'userB', 'userC']
13 locations = ['New York', 'London', 'Tokyo', 'San Francisco', 'Paris']
14 data_logins = pd.DataFrame({
15     'timestamp': pd.to_datetime('2025-01-01') + pd.to_timedelta(np.sort(np.random.rand(num_logins) * 30), unit='D'),
16     'user_id': np.random.choice(users, num_logins),
17     'login_location': np.random.choice(locations, num_logins)
18 })
19 data_logins = data_logins.sort_values(by=['user_id', 'timestamp']).reset_index(drop=True)
20
21 # Simulate 'impossible travel' for userA
22 # Login from New York, then shortly after from Tokyo
23 data_logins = pd.concat([
24     data_logins,
25     pd.DataFrame([
26         {'timestamp': pd.to_datetime('2025-01-05 10:00:00'), 'user_id': 'userA', 'login_location': 'New York'},
27         {'timestamp': pd.to_datetime('2025-01-05 10:01:00'), 'user_id': 'userA', 'login_location': 'Tokyo'} # Impossible travel
28     ])
29 ]).sort_values(by=['user_id', 'timestamp']).reset_index(drop=True)
30
31 # Encode locations numerically
32 le_location = LabelEncoder()
33 data_logins['location_encoded'] = le_location.fit_transform(data_logins['login_location'])
34
35 # Calculate time difference between consecutive logins for each user
36 data_logins['prev_timestamp'] = data_logins.groupby('user_id')['timestamp'].shift(1)
37 data_logins['time_diff_hours'] = (data_logins['timestamp'] - data_logins['prev_timestamp']).dt.total_seconds() / 3600
38 data_logins['time_diff_hours'] = data_logins['time_diff_hours'].fillna(0) # First login in sequence has 0 diff
39
40 # Create sequence features for each user
41 user_sequences = []
42 user_labels = [] # Label for impossible travel (1 if impossible, 0 if possible)
43 MAX_SEQ_LEN = 10 # Maximum number of login events per sequence
44
45 for user, group in data_logins.groupby('user_id'):
46     # Create sequence of (location_encoded, time_diff_hours)
47     seq_features = group[['location_encoded', 'time_diff_hours']].values
48     labels = np.zeros(len(group)) # Default to normal
49     # Mark 'impossible travel' as 1
50     idx = group[(group['login_location'] == 'New York') &
51                (group['timestamp'] == pd.to_datetime('2025-01-05 10:00:00'))].index
52     if not idx.empty:
53         idx_in_group = group.index.get_loc(idx)
54         if idx_in_group + 1 < len(group): # Check if there's a next login
55             if group.iloc[idx_in_group+1]['login_location'] == 'Tokyo' and \
56                group.iloc[idx_in_group+1]['time_diff_hours'] < 24: # Less than 24 hours
57                 labels[idx_in_group+1] = 1 # Mark the Tokyo login as impossible travel
58
59     # Pad or truncate sequences
60     if len(seq_features) > MAX_SEQ_LEN:
61         seq_features = seq_features[:MAX_SEQ_LEN]
62         labels = labels[:MAX_SEQ_LEN]
63     elif len(seq_features) < MAX_SEQ_LEN:
64         pad_len = MAX_SEQ_LEN - len(seq_features)
65         seq_features = np.vstack([seq_features, np.zeros((pad_len, seq_features.shape))])
66         labels = np.pad(labels, (0, pad_len), 'constant', constant_values=0)
67
68     user_sequences.append(seq_features)
69     # Classify sequence as 1 if it contains impossible travel
70     user_labels.append(1 if np.sum(labels) > 0 else 0)
71
72 X_seq_uba = np.array(user_sequences)
73 y_seq_uba = np.array(user_labels)
74
75 print(f"\nShape of X_seq_uba (num_users, 205, seq_len, num_features): {X_seq_uba.shape}")
76 print(f"Shape of y_seq_uba: {y_seq_uba.shape}")
77 print(f"User Sequence Labels Distribution: {Counter(y_seq_uba)}")
78
79 # Convert to PyTorch tensors
80 X_tensor_uba = torch.tensor(X_seq_uba, dtype=torch.float32)
81 y_tensor_uba = torch.tensor(y_seq_uba, dtype=torch.long)
82
83 # Split into train/test
84 X_train_uba, X_test_uba, y_train_uba, y_test_uba = train_test_split(
85     X_tensor_uba, y_tensor_uba, test_size=0.3, random_state=42, stratify=y_tensor_uba
86 )

```

```

1  import torch.nn as nn
2  import torch.nn.functional as F
3
4  input_size_uba = X_train_uba.shape # Number of features per event
5  hidden_size_uba = 32
6  num_layers_uba = 2
7  output_size_uba = 2 # 0: Normal, 1: Impossible Travel
8
9  class ImpossibleTravelLSTM(nn.Module):
10     def __init__(self, input_size, hidden_size, num_layers, output_size):
11         super(ImpossibleTravelLSTM, self).__init__()
12         self.hidden_size = hidden_size
13         self.num_layers = num_layers
14         self.lstm = nn.LSTM(input_size, hidden_size, num_layers, batch_first=True)
15         self.fc = nn.Linear(hidden_size, output_size)
16
17     def forward(self, x):
18         # Initialize hidden and cell states
19         h0 = torch.zeros(self.num_layers, x.size(0), self.hidden_size).to(x.device)
20         c0 = torch.zeros(self.num_layers, x.size(0), self.hidden_size).to(x.device)
21
22         # Pass input through LSTM
23         out, (hn, cn) = self.lstm(x, (h0, c0))
24
25         # Use the hidden state of the last time step for classification
26         out = self.fc(out[:, -1, :])
27         return out
28
29 model_uba_lstm = ImpossibleTravelLSTM(input_size_uba, hidden_size_uba, num_layers_uba, output_size_uba)
30 print("\nImpossible Travel LSTM Model Architecture:\n", model_uba_lstm)

```

Listing 69: تعريف نموذج LSTM بلغة PyTorch لتصنيف تسلسل تسجيلات الدخول.

```

1  import torch.optim as optim
2  from sklearn.metrics import classification_report, accuracy_score
3
4  device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
5  model_uba_lstm.to(device)
6  print(f"\nUsing device for UBA LSTM training: {device}")
7
8  criterion_uba = nn.CrossEntropyLoss()
9  optimizer_uba = optim.Adam(model_uba_lstm.parameters(), lr=0.005)
10
11 num_epochs_uba = 100
12 print("\n--- Starting PyTorch LSTM UBA Training ---")
13 for epoch in range(num_epochs_uba):
14     model_uba_lstm.train()
15     running_loss = 0.0
16     for batch_idx, (inputs, labels) in enumerate(train_loader_uba):
17         inputs, labels = inputs.to(device), labels.to(device)
18         optimizer_uba.zero_grad()
19         outputs = model_uba_lstm(inputs)
20         loss = criterion_uba(outputs, labels)
21         loss.backward()
22         optimizer_uba.step()
23         running_loss += loss.item()
24
25     if (epoch + 1) % 20 == 0:
26         print(f"Epoch [{epoch+1}/{num_epochs_uba}], Loss: {running_loss / len(train_loader_uba):.4f}")
27
28 print("\nPyTorch LSTM UBA Training complete.")
29
30 # Evaluation on test set
31 model_uba_lstm.eval()
32 all_preds_uba = []
33 all_labels_uba = []
34
35 with torch.no_grad():
36     for inputs, labels in test_loader_uba:
37         inputs, labels = inputs.to(device), labels.to(device)
38         outputs = model_uba_lstm(inputs)
39         _, predicted = torch.max(outputs.data, 1)
40         all_preds_uba.extend(predicted.cpu().numpy())
41         all_labels_uba.extend(labels.cpu().numpy())
42
43 print("\n--- UBA LSTM Model Evaluation on Test Set ---")
44 print(f"Test Accuracy: {accuracy_score(all_labels_uba, all_preds_uba):.4f}")
45 print("\nClassification Report:\n", classification_report(all_labels_uba, all_preds_uba, target_names=['Normal', 'Impossible']

```

Listing 70: تدريب نموذج UBA LSTM وتقييم أدائه.

```

1 import matplotlib.pyplot as plt
2
3 # Re-run prediction on the whole dataset to get results for plotting all points
4 df_viz = data_logins.copy()
5 model_uba_lstm.eval()
6 with torch.no_grad():
7     all_outputs = model_uba_lstm(X_tensor_uba.to(device))
8     _, all_predicted = torch.max(all_outputs.data, 1)
9
10 # Map user IDs to their sequence prediction
11 user_ids_in_order = data_logins['user_id'].unique()
12 user_to_prediction_map = {user: pred.item() for user, pred in zip(user_ids_in_order, all_predicted)}
13
14 # Apply this mapping to the visualization dataframe
15 df_viz['predicted_label'] = df_viz['user_id'].map(user_to_prediction_map)
16
17 # Filter for userA to visualize (as we manually injected impossible travel there)
18 df_userA = df_viz[df_viz['user_id'] == 'userA'].copy()
19
20 # Define location coordinates for plotting
21 location_coords = {
22     'New York': (40.71, -74.00), 'London': (51.50, -0.12),
23     'Tokyo': (35.68, 139.65), 'San Francisco': (37.77, -122.41), 'Paris': (48.85, 2.35)
24 }
25
26 # Get coordinates for plotting
27 df_userA['lat'] = df_userA['login_location'].map(lambda x: location_coords[x])
28 df_userA['lon'] = df_userA['login_location'].map(lambda x: location_coords[x])
29
30 plt.figure(figsize=(12, 8))
31 plt.plot(df_userA['lon'], df_userA['lat'], 'b-o', markersize=6, label='UserA Login Path')
32
33 # Highlight impossible travel predictions
34 # If userA's sequence was flagged, all their logins will be highlighted.
35 if df_userA['predicted_label'].iloc == 1:
36     plt.plot(df_userA['lon'], df_userA['lat'], 'rx', markersize=10, mew=3, linestyle='None', label='Predicted Impossible Travel')
37
38 for i, row in df_userA.iterrows():
39     plt.text(row['lon'], row['lat'], f"{row['login_location']} ({row['timestamp'].strftime('%H:%M')})", fontsize=8, ha='right')
40
41 plt.xlabel('Longitude')
42 plt.ylabel('Latitude')
43 plt.title('UserA Login Trajectory with Impossible Travel Detection')
44 plt.grid(True)
45 plt.legend()
46 plt.show()

```

71 Listing: تصوير مرئي لمواقع تسجيل دخول المستخدم وتسليط الضوء على السفر المستحيل المتوقع.

المختصرات والمصطلحات

AI الذكاء الاصطناعي Artificial Intelligence: هو مجال في علوم الحاسوب مخصص لإنشاء أنظمة يمكنها أداء المهام التي تتطلب عادةً ذكاءً بشرياً، مثل التعلم من البيانات وتحديد الأنماط.

APT التهديد المتقدم المستمر Advanced Persistent Threat: هو هجوم إلكتروني cyberattack طويل الأمد وموجه، حيث يتمكن المهاجم من الوصول غير المصرح به إلى شبكة ما ويبقى دون أن يتم اكتشافه لفترة طويلة.

C2 القيادة والتحكم Command and Control: هو خادم أو نظام يستخدمه المهاجم لإرسال الأوامر إلى أجهزة الحاسوب المخترقة واستلام البيانات المسروقة.

CERT فريق الاستجابة لطوارئ الحاسوب Computer Emergency Response Team: هو مجموعة من الخبراء تتولى التعامل مع حوادث أمن الحاسوب. يشير النص إلى الفريق في جامعة كارنيجي ميلون (CMU) المعروف بإنشاء مجموعات بيانات حول التهديد الداخلي insider threat.

Account Compromised الحساب المخترق Compromised Account: هو حساب مستخدم شرعي تم الاستيلاء عليه من قبل مهاجم غير مصرح له سرقة بيانات اعتماد المستخدم.

Drift Concept انحراف المفهوم Concept Drift: هي مشكلة في تعلم الآلة machine learning حيث تتغير الخصائص الإحصائية للمتغير المستهدف (مثل سلوك المستخدم الطبيعي) بمرور الوقت، مما يؤدي إلى أن يصبح النموذج المدرب أقل دقة.

Dilated Convolutions الالتفافات المتوسعة Dilated Convolutions: هي تقنية مستخدمة في الشبكات العصبونية الالتفافية الزمنية Temporal Convolutional Networks (TCNs) تسمح للنموذج بتحليل المعلومات من تاريخ طويل من الأحداث دون الحاجة إلى بنية شبكية عميقة جداً.

EDR كشف نقاط النهاية والاستجابة لها Endpoint Detection and Response: هو حل للأمن السيبراني cybersecurity يراقب بشكل مستمر أجهزة المستخدمين النهائيين (مثل أجهزة الكمبيوتر المحمولة والخوادم) للكشف عن التهديدات السيبرانية والاستجابة لها.

Vector Feature متجه السمات Feature Vector: هو قائمة من الأرقام تمثل عددياً حدثاً واحداً أو نقطة بيانات، مما يجعلها مفهومة لنموذج تعلم الآلة machine learning.

GRU وحدة البوابات المتكررة Gated Recurrent Unit: هي نوع من الشبكات العصبونية المتكررة (Recurrent Neural Network (RNN، تشبه ذاكرة طويلة قصيرة المدى LSTM، وهي مصممة لمعالجة تسلسل البيانات ولكن ببنية أبسط، مما يجعلها فعالة من الناحية الحسابية.

Travel Impossible السفر المستحيل Impossible Travel: هو شذوذ أمني كلاسيكي حيث يتم تسجيل دخول حساب مستخدم من موقعين جغرافيين متباعدين في إطار زمني قصير جداً لا يسمح بالسفر الفعلي المشروع.

Threat Insider التهديد الداخلي Insider Threat: هو خطر أمني يأتي من فرد داخل المنظمة، مثل موظف أو مقاول، لديه وصول مصرح به إلى الأنظمة والبيانات.

LSTM ذاكرة طويلة قصيرة المدى Long Short-Term Memory: هي نوع متقدم من الشبكات العصبونية المتكررة (Recurrent Neural Network (RNN تستخدم آلية بوابات خاصة لتعلم وتذكر الأنماط عبر تسلسلات طويلة من الأحداث.

Normalization التسوية Normalization: هي عملية تحجيم السمات الرقمية في مجموعة بيانات إلى نطاق قياسي (على سبيل المثال، متوسط 0 وانحراف معياري 1) لتحسين أداء واستقرار نموذج تعلم الآلة machine learning.

Padding/Truncation الحشو/القطع Padding/Truncation: هي تقنية لجعل جميع تسلسلات البيانات ذات طول موحد إما عن طريق إضافة قيم حشو (padding) إلى التسلسلات الأقصر أو قطع العناصر (truncating) من التسلسلات الأطول.

Injection Process حقن العمليات Process Injection: هي تقنية خبيثة يقوم فيها المهاجم بإدخال رمز برمجي في ذاكرة عملية شرعية قيد التشغيل لإخفاء نشاطه وتجنب الكشف.

Field Receptive الحقل الاستقبالي Receptive Field: في سياق الشبكات العصبونية الالتفافية الزمنية TCNs، يشير هذا إلى المدى الذي يمكن للنموذج أن ينظر به إلى الوراء في تسلسل ما لعمل تنبؤ للخطوة الزمنية الحالية.

RNN الشبكة العصبونية المتكررة Recurrent Neural Network: هي فئة من الشبكات العصبونية neural networks مصممة للعمل مع البيانات التسلسلية، مثل السلاسل الزمنية أو النصوص، عن طريق الحفاظ على حالة داخلية أو ذاكرة للمعلومات السابقة.

Mechanism Self-attention آلية الانتباه الذاتي Self-attention Mechanism: هي المكون الأساسي لنموذج المحولات Transformer model الذي يسمح له بتقييم أهمية

جميع العناصر الأخرى في التسلسل عند معالجة عنصر واحد، مما يمكنه من التقاط التبعيات المعقدة وطويلة المدى.

Sessionizing تكوين الجلسات Sessionizing: هي عملية تحضير البيانات التي يتم فيها تجميع أحداث السجل الأولية والفردية في تسلسلات ذات معنى (أو "جلسات") مرتبطة بكيان معين، مثل مستخدم أو مضيف.

TCN الشبكة العصبونية الالتفافية الزمنية Temporal Convolutional Network: هي نوع من الشبكات العصبونية neural network تطبق طبقات التفافية convolutional layers (تستخدم عادةً لمعالجة الصور) على بيانات التسلسل، وغالباً ما تحقق أداءً وكفاءة عالية.

Transformer المحول Transformer: هي بنية شبكة عصبونية ar-neural network chitecture حديثة، مصممة في الأصل لمعالجة اللغات الطبيعية natural language processing، تستخدم آلية الانتباه الذاتي self-attention mechanism لمعالجة تسلسلات كاملة مرة واحدة، مما يجعلها فعالة للغاية في التقاط العلاقات المعقدة في البيانات.

UBA تحليلات سلوك المستخدم User Behavior Analytics: هو مجال في الأمن السيبراني cybersecurity يركز على كشف التهديدات من خلال مراقبة نشاط المستخدمين وتحديد الانحرافات الكبيرة عن أنماط سلوكهم الطبيعية الراجعة.

باب 11

الطرق غير الخاضعة للإشراف لاكتشاف هجمات يوم الصفر والهجمات الجديدة

مقدمة

أهلاً بكم في طليعة أنظمة الدفاع الذكية. ركزت الفصول السابقة على التعلم الخاضع للإشراف supervised learning، وهو نهج قوي يتطلب بيانات مصنفة للتعلم من الهجمات السابقة. ومع ذلك، فإن أخطر التهديدات غالباً ما تكون تلك التي لم نرها من قبل. تُعرف هذه بالهجمات من نوع يوم الصفر zero-day attacks. إنها تستغل الثغرات الأمنية غير المعروفة، وبطبيعة الحال، ليس لدينا توقعات محددة مسبقاً أو أمثلة مصنفة لها. يتناول هذا الفصل هذا التحدي الحاسم بشكل مباشر.

سوف نستكشف طرق التعلم غير الخاضعة للإشراف unsupervised learning meth-ods، وهي فئة من الخوارزميات المصممة للعثور على الأنماط الخفية والحالات الشاذة في البيانات دون الحاجة إلى أي تصنيفات. الهدف الأساسي هو بناء نموذج لما يشكل السلوك "الطبيعي" ثم تحديد أي انحرافات كبيرة عن هذا المعيار كتهديدات محتملة. سوف نتعلم عن مجموعة من الخوارزميات القوية، من الأساليب الإحصائية الكلاسيكية إلى الشبكات العصبية الحديثة، وستفهم تطبيقاتها العملية في مجال الأمن السيبراني. بنهاية هذا الفصل، ستكون قادراً على بناء مشروع عملي لاكتشاف الحالات الشاذة، والجمع بين الرؤى من نماذج متعددة، وتطوير استراتيجيات لإدارة التحدي الحتمي المتمثل في الإنذارات الكاذبة false positives.

1.11 المشكلة: اكتشاف التهديدات غير المعروفة

الهدف الأساسي للكشف عن التهديدات غير الخاضع للإشراف unsupervised threat detection هو تحديد الحالات الشاذة النادرة التي لم يسبق رؤيتها وسلوكيات الهجوم الجديدة تماماً. هذا أمر حيوي في مجال الأمن السيبراني لأن منفذي التهديدات يطورون تكتيكاتهم باستمرار لتجنب الكشف. الأنظمة التقليدية القائمة على التوقيعات signature-based systems، مثل برامج مكافحة الفيروسات، تعمل عن طريق مطابقة الأنماط المعروفة للتعليمات البرمجية الضارة. نماذج التعلم الآلي الخاضعة للإشراف supervised machine learning محدودة أيضاً لأنها لا تستطيع اكتشاف سوى الهجمات المماثلة لتلك الموجودة في بيانات تدريبها.

تتجاوز تهديدات يوم الصفر Zero-day threats هذه الدفاعات تحديداً لأنها جديدة. لا توجد توقيعات حالية، ولا توجد أمثلة مصنفة لتدريب نموذج خاضع للإشراف. هنا يصبح التعلم غير الخاضع للإشراف ضرورياً. فبدلاً من تعلم شكل الهجوم، نقوم بتعليم النموذج كيف يبدو سلوك النظام أو الشبكة أو المستخدم الطبيعي. يتم وضع علامة على أي نشاط يخرف بشكل كبير عن هذا الخط الأساسي المكتسب كتهديد محتمل. يتيح لنا هذا النهج اكتشاف الهجمات الجديدة التي قد تمر دون أن يلاحظها أحد.

2.11 مجموعة من الخوارزميات غير الخاضعة للإشراف

يقدم التعلم غير الخاضع للإشراف مجموعة أدوات متنوعة لاكتشاف الحالات الشاذة anomaly detection. تتعامل كل خوارزمية مع المشكلة من منظور مختلف، مما يجعلها مناسبة لأنواع مختلفة من البيانات والتحديات الأمنية. دعنا نستكشف بعضاً من أكثر التقنيات فعالية.

1.2.11 آلة متجه الدعم أحادية الفئة (OCSVM)

آلة متجه الدعم أحادية الفئة One-Class Support Vector Machine (OCSVM) هي خوارزمية غير خاضعة للإشراف تتعلم حدود القرار decision boundary حول نقاط البيانات الطبيعية [70]. أثناء التدريب، يتعرض النموذج فقط للبيانات التي تمثل السلوك الطبيعي. يعمل عن طريق تحويل البيانات إلى فضاء ميزات عالي الأبعاد high-dimensional feature space وإيجاد مستوى فائق hyperplane يفصل نقاط البيانات عن الأصل بأقصى هامش. أي نقطة بيانات جديدة تقع على الجانب الآخر من هذه الحدود الفاصلة تعتبر حالة شاذة anomaly.

هذا يجعلها فعالة للغاية في إنشاء "فقاعة" واقية حول النشاط الطبيعي، حيث يتم الإبلاغ عن أي شيء خارجها للتحقيق. في لغة بايثون، يتم تنفيذ ذلك في ``sklearn.svm.OneClassSVM``.

2.2.11 غابة العزل (Isolation Forest)

كما ناقشنا سابقاً، فإن خوارزمية غابة العزل Isolation Forest فعالة بشكل استثنائي في اكتشاف الحالات الشاذة anomaly detection، خاصة في مجموعات البيانات عالية الأبعاد high-dimensional datasets [46]. تعمل عن طريق بناء مجموعة ensemble من أشجار القرار decision trees. لكل شجرة، تختار عشوائياً ميزة ثم تختار عشوائياً قيمة تقسيم بين القيم القصوى والدنيا لتلك الميزة. الفكرة الأساسية هي أن الحالات الشاذة "قليلة ومختلفة"، مما يجعل عزلها عن بقية البيانات أسهل. النقاط التي تتطلب عدداً أقل من الأقسام العشوائية لعزلها تحصل على درجة شذوذ anomaly score أعلى. هذه الطريقة فعالة من حيث الحساب ولا تعتمد على مقاييس المسافة أو الكثافة، مما يجعلها قوية. توفر مكتبة `scikit-learn` هذه الخوارزمية في ``sklearn.ensemble.IsolationForest``.

3.2.11 خوارزميات التجميع (Clustering Algorithms)

صُممت خوارزميات التجميع Clustering algorithms لتجميع نقاط البيانات المتشابهة معاً. في سياق الأمن السيبراني، يمكننا أن نفترض أن نقاط البيانات الطبيعية ستشكل تجمعات clusters كثيفة، بينما ستكون الحالات الشاذة anomalies نقاطاً معزولة لا تنتمي إلى أي مجموعة أو بعيدة جداً عن مركز أي مجموعة.

خوارزمية K-means

تقوم خوارزمية K-means بتقسيم البيانات إلى K مجموعات clusters مميزة وغير متداخلة. تعمل عن طريق تعيين كل نقطة بيانات إلى المجموعة ذات المتوسط الأقرب، أو ما يسمى بالمركز centroid. غالباً ما يتم تحديد القيم المتطرفة outliers على أنها نقاط بيانات لها مسافة كبيرة إلى مركز المجموعة المخصصة لها. نهج آخر هو تحديد المجموعات الصغيرة جداً والمعزولة، حيث قد تمثل هذه مجموعة صغيرة من الأحداث الشاذة المنسقة. ``sklearn.cluster.KMeans`` هو التنفيذ القياسي.

خوارزمية DBSCAN

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) هي خوارزمية تجمع قوية تقوم بتجميع النقاط المتقاربة في مناطق عالية الكثافة [23]. تقوم بتمييز النقاط التي تقع بمفردها في مناطق منخفضة الكثافة على أنها قيم متطرفة outliers، والتي تسميها "ضوضاء" noise. إحدى المزايا الرئيسية لـ DBSCAN هي أنها لا تتطلب من المستخدم تحديد عدد المجموعات مسبقاً. هذا يجعلها مناسبة تماماً لاكتشاف الأحداث الشاذة التي لا تتوافق مع أي سلوك جماعي متوقع. وهي متاحة في `sklearn.cluster.DBSCAN`.

نماذج الخليط الغاوسي (GMM)

نموذج الخليط الغاوسي (GMM) Gaussian Mixture Model هو نموذج احتمالي-probabilistic model يفترض أن البيانات يتم إنشاؤها من مزيج من عدد محدود من التوزيعات الغاوسية Gaussian distributions (المعروفة أيضاً بالتوزيعات الطبيعية). يمثل كل مكون غاوسي مجموعة cluster. بعد ملائمة نموذج GMM مع البيانات الطبيعية، يمكننا حساب كثافة الاحتمال probability density لأي نقطة بيانات جديدة. النقاط التي لها كثافة احتمالية منخفضة جداً بموجب النموذج المدرب تعتبر حالات شاذة anomalies، لأنه من غير المرجح أن تكون قد نشأت من نفس التوزيعات مثل البيانات الطبيعية. يتم تنفيذ هذا في `sklearn.mixture.GaussianMixture`.

4.2.11 خوارزميات كشف الحالات الشاذة الكلاسيكية

إلى جانب نماذج التعلم الآلي السائدة، لا تزال العديد من الطرق الإحصائية الكلاسيكية والقائمة على المسافة فعالة للغاية في اكتشاف القيم المتطرفة outlier detection.

عامل القيم المتطرفة المحلي (LOF)

تقيس خوارزمية عامل القيم المتطرفة المحلي (LOF) Local Outlier Factor الانحراف الكثافة المحلية local density لنقطة بيانات بالنسبة لجيرانها [10]. تقوم بتمييز النقاط كقيم متطرفة outliers إذا كانت كثافتها أقل بكثير من جيرانها. هذا مفيد بشكل خاص لمجموعات البيانات حيث تكون الكثافة غير منتظمة، لأنه يسمح للخوارزمية بتحديد الحالات الشاذة المحلية التي قد تفوتها الطرق العالمية. توفر فئة `sklearn.neighbors.LocalOutlierFactor` تنفيذاً لهذه الخوارزمية.

الدرجة المعيارية (Z-score) / الانحراف المعياري

الدرجة المعيارية Z-score هي مقياس إحصائي بسيط ولكنه فعال لتحديد القيم المتطرفة -outliers في البيانات أحادية المتغير univariate data (بيانات ذات ميزة واحدة). إنها تحدد كمياً عدد الانحرافات المعيارية standard deviations التي تبعد عنها نقطة البيانات عن المتوسط mean لتوزيعها. القاعدة الشائعة هي تمييز أي قيمة ذات درجة معيارية أكبر من 3 أو أقل من -3 كقيمة متطرفة. هذه الطريقة تكون أكثر فعالية عندما تتبع البيانات توزيعاً طبيعياً.

مسافة ماها لانوبيس (Mahalanobis Distance)

كما تم تغطيته في فصول سابقة، فإن مسافة ماها لانوبيس Mahalanobis distance هي طريقة للكشف عن القيم المتطرفة متعددة المتغيرات multivariate outlier detection method. تقيس المسافة بين نقطة وتوزيع، ولكن على عكس مسافة إقليدس Euclidean distance، فإنها تأخذ في الاعتبار الارتباطات بين الميزات. هذا يجعلها فعالة للغاية في تحديد النقاط البعيدة إحصائياً عن مركز توزيع بيانات متعدد المتغيرات، حتى لو لم تكن قيمًا متطرفة على طول أي بُعد واحد.

خوارزمية أقرب k من الجيران (k-NN)

يمكن تكييف خوارزمية أقرب k من الجيران k-Nearest Neighbors لاكتشاف الحالات الشاذة anomaly detection. الفكرة الأساسية هي أنه من المتوقع أن يكون للنقاط الطبيعية جيران قريبون، بينما ستكون الحالات الشاذة anomalies بعيدة عن النقاط الأخرى. يمكن حساب درجة الشذوذ anomaly score لنقطة بيانات على أنها المسافة إلى جارها الـ k الأقرب. تعتبر النقاط ذات المسافة الكبيرة إلى جارها الـ k حالات شاذة. يمكن استخدام وحدة sklearn.neighbors.NearestNeighbors لحساب هذه المسافات بكفاءة.

5.2.11 المرّز التلقائي (Autoencoder) لاكتشاف الحالات الشاذة

المرّز التلقائي Autoencoder هو نوع من الشبكات العصبية الاصطناعية artificial neural network يُستخدم لتعلم ترميزات بيانات فعالة بطريقة غير خاضعة للإشراف unsupervised manner [67]. بالنسبة لاكتشاف الحالات الشاذة anomaly detection، فإن قدرته على تعلم تمثيل مضغوط للبيانات الطبيعية مفيدة للغاية. تتضمن العملية أربع خطوات رئيسية.

(1) البنية: يتكون المرّز التلقائي من جزأين رئيسيين. تأخذ شبكة المرّز Encoder Net- work بيانات الإدخال وتضغطها في تمثيل منخفض الأبعاد. غالباً ما يطلق على هذا الشكل المضغوط "الفضاء الكامن" latent space أو طبقة "عنق الزجاجة" bottle-neck. ثم تأخذ شبكة المفكّك Decoder Network هذا التمثيل المضغوط وتحاول إعادة بناء بيانات الإدخال الأصلية منه.

(2) التدريب: الخطوة الأكثر أهمية هي تدريب المرّز التلقائي حصرياً على مجموعة بيانات تحتوي فقط على بيانات طبيعية وغير ضارة. أثناء التدريب، تقوم الشبكة بضبط أوزانها الداخلية لتقليل الفرق بين الإدخال الأصلي والمخرجات المعاد بناؤها. يُعرف هذا الفرق باسم خطأ إعادة البناء reconstruction error. من خلال تعلم إعادة بناء البيانات الطبيعية بدقة، يتعلم النموذج ضمناً الأنماط والهياكل الأساسية للسلوك الطبيعي.

(3) تسجيل درجة الشذوذ (خطأ إعادة البناء): بعد التدريب، يتم استخدام المرّز التلقائي للكشف. يتم تمرير نقطة بيانات جديدة وغير مرئية من خلال الشبكة المدربة. ثم نحسب خطأ إعادة البناء الخاص بها، عادةً باستخدام متوسط الخطأ التربيعي Mean Squared Error (MSE) بين الإدخال الأصلي والمخرجات المعاد بناؤها.

(4) تحديد العتبة: يشير خطأ إعادة البناء reconstruction error المرتفع إلى أن النموذج واجه صعوبة في إعادة بناء المدخلات بدقة. هذا يشير إلى أن بيانات الإدخال تحتوي على أنماط تختلف اختلافاً كبيراً عن البيانات الطبيعية التي تم تدريب النموذج عليها. لذلك، يمكننا تصنيف نقطة البيانات على أنها حالة شاذة anomaly. يتم تعيين عتبة threshold محددة مسبقاً على خطأ إعادة البناء؛ إذا تجاوز خطأ الإدخال هذه العتبة، يتم تمييزه كتهديد محتمل. أظهرت العديد من الدراسات فعالية هذا النهج في كشف التسلسل إلى الشبكات [?].

3.11 التطبيقات

يملك الكشف عن الحالات الشاذة غير الخاضع للإشراف مجموعة واسعة من التطبيقات العملية في أنظمة الدفاع السيبراني الحديثة.

• كشف الحالات الشاذة على نقاط النهاية (Endpoint Anomaly Detection): يمكن لهذه النماذج تحديد سلوكيات تنفيذ العمليات غير العادية على محطات العمل أو

الخوادم. تشمل الأمثلة برنامجاً شرعياً يصل فجأة إلى ملفات حساسة لم يلمسها من قبل، أو تعديلات مشبوهة يتم إجراؤها على سجل النظام.

• كشف الحالات الشاذة في تدفقات الشبكة (Detection): Anomaly Flow (Network) من خلال تحليل البيانات الوصفية لحركة مرور الشبكة، يمكن لهذه الطرق اكتشاف الأنماط غير الطبيعية. ويشمل ذلك الكشف عن بروتوكولات غير متوقعة على منافذ معينة، أو أحجام اتصالات غير عادية، أو إشارات القيادة والتحكم البطيئة والمنخفضة low-and-beaconing (C2) slow command and control، أو سلوك المسح الذي لا يتطابق مع توقعات الهجوم المعروفة.

• كشف الحالات الشاذة في السجلات (Detection): Anomaly (Log) يمكن للنماذج غير الخاضعة للإشراف العثور على تسلسلات أحداث نادرة أو مشبوهة في سجلات النظام (مثل سجلات أحداث ويندوز، وسجلات auditd في لينكس)، أو سجلات التطبيقات، أو سجلات الأجهزة الأمنية. يمكنها الكشف عن الانحرافات عن ترددات رسائل السجل العادية أو التسلسلات التي قد تشير إلى اختراق.

• تحديد متغيرات البرامج الضارة الجديدة (Identification): Variant Malware (New) يمكن أن تساعد هذه التقنيات في اكتشاف سلالات البرامج الضارة الجديدة. في حين أن متغيراً جديداً قد يتجنب الكشف القائم على التوقيع، فقد يُظهر ميزات شاذة في بنيته الثابتة (مثل الدوال المستوردة غير العادية) أو سلوكه الديناميكي (مثل إجراء استدعاءات API غير طبيعية) يمكن لنموذج غير خاضع للإشراف تمييزها.

• تغييرات التكوين المشبوهة (Suspicious): Configuration (Changes) يمكن إنشاء خط أساس للتكوينات الطبيعية للنظام والشبكة. يمكن للنماذج غير الخاضعة للإشراف بعد ذلك تحديد التعديلات غير المصرح بها أو غير العادية على هذه التكوينات، مثل التغييرات في قواعد جدار الحماية أو سياسات الأمان.

4.11 مجموعات البيانات المفتوحة للكشف غير الخاضع للإشراف

يعد العثور على البيانات المناسبة أمراً بالغ الأهمية لتطوير واختبار النماذج غير الخاضعة للإشراف. فيما يلي بعض المصادر الشائعة:

• UNSW-NB15: على الرغم من أن مجموعة البيانات هذه تحتوي على تسميات لهجمات مختلفة، إلا أنه يمكن استخدامها للبحث غير الخاضع للإشراف عن طريق تجاهل التسميات

أثناء مرحلة التدريب [57]. يمكن تدريب نموذج على مجموعة فرعية من البيانات تحتوي فقط على حركة مرور طبيعية لمحاكاة سيناريو واقعي. يمكن بعد ذلك تقييم قدرة النموذج على اكتشاف حالات الهجوم.

• **بيانات السجلات المحاكاة (Log (Simulated Data):** في كثير من الحالات، من المفيد إنشاء بيانات اصطناعية. يمكن للبرء إنشاء حجم كبير من السجلات التي تمثل أنشطة المستخدم العادية وأحداث النظام التوجيهية. بعد ذلك، يمكن إدخال حالات شاذة معروفة أو جديدة (مثل أنماط تسريب البيانات) في البيانات. يوفر هذا بيئة خاضعة للرقابة لاختبار أداء الكاشف.

• **بيانات القياس عن بعد لنقاط النهاية المتاحة للجمهور (Publicly Available Endpoint Telemetry):** توفر مصادر البيانات مثل سجلات Sysmon من ويندوز أو سجلات auditd من أنظمة لينكس بيانات قياس عن بعد مفصلة حول إنشاء العمليات واتصالات الشبكة ونشاط نظام الملفات. حجمها الكبير وتعقيدها يجعلها مرشحة ممتازة للتعلم غير الخاضع للإشراف لنمذجة سلوك نقطة النهاية الطبيعي.

• **مجموعات بيانات تدفق الشبكة (Network Flow (Datasets):** تحتوي مجموعات البيانات مثل CIC-IDS أو CTU-13 على مزيج من حركة المرور الطبيعية والهجومية. يمكن استخلاص حركة المرور الطبيعية لتدريب نموذج غير خاضع للإشراف، والذي يمكن بعد ذلك تقييمه مقابل حالات الهجوم التي لم يرها أثناء التدريب.

5.11 مشروع عملي: كشف الحالات الشاذة غير الخاضع للإشراف

في هذا المشروع، سنطبق العديد من التقنيات غير الخاضعة للإشراف لاكتشاف الحالات الشاذة في مجموعة بيانات اصطناعية. هدفنا هو فهم سير العمل، من إعداد البيانات إلى دمج مخرجات النماذج وإدارة الإنذارات الكاذبة.

1.5.11 إعداد البيانات

أولاً، نحتاج إلى مجموعة بيانات. نظراً لأن البيانات الواقعية غالباً ما تفتقر إلى تسميات واضحة لـ "الطبيعي" مقابل "الشاذ"، فسنقوم بإنشاء مجموعة بيانات اصطناعية. يتيح لنا ذلك الحصول على تسميات الحقيقة الأساسية للتقييم مع محاكاة سيناريو تدريب غير خاضع للإشراف. ستتألف مجموعة البيانات من مجموعة كبيرة من نقاط البيانات "الطبيعية" وعدد قليل من النقاط "الشاذة" المتناثرة. سندرب نماذجنا فقط على البيانات الطبيعية ثم نختبرها على مزيج من البيانات الطبيعية والشاذة. يقوم الكود في القائمة 72 بإعداد هذه البيانات.

```

1 import pandas as pd
2 import numpy as np
3 from sklearn.preprocessing import StandardScaler
4 from sklearn.model_selection import train_test_split
5 from sklearn.ensemble import IsolationForest
6 from sklearn.svm import OneClassSVM
7 from sklearn.metrics import classification_report, confusion_matrix
8 import matplotlib.pyplot as plt
9
10 # Create a synthetic dataset with normal data and a few outliers (anomalies)
11 np.random.seed(42)
12 # Normal data: centered around (0,0)
13 X_normal = np.random.randn(1000, 2) * 0.5
14
15 # Anomalies: far from normal data
16 X_anomalies = np.random.rand(20, 2) * 5 + 5 # 20 anomalies
17 # Add a few more anomalies from a different pattern
18 X_anomalies = np.vstack([X_anomalies, np.random.rand(10, 2) * -5 - 5])
19
20 # Combine and shuffle
21 X_data_unsupervised = np.vstack([X_normal, X_anomalies])
22 y_true_unsupervised = np.array([len(X_normal) * len(X_anomalies)])
23
24 # Shuffle data to mix normal and anomalies
25 shuffled_indices = np.random.permutation(len(X_data_unsupervised))
26 X_data_unsupervised = X_data_unsupervised[shuffled_indices]
27 y_true_unsupervised = y_true_unsupervised[shuffled_indices]
28
29 df_data_unsupervised = pd.DataFrame(X_data_unsupervised, columns=['feature_A', 'feature_B'])
30 df_data_unsupervised['true_label'] = y_true_unsupervised
31
32 # For unsupervised learning, we train on what we assume is 'normal'
33 X_train_data, X_test_mixed, _, y_test_mixed_true = train_test_split(
34     df_data_unsupervised, y_true_unsupervised, test_size=0.3, random_state=42, stratify=y_true_unsupervised
35 )
36
37 # Simulate training only on what we believe to be normal data
38 X_train_normal_only = X_train_data[X_train_data['true_label'] == 0].drop('true_label', axis=1)
39 X_test_mixed_features = X_test_mixed.drop('true_label', axis=1)
40
41 # It's good practice to scale features for these models
42 scaler_unsupervised = StandardScaler()
43 X_train_scaled = scaler_unsupervised.fit_transform(X_train_normal_only)
44 X_test_scaled = scaler_unsupervised.transform(X_test_mixed_features)
45
46 # Convert back to DataFrame for feature names
47 X_train_scaled_df = pd.DataFrame(X_train_scaled, columns=X_train_normal_only.columns)
48 X_test_scaled_df = pd.DataFrame(X_test_scaled, columns=X_test_mixed_features.columns)

```

Listing 72: إنشاء وإعداد مجموعة بيانات شاذة اصطناعية.

2.5.11 تطبيق نماذج متعددة غير خاضعة للإشراف

الآن بعد أن أصبح لدينا بيانات التدريب المقاسة (التي تحتوي فقط على نقاط طبيعية)، سندرب نموذجين مختلفين: غابة العزل Isolation Forest وآلة متجه الدعم أحادية الفئة One-Class SVM. سنستخدم بعد ذلك هذه النماذج المدربة لعمل تنبؤات على مجموعة الاختبار المختلطة الخاصة بنا. سيُخرج كلا النموذجين تنبؤاً (حيث يشير 1- عادةً إلى وجود حالة شاذة) ودرجة دالة القرار decision function score، والتي تشير إلى مدى شذوذ النقطة.

توضيح القائمة 73 كيفية تدريب وتطبيق نموذج غابة العزل.

```
1 # For demonstration, use a fixed contamination. In real scenario, tune or use a range.
2 model_iso = IsolationForest(random_state=42, contamination='auto')
3 model_iso.fit(X_train_scaled_df)
4
5 # Predict on the mixed test set
6 df_results_unsupervised = X_test_scaled_df.copy()
7 df_results_unsupervised['iso_anomaly_score'] = model_iso.decision_function(X_test_scaled_df)
8 df_results_unsupervised['iso_prediction'] = model_iso.predict(X_test_scaled_df) # -1 anomaly
9 df_results_unsupervised['true_label'] = y_test_mixed_true
```

Listing 73: تطبيق نموذج غابة العزل لاكتشاف الحالات الشاذة.

بعد ذلك، توضيح القائمة 74 نفس العملية لنموذج آلة متجه الدعم أحادية الفئة.

```
1 # nu is an upper bound on the fraction of training errors. It's an approximation of the fraction of outliers.
2 model_ocsvm = OneClassSVM(kernel='rbf', nu=0.05)
3 model_ocsvm.fit(X_train_scaled_df) # Train only on normal data
4
5 df_results_unsupervised['ocsvm_anomaly_score'] = model_ocsvm.decision_function(X_test_scaled_df)
6 df_results_unsupervised['ocsvm_prediction'] = model_ocsvm.predict(X_test_scaled_df) # -1 anomaly
```

Listing 74: تطبيق نموذج آلة متجه الدعم أحادية الفئة لاكتشاف الحالات الشاذة.

3.5.11 دمج درجات الشذوذ

قد تلتقط النماذج المختلفة جوانب مختلفة من السلوك الشاذ. لإنشاء كاشف أكثر قوة وتقليل الإنذارات الكاذبة false positives، يمكننا دمج درجات الشذوذ الخاصة بها. النهج البسيط هو حساب متوسط الدرجات. ومع ذلك، يجب علينا أولاً تطبيعها بحيث تكون على مقياس قابل للمقارنة. في حالتنا، ينتج كلا النموذجين درجات أقل للحالات الشاذة. سنقوم بعكس إشاراتها بحيث تعني الدرجة الأعلى دائماً "أكثر شذوذاً". يقوم الكود في القائمة 75 بتنفيذ هذا الدمج.

```
1 # To combine, we want "higher is more anomalous". Let's invert score signs.
2 df_results_unsupervised['iso_anomaly_score_norm'] = -df_results_unsupervised['iso_anomaly_score']
3 df_results_unsupervised['ocsvm_anomaly_score_norm'] = -df_results_unsupervised['ocsvm_anomaly_score']
4
5 df_results_unsupervised['combined_anomaly_score'] = (df_results_unsupervised['iso_anomaly_score_norm'] + df_results_unsupervised['ocsvm_anomaly_score_norm'])
```

Listing 75: تطبيع ودمج درجات الشذوذ من كلا النموذجين.

4.5.11 تحديد عتبات فعالة والتقييم

درجة الشذوذ في حد ذاتها ليست إنذاراً. نحتاج إلى تعيين عتبة threshold. سيتم تمييز أي نقطة بدرجة تتجاوز هذه العتبة على أنها حالة شاذة. يعد اختيار هذه العتبة بمثابة مفاضلة بين معدل الكشف (الاستدعاء) (recall) detection rate ومعدل الإنذارات الكاذبة false positive rate. ستلتقط العتبة المنخفضة المزيد من الحالات الشاذة ولكنها ستعلم أيضاً المزيد من الأحداث العادية بشكل غير صحيح.

الطريقة الجيدة لاختيار العتبة هي فحص توزيع الدرجات للبيانات الطبيعية المعروفة. على سبيل المثال، قد نحدد العتبة عند المئين الخامس والتسعين 95th percentile من الدرجات للبيانات الطبيعية، مما يعني أننا على استعداد لقبول معدل إنذار كاذب بنسبة 5%. يقوم الكود في القائمة 76 بتعيين عتبة، وتقييم أداء نموذجنا المدجج، وتصور النتائج. سيظهر الرسم البياني الأول توزيعات الدرجات للنقاط الطبيعية والشاذة، مما يجعل اختيار العتبة بديهياً. سيظهر الرسم البياني الثاني نقاط البيانات الأصلية، مع إبراز الحالات الشاذة الحقيقية وتلك التي توقعها نموذجنا.

5.5.11 استراتيجيات لتقليل الإنذارات الكاذبة

تمثل الإنذارات الكاذبة False positives تحدياً تشغيلياً كبيراً لأي نظام للكشف عن الحالات الشاذة. إذا كان النظام يولد الكثير من التنبيهات الكاذبة، فسيتعلم محلول الأمن بسرعة تجاهلها، مما يقوض الغرض من النظام. فيما يلي عدة استراتيجيات لإدارتها وتقليلها.

- طرق التجميع (Ensemble Methods): كما هو موضح في مشروعنا، يمكن أن يؤدي استخدام كاشفات شذوذ متعددة وطلب شكل من أشكال الإجماع إلى تحسين الدقة بشكل كبير. بدلاً من المتوسط البسيط، يمكن للمرء استخدام مخططات تصويت أكثر تقدماً، مثل تمييز حدث فقط إذا تم تحديده على أنه شاذ من قبل نموذجين على الأقل من بين ثلاثة نماذج مختلفة.

- التصفية السياقية (Contextual Filtering): يمكن إثراء تنبيهات التعلم الآلي الأولية بمعلومات سياقية. على سبيل المثال، قد يتم تصعيد تنبيه "تسجيل دخول غير عادي" من نموذج التعلم الآلي إلى محلل فقط إذا كان يفي أيضاً بمعايير قائمة على القواعد، مثل أن يكون مصدره عنوان IP غير عادي جغرافياً ويحدث خارج ساعات العمل العادية.

- حلقات التغذية الراجعة البشرية (Human Feedback): Loops من الضروري دمج التغذية الراجعة من محلي الأمن في النظام. عندما يحقق المحلل في تنبيه ويميزه على أنه إنذار

```

1  # Set a threshold at the 95th percentile of scores from the normal test data
2  # First, identify the combined scores for the truly normal points in the test set
3  normal_scores = df_results_unsupervised['combined_anomaly_score'][df_results_unsupervised['true_label'] == 0]
4  threshold = np.percentile(normal_scores, 95) # Top 5% of normal as threshold
5
6  df_results_unsupervised['combined_prediction'] = (df_results_unsupervised['combined_anomaly_score'] > threshold).astype(int)
7
8  print(f"\nCombined Model Evaluation (Threshold: {threshold:.2f}):")
9  # Note: For classification_report, 1 is anomaly, 0 is normal
10 print(classification_report(df_results_unsupervised['true_label'], df_results_unsupervised['combined_prediction'], target_names=['
11 cm_combined = confusion_matrix(df_results_unsupervised['true_label'], df_results_unsupervised['combined_prediction'])
12 print("\nCombined Confusion Matrix:\n", cm_combined)
13
14 # Visualize score distributions and threshold
15 plt.figure(figsize=(10, 6))
16 plt.hist(df_results_unsupervised['combined_anomaly_score'][df_results_unsupervised['true_label'] == 0], bins=50, alpha=0.7, label=
17 plt.hist(df_results_unsupervised['combined_anomaly_score'][df_results_unsupervised['true_label'] == 1], bins=50, alpha=0.7, label=
18 plt.axvline(x=threshold, color='blue', linestyle='--', label=f'Chosen Threshold ({threshold:.2f})')
19 plt.xlabel('Combined Anomaly Score (higher is more anomalous)'); plt.ylabel('Frequency')
20 plt.title('Distribution of Combined Anomaly Scores'); plt.legend(); plt.grid(True)
21 # In a real script you would call plt.show() here.
22
23 # Plot the data points themselves with predictions
24 plt.figure(figsize=(10, 8))
25 plt.scatter(X_test_mixed_features['feature_A'][df_results_unsupervised['true_label'] == 0],
26            X_test_mixed_features['feature_B'][df_results_unsupervised['true_label'] == 0],
27            c='blue', label='True Normal', alpha=0.6, s=20)
28 plt.scatter(X_test_mixed_features['feature_A'][df_results_unsupervised['true_label'] == 1],
29            X_test_mixed_features['feature_B'][df_results_unsupervised['true_label'] == 1],
30            c='red', marker='x', s=50, linewidth=2, label='True Anomaly')
31 # Highlight predicted anomalies
32 predicted_anomalies = X_test_mixed_features[df_results_unsupervised['combined_prediction'] == 1]
33 plt.scatter(predicted_anomalies['feature_A'], predicted_anomalies['feature_B'],
34            facecolors='none', edgecolors='black', s=100, linewidth=1.5, label='Predicted Anomaly')
35 plt.title('2D Feature Space with True and Predicted Anomalies'); plt.xlabel('Feature A')
36 plt.ylabel('Feature B'); plt.legend(); plt.grid(True)
37 # In a real script you would call plt.show() here.

```

Listing 76: تحديد العتبات وتقييم وتصور درجات الشذوذ المدججة.

كاذب، فإن هذه المعلومات قيمة للغاية. يمكن استخدام هذه التغذية الراجعة لتحسين عتبات الكشف، أو ضبط معلمات النموذج، أو الأهم من ذلك، إضافة هذا الحدث إلى مجموعة البيانات "الطبيعية" لإعادة تدريب النموذج في المستقبل.

- **معايرة العتبة (Threshold Calibration):** يجب ألا تكون عتبات الكشف ثابتة. يجب إعادة معايرتها باستمرار بناءً على معدل الإنذارات الكاذبة الملحوظ في بيئة الإنتاج. إذا أصبح المعدل مرتفعاً جداً، يمكن تعديل العتبة لتكون أكثر تحفظاً حتى تصل إلى مستوى مقبول لفريق العمليات.

الخاتمة

أظهر هذا الفصل الدور الحاسم للتعلم غير الخاضع للإشراف في اكتشاف تهديدات يوم الصفر zero-day والتهديدات الجديدة التي تفوتها الطرق الأمنية التقليدية. لقد استكشفنا مجموعة متنوعة

من الخوارزميات - من آلة متجه الدعم أحادية الفئة One-Class SVM وغابة العزل Iso-lation Forest إلى طرق التجميع clustering methods والمُرمّزات التلقائية Autoencoders - كل منها مصمم لبناء نموذج للحالة الطبيعية واكتشاف الانحرافات عنها. لقد تعلمت أن هذه التقنيات قابلة للتطبيق عبر مجالات الأمن السيبراني المتنوعة، بما في ذلك تحليل نقاط النهاية والشبكات والسجلات.

وضع المشروع العملي هذه المفاهيم موضع التنفيذ، موضحاً كيفية إعداد البيانات، وتطبيق نماذج متعددة، ودمج درجاتها للحصول على كشف أكثر قوة. والأهم من ذلك، أننا تناولنا الواقع التشغيلي لإدارة الإنذارات الكاذبة false positives من خلال معايرة العتبات، والتجميع ensembling، والتغذية الراجعة البشرية في الحلقة human-in-the-loop. من خلال إتقان هذه الأساليب غير الخاضعة للإشراف، فأنت تزود نفسك بالأدوات اللازمة لبناء أنظمة دفاع ذكية حقاً قادرة على تحديد تهديدات الغد غير المعروفة.

المختصرات والمصطلحات

درجة الشذوذ Anomaly Score قيمة تخصصها خوارزمية لنقطة بيانات، والتي تحدد كمياً مدى احتمالية كونها شاذة anomaly أو قيمة متطرفة outlier.

المُشفّر التلقائي Autoencoder نوع من الشبكات العصبونية neural network يُستخدم في التعلم غير الخاضع للإشراف unsupervised learning ويتكون من مُشفّر encoder ومُفكّك شفرة decoder. يتعلم تمثيل البيانات الطبيعية بكفاءة، ويتم اكتشاف الحالات الشاذة anomalies عندما تفشل الشبكة في إعادة بناء المدخلات بدقة.

خط الأساس (السلوكي) Baseline (Behavioral) نموذج لما يشكل نشاط النظام أو الشبكة أو المستخدم الطبيعي. يتم الإبلاغ عن الانحرافات عن خط الأساس baseline هذا كتهديدات محتملة.

القيادة والتحكم (C2 (Command and Control يشير إلى البنية التحتية وقنوات الاتصال التي يستخدمها المهاجم لإدارة الأنظمة المخترقة. يذكر النص اكتشاف إشارات القيادة والتحكم C2 beaconing كتطبيق لاكتشاف الشذوذ anomaly detection.

النقطة المركزية Centroid النقطة المركزية (المتوسط) لمجموعة عنقودية cluster. في خوارزميات مثل K-means، يمكن اعتبار النقاط البعيدة عن النقطة المركزية centroid لمجموعتها كقيم متطرفة outliers.

التجميع العنقودي Clustering تقنية تعلم آلي غير خاضعة للإشراف unsupervised machine learning لتجميع نقاط البيانات المتشابهة معاً. في مجال الأمن السيبراني cybersecurity، غالباً ما يتم تحديد الحالات الشاذة anomalies كنقاط لا تنتمي إلى أي مجموعة عنقودية cluster أو تشكل مجموعات صغيرة جداً.

خوارزمية DBSCAN (Density-Based Spatial Clustering of Applications with Noise) خوارزمية تجميع عنقودي clustering algorithm تجمع النقاط بناءً على الكثافة density. وهي فعالة في تحديد القيم المتطرفة outliers، التي تصنفها على أنها "ضوضاء" noise، دون الحاجة إلى تحديد عدد المجموعات العنقودية clusters مسبقاً.

شبكة مفكك الشفرة Decoder Network الجزء من المُشفّر التلقائي Autoencoder الذي يحاول إعادة بناء بيانات الإدخال الأصلية من تمثيلها المضغوط ومنخفض الأبعاد.

شبكة المُشفّر Encoder Network الجزء من المُشفّر التلقائي Autoencoder الذي يضغط بيانات الإدخال عالية الأبعاد إلى تمثيل منخفض الأبعاد يُعرف بالفضاء الكامن latent space.

الأساليب التجميعية Ensemble Methods تقنية تجمع بين تنبؤات نماذج تعلم الآلة -ma-chine learning models المتعددة لإنتاج نتيجة أكثر قوة ودقة من أي نموذج منفرد. الإنذار الكاذب False Positive تنبيه غير صحيح يصدره نظام أمني، حيث يتم تصنيف حدث حميد أو طبيعي عن طريق الخطأ على أنه خبيث أو شاذ.

نموذج الخليط الغاوسي (Gaussian Mixture Model) GMM نموذج احتمالي يفترض أن البيانات يتم إنشاؤها من مزيج من عدة توزيعات غاوسية (طبيعية) Gaussian (normal) distributions. تعتبر نقاط البيانات ذات الاحتمالية المنخفضة للانتماء إلى أي من هذه التوزيعات حالات شاذة anomalies.

الحقيقة المرجعية Ground Truth التصنيفات الفعلية والصحيحة لمجموعة بيانات dataset (على سبيل المثال، "طبيعي" أو "شاذ" anomaly)، والتي تُستخدم لتقييم أداء النموذج model.

المستوى الفائق Hyperplane في سياق آلة المتجهات الداعمة Support Vector Machine، هذا هو حد القرار decision boundary المستخدم لفصل نقاط البيانات في فضاء عالي الأبعاد.

غابة العزل Isolation Forest خوارزمية غير خاضعة للإشراف -unsupervised algo-rithm فعالة للغاية في اكتشاف الشذوذ anomaly detection. تعمل على مبدأ أن الحالات الشاذة anomalies "قليلة ومختلفة" وبالتالي يسهل عزلها عن بقية البيانات باستخدام أشجار القرار العشوائية random decision trees.

خوارزمية الجيران الأقرب (k-Nearest Neighbors) k-NN خوارزمية يمكن تكييفها لاكتشاف الشذوذ anomaly detection بناءً على فكرة أن نقاط البيانات الطبيعية سيكون لها جيران قريبون، بينما ستكون الحالات الشاذة anomalies بعيدة عن جيرانها.

الفضاء الكامن (أو عنق الزجاجة) Latent Space (or Bottleneck) التمثيل المضغوط ومنخفض الأبعاد للبيانات الذي تم إنشاؤه بواسطة شبكة المُشَفِّر encoder network في المُشَفِّر التلقائي Autoencoder.

عامل الانحراف المحلي LOF (Local Outlier Factor) خوارزمية تكتشف الحالات الشاذة anomalies عن طريق قياس الكثافة المحلية local density لنقطة البيانات بالنسبة لجيرانها. وهي فعالة في العثور على القيم المتطرفة outliers في مجموعات البيانات datasets ذات الكثافة المتفاوتة.

مسافة ماها لانوبيس Mahalanobis Distance مقياس إحصائي يستخدم لتحديد القيم المتطرفة متعددة المتغيرات multivariate outliers. يحسب المسافة بين نقطة وتوزيع، مع مراعاة الارتباطات بين الميزات المختلفة.

متوسط الخطأ التربيعي (MSE (Mean Squared Error مقياس شائع يستخدم لقياس الفرق بين مدخلات المُشَفِّر التلقائي Autoencoder الأصلية ومخرجاته المعاد بناؤها، ويعمل خطأ إعادة البناء reconstruction error.

آلة المتجهات الداعمة أحادية الفئة (OCSVM (One-Class Support Vector Machine خوارزمية غير خاضعة للإشراف unsupervised algorithm يتم تدريبها فقط على البيانات الطبيعية. تتعلم حدوداً تحيط بنقاط البيانات الطبيعية، وأي نقطة جديدة تقع خارج هذه الحدود يتم تصنيفها على أنها شاذة anomaly.

خطأ إعادة البناء Reconstruction Error في المُشَفِّر التلقائي Autoencoder، هذا هو الفرق بين المدخلات الأصلية والمخرجات المعاد بناؤها. يشير خطأ إعادة البناء recon-struction error المرتفع إلى أن المدخلات من المحتمل أن تكون شاذة anomaly لأن النموذج واجه صعوبة في إعادة إنتاجها.

الاكتشاف القائم على التوقيع Signature-based Detection نهج أمني تقليدي، تستخدمه برامج مكافحة الفيروسات antivirus software، يحدد التهديدات عن طريق مطابقتها مع قاعدة بيانات للأشكال الخبيثة المعروفة (التوقيعات signatures).

العتبة (لاكتشاف الشذوذ) Threshold (for anomaly detection) قيمة محددة مسبقاً يتم تطبيقها على درجة الشذوذ anomaly score. أي نقطة بيانات تتجاوز درجتها هذه العتبة threshold يتم الإبلاغ عنها كحالة شاذة anomaly.

التعلم غير الخاضع للإشراف Unsupervised Learning فئة من تعلم الآلة-machine learning حيث تتعلم الخوارزميات إيجاد الأنماط والهياكل والحالات الشاذة anomalies في البيانات دون إعطائها أي تصنيفات محددة مسبقاً.

الدرجة المعيارية Z-score مقياس إحصائي يشير إلى عدد الانحرافات المعيارية standard deviations التي تبعتها نقطة بيانات عن المتوسط. وهي طريقة بسيطة لتحديد القيم المتطرفة outliers في البيانات التي تتبع توزيعاً طبيعياً normal distribution.

هجوم يوم الصفر Zero-Day Attack هجوم سيبراني cyberattack يستغل ثغرة أمنية -vul-
nerability غير معروفة سابقًا وغير مصححة في البرامج أو الأجهزة. بحكم التعريف، لا
توجد توقعات signatures حالية لاكتشافه.

```latex

## باب 12

# الشبكات العصبونية البيانية لتحليل العلاقات وأمن سلسلة التوريد

### مقدمة

نادراً ما تكون الهجمات السيبرانية الحديثة أحداثاً فردية ومعزولة. بل هي في الغالب حملات معقدة تتضمن سلاسل من الإجراءات والتفاعلات الدقيقة بين كيانات عديدة داخل الشبكة. قد يقوم المهاجم باختراق حساب مستخدم، واستخدامه للوصول إلى مضيف، وتشغيل عملية لتنزيل ملف، ثم الاتصال بخادم خارجي. غالباً ما تفشل أدوات الأمان التقليدية التي تحلل البيانات في جداول مسطحة أو سجلات في كشف هذه الهجمات المتطورة متعددة الخطوات لأنها لا تستطيع نمذجة العلاقات المعقدة بين المستخدمين والأجهزة والعمليات والبيانات بفعالية. يستكشف هذا الفصل كيفية التغلب على هذا القصور من خلال تمثيل بيانات الأمان في شكلها الطبيعي المترابط: الرسم البياني graph.

سنقدم الشبكات العصبونية البيانية (GNNs) Graph Neural Networks، وهي فئة متخصصة من نماذج التعلم العميق deep learning مصممة للتعلم من البيانات ذات الهيكلية البيانية. من خلال تطبيق GNNs، يمكننا تحليل الأنماط العلائقية العميقة المتأصلة في الأحداث الأمنية، مما يمكننا من كشف التهديدات غير المرئية للطرق التقليدية. سيرشدك هذا الفصل خلال عملية بناء الرسوم البيانية الأمنية security graphs من البيانات الأولية، وفهم معماريات GNN المختلفة، وتطبيقها على تحديات أمنية واقعية مثل كشف التهديدات الداخلية insider threat detection، وتحليل الحركة الجانبية lateral movement analysis، وأمن سلسلة توريد البرمجيات software supply chain security.

عند إكمال هذا الفصل، ستكون قادراً على:



- فهم قصور هياكل البيانات التقليدية في تحليل البيانات الأمنية الغنية بالعلاقات.
- تعريف الشبكات العصبونية البيانية وشرح كيفية تعلمها من الهياكل البيانية.
- تحويل سجلات الأمان الأولية إلى تمثيلات بيانية للتحليل.
- بناء وتدريب وتقييم شبكة عصبونية بيانية بسيطة لمهمة تصنيف العقد - node classification على رسم بياني أمني.

## 1.12 المشكلة: العلاقات المعقدة في الهجمات

تُبنى الهجمات المتطورة على العلاقات. للبقاء دون اكتشاف، غالباً ما يستخدم المهاجمون أدوات وبيانات اعتماد شرعية في سلسلة من الخطوات الصغيرة التي تبدو حميدة. على سبيل المثال، قد يستخدم المهاجم حساب موظف مخترق لتشغيل PowerShell، وهي أداة إدارة نظام شرعية، لتنزيل نص برمجي من موقع لمشاركة الملفات. قد لا يؤدي كل إجراء فردي إلى إطلاق تنبيه. لا يتضح القصد الخبيث إلا عندما نحلل سلسلة الأحداث بأكملها: مستخدم معين، يشغل عملية معينة، على مضيف معين، يتصل بعنوان IP معين، للوصول إلى ملف معين. هياكل البيانات التقليدية، مثل الجداول في قاعدة بيانات علائقية relational database أو الصفوف في ملف سجل، غير مناسبة لهذا النوع من التحليل. فهي تمثل البيانات كمجموعة من الإدخالات المستقلة. إن الاستعلام عن علاقات معقدة متعددة القفزات multi-hop relationships في نظام كبير قائم على الجداول مكلف حسابياً وغالباً ما يكون غير عملي. هذه النظرة "المسطحة" للبيانات تجعل من الصعب رؤية الصورة الأكبر، مما يسمح للمهاجمين بإخفاء أنشطتهم وسط ضجيج العمليات اليومية. المشكلة الأساسية هي أن هياكل البيانات هذه تفشل في التقاط السياق الغني والمترابط لنشاط النظام والشبكة.

## 2.12 التقنية: الشبكات العصبونية البيانية للأمن

لتحليل العلاقات بفعالية، يجب أولاً استخدام بنية بيانات مصممة لها: الرسم البياني graph. بمجرد تمثيل بيانات الأمان كرسم بياني، يمكننا تطبيق نماذج تعلم آلي قوية مصممة خصيصاً لهذا الغرض.

## 1.2.12 الشبكات العصبونية البيانية (GNNs)

الشبكة العصبونية البيانية (GNN) Graph Neural Network هي فئة من نماذج التعلم العميق deep learning مصممة للعمل مباشرة على البيانات ذات الهيكلية البيانية [?]. على عكس الشبكات العصبونية التقليدية التي تفترض أن نقاط البيانات مستقلة عن بعضها البعض، فإن شبكات GNNs مصممة للتعلم من الروابط بين نقاط البيانات. تعمل عن طريق إنشاء تمثيل رقمي، أو تضمين embedding، لكل عقدة node في الرسم البياني. لا يلتقط هذا التضمين ميزات العقدة الخاصة بها فحسب، بل يلتقط أيضاً المعلومات المجمعة من جيرانها وموقعها داخل هيكل الرسم البياني العام. تتيح هذه العملية لشبكات GNNs تعلم أنماط علائقية معقدة ونشر المعلومات عبر الكيانات المتصلة، مما يجعلها مثالية للتحليل الأمني.

## 2.2.12 أنواع الشبكات العصبونية البيانية

توجد العديد من معماريات GNN، ولكل منها نهج مختلف للتعلم من البيانات البيانية. توفر مكتبات Python الشهيرة مثل PyTorch Geometric و Deep Graph Library (DGL) تطبيقات لهذه النماذج.

- **الشبكات الالتفافية البيانية: (GCN) (Graph Convolutional Networks)**  
توسع فكرة الالتفافات convolutions، المستخدمة بشكل مشهور في معالجة الصور، لتشمل الرسوم البيانية. تتعلم طبقة GCN ميزات العقدة عن طريق تجميع وتحويل معلومات الميزات من جيرانها المباشرين. تشبه هذه العملية تعميم الميزات عبر الرسم البياني، مما يسمح للنموذج بتعلم الأنماط الهيكلية المحلية.
- **شبكات الانتباه البيانية: (GAT) (Graph Attention Networks)**  
تحسن شبكات GCNs عن طريق إدخال آلية الانتباه attention mechanism. بدلاً من معاملة جميع الجيران على قدم المساواة أثناء التجميع، تتعلم GAT تعيين أوزان مختلفة، أو مستويات أهمية، لجيران مختلفين. هذا يمكّن النموذج من التركيز على الاتصالات الأكثر صلة عند تحديث تمثيل العقدة.
- **GraphSAGE (Sample Graph) and aggreGate): إطار GNN**  
هذا مصمم للتعلم على الرسوم البيانية الكبيرة جداً. بدلاً من استخدام جميع الجيران، تقوم GraphSAGE بأخذ عينات من جوار بحجم ثابت لكل عقدة وتتعلم دالة لتجميع المعلومات من هؤلاء الجيران الذين تم أخذ عينات منهم. هذا يجعلها فعالة ويسمح لها بتوليد تضمينات للعقد التي لم تتم رؤيتها أثناء التدريب، وهي عملية تُعرف بالتعلم الاستقرائي inductive learning.

### 3.2.12 الشبكات العصبونية للرسم البيانية الفائقة (HNNs)

بينما تمثل الرسوم البيانية القياسية علاقات ثنائية (حافة واحدة تربط بين عقدتين)، تتضمن بعض الأحداث الأمنية تفاعلات جماعية. على سبيل المثال، قد يتضمن حدث خبيث واحد ثلاثة مستخدمين، وعمليتين، وخمسة ملفات في وقت واحد. الشبكة العصبونية للرسم البياني الفائق (Hypergraph Neural Network (HNN هي نموذج متقدم يمكنه التقاط هذه العلاقات ذات الترتيب الأعلى. في الرسم البياني الفائق، يمكن "لحافة فائقة" hyperedge واحدة أن تربط أكثر من عقدتين، مما يجعلها أداة قوية لنمذجة الأحداث المعقدة متعددة الكيانات.

### 4.2.12 بناء الرسوم البيانية من البيانات الأمنية

لاستخدام شبكات GNNs، يجب علينا أولاً تحويل البيانات الأمنية الأولية إلى رسم بياني. يتضمن ذلك تحديد الكيانات الرئيسية (العقد) والتفاعلات بينها (الحواف). تؤدي أنواع مختلفة من البيانات الأمنية إلى أنواع مختلفة من الرسوم البيانية.

- **الرسوم البيانية للاعتمادية: Dependency Graphs** تظهر هذه الرسوم البيانية كيف تعتمد المكونات المختلفة على بعضها البعض. تمثل العقد مكتبات البرامج أو العمليات، وتمثل الحواف علاقة "تعتمد على". وهي حاسمة لتحليل أمن سلسلة التوريد.
- **الرسوم البيانية للنشاط: Activity Graphs** تمثل هذه الرسوم البيانية الإجراءات بمرور الوقت. يمكن أن تكون العقد مستخدمين، عمليات، ملفات، ومضيفين، بينما تمثل الحواف إجراءات مثل "المستخدم أطلق العملية س" أو "العملية س وصلت إلى الملف ع".
- **الرسوم البيانية للاتصالات: Communication Graphs** هنا، العقد هي كيانات شبكية مثل عناوين IP أو المضيفين. تمثل الحواف جلسات الاتصال بينها، وغالباً ما تكون مشروحة بتفاصيل مثل البروتوكول وحجم البيانات.
- **الرسوم البيانية للمنشأ: Provenance Graphs** تتبع هذه الرسوم البيانية أصل وتاريخ أحداث النظام، وتظهر السببية. غالباً ما يتم تمثيلها كرسم بياني موجه غير دوري Directed Acyclic Graph (DAG)، حيث تعني الحافة من العقدة أ إلى العقدة ب أن "الحدث أ تسبب في الحدث ب". على سبيل المثال، "العملية أ أنشأت الملف ب، الذي تمت قراءته بعد ذلك بواسطة العملية ج".
- **الرسوم البيانية للثغرات الأمنية: Vulnerability Graphs** في هذه الرسوم البيانية،

يمكن أن تكون العقد حزم برامج أو تكوينات نظام. تمثل الحواف الثغرات الأمنية أو الاعتماديات التي يمكن ربطها معاً لتشكيل مسار هجوم.

تحليل خوارزمي: بناء رسم بياني بسيط من سجلات الأمان

يمكن تقسيم عملية تحويل سجلات الأمان إلى رسم بياني إلى خمس خطوات منطقية:

- (1) تحديد العقد: Identify Nodes أولاً، حدد الكيانات الأساسية في بياناتك التي ستصبح عقداً. هذه هي "الأسماء" في سجلات الأمان الخاصة بك، مثل "المستخدم أ"، "المضيف س"، "الملف ع"، عنوان IP 4.3.2.1، أو معرف العملية 567. يجب تعيين معرف فريد لكل كيان فريد.
- (2) تحديد الحواف: Define Edges بعد ذلك، حدد التفاعلات التي تربط هذه الكيانات. هذه هي "الأفعال" في سجلاتك، مثل "سجل الدخول إلى"، "اتصل ب"، أو "تم تعديله بواسطة". يتم إنشاء حافة بين العقدتين المشاركتين في التفاعل (على سبيل المثال، حافة من المستخدم أ إلى المضيف س لحدث تسجيل الدخول).
- (3) تعيين سمات الحواف (اختياري): Assign Edge Attributes لجعل الرسم البياني أكثر إفادة، أضف خصائص إلى الحواف. توفر هذه السمات سياقاً حول التفاعل، مثل الطابع الزمني، أو نوع الوصول إلى الملف (قراءة/كتابة)، أو بروتوكول الشبكة المستخدم.
- (4) تعيين ميزات العقد (اختياري): Assign Node Features بالمثل، أضف ميزات إلى العقد لوصف خصائصها. بالنسبة لعقدة مستخدم، قد يكون هذا دوره أو أوقات تسجيل الدخول المعتادة. بالنسبة لعقدة مضيف، قد يكون نظام التشغيل أو مستوى التحديثات. توفر هذه الميزات المدخلات الأولية لشبكة GNN لتتعلم منها.
- (5) إنشاء قائمة المجاورة أو مصفوفة: Create Adjacency List or Matrix أخيراً، قم بتمثيل بنية الرسم البياني بتنسيق يمكن للحاسوب معالجته. تعد قائمة المجاورة adjacency list شائعة للرسوم البيانية المتفرقة (الرسوم البيانية ذات الاتصالات القليلة)، حيث تقوم بإدراج جميع جيران كل عقدة. أما مصفوفة المجاورة adjacency matrix فهي مصفوفة مربعة حيث يشير الإدخال في الموضع (i, j) إلى وجود حافة بين العقدة i والعقدة j.

## 3.12 التطبيقات

تتميز شبكات GNNs بأنها متعددة الاستخدامات ويمكن تطبيقها على مجموعة واسعة من تحديات الأمن السيبراني حيث يكون تحليل العلاقات أمراً أساسياً.

- **كشف التهديدات الداخلية: Insider Threat Detection** يمكن لشبكات GNNs نمذجة شبكة التفاعلات المعقدة بين المستخدمين والأجهزة والملفات. من خلال تحليل بنية هذه الرسوم البيانية للنشاط، يمكنها تحديد الأنماط الشاذة، مثل وصول مستخدم إلى ملفات غير عادية أو التعاون مع مجموعة غريبة من الزملاء، مما قد يشير إلى تسريب البيانات أو أي نشاط داخلي خبيث آخر.

- **كشف الحركة الجانبية ومسار الهجوم: Lateral Movement & Attack Path** Detection بعد الاختراق الأولي، غالباً ما يتحرك المهاجمون "جانبياً" عبر الشبكة للعثور على أصول قيمة. يمكن لشبكات GNNs نمذجة اتصالات الشبكة، وعلاقات العمليات (الأب-الابن)، وتدفقات المصادقة. وهذا يسمح لها بتحديد تسلسل الإجراءات التي تشكل حركة جانبية والتنوُّب بمسارات الهجوم المحتملة.

- **كشف شبكات الروبوت لإنترنت الأشياء: IoT Botnet Detection** غالباً ما تظهر أجهزة إنترنت الأشياء (IoT) المخترقة بواسطة شبكة روبوت أنماط اتصال جماعية غير عادية. يمكن لشبكات GNNs تحليل الرسم البياني للاتصالات لشبكة إنترنت الأشياء لتحديد مجموعات الأجهزة (الرسوم البيانية الفرعية subgraphs) التي تتصرف بشكل غير طبيعي، مثل الاتصال بخادم قيادة وتحكم command-and-control server معروف أو المشاركة في هجوم منسق.

- **تحليل مخاطر سلسلة توريد البرمجيات: Software Supply Chain Risk Analysis** تُبنى البرامج الحديثة من مئات المكتبات والاعتماديات مفتوحة المصدر. يمكن لشبكة GNN رسم خريطة لهذه الاعتماديات في رسم بياني، مما يسمح لها باكتشاف الثغرات الأمنية الخفية التي تنشأ من الاعتماديات المتعدية-transitive dependencies (اعتمادية تعتمد على اعتمادية أخرى). يمكنها أيضاً نمذجة كيفية انتشار المخاطر عبر سلسلة التوريد في حالة اختراق أحد المكونات.

- **تحليل الرسم البياني لتكوين الأمان: Security Configuration Graph Analysis** يمكن تمثيل تكوينات النظام، مثل قواعد جدار الحماية، وسياسات الوصول السحابي، وإعدادات أجهزة الشبكة، كرسم بياني. يمكن لشبكات GNNs تحليل هذا الرسم البياني

للغثور على التكوينات الخاطئة، أو انتهاكات السياسة، أو حقوق الوصول المتساهلة بشكل مفرط التي يمكن أن يستغلها المهاجم.

- تصنيف عائلات البرامج الضارة: Malware Family Classification يمكن تمثيل البرامج الضارة كرم بياني، مثل الرسم البياني لتدفق التحكم control flow graph (الذي يوضح مسارات التنفيذ داخل البرنامج) أو الرسم البياني لاستدعاءات الدوال function call graph. يمكن لشبكات GNNs أن تتعلم تصنيف البرامج الضارة إلى عائلات مختلفة بناءً على أوجه التشابه الهيكلية في هذه الرسوم البيانية، حتى لو تم تغيير كود البرامج الضارة لتجنب الكشف القائم على التوقيع signature-based detection.

## 4.12 مجموعات البيانات المفتوحة لشبكات GNNs في الأمن

قد يكون العثور على مجموعات بيانات أمنية مصنفة جيداً وذات طبيعة بيانية تحدياً. ومع ذلك، يمكن تكييف العديد من مجموعات البيانات الحالية للتحليل القائم على GNN.

- مجموعة بيانات اختبار التهديدات الداخلية من CERT: تحتوي مجموعة البيانات هذه على سجلات لنشاط المستخدم بمرور الوقت، بما في ذلك عمليات تسجيل الدخول، واستخدام الأجهزة، والوصول إلى الملفات، واتصالات البريد الإلكتروني. يمكن معالجتها لإنشاء رسوم بيانية لنشاط المستخدم مناسبة لنماذج كشف التهديدات الداخلية.
- الرسوم البيانية المحاكاة لاعتماديات البرامج: نظراً لأن الرسوم البيانية للاعتماديات واسعة النطاق في العالم الحقيقي غالباً ما تكون مملوكة، فكثيراً ما يقوم الباحثون بإنشاء رسوم بيانية اصطناعية تحاكي خصائص النظم البيئية للبرامج الحقيقية. هذه الرسوم البيانية قيمة لتطوير واختبار نماذج تحليل مخاطر سلسلة التوريد.
- بيانات طوبولوجيا الشبكة: يمكن أن تكون مجموعات البيانات العامة أو الخاصة التي تصف البنية المادية أو المنطقية للشبكة بمثابة رسم بياني أساسي. يمكن بعد ذلك وضع بيانات الأحداث الأمنية، مثل حركة مرور الشبكة أو التنبيهات، فوق هذه الطوبولوجيا لتوفير سياق لنماذج GNN.
- مجموعات بيانات تحليل البرامج الضارة: تتضمن بعض مجموعات بيانات أبحاث الأمان تقارير تحليل ديناميكي للبرامج الضارة، والتي غالباً ما تحتوي على تسلسلات استدعاء واجهة برمجة التطبيقات (API) أو السلوكيات المرصودة. يمكن تحويل هذه المعلومات إلى رسوم بيانية لاستدعاءات الدوال أو رسوم بيانية للمنشأ لتصنيف البرامج الضارة القائم على GNN.

## 5.12 مشروع عملي: شبكات GNNs للرسوم البيانية الأمنية

في هذا المشروع، سنقوم ببناء رسم بياني بسيط من سجلات أمان محاكاة وسنستخدم GNN لتصنيف العقد (العمليات) على أنها إما حميدة أو خبيثة. سنستخدم مكتبة `NetworkX` لبناء وتصور الرسم البياني ومكتبة `PyTorch` Geometric لبناء وتدريب نموذج GNN الخاص بنا.

### 1.5.12 الخطوة 1: بناء الرسم البياني

خطوتنا الأولى هي أخذ البيانات الأولية الشبيهة بالسجلات وتحويلها إلى بنية بيانية. سنقوم بمحاكاة سجلات إنشاء العمليات حيث يسجل كل إدخال عملية أصل تنشئ عملية فرعية. سندخل بعض الأحداث المشبوهة لإنشاء شذوذ يكتشفه نموذجنا. يوضح الكود في القائمة 77 هذه العملية. ينشئ هذا الكود رسماً بيانياً موجهاً حيث تكون العقد هي العمليات (المحددة بواسطة معرف العملية (PID) وتشير الحافة الموجهة من الأصل إلى الفرع إلى الإنشاء. يوضح التصور الذي تم إنشاؤه بواسطة هذا الكود بنية تشبه الشجرة، مع تلوين العقد الحميدة باللون الأزرق السماوي والعقد الخبيثة باللون الأحمر، مما يجعل سلسلة الهجوم واضحة بصرياً.

### 2.5.12 الخطوة 2: إعداد بيانات Geometric PyTorch

تتطلب مكثبات GNN مثل PyTorch Geometric أن تكون الرسوم البيانية بتنسيق معين. نحتاج إلى تحويل الرسم البياني `NetworkX` الخاص بنا إلى كائن `torch\_geometric.data.Data`. يتضمن ذلك إنشاء موتر tensor لميزات العقد، وموتر لاتصالية الرسم البياني، (`edge\_index`) وموتر للتصنيفات الحقيقية. بالنسبة لميزات العقد، سنستخدم ترميزاً بسيطاً أحادي التمثيل one-hot encoding لأسماء العمليات، كما هو موضح في القائمة 78.

### 3.5.12 الخطوة 3: تعريف نموذج GNN

الآن نحدد معمارية GNN الخاصة بنا. سنستخدم نموذجاً بسيطاً بطبقتين من الشبكة الالتفافية البيانية (`GCNConv`). ستأخذ الطبقة الأولى ميزات العقدة الأولية وتنتج تضمينات مخفية. ستأخذ الطبقة الثانية هذه التضمينات المخفية وتنتج مخرجات التصنيف النهائية (اللوغاريتمات logits لـ "حميد" و "خبيث"). يظهر الكود في القائمة 79.

---

```

1 import networkx as nx
2 import pandas as pd
3 import matplotlib.pyplot as plt
4 from collections import defaultdict
5
6 # Simulate process creation logs: parent_pid, child_pid, process_name, timestamp
7 # Add some suspicious activity: A legitimate process spawning a cmd.exe which then spawns an unusual process
8 process_logs = [
9 {'timestamp': '2025-08-16 10:00:00', 'parent_pid': 0, 'child_pid': 100, 'child_name': 'explorer.exe', 'is_malicious': 0},
10 {'timestamp': '2025-08-16 10:00:05', 'parent_pid': 100, 'child_pid': 101, 'child_name': 'chrome.exe', 'is_malicious': 0},
11 {'timestamp': '2025-08-16 10:00:10', 'parent_pid': 100, 'child_pid': 102, 'child_name': 'cmd.exe', 'is_malicious': 0}, # Normal
12 {'timestamp': '2025-08-16 10:00:15', 'parent_pid': 102, 'child_pid': 103, 'child_name': 'powershell.exe', 'is_malicious': 1},
13 {'timestamp': '2025-08-16 10:00:20', 'parent_pid': 103, 'child_pid': 104, 'child_name': 'malicious_script.exe', 'is_malicious': 1},
14 {'timestamp': '2025-08-16 10:00:25', 'parent_pid': 101, 'child_pid': 105, 'child_name': 'notepad.exe', 'is_malicious': 0},
15 {'timestamp': '2025-08-16 10:00:30', 'parent_pid': 100, 'child_pid': 106, 'child_name': 'firefox.exe', 'is_malicious': 0},
16]
17 df_processes = pd.DataFrame(process_logs)
18
19 # Create a directed graph from parent-child relationships
20 G = nx.DiGraph()
21
22 # Add nodes with features (process names, initial malicious label for training/testing)
23 # Ensure all PIDs (parent and child) are added as nodes
24 all_pids = set(df_processes['parent_pid']).union(set(df_processes['child_pid']))
25 process_names = {0: 'System'} # PID 0 for System/init process
26 is_malicious_map = {0: 0}
27
28 # Collect all process names and their latest malicious status
29 for _, row in df_processes.iterrows():
30 process_names[row['child_pid']] = row['child_name']
31 # Label the child process if it's malicious. If a process is ever malicious, it stays malicious.
32 if row['is_malicious'] == 1:
33 is_malicious_map[row['child_pid']] = 1
34 elif row['child_pid'] not in is_malicious_map: # Only set if not already marked malicious
35 is_malicious_map[row['child_pid']] = 0
36
37 # Add nodes to the graph
38 for pid in all_pids:
39 G.add_node(pid, name=process_names.get(pid, f'Unknown_{pid}'),
40 label=is_malicious_map.get(pid, 0)) # Default to 0 (benign) if no label
41
42 # Add edges
43 for _, row in df_processes.iterrows():
44 G.add_edge(row['parent_pid'], row['child_pid'], timestamp=row['timestamp'])
45
46 print(f"Nodes in graph: {G.nodes(data=True)}")
47 print(f"Edges in graph: {G.edges(data=True)}")
48
49 # Optional: Visualize the graph
50 plt.figure(figsize=(10, 8))
51 pos = nx.spring_layout(G, k=0.5, iterations=50) # Layout for visualization
52 node_labels = {node: G.nodes[node]['name'] for node in G.nodes()}
53 node_colors = ['red' if G.nodes[node]['label'] == 1 else 'skyblue' for node in G.nodes()]
54
55 nx.draw_networkx_nodes(G, pos, node_color=node_colors, node_size=2000)
56 nx.draw_networkx_edges(G, pos, edge_color='gray', arrows=True)
57 nx.draw_networkx_labels(G, pos, labels=node_labels, font_size=8, font_weight='bold')
58 plt.title("Process Creation Graph with Malicious Nodes Highlighted (Red)")
59 plt.axis('off')
60 plt.show()

```

---

Listing 77: بناء رسم بياني لإنشاء العمليات من السجلات باستخدام NetworkX.



#### 4.5.12 الخطوة 4: التدريب وكشف الشذوذ

مع تجهيز النموذج والبيانات، يمكننا الآن تدريب GNN. تتضمن عملية التدريب تغذية بيانات الرسم البياني للنموذج، وحساب خطأ التصنيف (الخسارة loss) على عقد التدريب المصنفة، وتحديث أوزان النموذج لتقليل هذا الخطأ. نستخدم مُحسِّن Adam ودالة خسارة الإنتروبيا المتقاطعة Cross-Entropy Loss، وهما خياران قياسيان لمهام التصنيف. أثناء التدريب، نقوم بتقييم أداء النموذج بشكل دوري على عقد الاختبار لمعرفة مدى جودة تعلمه للتعميم. تحتوي القائمة 80 على حلقة التدريب والتقييم.

#### 5.5.12 الخطوة 5: التفسير

يُظهر الناتج من حلقة التدريب أن النموذج يتعلم بسرعة تصنيف العقد الخبيثة في مجموعة الاختبار بشكل صحيح وبدقة عالية. الفكرة الرئيسية هنا هي كيفية تحقيق GNN لذلك. يتعلم النموذج تمثيل كل عقدة ليس فقط بميزاتها الخاصة (مثل اسم عملياتها) ولكن أيضاً من خلال النظر في ميزات جيرانها. في مثالنا، تكون عملية `cmd.exe` حميدة في العادة. ومع ذلك، في سلسلتنا الخبيثة، هي الأصل لـ `powershell.exe` والتي بدورها هي الأصل لـ `malicious\_script.exe`. يتعلم GNN أن السياق الذي يوفره هذا النمط المحدد للرسم البياني الفرعي - تسلسل إنشاء العمليات - هو مؤشر قوي على النشاط الخبيث. خلال عملية تمرير الرسائل في طبقات GCN، تنتشر إشارة "الخبيث" من `malicious\_script.exe` إلى الورا عبر الرسم البياني إلى أصلها وجدها. نتيجة لذلك، يتعلم النموذج ربط هذا النمط بأكمله، وليس فقط الملف التنفيذي الخبيث النهائي، بالتهديد. هذه القدرة على التعلم من بنية الرسم البياني المحلية هي ما يجعل شبكات GNNs قوية جداً في اكتشاف الهجمات المعقدة متعددة المراحل.

### خاتمة

أوضح هذا الفصل الأهمية الحاسمة لتحليل العلاقات في الأمن السيبراني الحديث. لقد رأينا كيف تفشل هياكل البيانات التقليدية المسطحة في كشف الهجمات المتطورة التي تحتجّ داخل تفاعلات معقدة. توفر الشبكات العصبونية البيانية Graph Neural Networks حلاً قوياً من خلال تمكين الآلات من التعلم مباشرة من البنية العلائقية لبيانات الأمان. لقد استكشفنا سير العمل بأكمله، من تحويل السجلات الأولية إلى رسوم بيانية أمنية ذات معنى، إلى بناء وتدريب وتفسير GNN لكشف التهديدات. أوضح المشروع العملي أن GNN

يمكن أن يتعلم تحديد النشاط الخبيث ليس من خلال النظر إلى أحداث معزولة، ولكن من خلال التعرف على الأنماط التي تشكلها الروابط بين الكيانات. من خلال إتقان هذه التقنيات، يمكن لمختبري الأمن السيبراني بناء أنظمة دفاع أكثر ذكاءً ووعياً بالسياق قادرة على كشف سلاسل الإجراءات الدقيقة والمتراكمة التي تحدد تهديدات اليوم المتقدمة.<sup>١١١</sup>

---

```

1 import torch
2 from torch_geometric.data import Data
3 from sklearn.preprocessing import LabelEncoder
4 from sklearn.model_selection import train_test_split
5
6 # Prepare node features: Convert process names to numerical features
7 # Simple one-hot encoding for demo. For real apps, use word embeddings or similar.
8 unique_process_names = list(set(node_labels.values()))
9 le_process = LabelEncoder()
10 le_process.fit(unique_process_names)
11
12 node_features_list = []
13 node_labels_list = []
14 node_to_idx = {node: i for i, node in enumerate(G.nodes())} # Mapping for consistent indexing
15
16 # Create feature matrix (X) and label vector (Y)
17 # Feature: one-hot encoding of process name
18 num_unique_process_names = len(unique_process_names)
19 for node_id in G.nodes():
20 process_name = G.nodes[node_id]['name']
21 encoded_name = le_process.transform([process_name])[0]
22 # Create a simple one-hot feature vector for the process name
23 feature_vector = [0] * num_unique_process_names
24 feature_vector[encoded_name] = 1
25 node_features_list.append(feature_vector)
26 node_labels_list.append(G.nodes[node_id]['label'])
27
28 x = torch.tensor(node_features_list, dtype=torch.float)
29 y = torch.tensor(node_labels_list, dtype=torch.long)
30
31 # Prepare edge_index: list of [source_node_idx, target_node_idx] pairs
32 edge_index_list = []
33 for u, v in G.edges():
34 edge_index_list.append([node_to_idx[u], node_to_idx[v]])
35 edge_index = torch.tensor(edge_index_list, dtype=torch.long).t().contiguous()
36
37 # Create PyTorch Geometric Data object
38 data_pg = Data(x=x, edge_index=edge_index, y=y)
39
40 print(f"\nPyTorch Geometric Data Object:\n{data_pg}")
41 print(f"Number of nodes: {data_pg.num_nodes}")
42 print(f"Number of edges: {data_pg.num_edges}")
43 print(f"Node features shape (x): {data_pg.x.shape}")
44 print(f"Edge index shape: {data_pg.edge_index.shape}")
45 print(f"Node labels shape (y): {data_pg.y.shape}")
46
47 # Create train/test masks for node classification
48 # Small dataset, so just split directly for demo
49 num_nodes = data_pg.num_nodes
50 train_mask = torch.zeros(num_nodes, dtype=torch.bool)
51 test_mask = torch.zeros(num_nodes, dtype=torch.bool)
52
53 # Manually set a few nodes for training and testing
54 train_indices = [node_to_idx[100], node_to_idx[101], node_to_idx[102], node_to_idx[105], node_to_idx[106]] # Benign for training
55 test_indices = [node_to_idx[103], node_to_idx[104]] # Malicious for testing
56 # For unseen anomalies, the model should generalize.
57 # In larger datasets, use train_test_split on node indices.
58
59 train_mask[train_indices] = True
60 test_mask[test_indices] = True
61
62 data_pg.train_mask = train_mask
63 data_pg.test_mask = test_mask
64
65 print(f"Train mask for nodes: {data_pg.train_mask.nonzero(as_tuple=True)[0].tolist()}")
66 print(f"Test mask for nodes: {data_pg.test_mask.nonzero(as_tuple=True)[0].tolist()}")

```

---

Geometric. PyTorch NetworkX إلى كائن بيانات

---

```

1 import torch.nn as nn
2 import torch.nn.functional as F
3 from torch_geometric.nn import GCNConv # For Graph Convolutional Network
4
5 class SimpleGCN(nn.Module):
6 def __init__(self, in_channels, hidden_channels, out_channels):
7 super().__init__()
8 # First GCN layer: processes input features and aggregates from neighbors
9 self.conv1 = GCNConv(in_channels, hidden_channels)
10 # Second GCN layer: takes hidden features and outputs final classification
11 self.conv2 = GCNConv(hidden_channels, out_channels)
12
13 def forward(self, x, edge_index):
14 # x: node feature matrix (num_nodes, in_channels)
15 # edge_index: graph connectivity in COO format (2, num_edges)
16
17 # First layer with ReLU activation
18 x = self.conv1(x, edge_index)
19 x = F.relu(x)
20 # Second layer (no activation here if used with CrossEntropyLoss which handles softmax internally)
21 x = self.conv2(x, edge_index)
22 return x # Logits for classification
23
24 # Instantiate the GCN model
25 # in_channels: number of features per node (from data_pg.x.shape[1])
26 # hidden_channels: number of features in the hidden layer (a design choice)
27 # out_channels: number of classes for node classification (e.g., 2 for benign/malicious)
28 in_channels = data_pg.x.shape[1]
29 num_classes_gnn = len(torch.unique(data_pg.y)) # Should be 2 (0 and 1)
30 model_gnn = SimpleGCN(in_channels=in_channels, hidden_channels=16, out_channels=num_classes_gnn)
31 print("\nGNN Model Architecture:\n", model_gnn)

```

---

Listing 79: تعريف نموذج GCN بسيط بطبقتين التفاضيتين.

---

```

1 import torch.optim as optim
2 from sklearn.metrics import classification_report
3
4 device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
5 model_gnn.to(device)
6 data_pg.to(device) # Move data to the same device as model
7
8 optimizer_gnn = optim.Adam(model_gnn.parameters(), lr=0.01)
9 criterion_gnn = nn.CrossEntropyLoss()
10
11 num_epochs_gnn = 200
12 print("\n--- Starting GNN Training ---")
13 for epoch in range(1, num_epochs_gnn + 1):
14 model_gnn.train()
15 optimizer_gnn.zero_grad()
16 out = model_gnn(data_pg.x, data_pg.edge_index)
17 loss = criterion_gnn(out[data_pg.train_mask], data_pg.y[data_pg.train_mask])
18 loss.backward()
19 optimizer_gnn.step()
20
21 # Evaluate on test set
22 model_gnn.eval()
23 with torch.no_grad():
24 pred = model_gnn(data_pg.x, data_pg.edge_index).argmax(dim=1)
25 correct = pred[data_pg.test_mask] == data_pg.y[data_pg.test_mask]
26 test_acc = int(correct.sum()) / int(data_pg.test_mask.sum())
27
28 if epoch % 20 == 0 or epoch == num_epochs_gnn:
29 print(f'Epoch: {epoch:03d}, Loss: {loss:.4f}, Test Accuracy: {test_acc:.4f}')
30
31 print("\nGNN Training complete.")
32
33 # Final evaluation on the test set
34 model_gnn.eval()
35 with torch.no_grad():
36 logits = model_gnn(data_pg.x, data_pg.edge_index)
37 final_preds = logits.argmax(dim=1)
38
39 # Get predictions and true labels for the test nodes
40 test_preds = final_preds[data_pg.test_mask].cpu().numpy()
41 test_true_labels = data_pg.y[data_pg.test_mask].cpu().numpy()
42
43 print("\n--- GNN Model Evaluation on Test Nodes ---")
44 print(classification_report(test_true_labels, test_preds, target_names=['Benign', 'Malicious']))
45
46 # Print predictions for all nodes
47 print("\nNode-level predictions:")
48 for i, node_id in enumerate(G.nodes()):
49 is_test_node = node_to_idx[node_id] in data_pg.test_mask.nonzero(as_tuple=True)[0].tolist()
50 label_status = " (Test Node)" if is_test_node else ""
51 true_label_text = "Malicious" if G.nodes[node_id]['label'] == 1 else "Benign"
52 predicted_label_text = "Malicious" if final_preds[node_to_idx[node_id]].item() == 1 else "Benign"
53 print(f" Process '{G.nodes[node_id]['name']}' (PID {node_id}){label_status}: True={true_label_text}, Predicted={predicted_label_text}")

```

---

Listing 80: تدريب نموذج GCN وتقييم أدائه.

## المختصرات والمصطلحات

قائمة التجاور Adjacency List طريقة لتمثيل الرسم البياني graph حيث يكون لكل عقدة node قائمة بجميع العقد المجاورة لها. وتعتبر هذه الطريقة فعالة للرسم البيانية ذات الاتصالات القليلة نسبياً.

مصفوفة التجاور Adjacency Matrix شبكة مربعة (مصفوفة matrix) تُستخدم لتمثيل الرسم البياني graph، حيث يشير الإدخال في صف وعمود معين إلى وجود حافة edge بين العقد nodes المقابلة.

خادم القيادة والتحكم Command-and-Control (C2) Server جهاز كمبيوتر مركزي يستخدمه المهاجم attacker لإرسال الأوامر إلى شبكة من الأجهزة المخترقة (شبكة بوتنت bot-net) واستقبال البيانات منها.

الرسم البياني لتدفق التحكم Control Flow Graph تمثيل بياني graph representation لجميع مسارات التنفيذ الممكنة التي يمكن لبرنامج كمبيوتر أن يسلكها أثناء تشغيله. وغالباً ما يُستخدم في تحليل البرمجيات الخبيثة malware analysis.

الرسم البياني الموجه غير الدوري DAG الرسم البياني الموجه غير الدوري Directed Acyclic Graph. وهو رسم بياني graph تكون فيه جميع الاتصالات (الحواف edges) في اتجاه واحد ولا توجد مسارات تسمح بالبدء من عقدة node والعودة إليها مرة أخرى. وغالباً ما يُستخدم في الرسوم البيانية للنشأ provenance graphs.

الرسم البياني للاعتمادية Dependency Graph رسم بياني graph يمثل كيفية اعتماد المكونات المختلفة، مثل مكتبات البرامج software libraries، على بعضها البعض. وجود حافة edge من A إلى B يعني أن A يعتمد على B ليعمل.

التضمين Embedding تمثيل رقمي (متجه vector من الأرقام) لعقدة node في رسم بياني graph. وهو مصمم لالتقاط سمات العقدة وعلاقتها بجيرانها وموقعها في هيكل الرسم البياني الكلي.

الرسم البياني لاستدعاء الدوال Function Call Graph رسم بياني graph يمثل علاقات الاستدعاء بين الدوال functions أو الإجراءات الفرعية subroutines في برنامج كمبيوتر. ويُستخدم لفهم كيفية تفاعل الأجزاء المختلفة من الشيفرة البرمجية.

شبكة الانتباه البيانية GAT شبكة الانتباه البيانية Graph Attention Network. نوع من الشبكات العصبية البيانية GNN يخصص مستويات مختلفة من الأهمية (أوزان الانتباه at-

(attention weights) لجيران العقدة node، مما يسمح للنموذج بالتركيز على الاتصالات الأكثر صلة أثناء التعلم.

الشبكة الالتفافية البيانية GCN الشبكة الالتفافية البيانية Graph Convolutional Net-work. نوع من الشبكات العصبية البيانية GNN يتعلم عن العقدة node من خلال تجميع ومعالجة معلومات السمات من جيرانها المباشرين، بطريقة تشبه عمل الالتفافات convolutions على وحدات البكسل في الصور.

الشبكة العصبية البيانية GNN الشبكة العصبية البيانية Graph Neural Network. فئة من نماذج التعلم العميق deep learning مصممة خصيصاً للعمل على البيانات المهيكلية على شكل رسم بياني graph والتعلم منها، مع التركيز على العلاقات والروابط بين نقاط البيانات.

إطار GraphSAGE إطار عمل للشبكات العصبية البيانية GNN framework مصمم للرسوم البيانية graphs الكبيرة جداً. يعمل عن طريق أخذ عينات من عدد ثابت من الجيران لكل عقدة node وتعلم كيفية تجميع المعلومات منها، مما يجعله عالي الكفاءة وقابلاً للتوسع.

الشبكة العصبية للرسوم البيانية الفائقة HNN الشبكة العصبية للرسوم البيانية الفائقة Hyper-graph Neural Network. شبكة عصبية neural network متقدمة يمكنها نمذجة العلاقات الجماعية، وليس فقط العلاقات الثنائية. وتستخدم "الحواف الفائقة" hyper-edges التي يمكنها ربط أكثر من عقدتين nodes في وقت واحد.

الحافة الفائقة Hyperedge حافة edge معممة في الرسم البياني الفائق hypergraph يمكنها ربط أي عدد من العقد nodes في وقت واحد، وتستخدم لتمثيل التفاعلات متعددة الكيانات.

التعلم الاستقرائي Inductive Learning قدرة نموذج تعلم الآلة machine learning، مثل GraphSAGE، على إجراء تنبؤات أو توليد تضمينات embeddings لعقد nodes جديدة لم يرها من قبل خلال مرحلة التدريب training phase.

التهديد الداخلي Insider Threat خطر أمني security risk ينشأ من داخل منظمة، مثل موظف خبيث أو مستخدم تم اختراق حسابه، يسبب استخدام وصوله المصرح به.

إنترنت الأشياء IoT إنترنت الأشياء Internet of Things. شبكة من الأجهزة المادية ("الأشياء") المدمج بها أجهزة استشعار sensors وبرامج وتقنيات أخرى لغرض الاتصال وتبادل البيانات مع الأجهزة والأنظمة الأخرى عبر الإنترنت.



الحركة الجانبية Lateral Movement التقنية التي يستخدمها المهاجمون attackers، بعد الحصول على وصول أولي إلى الشبكة، للانتقال من مضيف مخترق إلى آخر لتصعيد الامتيازات es-calate privileges والعثور على بيانات قيمة.

تصنيف العقد Node Classification مهمة شائعة في تعلم الآلة machine learning على الرسوم البيانية graphs حيث يكون الهدف هو التنبؤ وتعيين تسمية (على سبيل المثال، "خبث" أو "حميد") لكل عقدة node في الرسم البياني graph.

الترميز الأحادي الساخن One-Hot Encoding عملية تحويل البيانات الفئوية categori-cal data (مثل الكلمات أو التسميات) إلى متجهات vectors رقمية من الأصفار والآحاد، والتي يمكن استخدامها كسمات إدخال input features لنموذج تعلم الآلة machine learning model.

الرسم البياني المنشأ Provenance Graph رسم بياني graph يتتبع أصل وتاريخ البيانات أو أحداث النظام لإظهار السببية. وجود حافة edge من الحدث A إلى الحدث B يعني أن A تسبب في B.

الاعتماديات المتعدية Transitive Dependencies اعتمادية غير مباشرة في سلسلة التوريد supply chain؛ على سبيل المثال، إذا كان برنامجك يعتمد على المكتبة A، والمكتبة A تعتمد على المكتبة B، فإن برنامجك لديه اعتمادية متعددة transitive dependency على المكتبة B.

الرسم البياني للثغرات Vulnerability Graph رسم بياني graph تمثل فيه العقد nodes مكونات البرامج أو الأنظمة وتمثل الحواف edges الثغرات vulnerabilities المعروفة، مما يوضح كيف يمكن للمهاجم attacker ربط عدة استغلالات exploits معاً لإنشاء مسار هجوم attack path.

## باب 13

# النماذج الاحتمالية وتعلم الآلة البايزي Probabilistic Models & Bayesian ML لإدارة الثغرات وتحديد أولويات المخاطر

### مقدمة

في مجال الأمن السيبراني الحديث، تواجه المؤسسات فيضاً لا هوادة فيه من الثغرات البرمجية الجديدة. مع محدودية الوقت والموظفين والموارد، من المستحيل إصلاح كل ثغرة على الفور. وبالتالي، لا يمكن التحدي الرئيسي في تحديد الثغرات فحسب، بل في تحديد أولوياتها بفعالية. يقدم هذا الفصل النماذج الاحتمالية probabilistic models وتعلم الآلة البايزي Bayesian machine learning كأدوات قوية لمواجهة هذا التحدي. تتجاوز هذه الأساليب أنظمة التسجيل الثابتة القائمة على القواعد لتوفير نهج ديناميكي قائم على الأدلة للتنبؤ بمخاطر الثغرات. من خلال التعلم من مصادر بيانات متنوعة ونمذجة عدم اليقين بشكل صريح، تمكن هذه التقنيات فرق الأمن من اتخاذ قرارات أكثر ذكاءً ومبنية على المخاطر.

سيعلبك هذا الفصل كيفية استخدام التفكير الاحتمالي لإدارة الثغرات وتحديد أولوياتها. ستتعلم أساسيات الشبكات البايزية Bayesian Networks، وتفهم كيفية دمج مصادر مختلفة من استخبارات التهديدات threat intelligence، وسترى كيف تحدد هذه النماذج كمية عدم اليقين. بحلول النهاية، ستكون مشروعاً عملياً باستخدام لغة بايثون لبناء شبكة بايزية بسيطة

لتحديد أولويات مخاطر الثغرات، محولاً المفاهيم النظرية إلى مهارات دفاعية عملية.

### 1.13 المشكلة: التنبؤ بمخاطر الثغرات

الهدف الأساسي للإدارة الاستباقية للثغرات هو التنبؤ بقبالية الاستغلال، والخطورة، والتأثير الحقيقي للعيوب البرمجية. يسمح هذا للمؤسسة بتحديد أولويات جهود الإصلاح وإدارة المخاطر قبل وقوع الهجوم. ومع ذلك، هذه مهمة صعبة بسبب العديد من التحديات المتأصلة. أولاً، حجم الثغرات المفصح عنها هائل. لا يمكن لفرق الأمن معالجة كل ثغرة بنفس مستوى الإلحاح. ثانياً، غالباً ما تكون البيانات التاريخية للثغرات المكتشفة حديثاً محدودة. تتطور أساليب المهاجمين باستمرار، مما يجعل من الصعب التنبؤ بالعيوب التي سيستهدفونها بعد ذلك. أخيراً، يتطلب اتخاذ قرار مستنير الجمع بين مصادر معلومات متنوعة وغالباً ما تكون غير مكتملة. تشمل هذه المصادر درجات الخطورة الفنية، وموجزات استخبارات التهديدات حول الاستغلال النشطة، والسياق التجاري الداخلي للنظام المتأثر. يمكن التحدي الأساسي في التفكير في ظل هذا القدر الكبير من عدم اليقين لتخصيص الموارد حيث سيكون لها أكبر تأثير في تقليل المخاطر التنظيمية.

### 2.13 التقنية: النماذج الاحتمالية وتعلم الآلة البايزي Proba-bilistic & Bayesian ML

للتعامل مع عدم اليقين المتأصل في إدارة الثغرات، نلجأ إلى فئة معينة من نماذج تعلم الآلة المصممة للعمل مع الاحتمالات.

#### 1.2.13 النماذج الاحتمالية Probabilistic Models

النماذج الاحتمالية Probabilistic Models هي نماذج تعلم آلة تمثل عدم اليقين باستخدام رياضيات الاحتمالات. بدلاً من تقديم تنبؤ واحد وثابت (تقدير نقطي point estimate)، فإنها تُخرج توزيعاً احتمالياً probability distribution. يُظهر هذا التوزيع مجموعة من النتائج الممكنة واحتمالية كل منها. على سبيل المثال، بدلاً من التنبؤ بأن درجة خطورة الثغرة هي "5.8" بالضبط، قد يتنبأ النموذج الاحتمالي بوجود فرصة 60% بأن تكون المخاطرة "عالية"، وفرصة 30% بأن تكون "متوسطة"، وفرصة 10% بأن تكون "منخفضة". يوفر هذا أساساً أغنى بكثير لاتخاذ القرارات، لأنه يتضمن مقياساً للثقة في التنبؤ نفسه.

### 2.2.13 الشبكات البايزية Bayesian Networks

الشبكة البايزية Bayesian Network (المعروفة أيضاً باسم شبكة الاعتقاد Belief Network أو شبكة بايز Bayes Net) هي نوع شائع من النماذج الاحتمالية التي تمثل بصرياً الاعتماديات بين المتغيرات [64]. إنها نموذج رسومي احتمالي probabilistic graphical model، وهو مصطلح عام للنماذج التي تستخدم بنية الرسم البياني لإظهار العلاقات بين المتغيرات العشوائية [41].

في الشبكة البايزية، يتم تنظيم النموذج على شكل رسم بياني موجه غير دوري directed acyclic graph (DAG). الرسم البياني الموجه غير الدوري هو رسم بياني حيث يكون لجميع الاتصالات (الحواف) اتجاه (سهم) ومن المستحيل البدء عند عقدة، واتباع الأسهم، والانتهاؤ مرة أخرى عند نفس العقدة.

• العقد Nodes: تمثل كل عقدة في الرسم البياني متغيراً عشوائياً random variable. في سياقنا، يمكن أن يكون المتغير "وجود ثغرة"، أو "توفر استغلال"، أو "أهمية الأصل".

• الحواف Edges: تعني الحافة الموجهة (السهم) من العقدة A إلى العقدة B أن المتغير B يعتمد احتمالياً على المتغير A. بعبارة أخرى، معرفة حالة A تخبرنا شيئاً عن احتمالية حالة B.

تُعد الشبكات البايزية ممتازة لنمذجة العلاقات السببية المعقدة. فهي تسمح بالاستدلال "الأممي" (التنبؤ بتأثير معين بمعرفة السبب) والاستدلال "الخلفي" (تشخيص السبب المحتمل بمعرفة التأثير الملحوظ). على سبيل المثال، يمكننا استخدام شبكة للتنبؤ باحتمالية اختراق نظام ما بالنظر إلى العثور على ثغرة أمنية جديدة.

### 3.2.13 تقنيات تعلم الآلة البايزي - Bayesian Machine Learning Techniques

تعلم الآلة البايزي هو نموذج يطبق الإحصاء البايزي على مشاكل تعلم الآلة. يتميز بقدرته على التفكير في ظل عدم اليقين ودمج المعرفة المسبقة.

#### العمليات الغاوسية Gaussian Processes

العملية الغاوسية Gaussian Process هي طريقة بايزية غير بارامترية non-parametric تُستخدم غالباً في مهام الانحدار regression والتصنيف classification [65]. بدلاً من تعلم معاملات دالة واحدة، توفر العملية الغاوسية توزيعاً احتمالياً على فضاء كامل من الدوال الممكنة

التي تناسب البيانات. ميزتها الرئيسية هي أنها توفر تقديرات قوية لعدم اليقين مع تنبؤاتها. لأي مدخل معين، فإنها لا تنتج فقط مخرجاً متوقعاً ولكن أيضاً تبايناً، يخبرنا بمدى ثقة النموذج في هذا التنبؤ. هذا قيمٌ للغاية لتقييم المخاطر، حيث أن فهم ثقة النموذج لا يقل أهمية عن التنبؤ نفسه.

### التركيز على التفكير في ظل عدم اليقين

إحدى نقاط القوة الأساسية لجميع الأساليب البايزية هي قدرتها المتأصلة على تحديد كمية عدم اليقين. كما نوقش، فإنها تُخرج توزيعات احتمالية *probability distributions* بدلاً من تقديرات نقطية فردية *single-point estimates*. بالنسبة لمحلل أممي، يعني هذا تلقي درجة المخاطرة ليس كرقم واحد ولكن كمجموعة من الاحتمالات مع الاحتمالات المرتبطة بها. يسمح هذا باستراتيجية إدارة مخاطر أكثر دقة، تميز بين ثغرة من شبه المؤكد أنه سيتم استغلالها وأخرى تكون الأدلة فيها غامضة.

### دمج معرفة الخبراء (المسبقات Priors)

توفر الأساليب البايزية آلية رسمية لدمج معرفة الخبراء في النموذج من خلال ما يسمى بالتوزيعات المسبقة *prior distributions*، أو "المسبقات priors". المسبق هو توزيع احتمالي يمثل اعتقاداً أولياً حول متغير \*قبل\* ملاحظة أي بيانات. على سبيل المثال، قد يعتقد خبير في الأمن السيبراني أن الثغرات التي لها استغلال إثبات مفهوم عام يتم استغلالها في غضون 30 يوماً حوالي 70% من الوقت. يمكن ترميز هذا الاعتقاد كمسبق. عندما تصبح البيانات متاحة، يقوم النموذج بتحديث هذا الاعتقاد المسبق لتشكيل توزيع لاحق *posterior distribution*، والذي يجمع بين الاعتقاد الأولي للخبير والأدلة من البيانات. هذه الميزة لا تقدر بثمن في الأمن السيبراني، حيث تكون البيانات حول التهديدات الجديدة نادرة غالباً وحكم الخبراء أصل حاسم.

### 4.2.13 مكتبات بايثون

تتوفر العديد من مكتبات بايثون لبناء النماذج الاحتمالية والعمل معها. اثنتان من أبرزها هما:

- **pgmpy**: مكتبة مصممة خصيصاً للعمل مع النماذج الرسومية الاحتمالية *Probabilistic Graphical Models*. توفر أدوات لإنشاء الشبكات البايزية والنماذج الرسومية الأخرى، وتعلم معاملاتها، وإجراء الاستدلال عليها [6].

• **PyMC:** مكتبة قوية للبرمجة الاحتمالية probabilistic programming. تسمح للمستخدمين ببناء نماذج إحصائية بايزية مخصصة وإجراء الاستدلال باستخدام خوارزميات متقدمة مثل سلسلة ماركوف مونت كارلو (MCMC) Markov Chain Monte Carlo [69].

### 3.13 التطبيقات

تقدم النماذج الاحتمالية إطاراً مرناً لمعالجة العديد من التحديات الرئيسية في إدارة الثغرات وتقييم المخاطر.

#### 1.3.13 تحديد أولويات الثغرات

التطبيق الأساسي هو الترتيب الديناميكي للثغرات لتوجيه جهود الإصلاح. تتضمن هذه العملية دمج مصادر بيانات متعددة، يعمل كل منها كدليل في نموذج احتمالي:

• **CVE (الثغرات والمكاشف الشائعة - Common Vulnerabilities and Exposures):** ال CVE هو قاموس للثغرات والمكاشف الأمنية المعروفة علناً [75]. يتم تعيين معرف فريد لكل إدخال (على سبيل المثال، CVE-12345-2023) وغالباً ما يكون مصحوباً بدرجة CVSS (نظام تسجيل الثغرات الشائعة Common Vulnerability Scoring System)، وهو تمثيل رقمي لخطورتها الفنية.

• **Exploit-DB:** أرشيف للاستغلالات العامة والبرامج الضعيفة [62]. إن وجود استغلال وظيفي في قاعدة بيانات عامة مثل Exploit-DB يزيد بشكل كبير من احتمالية استغلال الثغرة من قبل المهاجمين.

• **KEV CISA (الثغرات المعروفة المستغلة - Known Exploited Vulnerabilities):** قائمة تحتفظ بها وكالة الأمن السيبراني وأمن البنية التحتية الأمريكية (CISA) للثغرات التي تم تأكيد استغلالها بنشاط في الواقع [19]. إدراج ثغرة في هذا الكatalog هو مؤشر قوي جداً على وجود خطر فوري وحقيقي.

• **Asset Criticality:** الأهمية التجارية للنظام المتأثر. قد تكون ثغرة ذات خطورة عالية على خادم تطوير غير حرج أولوية أقل من ثغرة ذات خطورة متوسطة على خادم قاعدة بيانات حيوي للمهام.

- استخبارات التهديدات واتجاهات وسائل التواصل الاجتماعي: يمكن للمعلومات في الوقت الفعلي، مثل الأحاديث في منتديات الويب المظلم أو مناقشات الباحثين الأمنيين، أن تكون بمثابة دليل لتحديث احتمالية استغلال ثغرة ما بشكل ديناميكي.

### 2.3.13 التنبؤ بثغرات سلسلة توريد البرمجيات

نادراً ما يتم بناء التطبيقات البرمجية من الصفر؛ فهي تعتمد على شبكة معقدة من المكتبات والتبعيات الخارجية. يمكن أن تؤدي ثغرة في أحد هذه المكونات إلى خلق مخاطر للتطبيق بأكمله. يمكن استخدام النماذج الاحتمالية لتقييم احتمالية ظهور الثغرات أو استغلالها في هذه التبعيات المتعددة، مما يساعد المؤسسات على فهم وتخفيف مخاطر سلسلة توريد البرمجيات الخاصة بها *software supply chain risk*.

### 3.3.13 التسجيل الديناميكي للمخاطر للحوادث والأصول الأمنية

المخاطر ليست ثابتة. يمكن للنموذج الاحتمالي تحديث درجة خطورة أصل أو حادث أمني مستمر باستمرار مع توفر معلومات جديدة. على سبيل المثال، إذا أظهر أصل به ثغرة معروفة نجاة حركة مرور شبكية غير طبيعية، يمكن للنموذج استيعاب هذا الدليل الجديد وزيادة درجة خطورة الأصل، مما يشير إلى الحاجة إلى تحقيق فوري. يسمح هذا بدفاع تكيفي يستجيب للظروف المتغيرة.

## 4.13 مجموعات البيانات المفتوحة لإدارة الثغرات

لبناء واختبار النماذج الاحتمالية لإدارة الثغرات، يمكن للممارسين استخدام العديد من مصادر البيانات المتاحة للجمهور.

- **NVD (قاعدة البيانات الوطنية للثغرات National Vulnerability Database):** المستودع الرئيسي لبيانات إدارة الثغرات في حكومة الولايات المتحدة [59]. يعزز المعلومات الأساسية في سجلات CVE بتفاصيل مثل درجات CVSS وتصنيفات الثغرات، وروابط للإرشادات. يمكن الوصول إليه عبر واجهة برمجة تطبيقات عامة.
- **موجزات بيانات Exploit-DB:** يمكن الوصول إلى أرشيف Exploit-DB لجمع بيانات حول الاستغلال المتاحة للجمهور، بما في ذلك نوع الاستغلال، والمنصة المتأثرة، ورمز إثبات المفهوم.

• كمالوج CISA: KEV هذا الكالوج متاح للجمهور ويوفر قائمة موجزة بالثغرات التي تمثل التهديدات الأكثر إلحاحاً، حيث من المعروف أنها تُستغل بنشاط.

• البيانات المحاكاة: لأغراض التدريب والتجريب، غالباً ما يكون من المفيد إنشاء مجموعات بيانات مخصصة. يمكن لهذه البيانات محاكاة جرد أصول المؤسسة (على سبيل المثال، عدد الخوادم، درجات أهميتها) وموجزات استخبارات التهديدات، مما يسمح بالممارسة في بيئة محكمة.

## 5.13 مشروع عملي: تحديد أولويات المخاطر الاحتمالية

في هذا المشروع، سنقوم ببناء شبكة بايزية بسيطة باستخدام مكتبة `pgmpy` لتقدير مخاطر استغلال ثغرة أمنية. سيوضح هذا النموذج كيفية الجمع بين قطع مختلفة من الأدلة لإنتاج تقييم احتمالي للمخاطر.

### 1.5.13 إعداد البيانات

أولاً، نحتاج إلى بيانات. نظراً لصعوبة الحصول على بيانات ثغرات حقيقية ومصنفة بشكل مثالي، سنقوم بإنشاء مجموعة بيانات اصطناعية باستخدام `pandas`. ستحاكي مجموعة البيانات هذه خصائص مختلفة للثغرات وستتضمن متغيرنا المستهدف: `Exploited_in_Wild` الكود في القائمة 81 يولد مجموعة البيانات الوهمية هذه.

### 2.5.13 نمذجة احتمالية بسيطة باستخدام `pgmpy`

بعد ذلك، سنحدد بنية شبكتنا البازية. سننشئ مجموعة من العلاقات السببية بناءً على معرفة الخبراء. على سبيل المثال، قد تجعل درجة CVE العالية من المرجح تطوير استغلال ونشره في `Exploit-DB`. المتغير النهائي الذي نريد التنبؤ به، `Exploited_in_Wild`، سيعتمد على العوامل الأخرى.

بعد تحديد البنية، يجب علينا تحديد توزيعات الاحتمالات الشرطية (CPDs) `Condi-tional Probability Distributions` لكل عقدة. يحدد CPD احتمالية وجود تلك العقدة في حالة معينة، بالنظر إلى حالات العقد الأصل. في هذا العرض التوضيحي، سنحدد CPDs يدوياً بناءً على افتراضات معقولة. في مشروع حقيقي ببيانات كافية، يمكنك تعلم هذه CPDs مباشرة من البيانات. الكود في القائمة 82 يبني النموذج ويستعلم عنه.



---

```

1 import pandas as pd
2 import numpy as np
3 from pgmpy.models import BayesianNetwork
4 from pgmpy.factors.discrete import TabularCPD
5 from pgmpy.inference import VariableElimination
6
7 # Create a dummy dataset for vulnerabilities
8 # Features:
9 # CVE_Score_High: 0 (Low/Medium), 1 (High/Critical)
10 # Exploit_DB_Exists: 0 (No), 1 (Yes) - Is there a public exploit?
11 # CISA_KEV_Listed: 0 (No), 1 (Yes) - Is it in CISA's Known Exploited Vulnerabilities?
12 # Asset_Criticality_High: 0 (Low/Medium), 1 (High) - Is the affected asset highly critical?
13 # Target:
14 # Exploited_in_Wild: 0 (No), 1 (Yes) - Has it been observed exploited?
15
16 data_vuln = pd.DataFrame({
17 'CVE_Score_High': np.random.choice(size=100, p=[0.6, 0.4]),
18 'Exploit_DB_Exists': np.random.choice(size=100, p=[0.7, 0.3]),
19 'CISA_KEV_Listed': np.random.choice(size=100, p=[0.9, 0.1]),
20 'Asset_Criticality_High': np.random.choice(size=100, p=[0.5, 0.5])
21 })
22
23 # Simulate 'Exploited_in_Wild' based on other factors
24 # Higher chance of being exploited if CVE is high, ExploitDB exists, or CISA KEV listed.
25 # This is a simple heuristic for generating dependent data.
26 data_vuln['Exploited_in_Wild'] = (
27 (data_vuln['CVE_Score_High'] == 1) * 0.4 +
28 (data_vuln['Exploit_DB_Exists'] == 1) * 0.5 +
29 (data_vuln['CISA_KEV_Listed'] == 1) * 0.9 +
30 (data_vuln['Asset_Criticality_High'] == 1) * 0.3 +
31 np.random.rand(100) * 0.2
32).apply(lambda x: 1 if x > 0.5 else 0)
33
34 print("Dummy Vulnerability Data Head:\n", data_vuln.head())
35 print(f"\nExploited_in_Wild distribution: {data_vuln['Exploited_in_Wild'].value_counts()}")

```

---

## Listing 81: إنشاء مجموعة بيانات وهمية للثغرات

### 3.5.13 عرض توضيحي لتحديد الأولويات

تُظهر المخرجات من استعلامات الاستدلال في القائمة 82 كيف يمكن استخدام النموذج لتحديد الأولويات. لأي ثغرة معينة، يمكننا إدخال خصائصها المعروفة (مثل درجة CVE الخاصة بها، وما إذا كان هناك استغلال موجود، وأهمية الأصل الذي تؤثر عليه) كأدلة. ثم يحسب النموذج الاحتمال المحدث بأن الثغرة سيتم استغلالها في الواقع.

سيستخدم فريق الأمن هذه المخرجات لإنشاء قائمة مرتبة للإصلاح. سيتم وضع الثغرات ذات الاحتمالية الأعلى لـ `Exploited\_in\_Wild` في أعلى القائمة. تكمن القوة العظمى لهذا النهج في طبيعته الديناميكية. إذا وصلت معلومة جديدة من استخبارات التهديدات—على سبيل المثال، تمت إضافة ثغرة إلى كتالوج CISA يمكننا KEV ببساطة إضافة هذا كدليل جديد إلى النموذج. سيقوم النموذج بتحديث تقديره الاحتمالي على الفور، مما قد ينقل تلك الثغرة إلى مستوى أعلى بكثير في قائمة الأولويات. يوفر هذا نظاماً لتحديد الأولويات أكثر تكيّفاً بكثير من النظام

الذي يعتمد على درجات CVSS الثابتة وحدها.

### 4.5.13 تحديد كمية عدم اليقين

توضح مخرجات `pgmpy` بشكل مباشر مفهوم تحديد كمية عدم اليقين -uncertainty quantification. بدلاً من إجابة بسيطة "نعم" أو "لا"، يقدم النموذج التوزيع الاحتمالي الكامل للمتغير. `Exploited\_in\_Wild` على سبيل المثال، قد يعيد  $P(\text{Exploited\_in\_Wild}=1)$  = 0.95 لثغرة واحدة و  $P(\text{Exploited\_in\_Wild}=1)$  = 0.55 لأخرى. إذا كنا نستخدم عتبة بسيطة قدرها 0.5، فقد يتم تصنيف كليهما على أنه "من المحتمل استغلاله". ومع ذلك، يمنح الإخراج الاحتمالي صانع القرار رؤية أكثر دقة. تمثل الحالة الأولى ثقة عالية في الاستغلال، وتتطلب إجراءً فورياً. تشير الحالة الثانية إلى عدم يقين كبير، في حين أن الاستغلال أكثر ترجيحاً بقليل من عدمه، فإن الأدلة ليست قاطعة. يسمح هذا لقادة الأمن باتخاذ قرارات مستنيرة بالمخاطر، وتخصيص مواردهم الأكثر أهمية للتهديدات حيث تكون الأدلة أقوى، وربما جدولة مراقبة إضافية للتهديدات حيث يظهر النموذج عدم يقين عالٍ.

## الخاتمة

استكشف هذا الفصل استخدام النماذج الاحتمالية وتعلم الآلة البايزي لإدارة الثغرات الحديثة. لقد رأينا أنه في مشهد يتميز بالبيانات الهائلة وعدم اليقين المتأصل، تقدم هذه التقنيات مساراً منظماً وذكياً للبضي قدماً. من خلال تمثيل العلاقات بين المتغيرات ودمج الأدلة الجديدة رسمياً، تسمح لنا الشبكات البايزية Bayesian Networks بالانتقال من درجات المخاطر الثابتة إلى عملية تحديد أولويات ديناميكية وتكيفية. إنها توفر إطاراً قوياً لدمج مصادر بيانات متباينة—من درجات CVE الفنية وأهمية الأصول إلى استخبارات التهديدات في الوقت الفعلي—في صورة مخاطر واحدة ومتناسكة.

الخلاصة الرئيسية هي أن نمذجة عدم اليقين بشكل صريح ليست نقطة ضعف بل قوة. فهي تسمح لنا بتحديد كمية ثقتنا، واتخاذ قرارات أكثر شفافية، وتخصيص موارد الأمن السيبراني المحدودة بشكل أكثر فعالية. أظهر المشروع العملي أنه باستخدام مكتبات مثل `pgmpy` يمكن ترجمة هذه المفاهيم المتقدمة إلى أدوات عملية تساعد في بناء أنظمة دفاع أكثر ذكاءً ومرونة.

---

```

1 # Define the network structure (causal relationships)
2 # Example: Exploit_DB_Exists and CISA_KEV_Listed depend on CVE_Score_High
3 # Exploited_in_Wild depends on all other factors
4 model_bn = BayesianNetwork([
5 ('CVE_Score_High', 'Exploit_DB_Exists'),
6 ('CVE_Score_High', 'CISA_KEV_Listed'),
7 ('Exploit_DB_Exists', 'Exploited_in_Wild'),
8 ('CISA_KEV_Listed', 'Exploited_in_Wild'),
9 ('Asset_Criticality_High', 'Exploited_in_Wild')
10])
11
12 # Define Conditional Probability Distributions (CPDs)
13 # For simplicity, we'll manually define some CPDs. In a real scenario, these could be learned from data.
14 # CPD for CVE_Score_High (prior probability)
15 cpd_cve = TabularCPD(variable='CVE_Score_High', variable_card=2, values=[[0.6], [0.4]]) # P(CVE=0), P(CVE=1)
16
17 # CPD for Exploit_DB_Exists given CVE_Score_High
18 cpd_exploitdb = TabularCPD(variable='Exploit_DB_Exists', variable_card=2,
19 values=[[0.9, 0.3], # P(ExploitDB=0 | CVE=0), P(ExploitDB=0 | CVE=1)
20 [0.1, 0.7]], # P(ExploitDB=1 | CVE=0), P(ExploitDB=1 | CVE=1)
21 evidence=['CVE_Score_High'], evidence_card=)
22
23 # CPD for CISA_KEV_Listed given CVE_Score_High
24 cpd_cisa_kev = TabularCPD(variable='CISA_KEV_Listed', variable_card=2,
25 values=[[0.99, 0.05], # P(KEV=0 | CVE=0), P(KEV=0 | CVE=1)
26 [0.01, 0.95]], # P(KEV=1 | CVE=0), P(KEV=1 | CVE=1)
27 evidence=['CVE_Score_High'], evidence_card=)
28
29 # CPD for Asset_Criticality_High (prior probability)
30 cpd_asset_crit = TabularCPD(variable='Asset_Criticality_High', variable_card=2, values=[[0.5], [0.5]])
31
32 # CPD for Exploited_in_Wild given its parents
33 # Order of parents: [Exploit_DB_Exists, CISA_KEV_Listed, Asset_Criticality_High]
34 # values: P(Exploited=0|...), P(Exploited=1|...) for all parent combinations
35 # EDB, KEV, AC: (0,0,0), (0,0,1), (0,1,0), (0,1,1), (1,0,0), (1,0,1), (1,1,0), (1,1,1)
36 cpd_exploited_in_wild = TabularCPD(variable='Exploited_in_Wild', variable_card=2,
37 values=[
38 [0.95, 0.80, 0.70, 0.20, 0.50, 0.10, 0.05, 0.01], # P(Exploited=0 | ...)
39 [0.05, 0.20, 0.30, 0.80, 0.50, 0.90, 0.95, 0.99] # P(Exploited=1 | ...)
40],
41 evidence=['Exploit_DB_Exists', 'CISA_KEV_Listed', 'Asset_Criticality_High'],
42 evidence_card=)
43
44 # Add CPDs to the model
45 model_bn.add_cpds(cpd_cve, cpd_exploitdb, cpd_cisa_kev, cpd_asset_crit, cpd_exploited_in_wild)
46
47 # Check if the model is valid
48 print(f"\nIs the Bayesian Network valid? {model_bn.check_model()}")
49
50 # Perform inference
51 infer = VariableElimination(model_bn)
52
53 # Example inference: What is the probability of Exploited_in_Wild if CVE is High and Exploit DB Exists?
54 print("\n--- Bayesian Network Inference Examples ---")
55 query_1 = infer.query(variables=['Exploited_in_Wild'],
56 evidence={'CVE_Score_High': 1, 'Exploit_DB_Exists': 1})
57 print(f"\nP(Exploited_in_Wild | CVE_High=1, Exploit_DB_Exists=1):\n{query_1}")
58
59 # Example inference: What is the probability of Exploited_in_Wild if Asset Criticality is High?
60 query_2 = infer.query(variables=['Exploited_in_Wild'],
61 evidence={'Asset_Criticality_High': 1})
62 print(f"\nP(Exploited_in_Wild | Asset_Criticality_High=1):\n{query_2}")
63
64 # Example inference: What is the probability of Exploited_in_Wild if CISA KEV is Listed but CVE is Low?
65 query_3 = infer.query(variables=['Exploited_in_Wild'],
66 evidence={'CISA_KEV_Listed': 1, 'CVE_Score_High': 0})
67 print(f"\nP(Exploited_in_Wild | CISA_KEV_Listed=1, CVE_Score_High=0):\n{query_3}")

```

---

## Listing 82: بناء واستعلام شبكة بايزية بسيطة باستخدام pgmpy

## المختصرات والمصطلحات

تعلم الآلة البايزي Bayesian Machine Learning نموذج في تعلم الآلة-machine learning  
ing يطبق الإحصاءات البايزية Bayesian statistics للاستدلال في ظل عدم اليقين  
ودمج المعرفة الخبيرة المسبقة رسمياً في النموذج.

الشبكة البايزية Bayesian Network (Bayes Net) نوع من النماذج الاحتمالية-proba-  
bilistic model التي تستخدم رسماً بيانياً graph (تحديداً، رسماً بيانياً موجهاً غير دوري  
Directed Acyclic Graph) لتمثيل التبعيات الاحتمالية بين مجموعة من المتغيرات  
بشكل مرئي.

وكالة الأمن السيبراني وأمن البنية التحتية CISA وكالة الأمن السيبراني وأمن البنية التحتية Cy-  
bersecurity and Infrastructure Security Agency: وكالة حكومية أمريكية،  
من بين واجباتها، تحتفظ بكتالوج للثغرات الأمنية المعروفة والمستغلة known exploited  
vulnerabilities لمساعدة المؤسسات على تحديد أولويات التهديدات.

توزيع الاحتمال الشرطي (Conditional Probability Distribution) CPD جدول أو  
دالة في الشبكة البايزية Bayesian Network تحدد احتمالية حالة متغير معين بناءً على  
حالات المتغيرات الأصلية له.

الثغرات والمكاشف الشائعة (Common Vulnerabilities and Exposures) CVE قاموس  
موحد للثغرات الأمنية السيبرانية cybersecurity vulnerabilities والمكاشف المعروفة  
للعمامة، حيث يتم تعيين معرف فريد لكل إدخال.

نظام تسجيل الثغرات المشترك (Common Vulnerability Scoring System) CVSS  
نظام تسجيل رقمي يستخدم لتمثيل الخطورة التقنية لثغرة برمجية-software vulnerabil-  
ity.

الرسم البياني الموجه غير الدوري DAG (Directed Acyclic Graph) رسم بياني graph  
تكون فيه جميع الاتصالات (الحواف edges) موجهة (تحتوي على سهم) ولا تحتوي على  
دورات cycles، مما يشكل الأساس الهيكلي للشبكة البايزية Bayesian Network.

العملية الغاوسية Gaussian Process طريقة بايزية لا معلمية non-parametric Bayesian  
method توفر توزيعاً احتمالياً probability distribution على الدوال، وهي مفيدة  
للغاية لتوليد التنبؤات مصحوبة بتقديرات عدم اليقين uncertainty estimates القابلة  
للقياس الكمي.

كتالوج الثغرات الأمنية المعروفة والمستغلة KEV (Known Exploited Vulnerabilities) Catalog  
قائمة تحتفظ بها وكالة الأمن السيبراني وأمن البنية التحتية CISA تحدد الثغرات الأمنية

vulnerabilities التي تم التأكد من استغلالها بنشاط وحالياً من قبل المهاجمين، مما يشير إلى مستوى عالٍ جداً من المخاطر في العالم الحقيقي.

سلسلة ماركوف مونت كارلو (MCMC (Markov Chain Monte Carlo فئة من الخوارزميات probabilistic algorithms المتقدمة التي تستخدمها مكتبات البرمجة الاحتمالية -probabilistic programming مثل PyMC لإجراء الاستدلال inference والحسابات في النماذج البايزية Bayesian models المعقدة.

قاعدة بيانات الثغرات الوطنية (NVD (National Vulnerability Database المستودع الرئيسي لحكومة الولايات المتحدة لبيانات إدارة الثغرات الأمنية -vulnerability management، والذي يثري إدخلالات CVE بتفاصيل إضافية مثل درجات نظام تسجيل الثغرات المشترك CVSS scores والتصنيفات.

pgmpy مكتبة بايثون Python library مصممة خصيصاً لإنشاء وتحليل وإجراء الاستدلال inference على النماذج الرسومية الاحتمالية Probabilistic Graphical Models، بما في ذلك الشبكات البايزية Bayesian Networks.

التقدير النقطي Point Estimate تنبؤ بقيمة واحدة ثابتة (على سبيل المثال، درجة مخاطرة risk score تبلغ "5.8") لا يعكس أي معلومات حول عدم اليقين uncertainty أو الثقة confidence في التنبؤ.

التوزيع اللاحق Posterior Distribution في إحصاءات بايزي Bayesian statistics، هذا هو التوزيع الاحتمالي probability distribution المحدث لمتغير بعد أخذ البيانات أو الأدلة الجديدة في الاعتبار. يتم تشكيله من خلال دمج التوزيع المسبق prior distribution مع البيانات المرصودة.

التوزيع المسبق Prior Distribution (Prior) توزيع احتمالي probability distribution tion يمثل اعتقاداً أولياً حول احتمالية متغير ما قبل ملاحظة أي دليل أو بيانات جديدة. وهي طريقة لدمج المعرفة الخبيرة رسمياً في النموذج.

النموذج الرسومي الاحتمالي (PGM) Probabilistic Graphical Model فئة عامة من النماذج التي تستخدم بنية رسم بياني graph لتمثيل العلاقات الاحتمالية بين المتغيرات العشوائية random variables. تعتبر الشبكات البايزية Bayesian Networks نوعاً بارزاً من النماذج الرسومية الاحتمالية PGM.

النماذج الاحتمالية Probabilistic Models نماذج تعلم الآلة Machine learning التي

تمثل عدم اليقين uncertainty من خلال إخراج توزيع احتمالي probability distribution على مجموعة من النتائج المحتملة، بدلاً من تنبؤ واحد ثابت.

PyMC مكتبة بايثون Python library قوية للبرمجة الاحتمالية probabilistic programming تمكن المستخدمين من بناء نماذج إحصائية بايزية Bayesian statistical models مخصصة وإجراء الاستدلال inference.

```latex


القسم 3

عمليات الذكاء الاصطناعي في الأمن السيبراني

يتناول هذا الجزء الجوانب الحاسمة لنشر وإدارة وصيانة وحوكمة أنظمة AI بمسؤولية في بيئات Cybersecurity عالية المخاطر. ويغطي الجوانب العملية لـ MLOps، والتقييم المستمر، والاعتبارات الأخلاقية، والاتجاهات المستقبلية المثيرة. ``

باب 14

من النموذج الأولي إلى الإنتاج: عمليات تعلم الآلة في الأمن السيبراني

مقدمة

إن بناء نموذج تعلم آلة يعمل بشكل جيد على مجموعة بيانات ثابتة هو مجرد خطوة أولى. لخلق قيمة حقيقية في مجال الأمن السيبراني، يجب نشر هذا النموذج في بيئة حية وديناميكية، وغالباً ما تكون معادية. يمثل هذا الانتقال من نموذج أولي إلى نظام إنتاج موثوق تحدياً كبيراً. يقدم هذا الفصل عمليات تعلم الآلة (MLOps) Machine Learning Operations، وهو تخصص مصمم لسد هذه الفجوة. سوف نستكشف الصعوبات الفريدة لتفعيل الذكاء الاصطناعي في مجال الأمن، بدءاً من التهديدات المتطورة باستمرار إلى المخاطر العالية للتنبؤات غير الصحيحة. سنتعلم المكونات الأساسية لدورة حياة MLOps، بما في ذلك خطوط أنابيب البيانات المؤتمتة، وإصدار النماذج، والتكامل والتسليم المستمر (CI/CD)، واستراتيجيات المراقبة الحرجة. كما سنغطي تقنيات النشر الحديثة، وقابلية المراقبة observability، والممارسات الأمنية الأساسية لحماية أنظمة الذكاء الاصطناعي نفسها. بنهاية هذا الفصل، ستفهم المبادئ والممارسات اللازمة لبناء ونشر وصيانة أنظمة دفاع قوية وقابلة للتطوير وآمنة مدعومة بالذكاء الاصطناعي.

1.14 تفعيل تعلم الآلة في الأمن: تحديات فريدة

يمثل نشر نماذج تعلم الآلة في مجال الأمن السيبراني صعوبات لا توجد دائماً في نشر البرمجيات التقليدية. فالبيئة ديناميكية، وعدائية، ولديها قدرة تحمل منخفضة جداً للخطأ. إن فهم هذه التحديات هو الخطوة الأولى نحو بناء ممارسات MLOps فعالة.

1.1.14 الانحراف المستمر للمفهوم (Continuous Drift) Concept

في تعلم الآلة، يصف انحراف المفهوم concept drift ظاهرة تغير الخصائص الإحصائية للمتغير المستهدف، الذي يحاول النموذج التنبؤ به، بمرور الوقت [27]. في الأمن السيبراني، هذا واقع مستمر. حيث يقوم المهاجمون باستمرار بإنشاء برامج ضارة جديدة، وتطوير تقنيات اختراق مبتكرة، وتغيير تكتيكاتهم لتجنب الكشف. في الوقت نفسه، يتطور سلوك المستخدم والنظام الطبيعي أيضاً مع اعتماد برامج جديدة وتغير العمليات التجارية. قد يكون النموذج الذي تم تدريبه على تحديد تهديدات الأمس غير فعال تماماً ضد تهديدات الغد. وهذا يتطلب أنظمة يمكنها اكتشاف الانحراف وتسهيل إعادة تدريب النماذج بشكل متكرر وتلقائي.

2.1.14 القدرة المنخفضة على تحمل الأخطاء (مخاطر عالية)

إن عواقب أخطاء النماذج في التطبيقات الأمنية وخيمة، مما يخلق عملية توازن صعبة.

- الإيجابيات الكاذبة (False Positives - FP): تحدث الإيجابية الكاذبة عندما يصنف النموذج بشكل غير صحيح نشاطاً حميداً على أنه ضار. يؤدي العدد الكبير جداً من الإيجابيات الكاذبة إلى إرهاق التنبيهات alert fatigue، حيث يصبح محلولو الأمن غارقين في سيل مستمر من التنبيهات غير الصحيحة. يمكن أن يتسبب هذا في تجاهلهم للإشعارات، بما في ذلك تلك الخاصة بالهجمات الحقيقية، ويهدر موارد بشرية قيمة في تحقيقات لا طائل من ورائها [?].

- السلبيات الكاذبة (False Negatives - FN): السلبيات الكاذبة هي أخطر أنواع الأخطاء، حيث يفشل النموذج في اكتشاف هجوم فعلي. يمكن أن يؤدي تهديد واحد لم يتم اكتشافه إلى خرق كارثي للبيانات، وخسارة مالية كبيرة، وضرر دائم لسمعة المؤسسة. غالباً ما تكون تكلفة السلبيات الكاذبة أعلى بأضعاف مضاعفة من تكلفة الإيجابيات الكاذبة.

3.1.14 الهجمات العدائية على الذكاء الاصطناعي

على عكس العديد من المجالات الأخرى، فإن نماذج الذكاء الاصطناعي في الأمن السيبراني مستهدفة بشكل نشط من قبل خصوم أذكياء. يقوم المهاجمون بصياغة مدخلات بشكل متعمد لخداع النماذج، وهي ممارسة تُعرف باسم الهجوم العدائي adversarial attack [?]. قد يقومون بتغيير طفيف في كود البرامج الضارة لتجاوز نموذج الكشف أو إنشاء حركة مرور شبكية تحاكي الأنماط الحميدة لتجنب نظام كشف التسلل. هذا يعني أنه يجب بناء النماذج الأمنية مع مراعاة المتانة، بحيث تكون قادرة على مقاومة محاولات التلاعب.

4.1.14 الأداء في الوقت الفعلي

تتطلب العديد من المهام الأمنية اتخاذ إجراءات فورية. على سبيل المثال، يجب على بوابة أمان البريد الإلكتروني أن تقرر ما إذا كانت ستحظر محاولة تصيد احتيالي في أجزاء من الثانية، قبل أن تصل إلى صندوق بريد المستخدم. يجب على نظام منع التسلل الشبكي تحديد وحظر حزمة ضارة قبل أن تخترق الخادم. تضع هذه الحاجة إلى تنبؤات بزمن انتقال منخفض جداً قيوداً كبيرة على تعقيد النموذج والبنية التحتية وهندسة النشر.

5.1.14 ندرة البيانات وعدم توازنها

غالباً ما تكون البيانات عالية الجودة والمصنفة للهجمات الجديدة والمتطورة نادرة. بحلول الوقت الذي يتم فيه جمع وتصنيف مجموعة بيانات كبيرة لهجوم معين، قد يكون المهاجمون قد انتقلوا بالفعل إلى تقنية جديدة. علاوة على ذلك، تكون مجموعات البيانات الأمنية دائماً غير متوازنة إلى حد كبير. فالأحداث الضارة نادرة مقارنة بالحجم الهائل للنشاط الحميد. هذا الخلل يجعل من الصعب تدريب نماذج يمكنها تحديد فئة الهجوم النادرة بفعالية دون توليد عدد كبير جداً من الإيجابيات الكاذبة.

6.1.14 الحاجة إلى قابلية التفسير

عندما يبلغ نموذج ذكاء اصطناعي عن حدث ما، فإن وظيفة محلل الأمن هي التحقيق. للقيام بذلك بفعالية، غالباً ما يحتاجون إلى فهم *لماذا* اتخذ النموذج قراره. إن نموذج الصندوق الأسود black-box model الذي يخرج ببساطة "ضار" هو أقل فائدة من نموذج يمكنه توفير سياق، مثل "تم الإبلاغ عن هذا الملف لأنه يستخدم PowerShell لتنفيذ تقنية حقن الذاكرة المرتبطة

عادةً بالبرامج الضارة". دفعت هذه الحاجة إلى اعتماد تقنيات الذكاء الاصطناعي القابل للتفسير (Explainable AI - XAI) في العمليات الأمنية.

2.14 دورة حياة عمليات تعلم الآلة (MLOps)

لمواجهة تحديات نشر تعلم الآلة في الأمن السيبراني، يلزم اتباع نهج منهجي. هذا هو دور عمليات تعلم الآلة (MLOps).

1.2.14 عمليات تعلم الآلة (MLOps)

MLOps هي مجموعة من الممارسات التي تجمع بين تعلم الآلة (ML)، وعمليات التطوير (DevOps)، وهندسة البيانات لإدارة دورة الحياة الكاملة لنموذج تعلم الآلة. الهدف من *MLOps* هو أتمتة وتبسيط عملية بناء ونشر ومراقبة وتحديث نماذج تعلم الآلة بطريقة موثوقة وقابلة للتطوير [42]. إنها تحول تطوير النماذج من عملية يدوية موجهة نحو البحث إلى سير عمل إنتاجي متكرر ومؤتمت ومتكامل.

2.2.14 المكونات الرئيسية لعمليات تعلم الآلة (MLOps)

يُبنى إطار *MLOps* الفعال على عدة مكونات رئيسية تعمل معاً لإنشاء دورة حياة متماسكة.

بناء خطوط أنابيب البيانات

خط أنابيب البيانات *data pipeline* هو سير عمل مؤتمت يتعامل مع جميع الخطوات المطلوبة لإعداد البيانات لنموذج تعلم الآلة. ويشمل ذلك استيعاب البيانات الأمنية الأولية (مثل سجلات النظام، والتقاط حزم الشبكة، وبيانات القياس عن بعد من نقاط النهاية)، متبوعاً بتنظيفها لإزالة الأخطاء، وتحويلها إلى تنسيق متسق، وأخيراً هيكلتها في ميزات مناسبة للتدريب والاستدلال. تعتبر خطوط أنابيب البيانات القوية ضرورية لضمان جودة واتساق البيانات التي تغذي نماذج تعلم الآلة.

إنشاء متاجر الميزات

متجر الميزات feature store هو مستودع مركزي لتخزين وإدارة وتقديم ميزات تعلم الآلة. يوفر مجموعة موحدة ومصنفة من الميزات التي يمكن استخدامها باستقرار عبر نماذج و فرق مختلفة. تتمثل إحدى الفوائد الرئيسية لمتجر الميزات في أنه يضمن تطبيق نفس منطق هندسة الميزات أثناء كل من تدريب النموذج والاستدلال في الوقت الفعلي. يساعد هذا في منع انحراف التدريب-الخدمة training-serving skew، وهي مشكلة شائعة حيث تؤدي الاختلافات بين بيئات التدريب والإنتاج إلى ضعف أداء النموذج [20].

إصدار النماذج

تماماً كما يتم إصدار الكود في Git، يجب أيضاً إصدار نماذج تعلم الآلة. يتضمن إصدار النماذج Model versioning تتبع كل إصدار من النموذج مع البيانات الوصفية المرتبطة به، بما في ذلك الكود المستخدم لتدريبه، والإصدار المحدد لمجموعة البيانات التي تم تدريبه عليها، والمعاملات الفائقة، ومقاييس الأداء. تضمن هذه الممارسة إمكانية إعادة الإنتاج، وتسمح بالعودة السهلة إلى إصدار مستقر سابق إذا فشل نموذج جديد، وتوفر مسار تدقيق واضحاً.

أتمتة اختبار النماذج

قبل نشر إصدار نموذج جديد، يجب اختباره بدقة. يتجاوز الاختبار الآلي في خط أنابيب MLOps فحوصات الدقة البسيطة. ويشمل تقييم الأداء على قطاعات البيانات الرئيسية، واختبار العدالة والتحيز، وتقييم متانة النموذج ضد المدخلات العدائية المحتملة. تضمن هذه الفحوصات الآلية ترقية النماذج عالية الجودة والموثوقة فقط إلى الإنتاج.

التكامل المستمر/التسليم المستمر (CI/CD) لنماذج تعلم الآلة

التكامل المستمر (CI) والتسليم المستمر (CD) هما ممارستان من DevOps تعملان على أتمتة عملية بناء واختبار وإصدار البرامج. في MLOps، يتم تكييف خطوط أنابيب CI/CD لتعلم الآلة. يقوم خط أنابيب CI/CD لتعلم الآلة تلقائياً بتشغيل إعادة تدريب النموذج عند توفر كود أو بيانات جديدة، وإجراء اختبارات آلية، وإصدار النموذج الناتج، ونشره في الإنتاج، مما يتيح تحديثات سريعة وموثوقة للنماذج.

خط أنابيب CI/CD مبسط لعمليات MLOps

توضح الخطوات التالية خط أنابيب CI/CD آلياً نموذجياً لنموذج تعلم الآلة في سياق أمني:

- (1) إيداع الكود: يقوم مهندس بإيداع كود جديد، مثل نص برمجي محسن لهندسة الميزات أو خوارزمية جديدة، في مستودع التحكم في الإصدارات مثل Git.
- (2) تشغيل CI: يؤدي هذا الإيداع إلى تشغيل خط أنابيب التكامل المستمر (CI) تلقائياً باستخدام أداة مثل Jenkins أو GitHub Actions.
- (3) سحب البيانات والتحقق منها: يسحب خط الأنابيب أحدث البيانات الأمنية المصنفة من بحيرة بيانات أو متجر ميزات ويجري فحوصات التحقق لضمان جودتها وسلامتها.
- (4) إعادة تدريب النموذج: يقوم النظام تلقائياً بإعادة تدريب نموذج تعلم الآلة باستخدام الكود الجديد والبيانات الحديثة.
- (5) اختبار وتقييم النموذج: يتم تقييم النموذج المدرب حديثاً على مجموعة اختبار مخصصة. تتم مقارنة مقاييس أدائه (مثل درجة F1 ومعدل الإيجابيات الكاذبة) مقابل عتبات محددة مسبقاً وأداء النموذج المنشور حالياً.
- (6) إصدار وتسجيل النموذج: إذا اجتاز النموذج الجديد جميع الاختبارات، يتم إصدار وتسجيله في سجل نماذج model registry مثل MLflow. يقوم السجل بتخزين قطعة أثرية النموذج وجميع البيانات الوصفية المرتبطة بها.
- (7) نشر CD: يتم نشر النموذج الذي تم التحقق منه تلقائياً. قد يكون هذا في بيئة اختبار مرحلي لمزيد من الاختبارات (مثل الإصدار الكئاري canary release) أو مباشرة إلى بيئة الإنتاج.
- (8) المراقبة: بمجرد دخوله مرحلة الإنتاج، تتم مراقبة النموذج باستمرار بحثاً عن تدهور الأداء، وانحراف البيانات، وغيرها من المشكلات. إذا تم اكتشاف مشكلة، يتم إطلاق تنبيه، وقد تبدأ الدورة مرة أخرى.

3.14 مراقبة النماذج والتنبيه

إن نشر النموذج ليس نهاية دورة حياة MLOps. المراقبة المستمرة أمر بالغ الأهمية لضمان بقاء النموذج فعالاً وموثوقاً بمرور الوقت، خاصة في المشهد سريع التغير للأمن السيبراني.

1.3.14 اكتشاف انحراف النموذج

يشير انحراف النموذج Model drift إلى تدهور القوة التنبؤية للنموذج بسبب التغيرات في البيئة بعد النشر. تعتبر المراقبة بحثاً عن الانحراف مهمة أساسية.

- انحراف البيانات (Data Drift): يحدث هذا عندما يتغير توزيع بيانات الإدخال. على سبيل المثال، قد يرى نظام كشف التسلسل الشبكي نوعاً جديداً من حركة المرور المشفرة لم يتم تدريبه عليها. تتضمن مراقبة انحراف البيانات data drift مقارنة الخصائص الإحصائية (مثل المتوسط والانحراف المعياري) لبيانات الإدخال الحية ببيانات التدريب.

- انحراف المفهوم (Concept Drift): كما ذكرنا سابقاً، يحدث هذا عندما تتغير العلاقة بين المدخلات والمخرجات. قد يجد المهاجم طريقة جديدة لتعبئة حمولة ضارة تخضع مصنف البرامج الضارة الحالي. يصعب اكتشاف انحراف المفهوم مباشرة ولكنه غالباً ما يُستدل عليه من تدهور الأداء.

- تدهور الأداء (Performance Degradation): هذا هو المؤشر الأكثر مباشرة على وجود مشكلة. يتضمن تتبع مقاييس الأداء الرئيسية مثل درجة، F1 والدقة، والاستدعاء، ومعدل الإيجابيات الكاذبة بمرور الوقت. يعتبر الانخفاض المستمر في هذه المقاييس إشارة واضحة على أن النموذج لم يعد يعمل كما هو متوقع وقد يحتاج إلى إعادة تدريب.

2.3.14 مشكلات جودة البيانات

يمكن أن تقوض جودة البيانات الرديئة أداء النموذج بصمت. يجب أن تكون أنظمة المراقبة قادرة على اكتشاف المشكلات في تدفقات البيانات الواردة، مثل الزيادة المفاجئة في القيم المفقودة من مصدر سجل، أو حقول البيانات التالفة، أو أنواع البيانات غير المتوقعة. يمكن أن تسبب هذه المشكلات في قيام النموذج بعمل تنبؤات غير موثوقة ويجب معالجتها بسرعة.

3.3.14 أنظمة التنبيه الآلية

ليس من العملي أن يراقب البشر لوحات المعلومات على مدار الساعة طوال أيام الأسبوع. يجب أن يشمل نظام MLOps على تنبيه آلي. يتم تكوين المشغلات لإرسال إشعارات فورية إلى محلي الأمن ومهندسي MLOps عند تجاوز عتبات محددة مسبقاً. على سبيل المثال، قد يتم إرسال تنبيه إذا انخفضت درجة F1 للنموذج إلى ما دون 85.0 أو إذا تجاوز معدل القيم المفقودة في مصدر بيانات رئيسي 5%. يتيح ذلك استجابة سريعة لإصلاح المشكلة أو تشغيل خط أنابيب إعادة التدريب.

يوضح التصور في 83 كيف يمكن أن يتدهور أداء النموذج بمرور الوقت، مما يوضح سبب أهمية المراقبة والتنبيه المستمرين.

```
1 import matplotlib.pyplot as plt
2 import pandas as pd
3 import numpy as np
4
5 # Simulate dates over several months
6 dates = pd.to_datetime(pd.date_range(start='2025-01-01', periods=20, freq='W'))
7
8 # Simulate F1-scores, showing a period of stability followed by degradation
9 # and then a recovery after a hypothetical retraining
10 f1_scores = np.array([
11     0.92, 0.91, 0.90, 0.92, 0.89, 0.88, 0.87, 0.85, # Stable period
12     0.80, 0.75, 0.70, 0.65, 0.60, # Degradation
13     0.62, 0.65, 0.70, 0.80, 0.88, 0.91, 0.92 # Recovery after retraining
14 ])
15
16 plt.figure(figsize=(12, 6))
17 plt.plot(dates, f1_scores, marker='o', linestyle='-', color='blue', label='Model F1-score')
18 plt.axhline(y=0.85, color='red', linestyle='--', label='Alert Threshold (F1 < 0.85)')
19 plt.axvline(x=dates, color='green', linestyle=':', label='Start of Degradation')
20 plt.axvline(x=dates, color='purple', linestyle=':', label='Hypothetical Retraining Point')
21 plt.text(dates, 0.95, 'Concept Drift Starts', rotation=0, va='center', ha='left', color='green', fontsize=9)
22 plt.text(dates, 0.95, 'Retraining Deployed', rotation=0, va='center', ha='left', color='purple', fontsize=9)
23
24
25 plt.title('AI Model F1-score Over Time in Production (Simulated)')
26 plt.xlabel('Date'); plt.ylabel('F1-score')
27 plt.grid(True, linestyle=':', alpha=0.7)
28 plt.legend()
29 plt.tight_layout()
30 plt.show()
```

Listing 83: محاكاة وتصور تدهور أداء النموذج بمرور الوقت

4.14 استراتيجيات النشر

يتطلب نشر إصدار نموذج جديد بأمان في بيئة أمنية حية استراتيجية دقيقة لتقليل المخاطر والتحقق من الأداء. يتم استخدام العديد من الاستراتيجيات الشائعة.

1.4.14 الحاويات (Containerization)

الحاويات Containerization هي تقنية تقوم بتعبئة تطبيق، جنباً إلى جنب مع كل ما لديه من كود ومكتبات وتبعيات، في وحدة واحدة معزولة تسمى حاوية. وهذا يضمن تشغيل النموذج في نفس البيئة تماماً أثناء التطوير والاختبار والإنتاج، مما يمنع مشاكل "إنه يعمل على جهازي".

• الأدوات: Docker هي الأداة الأكثر شيوعاً لإنشاء وإدارة الحاويات [22]. Kubernetes

هي منصة تنسيق تستخدم لنشر وتوسيع وإدارة التطبيقات المعبأة في حاويات تلقائياً عبر مجموعة من الأجهزة [74].

• الفوائد: توفر الحاويات الاتساق وقابلية النقل وقابلية التوسع، مما يجعلها تقنية أساسية لعمليات MLOps الحديثة.

2.4.14 اختبار A/B

في اختبار A/B ، يتم تشغيل إصدارين مختلفين من النموذج (على سبيل المثال، النموذج الحالي 'A' ونموذج جديد 'B') بالتوازي في بيئة الإنتاج. يتم توجيه جزء من حركة المرور الحية إلى كل نموذج، ويتم مقارنة أدائهما جنباً إلى جنب على مقاييس مثل معدل الكشف ومعدل الإيجابيات الكاذبة. يسمح هذا باتخاذ قرار قائم على البيانات حول ما إذا كان النموذج الجديد أفضل حقاً في بيئة العالم الحقيقي قبل أن يحل محل القديم بالكامل.

3.4.14 عمليات النشر الكاري (Canary Deployments)

النشر الكاري canary deployment هو استراتيجية لتقليل المخاطر عن طريق طرح إصدار نموذج جديد ببطء. في البداية، يتم إصدار النموذج الجديد لمجموعة صغيرة جداً من المستخدمين أو حركة المرور (مجموعة "الكاري"). يراقب فريق MLOps أدائه عن كثب. إذا كان أدائه جيداً ولم يسبب أي مشاكل، يتم زيادة تعرضه تدريجياً حتى يتعامل مع كل حركة المرور. إذا ظهرت مشاكل، يمكن عكس الطرح بسرعة، مما يحد من التأثير على مجموعة الكاري الصغيرة فقط.

4.4.14 عمليات النشر الزرقاء/الخضراء (Blue/Green Deployments)

تتضمن هذه الاستراتيجية الحفاظ على بيئتي إنتاج متطابقتين، تسمى "الزرقاء" و "الخضراء". تقوم البيئة الزرقاء بتشغيل إصدار النموذج المستقر الحالي وتتعامل مع كل حركة المرور الحية. يتم نشر إصدار النموذج الجديد في البيئة الخضراء، حيث يمكن اختباره بالكامل دون التأثير على المستخدمين. بمجرد التحقق من صحة البيئة الخضراء، يتم تحويل حركة المرور على الفور من الزرقاء إلى الخضراء. يتم الاحتفاظ بالبيئة الزرقاء في وضع الاستعداد، مما يسمح بالعودة الفورية إذا أظهر النموذج الجديد مشاكل. يقلل هذا النهج من وقت التوقف عن العمل أثناء عمليات النشر.

5.4.14 استراتيجيات موثوقة للتراجع عن النماذج

جزء حاسم من أي استراتيجية نشر هو القدرة على التراجع بسرعة وأمان إلى إصدار نموذج سابق ومستقر. يجب أن تكون هذه عملية آلية، مدمجة في خط أنابيب CI/CD. إذا اكتشفت أنظمة المراقبة مشكلة خطيرة في نموذج تم نشره حديثاً (على سبيل المثال، ارتفاع حاد في الإيجابيات الكاذبة أو فشل حرج)، يجب أن يقوم خط الأنابيب تلقائياً بتشغيل التراجع إلى آخر إصدار جيد معروف، مما يضمن الحفاظ على الموقف الأمني.

5.14 قابلية المراقبة لأنظمة تعلم الآلة

تتعلق قابلية المراقبة بالقدرة على فهم الحالة الداخلية للنظام من خلال فحص مخرجاته الخارجية. بالنسبة لنظام تعلم آلة معقد في الإنتاج، يعني هذا جمع بيانات قياس عن بعد مفصلة لتشخيص المشكلات واستكشاف الأخطاء وإصلاحها. الركائز الثلاث لقابلية المراقبة هي السجلات logs، والتتبعات traces، والمقاييس metrics.

1.5.14 التسجيل (Logging)

يتضمن التسجيل Logging تسجيل أحداث مفصلة ومختومة زمنياً من نظام تعلم الآلة. لا يشمل هذا الأخطاء والتحذيرات فحسب، بل يشمل أيضاً معلومات قيمة لتصحيح الأخطاء والتدقيق، مثل ميزات الإدخال المحددة لتنبؤ عالي المخاطر، ومخرجات النموذج، وزمن انتقال الاستدلال. السجلات جيدة التنظيم ضرورية لتحليل الإخفاقات بعد وقوعها.

2.5.14 التتبع (Tracing)

يوفر التتبع Tracing طريقة لمتابعة طلب واحد أو نقطة بيانات واحدة أثناء تحركها عبر جميع مكونات نظام تعلم الآلة. على سبيل المثال، يمكن أن يظهر التتبع رحلة حزمة شبكة من استيعابها بواسطة خط أنابيب البيانات، مروراً باستخراج الميزات، إلى خدمة استدلال النموذج، وأخيراً إلى نظام التنبيه. يعد التتبع لا يقدر بثمن لتحديد اختناقات الأداء وفهم التفاعلات المعقدة في الأنظمة الموزعة.

3.5.14 المقاييس (Metrics)

المقاييس Metrics هي قياسات كمية ومجموعة لصحة النظام وأدائه بمرور الوقت. ويشمل ذلك المقاييس التشغيلية مثل استخدام وحدة المعالجة المركزية/وحدة معالجة الرسومات، واستخدام الذاكرة، ومعدل نقل الطلبات، بالإضافة إلى المقاييس الخاصة بالنموذج مثل معدل التنبؤات التي يتم إنشاؤها أو متوسط درجة ثقة التنبؤ. يتم عادةً جمع المقاييس في قاعدة بيانات سلاسل زمنية وتصورها في لوحات معلومات لتوفير نظرة عامة عالية المستوى على حالة النظام.

6.14 ممارسات عمليات تعلم الآلة الآمنة

بما أن أنظمة MLOps جزء أساسي من دفاعات المؤسسة، فيجب تأمينها. تتضمن عمليات MLOps الآمنة دمج أفضل الممارسات الأمنية في جميع مراحل دورة حياة تعلم الآلة.

1.6.14 مصدر البيانات وسلامتها

مصدر البيانات Data provenance يعني القدرة على تتبع البيانات وصولاً إلى أصلها. من الأهمية بمكان التأكد من أن بيانات التدريب تأتي من مصادر موثوقة ولم يتم العبث بها. يمكن الحفاظ على سلامة البيانات باستخدام التجزئة المشفرة للتحقق من أن البيانات لم يتم تغييرها أثناء التخزين أو النقل. كما أن ضوابط الوصول القوية على بحيرات البيانات ومتاجر الميزات ضرورية لمنع التعديلات غير المصرح بها.

2.6.14 تأمين خطوط أنابيب البيانات

يجب حماية البيانات أثناء النقل وفي حالة السكون. يجب تشفير جميع عمليات نقل البيانات داخل خط أنابيب تعلم الآلة، مثل من جامع السجلات إلى بحيرة البيانات، باستخدام بروتوكولات مثل TLS/SSL. يجب تشفير البيانات المخزنة في قواعد البيانات أو بحيرات البيانات أو التخزين السحابي في حالة السكون لحمايتها من الوصول غير المصرح به.

3.6.14 توقيع النماذج

بعد تدريب النموذج والتحقق من صحته، يجب توقيعه بشكل مشفر. ينشئ توقيع النموذج Model signing توقيعاً رقمياً لقطعة أثرية النموذج. عند تحميل النموذج للنشر، يمكن للنظام التحقق من هذا التوقيع للتأكد من أن النموذج أصلي ولم يتم استبداله بشكل ضار بإصدار مخترق.

4.6.14 النشر الآمن للنماذج

يجب نشر النماذج في بيئات معزولة ومحصنة. يعد استخدام صور حاويات أساسية آمنة، وتقليل سطح الهجوم عن طريق إزالة البرامج غير الضرورية أو المنافذ المفتوحة، وتشغيل الحاويات بأقل الامتيازات المطلوبة، كلها ممارسات حاسمة. يمكن أن يوفر وضع الحماية طبقة إضافية من العزل لعمليات استدلال النموذج.

5.6.14 الدفاعات ضد هجمات تسميم بيانات التدريب

هجوم تسميم البيانات data poisoning attack هو حيث يقوم الخصم بحقن بيانات ضارة عمداً في مجموعة التدريب لإفساد النموذج [7]. قد يؤدي هذا إلى إنشاء باب خلفي للمهاجم أو ببساطة تدهور أداء النموذج. تشمل الدفاعات التحقق الصارم من البيانات وتنقيتها، واستخدام خوارزميات كشف القيم الشاذة لتحديد عينات التدريب الشاذة، واستخدام طرق تجميع قوية في إعدادات التعلم الموزعة مثل التعلم الفيدرالي federated learning.

6.6.14 التحكم في الوصول للبنية التحتية لتعلم الآلة

التحكم الصارم في الوصول أمر أساسي. يجب استخدام التحكم في الوصول المستند إلى الأدوار (Role-Based Access Control - RBAC) لضمان أن المستخدمين والخدمات لديهم فقط الأذونات التي يحتاجونها تماماً لأداء وظائفهم. ينطبق هذا على جميع مكونات البنية التحتية لتعلم الآلة، بما في ذلك مستودعات الكود ومخازن البيانات وسجلات النماذج وبيئات النشر.

7.6.14 أمان سلسلة التوريد لتعلم الآلة

تشمل سلسلة توريد تعلم الآلة كل شيء من المكتبات مفتوحة المصدر والنماذج المدربة مسبقاً إلى مصادر البيانات وأدوات النشر. يمكن أن تؤدي ثغرة أمنية في أي من هذه المكونات إلى تعريض النظام بأكمله للخطر. تعد الممارسات مثل فحص التبعيات بحثاً عن الثغرات الأمنية المعروفة، واستخدام مصادر موثوقة للنماذج المدربة مسبقاً، والتحقق من سلامة جميع مكونات الطرف الثالث ضرورية لتأمين سلسلة التوريد.

7.14 مشروع عملي: خط أنابيب MLOps أساسي

يوضح هذا المشروع كيفية إعداد خط أنابيب MLOps مبسط لنموذج أمني باستخدام أداتين شائعتين مفتوحتي المصدر: DVC لإصدار البيانات و MLflow لتتبع التجارب وإدارة النماذج. سوف نستخدم مفهوم كاشف التصيد الاحتيالي من الفصول السابقة.

1.7.14 الخطوة 1: إعداد المشروع وإصدار البيانات باستخدام DVC

أولاً، نستخدم DVC (التحكم في إصدار البيانات) لتتبع مجموعة بياناتنا. يعمل DVC جنباً إلى جنب مع Git لإصدار الملفات الكبيرة، مثل مجموعات البيانات والنماذج، دون إنقال مستودع Git [34]. توضح الأوامر في 84 كيفية تهيئة DVC والبدء في تتبع ملف بيانات. يتتبع Git ملف البيانات الوصفية الصغير .dvc، بينما يدير DVC البيانات الفعلية في موقع تخزين بعيد.

```
1 # Assume you have your dataset in a folder, e.g., 'data/emails.csv'
2 # 1. Initialize DVC in your project root
3 dvc init
4
5 # 2. Add your data file to DVC tracking. This creates a .dvc file
6 # and adds the data file to .gitignore, so Git only tracks the .dvc file.
7 dvc add data/emails.csv
8
9 # 3. Commit the .dvc file (and .gitignore) to Git
10 git add data/emails.csv.dvc .gitignore
11 git commit -m "Add raw email data with DVC"
12
13 # 4. (Optional) If you modify data, run 'dvc add' again and commit to Git
14 # dvc add data/emails.csv
15 # git commit -m "Update email data (v2)"
```

Listing 84: أوامر تهيئة DVC وإصدار البيانات

2.7.14 الخطوة 2: تتبع التجارب باستخدام MLflow

بعد ذلك، نستخدم MLflow لتسجيل تجارب تدريب نماذجنا. MLflow هي منصة مفتوحة المصدر لإدارة دورة حياة تعلم الآلة، بما في ذلك تتبع التجارب، وتعبئة الكود، ونشر النماذج [76]. يوضح البرنامج النصي في 85 كيفية استخدام MLflow لتسجيل الملاحظات الفائقة (مثل `n\_estimators` ومقاييس الأداء (مثل `f1\_score` والنموذج المدرب نفسه كقطعة أثرية. يؤدي هذا إلى إنشاء سجل مفصل لكل جولة تدريب، مما يسهل مقارنة النتائج.

```

1 import mlflow
2 import mlflow.sklearn # For logging scikit-learn models
3 from sklearn.ensemble import RandomForestClassifier
4 from sklearn.metrics import f1_score
5 from sklearn.model_selection import train_test_split
6 import pandas as pd
7 import numpy as np
8
9 # Assume a dummy dataset for demonstration (replace with your actual data)
10 X_dummy = np.random.rand(100, 10)
11 y_dummy = np.random.randint(0, 2, 100)
12 X_train, X_test, y_train, y_test = train_test_split(X_dummy, y_dummy, test_size=0.2, random_state=42)
13
14 # Set an MLflow experiment name
15 mlflow.set_experiment("Phishing Detector Training")
16
17 # Start an MLflow run
18 with mlflow.start_run():
19     # Log parameters
20     n_estimators = 100
21     max_depth = 10
22     mlflow.log_param("n_estimators", n_estimators)
23     mlflow.log_param("max_depth", max_depth)
24
25     # Train your model (e.g., RandomForestClassifier from previous chapters)
26     model = RandomForestClassifier(n_estimators=n_estimators, max_depth=max_depth, random_state=42)
27     model.fit(X_train, y_train)
28
29     # Make predictions and log metrics
30     y_pred = model.predict(X_test)
31     f1 = f1_score(y_test, y_pred)
32     mlflow.log_metric("f1_score", f1)
33     print(f"MLflow Run ID: {mlflow.active_run().info.run_id}")
34     print(f"Logged F1-score: {f1:.2f}")
35
36     # Log the trained model as an artifact
37     mlflow.sklearn.log_model(model, "phishing_model")
38
39 print("\nMLflow run completed. To view runs, run 'mlflow ui' in your terminal.")

```

Listing 85: تسجيل جولة تدريب باستخدام MLflow

3.7.14 الخطوة 3: سجل النماذج

بعد إجراء العديد من التجارب، يمكنك تحديد أفضل النماذج أداءً وتسجيله في Model MLflow Registry. السجل هو مركز مركزي لإدارة دورة حياة النماذج، مما يسمح لك بإصدارها ونقلها عبر مراحل مثل Staging و Production. يوضح الكود في 86 كيفية تسجيل نموذج من جولة تدريب، وإعطائه اسمًا للاستخدام في الإنتاج.

4.7.14 الخطوة 4: النشر الأساسي (مفاهيمي)

مع وجود نموذج مسجل، فإن الخطوة الأخيرة هي النشر. في سيناريو العالم الحقيقي، سيتضمن ذلك تعبئة النموذج في حاوية Docker ونشره على منصة قابلة للتطوير مثل Kubernetes. لهذا

```

1  # For this demo, let's re-run the previous training and register it automatically
2  # (Assuming the setup from the previous script exists)
3  with mlflow.start_run(run_name="Register_Best_Model") as run:
4      n_estimators = 150
5      max_depth = 12
6      mlflow.log_param("n_estimators", n_estimators)
7      mlflow.log_param("max_depth", max_depth)
8
9      model = RandomForestClassifier(n_estimators=n_estimators, max_depth=max_depth, random_state=42)
10     model.fit(X_train, y_train)
11     y_pred = model.predict(X_test)
12     f1 = f1_score(y_test, y_pred)
13     mlflow.log_metric("f1_score", f1)
14
15     # Register the model directly from this run
16     registered_model = mlflow.sklearn.log_model(
17         sk_model=model,
18         artifact_path="phishing_model",
19         registered_model_name="PhishingDetectorProd" # Name in the Model Registry
20     )
21     print(f"\nModel version {registered_model.version} of 'PhishingDetectorProd' registered.")
22     print(f"Access registered model at: {registered_model.model_uri}")
23
24     # To transition the model's stage (e.g., to Production) in a real workflow:
25     # from mlflow.tracking import MlflowClient
26     # client = MlflowClient()
27     # client.transition_model_version_stage(
28     #     name="PhishingDetectorProd",
29     #     version=registered_model.version,
30     #     stage="Production"
31     # )
32     # print(f"Model version {registered_model.version} transitioned to Production.")

```

Listing 86: تسجيل نموذج في سجل نماذج MLflow

العرض التوضيحي، يوضح البرنامج النصي في 87 العملية المفاهيمية لتحميل النموذج المسجل من السجل لعمل تنبؤ على بيانات جديدة. هذا هو المنطق الأساسي الذي سيتم تضمينه في تطبيق خدمة النماذج، مثل نقطة نهاية Flask API.

5.7.14 الخطوة 5: التدريب المستمر (مفاهيمي)

الجزء الأخير من لغز MLOps هو أتمتة سير العمل هذا بالكامل. في نظام إنتاجي، سيتم تكوين منصة CI/CD مثل GitHub Actions لمراقبة مستودع Git. عند اكتشاف تغيير — إما دفع كود جديد أو إصدار بيانات جديد (يشار إليه بتغيير في ملف dvc). — سيتم تشغيل خط أنابيب CI/CD تلقائياً. وسيقوم بما يلي:

- (1) استخدام pull dvc لاسترداد أحدث البيانات.
- (2) تنفيذ البرنامج النصي للتدريب، الذي يسجل التجربة في MLflow.
- (3) تشغيل برنامج نصي للتقييم لاختبار النموذج الجديد.

```

1 # This part is conceptual for demonstration of loading a model
2 # In a real Flask/FastAPI app, you would load the model once at startup.
3 import mlflow.pyfunc
4 import pandas as pd
5 import numpy as np
6
7 # This would normally be defined in your app's environment
8 # registered_model_uri = "models:/PhishingDetectorProd/1" # Example URI
9 # For this demo, we use the object from the previous step
10 registered_model_uri = registered_model.model_uri
11
12 # Load the model from the MLflow Model Registry
13 latest_production_model = mlflow.pyfunc.load_model(registered_model_uri)
14
15 # Simulate a new incoming email feature vector for prediction
16 # (Ensure the shape matches the training data)
17 num_features = X_train.shape
18 new_email_features = pd.DataFrame(np.random.rand(1, num_features), columns=[f'feature_{i}' for i in range(num_features)])
19
20 prediction = latest_production_model.predict(new_email_features)
21 print(f"\nPrediction for new email: {prediction} (0: Legitimate, 1: Phishing)")
22
23 print("\nConceptual deployment: Model loaded and ready for inference.")
24 print("In a real setup, this would be an API endpoint.")

```

Listing 87: تحميل نموذج مسجل للاستدلال

(4) إذا كان النموذج الجديد متفوقاً، فسيتم تسجيله وربما نشره تلقائياً، لإكمال الدورة.

الخلاصة

يعد نقل نموذج تعلم الآلة من نموذج أولي بحثي إلى أداة للأمن السيبراني على مستوى الإنتاج عملية معقدة ولكنها ضرورية. أثبت هذا الفصل أن MLOps توفر الإطار اللازم لهذه الرحلة. لقد استكشفنا التحديات الفريدة والشديدة في مجال الأمن، بما في ذلك الانحراف المستمر للمفهوم concept drift، والتكلفة العالية للأخطاء، والهجمات العدائية المباشرة. لمعالجة هذه التحديات، قمنا بتفصيل دورة حياة MLOps، التي تغطي خطوط أنابيب البيانات، ومتاجر الميزات، والتكامل والتسليم المستمر لتعلم الآلة CI/CD for ML، ومراقبة النماذج الحاسمة. من خلال اعتماد استراتيجيات النشر مثل الإصدارات الكارية canary releases وبناء أنظمة قابلة للمراقبة وآمنة، يمكنك إنشاء دفاعات ذكاء اصطناعي ليست ذكية فحسب، بل أيضاً قوية وموثوقة وقابلة للتطوير. قدم المشروع العملي لمحة عن قوة أدوات مثل DVC و MLflow، التي تعمل على أتمتة الإصدار والتتبع، وتشكل أساس سير عمل MLOps حديث وفعال.

المختصرات والمصطلحات

اختبار أ/ب A/B Testing استراتيجية نشر يتم فيها تشغيل نسختين مختلفتين من النموذج بالتوازي على حركة المرور الحية لمقارنة أدائهما بشكل مباشر قبل اتخاذ قرار نهائي.

الهجوم التخاصمي Adversarial Attack تقنية يستخدمها المهاجمون لإنشاء مدخلات خادعة بشكل متعمد (مثل برمجيات خبيثة malware معدلة قليلاً) مصممة لخداع نموذج الذكاء الاصطناعي AI model ودفعه للقيام بتنبؤ خاطئ.

النشر الأزرق/الأخضر Blue/Green Deployment استراتيجية نشر تستخدم بيئتي إنتاج متطابقتين. يتم نشر نموذج جديد في البيئة غير النشطة ("الخضراء") للاختبار، وبمجرد التحقق منه، يتم تحويل حركة المرور الحية من البيئة القديمة ("الزرقاء")، مما يقلل من وقت التوقف عن العمل.

النشر التدريجي (كناري) Canary Deployment استراتيجية لتقليل المخاطر حيث يتم إطلاق نموذج جديد أولاً لمجموعة فرعية صغيرة من المستخدمين أو حركة المرور. إذا كان أدائه جيداً، يتم تعميمه تدريجياً للجميع.

التكامل المستمر والتسليم المستمر CI/CD التكامل المستمر والتسليم المستمر. مجموعة من الممارسات الآلية لبناء واختبار ونشر البرمجيات، والتي تم تكييفها في عمليات تعلم الآلة MLOps لأتمتة دورة حياة نموذج تعلم الآلة machine learning model بأكملها.

انحراف المفهوم Concept Drift ظاهرة تتغير فيها العلاقة بين مدخلات النموذج والمخرجات الصحيحة بمرور الوقت. في مجال الأمن السيبراني cybersecurity، يحدث هذا باستمرار حيث يبتكر المهاجمون تكتيكات جديدة.

الحوسنة Containerization عملية تجميع تطبيق وجميع تبعياته (الكود، المكتبات، الإعدادات) في وحدة واحدة معزولة تسمى حاوية container (باستخدام دوكر Docker مثلاً)، مما يضمن تشغيله بشكل متسق في أي مكان.

انحراف البيانات Data Drift نوع من تدهور أداء النموذج يحدث عندما تختلف الخصائص الإحصائية للبيانات الحية التي يراها النموذج في بيئة الإنتاج عن البيانات التي تم تدريبه عليها.

مسار البيانات Data Pipeline سير عمل آلي يقوم بابتلاع البيانات الأولية وتنظيفها وتحويلها وإعدادها، مما يجعلها جاهزة للاستخدام من قبل نموذج تعلم الآلة machine learning model.

هجوم تسميم البيانات Data Poisoning Attack تهديد أمني يقوم فيه المهاجم بحقن بيانات خبيثة أو مضللة سراً في مجموعة تدريب النموذج لإفساده، مما يخلق أبواباً خلفية backdoors أو يؤدي إلى تدهور أدائه.

مصدرية البيانات Data Provenance التاريخ الموثق لبيانات معينة، من أصلها إلى حالتها الحالية، وهو أمر بالغ الأهمية لضمان سلامة البيانات data integrity وموثوقيتها.

التحكم في إصدارات البيانات DVC التحكم في إصدارات البيانات. أداة تستخدم جنباً إلى جنب مع جت Git لإدارة وإصدار مجموعات البيانات الكبيرة وملفات النماذج، دون تخزينها مباشرة في مستودع جت Git repository.

الذكاء الاصطناعي القابل للتفسير Explainable AI (XAI) مجال من مجالات الذكاء الاصطناعي artificial intelligence يركز على تطوير نماذج يمكنها شرح قراراتها بطريقة مفهومة للبشر، وهو أمر حيوي لمخالي الأمن الذين يحتاجون إلى التحقق من صحة التنبؤات.

السلبي الخاطئ False Negative (FN) خطأ حرج حيث يفشل نموذج أمني في اكتشاف تهديد حقيقي، مثل السماح بمرور بريد إلكتروني للتصيد الاحتيالي phishing email أو عدم كشف اختراق للشبكة network intrusion.

الإيجابي الخاطئ False Positive (FP) خطأ يقوم فيه نموذج أمني بتصنيف نشاط حميد (غير ضار) بشكل غير صحيح على أنه خبيث، مما قد يؤدي إلى إرهاق التنبيهات alert fatigue لمخالي الأمن.

مخزن السمات Feature Store نظام مركزي لتخزين وإدارة ومشاركة السمات features المنسقة لنماذج تعلم الآلة machine learning models، مما يضمن الاتساق بين بيئات التدريب والإنتاج.

إم إل فلو MLflow منصة مفتوحة المصدر مصممة لإدارة دورة حياة تعلم الآلة machine learning lifecycle من البداية إلى النهاية، بما في ذلك تتبع التجارب وتغليف النماذج وإدارة عمليات النشر.

عمليات تعلم الآلة MLOps عمليات تعلم الآلة. تخصص يجمع بين تعلم الآلة machine learning، وتطوير البرمجيات (DevOps)، وهندسة البيانات data engineering لأتمتة وتبسيط عملية بناء ونشر وصيانة نماذج تعلم الآلة ML models في بيئة الإنتاج.

انحراف النموذج Model Drift الانخفاض العام في أداء النموذج بمرور الوقت بعد نشره، وغالباً ما يكون سببه تغيرات في البيانات أو البيئة الأساسية (أي انحراف البيانات data drift أو انحراف المفهوم concept drift).

سجل النماذج Model Registry مستودع مركزي لتخزين نماذج تعلم الآلة-machine learning models المدربة وإصدارها وإدارتها، مما يسمح بالتتبع والنشر المنهجي.

توقيع النموذج Model Signing ممارسة أمنية تتمثل في تطبيق توقيع إلكتروني cryptographic signature على نموذج مدرب للتحقق من أصالته والتأكد من عدم العبث به قبل نشره. القابلية للمراقبة Observability القدرة على فهم الحالة الداخلية لنظام ما عن طريق تحليل مخرجاته الخارجية، ويتم تحقيق ذلك عادةً عن طريق جمع ومراقبة السجلات logs، والمقاييس metrics، والآثار traces.

التحكم في الوصول المستند إلى الأدوار RBAC التحكم في الوصول المستند إلى الأدوار. سياسة أمنية تقيد الوصول إلى النظام للمستخدمين المصرح لهم بناءً على أدوارهم ومسؤولياتهم المحددة داخل المؤسسة.

None

باب 15

تقييم النموذج والتحقق المستمر في مراكز العمليات الأمنية

مقدمة

بمجرد بناء نموذج ذكاء اصطناعي واجتيازه للاختبارات الأولية، فإن رحلته لم تنته بعد. ففي البيئة الديناميكية لمركز العمليات الأمنية (SOC) --- وهي وحدة مركزية تتعامل مع القضايا الأمنية --- يمكن أن يتغير أداء النموذج بمرور الوقت. فالتحديات التي تم تدريبه على اكتشافها يمكن أن تتطور، وخصائص حركة مرور الشبكة العادية يمكن أن تتغير. يتجاوز هذا الفصل مقاييس الأداء الأولية لاستكشاف الممارسات الحاسمة للتقييم والتحقق المستمر لنماذج الذكاء الاصطناعي في بيئة تشغيلية حية لمركز العمليات الأمنية. سنركز على الاستراتيجيات التي تقيس التأثير التشغيلي الحقيقي للنموذج وقدرته على التكيف. سنتعلم تقنيات التحقق المتقدمة، وطرق اكتشاف انحراف المفهوم والتكيف معه، وأهمية إبقاء العنصر البشري في الحلقة، وكيفية استخدام الذكاء الاصطناعي القابل للتفسير (XAI) لبناء الثقة وتحسين التحقيقات.

1.15 تقنيات التحقق المتقدمة

يتطلب نشر نموذج في بيئة الإنتاج أكثر من مجرد أداء جيد على مجموعة اختبار ثابتة. نحن بحاجة إلى طرق لتقييم أدائه بأمان على البيانات الحية ومقارنته بالحوادث التاريخية.

1.1.15 الاختبار الكاري Canary Testing

الاختبار الكاري (Canary testing) هو استراتيجية لإطلاق نموذج جديد على مجموعة فرعية صغيرة ومراقبة من حركة مرور الإنتاج قبل النشر الكامل. على سبيل المثال، يمكن لفريق أمني توجيه 5% من حركة مرور الشبكة الخاصة به من خلال نموذج جديد لنظام كشف التسلسل الشبكي (NIDS). ثم يقوم الفريق بمقارنة دقيقة لحجم التنبيهات، ومعدل الإيجابيات الحقيقية، ومعدل الإيجابيات الخاطئة مع النموذج الحالي الموثوق به. يتيح هذا التعرض المحدود مراقبة الأداء في العالم الحقيقي مع تقليل مخاطر التأثيرات السلبية واسعة النطاق، مثل فيضان هائل من التنبيهات الخاطئة.

2.1.15 اختبار أ/ب A/B Testing

يتضمن اختبار أ/ب (A/B testing) تشغيل نسختين أو أكثر من النماذج بالتوازي، حيث تتلقى كل منها شريحة مختلفة من حركة المرور الحية. على سبيل المثال، قد يتعامل النموذج "أ" (النموذج الحالي في الإنتاج) مع 50% من حركة المرور، بينما يتعامل النموذج "ب" (النموذج المرشح الجديد) مع الـ 50% الأخرى. تتيح هذه المقارنة المباشرة للمشغلين قياس مقاييس الأداء الرئيسية جنباً إلى جنب. يمكن أن تشمل هذه المقاييس دقة الكشف، والدقة، والاستدعاء للأحداث الخبيثة، ومعدلات الإيجابيات الخاطئة، وزمن الاستدلال. يوفر اختبار أ/ب دليلاً كميًا واضحاً لتحديد أي نموذج يؤدي بشكل أفضل في ظل ظروف العالم الحقيقي المتطابقة.

3.1.15 الاختبار الرجعي بالبيانات التاريخية Backtesting with Historical Data

يقوم الاختبار الرجعي (Backtesting) بتقييم أداء نموذج جديد على بيانات أمنية تاريخية لم يرها أثناء التدريب. يجب أن تتضمن هذه البيانات حوادث أمنية سابقة ذات نتائج معروفة، مما يسمح بحاكاة بأثر رجعي لفعالية النموذج. على سبيل المثال، يمكنك تغذية النموذج ببيانات سجلات لمدة عام كامل، بما في ذلك العديد من حوادث الاختراق المؤكدة. تساعد هذه العملية في الإجابة على السؤال: "لو كان لدينا هذا النموذج قبل عام، فهل كان سيكتشف هذه التهديدات المعروفة؟" إنها طريقة قوية للتحقق من صحة النموذج ضد هجمات حقيقية ومؤكدة من الماضي.

4.1.15 مراقبة الأداء في الوقت الفعلي مقابل الحقائق المؤكدة Verified Ground Truth

إن المقياس الأكثر دقة لأداء النموذج الحالي يأتي من المقارنة المستمرة لتنبؤاته الحية مع حقيقة مؤكدة (verified ground truth). هذه الحقيقة المؤكدة هي النتيجة المؤكدة التي تقدمها مصادر موثوقة مثل محلي الأمن البشريين، أو تحليل الملفات المشبوهة في بيئة معزولة (sandbox analysis)، أو خلاصات استخبارات التهديدات الخارجية. توفر هذه التقنية بطاقة أداء شبه فورية لدقة النموذج في بيئة الإنتاج. ومع ذلك، يوجد تحد كبير: غالباً ما يكون الحصول على تسميات الحقائق المؤكدة في الوقت الفعلي أمراً صعباً وبطيئاً. يستغرق التحليل البشري وقتاً، مما يعني غالباً وجود تأخير بين تنبؤ النموذج والتحقق منه.

2.15 اكتشاف انحراف المفهوم والتكيف معه Concept Drift Detection & Adaptation

في مجال الأمن السيبراني، يتغير مشهد البيانات باستمرار. يبتكر المهاجمون تقنيات جديدة، ويتطور سلوك المستخدم العادي. تُعرف هذه الظاهرة باسم انحراف المفهوم *concept drift*، حيث تتغير الخصائص الإحصائية للبيانات بمرور الوقت، مما يؤدي إلى تدهور أداء النموذج. من الأهمية بمكان اكتشاف هذا الانحراف وتكييف النموذج للحفاظ على فعاليته.

1.2.15 استراتيجيات أعمق لاكتشاف الانحراف

توجد العديد من الخوارزميات والأساليب المتقدمة للكشف بشكل منهجي عن انحراف المفهوم [27].

- **ADaptive ADWIN (ADaptive ADWIN):** تحافظ هذه الخوارزمية على نافذة منزقة من البيانات الحديثة. تكتشف الانحراف عن طريق المقارنة الإحصائية بين نافذتين فرعيتين داخل هذه النافذة الأكبر. إذا كانت خصائص النافذتين الفرعيتين (مثل المتوسط أو التباين) مختلفة بشكل كبير، فإن ADWIN تشير إلى حدوث انحراف. ثم تقوم بتقليص نافذتها إلى البيانات الأحدث، متناسبة بشكل فعال الأنماط الأقدم والقديمة.

- **DDM (Drift Detection Method):** تركز DDM على مراقبة معدل خطأ النموذج بمرور الوقت. تحسب المتوسط المتحرك والانحراف المعياري للخطأ. إذا ارتفع معدل الخطأ الحالي فوق عتبة معينة (على سبيل المثال، انحرافان معياريان أو ثلاثة فوق المتوسط)، فإنها

تطلق تحذيراً أو إشارة اكتشاف انحراف. يشير هذا إلى أن النموذج أصبح أقل دقة وقد يحتاج إلى إعادة تدريب.

• التحكم الإحصائي في العمليات (SPC): تُستخدم تقنيات التحكم الإحصائي في العمليات (Statistical Process Control - SPC)، المقتبسة من مراقبة الجودة الصناعية، مخططات التحكم لمراقبة مقاييس أداء النموذج وتوزيعات البيانات. يرسم مخطط التحكم مقياساً بمرور الوقت ويتضمن حدود تحكم عليا وسفلى. إذا تحرك المقياس خارج هذه الحدود، فإنه يشير إلى عملية خارجة عن السيطرة، وهو ما يعني في هذا السياق انحراف المفهوم.

• مراقبة توزيع البيانات مباشرة: تقارن هذه الطريقة مباشرة توزيع سمات البيانات الواردة مع توزيع بيانات التدريب الأصلية. يمكن للاختبارات الإحصائية، مثل اختبار كولموغوروف-سميرنوف (Kolmogorov-Smirnov test)، تحديد ما إذا كان التوزيعان مختلفين بشكل كبير. يشير الاختلاف الكبير إلى أن البيانات الأساسية قد تغيرت، وهو مؤشر قوي على الانحراف.

2.2.15 مشغلات إعادة التدريب الآلية والتعلم التكيفي

عندما يتم الكشف عن انحراف المفهوم، غالباً ما تكون الاستجابة اليدوية بطيئة جداً. تستخدم الأنظمة الحديثة مشغلات إعادة تدريب آلية لبدء عملية إعادة تدريب النموذج تلقائياً. يضمن هذا تحديث النموذج ببيانات جديدة لمواجهة تدهور الأداء. إلى جانب إعادة التدريب الكاملة، يوفر التعلم التكيفي *adaptive learning* (المعروف أيضاً بالتعلم عبر الإنترنت أو التعلم التزايدى) نهجاً أكثر ديناميكية. فبدلاً من إعادة التدريب من الصفر، تقوم النماذج التكيفية بتحديث معلماتها باستمرار مع كل قطعة بيانات جديدة تعالجها. يسمح هذا بتكيف أسرع وأكثر كفاءة مع التغييرات الطفيفة والمستمرة في تدفق البيانات. يقدم تحليل خوارزمي مستوحى من طريقة اكتشاف الانحراف (DDM) مثلاً واضحاً لكيفية عمل ذلك.

(1) مراقبة معدل الخطأ: يتتبع النظام باستمرار معدل خطأ تنبؤ النموذج على نقاط البيانات الواردة التي تصبح حقيقتها المؤكدة متاحة.

(2) الحفاظ على مستويات التحذير/الانحراف: يقوم بحساب متوسط الانحراف المعياري لمعدل الخطأ على مدى نافذة تاريخية مستقرة.

(3) اكتشاف التحذير: إذا تجاوز معدل الخطأ الحالي المتوسط بمضاعف معين للانحراف المعياري (على سبيل المثال، انحرافان معياريان)، يصدر النظام تحذيراً. يشير هذا إلى احتمال حدوث انحراف ويستدعي مراقبة أدق.

(4) اكتشاف الانحراف: إذا تجاوز معدل الخطأ عتبة أعلى (على سبيل المثال، 3 انحرافات معيارية) أو بقي في منطقة التحذير لفترة طويلة، يعلن النظام رسمياً عن انحراف المفهوم. يؤدي هذا الإجراء إلى استجابة آلية، مثل إعادة تدريب النموذج أو إرسال تنبيه إلى مهندس عمليات تعلم الآلة (MLOps).

3.15 التحقق بوجود الإنسان في الحلقة Human-in-the-Loop (HITL) Validation

نماذج الذكاء الاصطناعي هي أدوات قوية، لكنها ليست معصومة من الخطأ. يعد دمج الخبرة البشرية في سير عمل تعلم الآلة أمراً ضرورياً لتحسين أداء النموذج، والتحقق من صحة التنبؤات، وبناء الثقة في النظام.

1.3.15 الإنسان في الحلقة وآليات التغذية الراجعة

الإنسان في الحلقة *Human-in-the-Loop (HITL)* هو عملية تجمع بين ذكاء الآلة والذكاء البشري. في مركز العمليات الأمنية، يعني هذا أن محلي الأمن يراجعون ويتحققون ويصححون مخرجات نموذج الذكاء الاصطناعي بنشاط. هذا مهم بشكل خاص للأحداث عالية الخطورة أو للتنبؤات التي يكون فيها لدى النموذج ثقة منخفضة. لإنجاح ذلك، نحتاج إلى حلقات تغذية راجعة فعالة تسمح للمحللين بما يلي:

- تمييز الإيجابيات/السلبيات الخاطئة: يمكن للمحللين تصحيح أخطاء النموذج عن طريق تمييز تنبيه بأنه إيجابي خاطئ (حميد) أو تحديد هجوم تم تفويته (سلي خاطئ).
- تصحيح التصنيفات غير الصحيحة: بالنسبة لنموذج يصنف الهجمات إلى أنواع مختلفة، يمكن للمحلل تغيير تصنيف غير صحيح (على سبيل المثال، تغيير "برامج الفدية" إلى "تسريب البيانات").
- توفير تسميات جديدة: يمكن للمحللين توفير تسميات الحقيقة المؤكدة لنقاط بيانات جديدة أو غير مؤكدة، مما يخلق أمثلة لتدريب جديدة قيمة.

- تحديد أولويات المراجعة: يمكن للنظام تقديم التنبيهات للمحللين مرتبة حسب درجة ثقة النموذج، مما يسمح لهم بتركيز انتباههم على التنبؤات الأكثر غموضاً أو خطورة.

2.3.15 التعلم النشط Active Learning

التعلم النشط *Active learning* هو نهج في تعلم الآلة يجعل عملية الإنسان في الحلقة (HITL) أكثر كفاءة. فبدلاً من اختيار البيانات بشكل عشوائي للمراجعة البشرية، يستعلم النموذج بذكاء من محلل بشري لتسمية نقاط البيانات الجديدة التي ستكون الأكثر إفادة لتحسينه. قد يطلب النموذج تسميات لأمثلة هو الأكثر تردداً بشأنها أو تلك التي تقع بالقرب من حدود قراره. تقلل هذه الاستراتيجية بشكل كبير من تكلفة وجهد التسمية اليدوية من خلال تركيز الخبرة البشرية بالضبط حيث تشتد الحاجة إليها، مما يسرع من تحسين النموذج. من خلال قياس تأثير هذه التغذية الراجعة البشرية على أداء النموذج بمرور الوقت (على سبيل المثال، تتبع انخفاض الإيجابيات الخاطئة بعد شهر من تصحيحات HITL)، يمكن للمؤسسات تبرير الاستثمار في هذه العمليات التعاونية.

4.15 الذكاء الاصطناعي القابل للتفسير للأمن Explainable AI (XAI) for Security

لكي يثق محللو الأمن في نموذج الذكاء الاصطناعي ويستخدمونه بفعالية، يحتاجون إلى فهم لماذا يتخذ قراراً معيناً. هذا هو هدف الذكاء الاصطناعي القابل للتفسير (XAI). الذكاء الاصطناعي القابل للتفسير (*Explainable AI - XAI*) هو مجموعة من الأساليب التي تساعد المستخدمين البشريين على فهم وتفسير مخرجات نماذج تعلم الآلة، خاصة النماذج المعقدة "الصندوق الأسود" ("black-box" models) مثل الشبكات العصبية العميقة.

1.4.15 تقنيات للسياق الأمني

تعتبر العديد من تقنيات XAI القوية مفيدة بشكل خاص في السياق الأمني.

- **exPlanations): Additive (SHapley SHAP** استناداً إلى نظرية الألعاب، تشرح SHAP مخرجات أي نموذج تعلم آلة عن طريق تعيين قيمة أهمية لكل سمة لتنبؤ معين [48]. تُظهر هذه القيمة، التي تسمى قيمة شابلي، مدى مساهمة كل سمة في دفع التنبؤ نحو

نتيجة معينة (على سبيل المثال، "خيث"). يمكن ل SHAP تقديم تفسيرات محلية (لتنبؤ واحد) وتفسيرات عالمية (أهمية السمات بشكل عام).

• **Explanations): Model-agnostic Interpretable (Local LIME** تشرح

LIME تنبؤاً عن طريق إنشاء نموذج أبسط وقابل للتفسير (مثل نموذج خطي) يقارب سلوك النموذج المعقد في المنطقة المجاورة لذلك التنبؤ المحدد [66]. تعمل عن طريق تغيير سمات الإدخال قليلاً وملاحظة كيفية تغير التنبؤ، مما يكشف عن السمات الأكثر تأثيراً في ذلك القرار الفردي.

• **التفسيرات المضادة للواقع (Counterfactual Explanations):** تصف هذه التفسيرات

أصغر تغيير في المدخلات من شأنه أن يغير تنبؤ النموذج. على سبيل المثال: "تم تمييز هذا الاتصال الشبكي على أنه خيث، ولكن لو كان منفذ الوجهة 80 بدلاً من 445، لكان قد تم تصنيفه على أنه حميد." يساعد هذا المحللين على فهم حدود قرار النموذج.

جانب رئيسي آخر من XAI هو توصيل عدم اليقين في النموذج. فبدلاً من مجرد إظهار تنبؤ ثنائي ("خيث" أو "حميد")، يجب أن يقدم النظام أيضاً درجة ثقة (على سبيل المثال، "ثقة بنسبة 95%"). يتيح هذا للمحللين تحديد أولويات تحقيقاتهم، مع التركيز على التنبؤات منخفضة الثقة حيث يكون النموذج أقل يقيناً وتكون الرقابة البشرية أكثر قيمة. يوضح الكود في القائمة 88 كيفية تطبيق SHAP لتفسير تنبؤات نموذج الغابة العشوائية Random Forest.

5.15 التكامل مع الأدوات الأمنية

تتحقق القيمة الحقيقية لنموذج الذكاء الاصطناعي عندما يتم دمج مخرجاته بسلسلة في سير عمل مركز العمليات الأمنية (SOC) الحالي. الهدف هو ضمان أن تكون الرؤى الناتجة عن الذكاء الاصطناعي قابلة للاستهلاك والتنفيذ بسهولة من قبل الأدوات الأمنية، مما يزيد من الأتمتة ويقلل من الجهد اليدوي.

1.5.15 تكامل SIEM

تقوم أنظمة إدارة معلومات وأحداث الأمان (Security Information and Event Management - SIEM) بتجميع وتحليل بيانات السجلات من جميع أنحاء البنية التحتية لتكنولوجيا المعلومات في المؤسسة. يمكن لنماذج الذكاء الاصطناعي تعزيز قدرات SIEM بشكل كبير من خلال:

```

1 import shap
2 import numpy as np
3 import pandas as pd
4 from sklearn.ensemble import RandomForestClassifier
5 from sklearn.model_selection import train_test_split
6 from sklearn.datasets import make_classification
7 import matplotlib.pyplot as plt
8
9 # 1. Create a synthetic cybersecurity dataset
10 X, y = make_classification(n_samples=200, n_features=10, n_informative=5,
11                           n_redundant=0, n_classes=2, random_state=42)
12 feature_names = [f"feature_{i}" for i in range(X.shape)]
13 X_df = pd.DataFrame(X, columns=feature_names)
14 y_df = pd.Series(y)
15
16 # Split data
17 X_train, X_test, y_train, y_test = train_test_split(X_df, y_df, test_size=0.2, random_state=42)
18
19 # 2. Train a simple sklearn model
20 model_xai = RandomForestClassifier(n_estimators=100, random_state=42)
21 model_xai.fit(X_train, y_train)
22 print("RandomForestClassifier trained for XAI demo.")
23
24 # 3. Create a SHAP explainer
25 explainer = shap.TreeExplainer(model_xai)
26
27 # 4. Calculate SHAP values for the test set
28 shap_values = explainer.shap_values(X_test)
29 print(f"\nSHAP values shape (for class 0, class 1): {shap_values.shape}, {shap_values.shape}")
30 print("Expected value for class 1 (malicious):", explainer.expected_value)
31
32 # 5. Visualize a single prediction's explanation
33 test_preds = model_xai.predict(X_test)
34 malicious_indices = np.where(test_preds == 1)
35 if len(malicious_indices) > 0:
36     example_idx = malicious_indices # Take the first one
37     print(f"\nExplaining prediction for test example index {example_idx} (True Label: {y_test.iloc[example_idx]}, Predicted Label
38
39     shap.initjs()
40     shap.force_plot(explainer.expected_value, shap_values[example_idx,:], X_test.iloc[example_idx,:],
41                     matplotlib=True, show=False)
42     plt.title(f"SHAP Force Plot for a Malicious Prediction (Example {example_idx})")
43     plt.tight_layout()
44     plt.show()
45
46 # 6. Summary plot (overall feature importance)
47 print("\n--- SHAP Summary Plot (Overall Feature Importance) ---")
48 shap.summary_plot(shap_values, X_test, plot_type="bar", show=False)
49 plt.title("SHAP Feature Importance for Malicious Class (Class 1)")
50 plt.tight_layout()
51 plt.show()
52
53 shap.summary_plot(shap_values, X_test, show=False)
54 plt.title("SHAP Beeswarm Plot for Malicious Class (Class 1)")
55 plt.tight_layout()
56 plt.show()

```

Listing 88: تطبيق SHAP لشرح تنبؤات النموذج، مع عرض تفصيلي لتنبؤ واحد وأهمية السمات بشكل عام.

- إثراء التنبيهات: يمكن لنموذج الذكاء الاصطناعي إضافة سياق قيم إلى تنبيه SIEM الخاطئ. على سبيل المثال، قد يضيف معلومات مثل "likely phishing"، "classfication" أو

port". destination "unusual "explanation": أو: 95.0 "anomaly_score":
يمنح هذا المحللين رؤية فورية قابلة للتنفيذ.

• تحديد أولويات الفرز: يمكن استخدام الدرجات التي يولدها الذكاء الاصطناعي (مثل درجات الثقة أو المخاطر) لتحديد أولويات قائمة انتظار التنبيهات تلقائياً. يتم دفع التنبيهات عالية الثقة إلى الأعلى للمراجعة البشرية الفورية، مما يساعد على مكافحة إرهاب التنبيهات ويضمن معالجة الحوادث الحرجة أولاً.

2.5.15 تكامل كتيبات التشغيل SOAR

تُستخدم منصات تنسيق وأتمتة واستجابة الأمان (*Security Orchestration, Automation, and Response - SOAR*) لأتمتة سير العمل الأمني، المعروف باسم كتيبات التشغيل (playbooks). يمكن أن تكون تنبؤات الذكاء الاصطناعي بمثابة مشغلات ومدخلات قوية لكتيبات التشغيل هذه.

• تشغيل إجراءات الاستجابة الآلية: يمكن لتنبؤ عالي الثقة من نموذج ذكاء اصطناعي أن يشغل مباشرة كتيب تشغيل SOAR. على سبيل المثال، إذا اكتشف نموذج ذكاء اصطناعي برامج ضارة على نقطة نهاية بثقة 99%، فيمكن لكتيب التشغيل عزل هذا المضيف تلقائياً عن الشبكة لمنع انتشار التهديد.

• إثراء كتيبات التشغيل: SOAR يمكن تغذية الرؤى الناتجة عن الذكاء الاصطناعي، مثل نوع الهجوم المتوقع أو تقنيات MITRE ATT&CK المرتبطة به، في كتيبات التشغيل SOAR لتمكين استجابات أكثر ذكاءً وديناميكية.

يمتد هذا التكامل إلى أدوات أخرى أيضاً. يمكن لتنبيهات الذكاء الاصطناعي إنشاء تذاكر تلقائياً في أنظمة إدارة الحوادث، ويمكن تغذية مؤشرات الاختراق (Indicators of Compromise - IOCs) المكتشفة حديثاً في منصات استخبارات التهديدات لتحسين الكشف عبر النظام البيئي الأمني بأكمله.

6.15 مشروع عملي: اكتشاف الانحراف و XAI

يوضح هذا المشروع كيفية محاكاة انحراف المفهوم، وتنفيذ طريقة إحصائية بسيطة لاكتشافه، واستخدام XAI لفهم سبب فشل النموذج في التعامل مع البيانات المنحرفة.

1.6.15 الخطوة 1: محاكاة انحراف المفهوم

أولاً، نقوم بإنشاء مجموعتي بيانات. تمثل `baseline\_data` الظروف العادية التي تم تدريب النموذج عليها. وتمثل `drifted\_data` بيئة جديدة تغيرت فيها أنماط البيانات الأساسية. توضح القائمة 89 كيفية إنشاء هذه البيانات وتوضح أن النموذج المدرب على البيانات الأساسية يكون أداؤه ضعيفاً على البيانات المنحرفة.

```
1 import numpy as np
2 import pandas as pd
3 from sklearn.ensemble import RandomForestClassifier
4 from sklearn.metrics import accuracy_score, f1_score
5 from scipy.stats import ks_2samp
6 import matplotlib.pyplot as plt
7
8 # 1. Generate baseline data (e.g., normal network traffic features)
9 np.random.seed(0)
10 baseline_data = np.random.normal(loc=0, scale=1, size=(500, 5))
11 baseline_labels = np.random.randint(0, 2, 500)
12
13 # 2. Generate drifted data (new attack patterns, different feature distribution)
14 drifted_data = np.random.normal(loc=1, scale=1.5, size=(500, 5))
15 drifted_data[:, 0] += 2 # Significant shift in feature_0
16 drifted_data[:, 1] -= 1 # Shift in feature_1
17 drifted_labels = np.where(drifted_data[:, 0] > 1, 1, 0) # Feature_0 now determines label
18
19 # Combine into DataFrames for clarity
20 feature_names_drift = [f'f_{i}' for i in range(5)]
21 df_baseline = pd.DataFrame(baseline_data, columns=feature_names_drift)
22 df_drifted = pd.DataFrame(drifted_data, columns=feature_names_drift)
23
24 # Train a model on baseline data
25 model_drift_detection = RandomForestClassifier(n_estimators=50, random_state=42)
26 model_drift_detection.fit(df_baseline, baseline_labels)
27 print("Model trained on baseline data.")
28
29 # Evaluate model performance on baseline and then on drifted data
30 baseline_pred = model_drift_detection.predict(df_baseline)
31 print(f"\nPerformance on Baseline Data - Accuracy: {accuracy_score(baseline_labels, baseline_pred):.2f}, F1: {f1_score(baseline_labels, baseline_pred):.2f}")
32
33 drifted_pred = model_drift_detection.predict(df_drifted)
34 print(f"Performance on Drifted Data - Accuracy: {accuracy_score(drifted_labels, drifted_pred):.2f}, F1: {f1_score(drifted_labels, drifted_pred):.2f}")
35 print("\nNoticeable drop in performance on drifted data.")
```

Listing 89: محاكاة انحراف المفهوم عن طريق إنشاء مجموعتي بيانات مختلفتين وإظهار انخفاض الأداء.

2.6.15 الخطوة 2: تنفيذ اكتشاف الانحراف

بعد ذلك، نقوم بتنفيذ آلية بسيطة لاكتشاف الانحراف. نستخدم اختبار كولموغوروف-سميرنوف (KS test) لعينتين لمقارنة توزيع كل سمة بين مجموعتي البيانات الأساسية والمنحرفة. تشير قيمة p-value الصغيرة (عادةً $0.05 > p$) إلى أن العينتين مأخوذتان من توزيعات مختلفة، مما يدل على

وجود انحراف. نراقب أيضًا أي انخفاض كبير في درجة F1 للنموذج. الكود موضح في القائمة 90.

```

1 # Compare feature distributions between baseline and drifted data
2 print("\n--- Drift Detection (Kolmogorov-Smirnov Test) ---")
3 for i, feature in enumerate(feature_names_drift):
4     statistic, p_value = ks_2samp(df_baseline[feature], df_drifted[feature])
5     print(f"Feature '{feature}': KS Statistic={statistic:.3f}, P-value={p_value:.3f}")
6     if p_value < 0.05: # Common threshold for statistical significance
7         print(f" -> Drift detected in feature '{feature}' (p-value < 0.05).")
8
9 # Visualize feature distribution shift
10 plt.figure(figsize=(12, 5))
11 plt.subplot(1, 2, 1)
12 plt.hist(df_baseline['f_0'], bins=30, alpha=0.7, label='Baseline f_0', color='blue')
13 plt.hist(df_drifted['f_0'], bins=30, alpha=0.7, label='Drifted f_0', color='red')
14 plt.title('Distribution of Feature_0 (Drift)')
15 plt.legend()
16 plt.subplot(1, 2, 2)
17 plt.hist(df_baseline['f_1'], bins=30, alpha=0.7, label='Baseline f_1', color='blue')
18 plt.hist(df_drifted['f_1'], bins=30, alpha=0.7, label='Drifted f_1', color='red')
19 plt.title('Distribution of Feature_1 (Drift)')
20 plt.legend()
21 plt.tight_layout()
22 plt.show()
23
24 # Basic performance degradation monitoring
25 initial_f1 = f1_score(baseline_labels, baseline_pred)
26 current_f1 = f1_score(drifted_labels, drifted_pred)
27 f1_drop = initial_f1 - current_f1
28 if f1_drop > 0.1: # Example threshold for significant drop
29     print(f"\nSignificant F1-score drop detected: {f1_drop:.2f}. Consider retraining!")

```

Listing 90: استخدام اختبار كولموغوروف-سميرنوف ومراقبة الأداء للكشف عن انحراف المفهوم.

3.6.15 الخطوة 3: تطبيق XAI على التصنيفات الخاطئة

الآن، نستخدم SHAP لفهم لماذا يفشل النموذج في التعامل مع البيانات المنحرفة. من خلال شرح تنبؤ تم تصنيفه بشكل خاطئ، يمكننا أن نرى السمات التي تترك النموذج بسبب الانحراف. يوفر هذا رؤية حاسمة لتصحيح الأخطاء وتحسين النموذج. توضح القائمة 91 كيفية إنشاء تفسير SHAP لتنبؤ معين تم إجراؤه على البيانات المنحرفة.

4.6.15 الخطوة 4: تكامل SIEM/SOAR المفاهيمي

أخيرًا، ننظر في كيفية دمج هذه النتائج في سير عمل مركز العمليات الأمنية.

- تنبيه الانحراف إلى SIEM: عندما يكتشف اختبار KS انحرافًا (على سبيل المثال، $p_value > 0.05$)، يمكن إرسال تنبيه آلي إلى نظام SIEM. هذا التنبيه، المنسق

```

1 import shap # Ensure shap is imported
2 # This code assumes model_drift_detection, df_drifted, drifted_labels
3 # and drifted_pred are available from the previous steps.
4 test_preds = drifted_pred # Use predictions on drifted data
5 misclassified_idx = np.where((test_preds == 1) & (drifted_labels == 0))
6 if len(misclassified_idx) == 0:
7     misclassified_idx = np.where((test_preds == 0) & (drifted_labels == 1))
8     if len(misclassified_idx) == 0:
9         print("\nNo misclassified examples found for XAI.")
10        example_to_explain_idx = 0
11        is_misclassified = "Not explicitly misclassified"
12    else:
13        example_to_explain_idx = misclassified_idx
14        is_misclassified = "Misclassified (Predicted Benign, True Malicious)"
15 else:
16     example_to_explain_idx = misclassified_idx
17     is_misclassified = "Misclassified (Predicted Malicious, True Benign)"
18
19 print(f"\n--- Explaining a Prediction on Drifted Data ({is_misclassified}) ---")
20 explainer_xai = shap.TreeExplainer(model_drift_detection)
21 shap_values_example = explainer_xai.shap_values(df_drifted.iloc[[example_to_explain_idx]])
22
23 shap.initjs()
24 shap.force_plot(explainer_xai.expected_value, shap_values_example, df_drifted.iloc[example_to_explain_idx],
25                 matplotlib=True, show=False)
26 plt.title(f"SHAP Force Plot for Drifted Data Example (Index {example_to_explain_idx})")
27 plt.tight_layout()
28 plt.show()
29
30 print(f"\nOriginal Feature Values for Example {example_to_explain_idx}: \n{df_drifted.iloc[example_to_explain_idx]}")
31 print(f"Model Predicted: {model_drift_detection.predict(df_drifted.iloc[[example_to_explain_idx]])}, True: {drifted_labels[example_to_explain_idx]}")
32 print("\nObservation: XAI helps show why the model made its prediction on drifted data.")

```

Listing 91: استخدام SHAP لشرح تنبؤ النموذج على البيانات المنحرفة، مع تسليط الضوء على سبب احتمال تصنيفه الخاطئ.

بـ JSON، يمكن أن يبدو كالتالي:

```

{
  "alert_type": "Detected",
  "model_id": "NIDS_Model_v2.1",
  "feature_affected": "f_0",
  "p_value": 0.001,
  "recommendation": "Investigate retraining."
}

```

• إثراء XAI لـ SOAR: عندما يميز النموذج حدثاً بأنه خبيث، يمكن إضافة تفسير XAI إلى التنبيه المرسل إلى أدوات SIEM و SOAR. هذا يثري بيانات الحادث للمحللين وكتيبات التشغيل الآلية.

```

{
  "event_id": "EVT45678",
  "model_prediction": "Malicious"
}

```

```

,88.0 "confidence":
  { "explanation":
    , "high_positive_contribution" "f_0":
    , "moderate_positive_contribution" "f_2":
    "low_negative_contribution" "f_3":
  },
  { "f_0": "raw_features":
    { "f_1": -5.0, ...
  }
}

```

يحول هذا التكامل مخرجات المراقبة إلى استخبارات قابلة للتنفيذ، مما يتيح عمليات أمنية أسرع وأكثر استنارة.

خاتمة

إن تقييم نموذج الذكاء الاصطناعي ليس مهمة لمرة واحدة، بل هو عملية مستمرة وحيوية لنجاحه في مركز عمليات أمنية حقيقي. أوضح هذا الفصل أن تجاوز مقاييس النشر الأولية أمر ضروري للحفاظ على دفاع فعال وموثوق به مدعوم بالذكاء الاصطناعي. لقد استكشفنا تقنيات التحقق المتقدمة مثل اختبار أ/ب (A/B testing) والاختبار الرجعي (backtesting)، والتي توفر تقييمات واقعية للأداء. وفصلنا الأساليب المنهجية لاكتشاف انحراف المفهوم (concept drift)، مما يضمن تكيف النماذج مع مشهد التهديدات المتغير باستمرار. علاوة على ذلك، أكدنا على الدور الحاسم للتحقق بوجود الإنسان في الحلقة (Human-in-the-Loop) والذكاء الاصطناعي القابل للتفسير (XAI) في بناء الثقة، وتحسين الدقة، وتمكين التحقيقات الأعمق. من خلال دمج استراتيجيات التحقق المستمر هذه في سلاسل الأدوات الأمنية، يمكن للمؤسسات ضمان بقاء أنظمة الذكاء الاصطناعي الخاصة بها قوية وموثوقة، وأن تكون شركاء أذكياء حقاً في الدفاع السيبراني. None

المختصرات والمصطلحات

اختبار A/B Testing طريقة لمقارنة نسختين أو أكثر من نموذج ذكاء اصطناعي AI model بشكل متوازٍ على البيانات الحية لتحديد أيها يقدم أداءً أفضل. يتم عرض كل نسخة لشريحة مختلفة من المستخدمين أو حركة البيانات.

التعلم النشط Active Learning استراتيجية في التعلم الآلي machine learning حيث يطلب النموذج بذكاء من الإنسان تصنيف نقاط البيانات الجديدة الأكثر إفادة، مما يجعل عملية التدريب أكثر كفاءة.

التعلم التكيفي Adaptive Learning نهج يقوم فيه نموذج الذكاء الاصطناعي AI model بتحديث نفسه باستمرار بالبيانات الجديدة فور وصولها، بدلاً من انتظار إعادة تدريب كاملة. يُعرف هذا أيضاً بالتعلم عبر الإنترنت أو التعلم التزايدى online or incremental learning.

أدوين ADWIN (النوافذ التكيفية ADaptive WINdowing): خوارزمية تستخدم للكشف عن انحراف المفهوم concept drift من خلال مقارنة الخصائص الإحصائية للبيانات الحديثة (ضمن "نافذة") مع البيانات الأقدم.

الاختبار الرجعي Backtesting عملية تقييم أداء نموذج ما على البيانات التاريخية، بما في ذلك الحوادث الأمنية السابقة، لمحاكاة كيفية أدائه في الماضي.

الاختبار الكاري Canary Testing استراتيجية لإصدار نموذج جديد بأمان عن طريق توجيه مجموعة فرعية صغيرة من حركة البيانات الحية إليه أولاً لمراقبة أدائه قبل الإطلاق الكامل. انحراف المفهوم Concept Drift ظاهرة تتغير فيها الخصائص الإحصائية للبيانات بمرور الوقت، مما يؤدي إلى تدهور أداء نموذج الذكاء الاصطناعي AI model المدرب لأن البيانات الجديدة لم تعد تتطابق مع ما تم تدريبه عليه.

التفسيرات المضادة للواقع Counterfactual Explanations نوع من التفسير يصف أصغر تغيير مطلوب في المدخلات لعكس قرار النموذج (على سبيل المثال، من "ضار" إلى "غير ضار").

طريقة كشف الانحراف DDM (طريقة كشف الانحراف Drift Detection Method): خوارزمية تراقب معدل خطأ النموذج وتشير إلى حدوث انحراف المفهوم concept drift إذا زاد معدل الخطأ بشكل كبير.

التدخل البشري في الحلقة HITL (التدخل البشري في الحلقة Human-in-the-Loop):
عملية تجمع بين ذكاء الآلة والذكاء البشري، حيث يقوم محللو الأمن بمراجعة والتحقق
من صحة وتصحيح مخرجات نموذج الذكاء الاصطناعي AI model.

مؤشرات الاختراق IOCs (مؤشرات الاختراق Indicators of Compromise): أجزاء
من الأدلة الرقمية، مثل عناوين IP أو تجزئات الملفات أو أسماء النطاقات، التي تشير إلى
أن شبكة حاسوبية قد تم اختراقها.

اختبار كولموغوروف-سميرنوف KS Test (اختبار كولموغوروف-سميرنوف Kolmogorov-Smirnov Test): اختبار إحصائي يستخدم لتحديد ما إذا كانت عينتان من البيانات
تأتيان من نفس التوزيع، وغالباً ما يستخدم للكشف عن الانحراف المفهوم concept
drift من خلال مقارنة البيانات الجديدة ببيانات التدريب الأصلية.

لايم LIME (تفسيرات محلية قابلة للتفسير ومستقلة عن النموذج Local Interpretable
Model-agnostic Explanations): تقنية من تقنيات الذكاء الاصطناعي القابل
للتفسير XAI تشرح تنبؤاً واحداً لنموذج معقد عن طريق إنشاء نموذج أبسط ومفهوم حول
تلك الحالة المحددة.

نظام كشف التسلل الشبكي NIDS (نظام كشف التسلل الشبكي Network Intrusion
Detection System): نظام يراقب حركة مرور الشبكة بحثاً عن الأنشطة المشبوهة
ويصدر تنبيهات عند اكتشاف مثل هذا النشاط.

شاب SHAP (التفسيرات المضافة لشابي SHapley Additive exPlanations): تقنية
من تقنيات الذكاء الاصطناعي القابل للتفسير XAI تستند إلى نظرية الألعاب وتشرح تنبؤ
النموذج من خلال تعيين قيمة أهمية لكل ميزة، مما يوضح مدى مساهمتها في النتيجة النهائية.

سييم SIEM (إدارة معلومات وأحداث الأمان Security Information and Event
Management): نظام يجمع ويصنف ويحلل بيانات السجلات من مصادر مختلفة عبر
البنية التحتية لتكنولوجيا المعلومات للكشف عن التهديدات الأمنية.

سوار SOAR (التنسيق والأتمتة والاستجابة الأمنية - Automation, Orchestration, and Response): منصة تساعد على أتمتة تدفقات العمل الأمنية (كتيبات
التشغيل playbooks) من خلال دمج مختلف الأدوات والعمليات الأمنية.

مركز عمليات الأمن SOC (مركز عمليات الأمن Security Operations Center): مركز
قيادة مركزي حيث يقوم فريق من محترفي الأمن السيبراني cybersecurity بمراقبة
الحوادث الأمنية وكشفها وتحليلها والاستجابة لها.

المراقبة الإحصائية للعمليات SPC (المراقبة الإحصائية للعمليات Statistical Process Control): طريقة، مستوحاة أصلاً من مراقبة الجودة، تستخدم مخططات التحكم لمراقبة مقاييس أداء النموذج بمرور الوقت والإشارة إلى انحراف المفهوم concept drift إذا خرجت عن الحدود المتوقعة.

الذكاء الاصطناعي القابل للتفسير XAI (الذكاء الاصطناعي القابل للتفسير Explainable AI): مجال في الذكاء الاصطناعي artificial intelligence يركز على إنشاء طرق وتقنيات تسمح للمستخدمين البشر بفهم النتائج والمخرجات التي تنشأها نماذج التعلم الآلي machine learning models والثقة بها.

باب 16

الخصوصية والأخلاقيات والحوكمة في الذكاء الاصطناعي للأمن السيبراني

مقدمة

بينما ندمج الذكاء الاصطناعي في صميم أنظمتنا الدفاعية، يجب علينا أيضاً معالجة التحديات العميقة المتعلقة بالأخلاقيات والقانون والخصوصية التي تنشأ. إن أي نظام ذكاء اصطناعي، حتى لو كان متقناً من الناحية التقنية، ولكنه معيب أخلاقياً، يمكن أن يخلق مخاطر أكثر مما يحل. يمكن أن يؤدي إلى تآكل الثقة، وإدخال تحيز منهجي، وانتهاك الحقوق الأساسية. يستكشف هذا الفصل الأبعاد غير التقنية الحاسمة لبناء أنظمة دفاع ذكية. سندرس كيفية إدارة المخاطر الأخلاقية، والتعامل مع الأطر القانونية المعقدة، وإنشاء حوكمة قوية لضمان أن حلول الأمن السيبراني المدعومة بالذكاء الاصطناعي ليست فعالة فحسب، بل مسؤولة وجديرة بالثقة أيضاً. تمثل أهداف التعلم الرئيسية في فهم المخاطر الأخلاقية للذكاء الاصطناعي في الأمن السيبراني، والامتثال للأطر القانونية والسياساتية الرئيسية، وتنفيذ هياكل الحوكمة الداخلية، واختبار أنظمة الذكاء الاصطناعي بشكل استباقي بحثاً عن الثغرات الأخلاقية.

1.16 إدارة المخاطر الأخلاقية

يثير نشر الذكاء الاصطناعي في مجال الأمن السيبراني اعتبارات أخلاقية فريدة تتطلب إدارة دقيقة. تعمل هذه الأنظمة على مجموعات بيانات ضخمة وتتخذ قرارات آلية يمكن أن تكون لها

عواقب وخيمة. يتطلب التنفيذ المسؤول نهجاً استباقياً لتحديد المخاطر الأخلاقية المحتملة وتحليلها والتخفيف من حدتها.

1.1.16 الآثار المترتبة على الاستخدام المزدوج

أحد التحديات الأخلاقية المركزية في استخدام الذكاء الاصطناعي للأمن هو مشكلة الاستخدام المزدوج. يشير هذا إلى كيفية إمكانية إعادة توظيف التقنيات المطورة للدفاع لأغراض هجومية. خير مثال على ذلك هو نموذج ذكاء اصطناعي مصمم لاكتشاف ثغرات البرامج تلقائياً لمساعدة المطورين على إصلاحها. في يد جهة فاعلة خبيثة، تصبح هذه الأداة نفسها سلاحاً قوياً للعثور على ثغرات يوم الصفر zero-day exploits لاستخدامها في الهجمات. وبالمثل، يمكن إجراء هندسة عكسية لخوارزميات كشف التصيد الاحتيالي phishing detection algorithms المدفوعة بالذكاء الاصطناعي لإنشاء حملات تصيد احتيالي phishing campaigns أكثر تطوراً ومراوغة [?]. لإدارة هذه المخاطر، طور مجتمع الأمن السيبراني ممارسات أخلاقية للتعامل مع المعلومات الحساسة.

الإفصاح المسؤول Responsible Disclosure هو ممارسة الإبلاغ عن ثغرة برمجية مكتشفة حديثاً بشكل مباشر وخاص إلى المورد أو مالك النظام المتأثر. يمنح هذا المورد وقتاً لإنشاء وإصدار تصحيح قبل أن تصبح الثغرة معروفة للعامة. يوازن هذا النهج بين هدف تأمين الأنظمة والحاجة إلى منع استغلال الثغرة على نطاق واسع من قبل المهاجمين.

مشاركة الثغرات Vulnerability Sharing تتضمن التبادل المنظم لمعلومات التهديدات وبيانات الثغرات داخل مجموعات موثوقة. توفر منظمات مثل مراكز تبادل المعلومات والتحليل Information Sharing and Analysis Centers (ISACs) منصة للشركات في قطاع معين (مثل التمويل والرعاية الصحية) لتبادل المعلومات حول التهديدات التي تواجهها. يعزز هذا النهج الجماعي دفاعات جميع الأعضاء.

2.1.16 مخاوف الخصوصية

غالباً ما تحتاج أنظمة الأمن المدعومة بالذكاء الاصطناعي إلى معالجة كميات هائلة من البيانات لتكون فعالة، بما في ذلك حركة مرور الشبكة network traffic، وسجلات نشاط المستخدم user activity logs، وأحداث النظام system events. يمكن أن تحتوي هذه البيانات على معلومات شخصية حساسة، مما يخلق مخاطر كبيرة على الخصوصية. أحد المخاوف الرئيسية هو القدرة على إعادة تحديد هوية الأفراد re-identify individuals من البيانات التي كان يُعتقد أنها مجهولة الهوية. قد لا تكشف البيانات الأمنية مثل عناوين

IP أو معرفات الأجهزة device identifiers أو الطوابيع الزمنية لتسجيل الدخول login timestamps عن هوية الشخص بشكل مباشر. ومع ذلك، من خلال دمج مجموعات البيانات هذه مع معلومات أخرى متاحة للجمهور، يمكن للمهاجم ربط النشاط بفرد معين، مما يؤدي إلى خرق خطير للخصوصية.

لمواجهة هذه المخاطر، يجب على المنظمات تبني مبادئ الخصوصية الأساسية:

- **تقليل البيانات Data Minimization:** ينص هذا المبدأ على أنه يجب عليك فقط جمع ومعالجة الحد الأدنى المطلق من البيانات الشخصية اللازمة لتحقيق هدف أمني محدد. إذا كان يمكن إنجاز مهمة ما دون بيانات شخصية، فيجب القيام بذلك.
- **إخفاء الهوية Anonymization:** هي عملية تغيير البيانات بحيث لا يمكن التعرف على الأفراد منها، بشكل مباشر أو غير مباشر. يجب أن يكون هذا التحويل غير قابل للعكس. على سبيل المثال، إزالة جميع المعرفات الشخصية تماماً من مجموعة البيانات.
- **الاسم المستعار Pseudonymization:** تستبدل هذه العملية المعلومات التعريفية بمعرفات اصطناعية، أو أسماء مستعارة. على سبيل المثال، استبدال اسم المستخدم برقم تعريف عشوائي مثل `User-1138` لا يزال من الممكن تحليل البيانات بحثاً عن أنماط، ولكن هوية الفرد تظل محمية. على عكس إخفاء الهوية، هذه العملية قابلة للعكس إذا كان لدى الشخص حق الوصول إلى المفتاح الذي يربط الاسم المستعار بالمعرف الأصلي.
- **الامتثال لمبادئ حماية البيانات:** بالإضافة إلى ما سبق، يجب أن تلتزم العمليات الأمنية بالمبادئ الراضية مثل تحديد الغرض (لا ينبغي استخدام البيانات التي تم جمعها للأمن لأغراض التسويق)، وتحديد التخزين (يجب حذف البيانات عند عدم الحاجة إليها)، والمساءلة (يجب أن تكون المنظمة قادرة على إثبات الامتثال).

3.1.16 التحيز الخوارزمي

يحدث التحيز الخوارزمي Algorithmic bias عندما ينتج نموذج ذكاء اصطناعي نتائج متحيزة بشكل منهجي وغير عادلة لأفراد أو مجموعات معينة. غالباً ما ينشأ هذا التحيز من البيانات المستخدمة لتدريب النموذج. إذا كانت بيانات التدريب تعكس التحيزات البشرية التاريخية أو لم تكن ممثلة للعالم الحقيقي، فسوف يتعلم نموذج الذكاء الاصطناعي تلك التحيزات ويعززها.

في سياق الأمن السيبراني، يمكن أن يكون لذلك عواقب وخيمة. على سبيل المثال:

- قد يقوم نموذج كشف التهديدات الداخلية insider threat detection model الذي تم تدريبه على بيانات تاريخية متحيزة بتمييز الموظفين من فئة ديموغرافية أو موقع

جغرافي معين بشكل غير متناسب على أنهم ذوو مخاطر عالية. يؤدي هذا إلى تدقيق غير عادل وبيئة عمل معادية.

• قد يتم تدريب نظام كشف اختراق الشبكة-network intrusion detection sys-tem بشكل أساسي على بيانات من النشاط النموذجي لقسم واحد. نتيجة لذلك، يمكن أن يولد عدداً كبيراً من الإنذارات الكاذبة false positives لقسم آخر بأنماط عمل مختلفة، مما يؤدي إلى إرهاق التنبيهات alert fatigue المحلي للأمن.

• قد يفشل النموذج في اكتشاف التهديدات الحقيقية من المجموعات أو الأنظمة غير الممثلة بشكل كافٍ لأنه لم يتعلم أبداً كيف يبدو نشاطها الشاذ.

بعد التخفيف من التحيز الخوارزمي جزءاً مهماً من تطوير الذكاء الاصطناعي الأخلاقي. تشمل الاستراتيجيات الرئيسية استخدام بيانات تدريب متنوعة ومثلة، واستخدام مقاييس الإنصاف fairness metrics لمراجعة النماذج بحثاً عن النتائج المتحيزة، واختيار الخوارزميات المصممة لتعزيز الإنصاف، والتأكد من أن الفرق التي تبني هذه الأنظمة متنوعة هي نفسها [?].

2.16 الأطر السياسية والقانونية

لا يتم نشر الذكاء الاصطناعي في مجال الأمن السيبراني في فراغ. فهو محكوم بعدد متزايد من القوانين واللوائح التي تركز على خصوصية البيانات وأمنها والاستخدام المسؤول للذكاء الاصطناعي. إن الالتزام بهذه الأطر ليس اختياريًا؛ بل هو مطلب قانوني.

1.2.16 التنقل بين متطلبات الامتثال

يمثل التنقل في المشهد المعقد للوائح العالمية تحدياً كبيراً للمنظمات التي تستخدم الذكاء الاصطناعي. غالباً ما يكون لهذه القوانين نطاق خارج الحدود الإقليمية extraterritorial reach، مما يعني أن شركة في بلد ما قد تخضع لقوانين بلد آخر إذا كانت تعالج بيانات تخص مواطني ذلك البلد. سنستعرض بعض الأطر القانونية الأكثر تأثيراً.

2.2.16 اللائحة العامة لحماية البيانات (GDPR)

اللائحة العامة لحماية البيانات (General Data Protection Regulation - GDPR) هي قانون تاريخي لحماية البيانات من الاتحاد الأوروبي [25]. تنطبق على أي منظمة في جميع أنحاء

العالم تعالج البيانات الشخصية للأفراد المقيمين في الاتحاد الأوروبي. بالنسبة للذكاء الاصطناعي في الأمن السيبراني، فإن تأثير اللائحة العامة لحماية البيانات كبير. فهي تفرض مبادئ مثل "الخصوصية حسب التصميم" privacy by design، التي تتطلب بناء حماية الخصوصية في الأنظمة منذ البداية. كما تمنح الأفراد "الحق في التفسير" right to explanation للقرارات التي تتخذها الأنظمة الآلية، مما له آثار مباشرة على قابلية التفسير explainability لنماذج الذكاء الاصطناعي التي قد تمنع وصول المستخدم أو تصنف سلوكه على أنه ضار.

3.2.16 قانون خصوصية المستهلك في كاليفورنيا (CCPA)

قانون خصوصية المستهلك في كاليفورنيا (California Consumer Privacy Act - CCPA)، وخليفته قانون حقوق الخصوصية في كاليفورنيا (California Privacy Rights Act - CPRA)، هو قانون على مستوى الولاية في الولايات المتحدة يمنح المستهلكين سيطرة كبيرة على معلوماتهم الشخصية [72]. يمنح سكان كاليفورنيا الحق في معرفة البيانات الشخصية التي يتم جمعها عنهم، والحق في حذف تلك البيانات، والحق في إلغاء الاشتراك في بيعها. يجب تصميم أنظمة الأمن التي تجمع وتحلل بيانات المستخدم لتلبية حقوق المستهلكين هذه.

4.2.16 توجيه أمن الشبكات والمعلومات 2 (NIS2)

توجيه أمن الشبكات والمعلومات 2 (Network and Information Security Directive - NIS2) هو قانون على مستوى الاتحاد الأوروبي يهدف إلى تعزيز وضع الأمن السيبراني للبنية التحتية الحيوية critical infrastructure والخدمات الأساسية essential services [26]. يفرض على المنظمات في قطاعات مثل الطاقة والنقل والرعاية الصحية والبنية التحتية الرقمية تنفيذ تدابير أمنية أقوى والإبلاغ عن الحوادث الهامة. تلعب أنظمة الذكاء الاصطناعي دوراً رئيسياً في تلبية هذه المتطلبات، على سبيل المثال، من خلال توفير قدرات متقدمة للكشف عن التهديدات advanced threat detection والاستجابة الآلية للحوادث automated incident response. يخلق توجيه NIS2 فعلياً دافعاً قانونياً لتبني تقنيات أمنية أكثر تطوراً، بما في ذلك الذكاء الاصطناعي.

5.2.16 قانون الذكاء الاصطناعي (الاتحاد الأوروبي)

قانون الذكاء الاصطناعي المقترح من الاتحاد الأوروبي (EU AI Act) هو أول محاولة رئيسية لإنشاء إطار قانوني شامل للذكاء الاصطناعي [24]. يتبنى نهجاً قائماً على المخاطر risk-based approach، حيث يصنف أنظمة الذكاء الاصطناعي إلى مستويات مختلفة من المخاطر (غير

مقبولة، عالية، محدودة، ضئيلة). من المرجح أن يتم تصنيف العديد من تطبيقات الذكاء الاصطناعي في مجال الأمن السيبراني، خاصة تلك المستخدمة في البنى التحتية الحيوية أو إنفاذ القانون، على أنها "عالية المخاطر" high-risk. ستواجه هذه الأنظمة متطلبات صارمة تتعلق بحوكمة البيانات، والإشراف البشري، والمتانة، والدقة، والأمن قبل أن يتم نشرها في سوق الاتحاد الأوروبي.

6.2.16 الآثار القانونية لاتخاذ القرارات الآلية

هناك خيط مشترك في هذه اللوائح وهو التركيز على اتخاذ القرارات الآلية-automated decision-making. عندما يتخذ نظام ذكاء اصطناعي قراراً له تأثير قانوني أو تأثير مماثل على شخص ما - مثل نموذج ذكاء اصطناعي يقفل تلقائياً حساب موظف بناءً على نشاط مشتببه به كتهديد داخلي - يتم تفعيل الالتزامات القانونية. غالباً ما تشمل هذه الحاجة إلى الشفافية transparency (إبلاغ الشخص بأنه تم اتخاذ قرار آلي)، وقابلية التفسير explainability (توفير المنطق وراء القرار)، والحق في التدخل البشري right to human intervention (السماح للشخص بالطعن في القرار أمام إنسان).

3.16 أطر الحوكمة

إلى جانب الامتثال القانوني، تنشئ المنظمات المسؤولة أطر حوكمة داخلية لتوجيه تطوير ونشر الذكاء الاصطناعي. تتكون هذه الأطر من مبادئ وعمليات ووثائق تترجم الأهداف الأخلاقية عالية المستوى إلى ممارسات تشغيلية ملموسة.

1.3.16 مبادئ الذكاء الاصطناعي المسؤول في الأمن السيبراني

يُبنى إطار الحوكمة القوي على مجموعة من المبادئ الأساسية التي توجه كل مرحلة من مراحل دورة حياة الذكاء الاصطناعي. يجب أن تشمل هذه المبادئ:

- الإنصاف Fairness: يجب ألا ينتج النظام نتائج متحيزة أو تمييزية.
- الشفافية Transparency: يجب أن تكون عمليات وبيانات وقرارات نظام الذكاء الاصطناعي مفهومة وقابلة للتدقيق.
- المساءلة Accountability: يجب أن تكون هناك خطوط واضحة للمسؤولية عن نتائج نظام الذكاء الاصطناعي، خاصة عندما يفشل.

- المتانة Robustness: يجب أن يكون النظام آمناً ومرناً، وقادراً على تحمل الهجمات العدائية والأداء بشكل موثوق في ظل ظروف غير متوقعة.
- قابلية التفسير Explainability: يجب أن يكون النظام قادراً على شرح قراراته بطريقة يمكن للبشر فهمها، وهو مفهوم ناقشناه في هذا الكتاب.
- الخصوصية Privacy: يجب تصميم النظام لحماية البيانات الشخصية منذ البداية، باستخدام تقنيات مثل تقليل البيانات والأسماء المستعارة.
- السلامة Safety: يجب ألا يسبب النظام ضرراً غير مقصود، خاصة عند نشره في بيئات حرجية.

2.3.16 تطوير "بطاقات النماذج"

- بطاقة النموذج Model Card هي وثيقة قصيرة وموحدة توفر معلومات أساسية حول نموذج تعلم الآلة [55]. تشبه إلى حد كبير ملصق القيمة الغذائية على الطعام، تقدم بطاقة النموذج شفافية حول مكونات النموذج وأدائه. تتضمن عادةً:
- الاستخدامات المقصودة للنموذج والقيود المفروضة عليه.
 - مقاييس الأداء Performance metrics، مفصلة حسب مجموعات المستخدمين المختلفة أو الظروف للكشف عن التحيزات المحتملة.
 - تفاصيل حول بيانات التدريب training data المستخدمة.
 - الاعتبارات الأخلاقية والمخاطر المحتملة.
- تُعد بطاقات النماذج أداة عملية لتعزيز الشفافية والمساءلة في تطوير الذكاء الاصطناعي.

3.3.16 "صحائف البيانات لمجموعات البيانات"

تماماً كما تحتاج النماذج إلى توثيق، كذلك تحتاج مجموعات البيانات المستخدمة لتدريبها. صحيفة البيانات لمجموعة البيانات Datasheet for a Dataset هي وثيقة تفصل الدافع والتكوين وعملية الجمع والاستخدامات الموصى بها لمجموعة البيانات [29]. تشجع هذه الممارسة منشئي مجموعات البيانات على أن يكونوا شفافين بشأن كيفية جمع البيانات وتنظيفها، وتبسيط الضوء على أي تحيزات أو قيود محتملة. بالنسبة للذكاء الاصطناعي في الأمن السيبراني، حيث تكون جودة البيانات data quality ذات أهمية قصوى، تعد صحائف البيانات أداة حوكمة لا تقدر بثمن لضمان سلامة البيانات وملاءمتها.

4.3.16 إجراء "تقييمات تأثير الذكاء الاصطناعي"

تقييم تأثير الذكاء الاصطناعي AI Impact Assessment هو عملية منظمة لتحديد وتحليل وتخفيف الآثار المجتمعية والأخلاقية والقانونية المحتملة لنظام الذكاء الاصطناعي قبل نشره. إنه تمرين استباقي لإدارة المخاطر يجبر الفرق على التفكير في العواقب المحتملة لعملهم. بالنسبة لنظام ذكاء اصطناعي آمن، قد يتضمن ذلك تقييم مخاطر الاتهامات الباطلة من نموذج التهديد الداخلي insider threat model أو الآثار المترتبة على الخصوصية لطريقة جديدة لجمع البيانات.

4.16 الاختبار الأحمر المسؤول للذكاء الاصطناعي

يتضمن الاختبار الأحمر red-teaming التقليدي في الأمن السيبراني محاكاة الهجمات لاختبار دفاعات المنظمة. يكيّف الاختبار الأحمر المسؤول للذكاء الاصطناعي Responsible AI Red-Teaming هذا المفهوم لاختبار أنظمة الذكاء الاصطناعي بشكل استباقي بحثاً عن الثغرات الأخلاقية ethical vulnerabilities، والتحييزات biases، وإمكانية سوء الاستخدام. يتجاوز الاختبار التقني البحث لاستكشاف كيف يمكن أن يفشل النظام بطرق تسبب ضرراً مجتمعياً أو أخلاقياً.

1.4.16 التحديد الاستباقي

الهدف من الاختبار الأحمر للذكاء الاصطناعي هو تحديد نقاط الضعف بشكل استباقي قبل أن يمكن استغلالها أو التسبب في ضرر. هذه عملية عدائية adversarial process حيث يحاول فريق مخصص كسر نظام الذكاء الاصطناعي من منظور أخلاقي. على سبيل المثال، قد يحاول فريق أحمر صياغة بيانات إدخال تخدع النموذج لاتخاذ قرار متحيز أو إيجاد طرق لإزالة إخفاء هوية البيانات التي يستخدمها النظام.

2.4.16 تخطيط السيناريوهات

أحد الأنشطة الرئيسية في الاختبار الأحمر للذكاء الاصطناعي هو تخطيط السيناريوهات scenario planning. يقوم الفريق بتطوير وتحليل مواقف افتراضية يمكن فيها إساءة استخدام نظام الذكاء الاصطناعي أو أن يؤدي إلى نتائج سلبية. قد يشمل ذلك:

- سيناريو يستغل فيه مهاجم الطبيعة المزدوجة الاستخدام لما سخ الثغرات vulnerability scanner لشن هجوم واسع النطاق.

- سيناريو يتلاعب فيه جهة فاعلة تابعة لدولة ببيانات التدريب لمنصة استخبارات التهديدات threat intelligence platform لإخفاء أنشطتها.
- سيناريو يقوم فيه نظام كشف الشذوذ anomaly detection system الذي تم تكوينه بشكل خاطئ بإغلاق معدات المستشفى الحيوية أثناء حالة طوارئ.
- تساعد هذه العملية المنظمات على توقع المخاطر غير التقنية البحتة والاستعداد لها.

3.4.16 تحليل الأضرار

أخيراً، يتضمن الاختبار الأحمر للذكاء الاصطناعي تحليلاً منهجياً للأضرار harm analysis. لكل سيناريو فشل أو سوء استخدام محتمل تم تحديده، يقوم الفريق بتقييم العواقب السلبية المحتملة. يشمل ذلك النظر في مجموعة واسعة من الأضرار، مثل خروقات الخصوصية privacy breaches، والنتائج التمييزية ضد مجموعات معينة من المستخدمين، والانتهاكات الباطلة التي تؤدي إلى فقدان الوظائف، والفشل في البنى التحتية الحيوية critical infrastructure. من خلال فهم احتمالية الضرر، يمكن للمنظمات تحديد أولويات جهود التخفيف وبناء أنظمة ذكاء اصطناعي أكثر أماناً ومسؤولية.

الخاتمة

يتطلب بناء أنظمة أمن سيرياني فعالة مدعومة بالذكاء الاصطناعي أكثر من مجرد خوارزميات algorithms متطورة ومجموعات بيانات ضخمة. إنه يتطلب التزاماً عميقاً ومستمرًا بالمبادئ الأخلاقية وحماية الخصوصية والحوكمة القوية. كما رأينا، فإن الطبيعة المزدوجة الاستخدام لأدوات الذكاء الاصطناعي، وإمكانية التحيز الخوارزمي algorithmic bias، والمخاطر التي تهدد خصوصية الأفراد ليست حالات هامشية؛ بل هي تحديات مركزية يجب معالجتها منذ بداية دورة حياة التطوير.

لم يعد التنقل في الشبكة المعقدة للأطر القانونية مثل اللائحة العامة لحماية البيانات GDPR وقانون الذكاء الاصطناعي AI Act القادم اختياريًا، بل أصبح مطلباً أساسياً للعمل على مستوى العالم. ومع ذلك، فإن المسؤولية الحقيقية تتجاوز مجرد الامتثال. إنها تنطوي على خلق ثقافة من الوعي الأخلاقي مدعومة بأدوات حوكمة عملية مثل بطاقات النماذج model cards وصحائف البيانات لمجموعات البيانات datasheets for datasets والاختبار الأحمر الاستباقي للذكاء الاصطناعي AI red-teaming. من خلال دمج هذه الممارسات في سير عملنا، يمكننا بناء أنظمة دفاع ذكية ليست قوية فحسب، بل عادلة وشفافة وجديرة بالثقة التي نضعها فيها. يعتمد مستقبل الأمن السيرياني على قدرتنا على الابتكار بمسؤولية.

المختصرات والمصطلحات

قانون الذكاء الاصطناعي AI Act إطار قانوني مقترح من الاتحاد الأوروبي للذكاء الاصطناعي artificial intelligence يصنف أنظمة الذكاء الاصطناعي AI systems حسب مستوى المخاطر ويفرض متطلبات صارمة على التطبيقات عالية المخاطر، والتي يقع الكثير منها في مجال الأمن السيبراني cybersecurity.

تقييم تأثير الذكاء الاصطناعي AI Impact Assessment عملية منظمة لتحديد وتحليل وتخفيف الآثار المجتمعية والأخلاقية والقانونية المحتملة لنظام الذكاء الاصطناعي AI system قبل نشره.

التحيز الخوارزمي Algorithmic Bias ظاهرة ينتج فيها نموذج الذكاء الاصطناعي AI model نتائج متحيزة بشكل منهجي، غالباً لأنه تم تدريبه على بيانات متحيزة أو غير ممثلة. إخفاء الهوية Anonymization العملية غير القابلة للعكس لتغيير البيانات لضمان عدم إمكانية التعرف على الأفراد منها، سواء بشكل مباشر أو غير مباشر.

اتخاذ القرار الآلي Automated Decision-Making قرار يتخذه نظام الذكاء الاصطناعي AI system دون تدخل بشري ويكون له تأثير قانوني أو تأثير كبير مماثل على شخص ما، مما يؤدي إلى التزامات قانونية محددة.

قانون خصوصية المستهلك في كاليفورنيا CCPA قانون خصوصية المستهلك في كاليفورنيا Cal-ifornia Consumer Privacy Act؛ قانون ولاية أمريكي يمنح المستهلكين حقوقاً على معلوماتهم الشخصية، بما في ذلك الحق في معرفة البيانات التي يتم جمعها وطلب حذفها.

قانون حقوق الخصوصية في كاليفورنيا CPRA قانون حقوق الخصوصية في كاليفورنيا Cali-fornia Privacy Rights Act؛ وهو القانون الذي خلف قانون خصوصية المستهلك في كاليفورنيا CCPA، حيث يوسع حقوق المستهلكين ولوائح الخصوصية في كاليفورنيا.

تقليل البيانات Data Minimization مبدأ الخصوصية الذي ينص على أنه يجب على المؤسسات جمع ومعالجة الحد الأدنى المطلق من البيانات الشخصية اللازمة لتحقيق غرض محدد ومصرح به.

صحيفة بيانات مجموعة بيانات Datasheet for a Dataset وثيقة توفر معلومات مفصلة حول دوافع مجموعة البيانات dataset وتكوينها وعملية جمعها واستخدامها المقصود لتعزيز الشفافية والتعامل السليم.

مشكلة الاستخدام المزدوج Dual-Use Problem التحدي الأخلاقي حيث يمكن إعادة توظيف تقنية تم تطويرها لغرض دفاعي (على سبيل المثال، ذكاء اصطناعي AI للعثور على الثغرات الأمنية vulnerabilities من أجل تصحيحها) لأغراض هجمات خبيثة وهجومية.

اللائحة العامة لحماية البيانات GDPR اللائحة العامة لحماية البيانات - General Data Protection Regulation؛ قانون تاريخي للاتحاد الأوروبي ينظم معالجة البيانات الشخصية للأفراد داخل الاتحاد الأوروبي، مما يؤثر على أي منظمة تتعامل مع هذه البيانات.

تحليل الضرر Harm Analysis تقييم منهجي للعواقب السلبية المحتملة، مثل خروقات الخصوصية أو التمييز، التي يمكن أن تنجم عن فشل أو سوء استخدام نظام الذكاء الاصطناعي - AI system.

مراكز تبادل المعلومات وتحليلها ISACs مراكز تبادل المعلومات وتحليلها - Information Sharing and Analysis Centers؛ منظمات توفر منصة للشركات في صناعة معينة لتبادل معلومات التهديدات threat intelligence وبيانات الثغرات الأمنية vulnerability data في بيئة موثوقة.

بطاقة النموذج Model Card وثيقة موحدة توفر معلومات أساسية وشفافة حول نموذج تعلم الآلة machine learning model، بما في ذلك استخداماته المقصودة ومقاييس الأداء والاعتبارات الأخلاقية.

توجيه أمن الشبكات والمعلومات NIS2 Directive 2 توجيه أمن الشبكات والمعلومات 2 Network and Information Security Directive 2؛ قانون على مستوى الاتحاد الأوروبي يعزز متطلبات الأمن السيبراني cybersecurity للبنية التحتية الحيوية والخدمات الأساسية، مما يدفع إلى اعتماد حلول أمنية متقدمة مثل الذكاء الاصطناعي AI.

الخصوصية حسب التصميم Privacy by Design مبدأ، يفرضه لوائح مثل اللائحة العامة لحماية البيانات GDPR، يتطلب بناء ميزات الخصوصية وحماية البيانات في تصميم الأنظمة والعمليات منذ البداية.

إخفاء الهوية الزائفة (التسمية المستعارة) Pseudonymization عملية استبدال المعلومات الشخصية بمعرفات اصطناعية (أسماء مستعارة). على عكس إخفاء الهوية anonymization، هذه العملية قابلة للعكس عند الوصول إلى مفتاح منفصل.

إعادة تحديد الهوية Re-identification عملية ربط البيانات التي يفترض أنها مجهولة الهوية مرة أخرى بفرد معين من خلال دمجها مع مجموعات بيانات datasets أخرى متاحة.

الفريق الأحمر المسؤول للذكاء الاصطناعي Responsible AI Red-Teaming ممارسة اختبار نظام ذكاء اصطناعي AI system بشكل استباقي بحثاً عن الثغرات الأخلاقية والتحيزات واحتمالية سوء الاستخدام من خلال محاكاة السيناريوهات العدائية - adversarial scenarios من منظور أخلاقي، وليس فنياً فقط.

الإفصاح المسؤول Responsible Disclosure الممارسة الأخلاقية المتمثلة في الإبلاغ بشكل

خاص عن ثغرة أمنية vulnerability مكتشفة حديثاً في البرامج إلى المورد، مما يمنحهم الوقت لإصدار تصحيح برمجي patch قبل الإعلان عن الثغرة للعامة.

الحق في التفسير Right to Explanation حق قانوني يُمنح للأفراد بموجب لوائح مثل اللائحة

العامة لحماية البيانات GDPR لتلقي تفسير مفيد للقرارات التي يتخذها بشأنهم نظام آلي automated system.

تخطيط السيناريوهات Scenario Planning نشاط ضمن الفريق الأحمر للذكاء الاصطناعي

AI red-teaming يتضمن تطوير وتحليل مواقف افتراضية يمكن فيها إساءة استخدام

نظام الذكاء الاصطناعي AI system أو أن يؤدي إلى عواقب سلبية غير مقصودة.

مشاركة الثغرات الأمنية Vulnerability Sharing التبادل المنظم لمعلومات التهديدات threat

intelligence والبيانات حول الثغرات الأمنية vulnerabilities في البرامج بين المنظمات

الموثوقة لتعزيز الدفاعات الجماعية.

```latex

## باب 17

# الاتجاهات المستقبلية والمواضيع المتقدمة

### مقدمة

بينما نتطلع إلى أفق الأمن السيبراني، لا يمثل الذكاء الاصطناعي مجرد أداة، بل هو عنصر أساسي يشكل مستقبل الدفاع الرقمي. يتجاوز هذا الفصل التطبيقات المعروفة لاستكشاف أحدث ما توصلت إليه التقنيات الأمنية المدعومة بالذكاء الاصطناعي. سندرس النماذج الناشئة التي تبشر بأنظمة دفاع أكثر تعاونية ومرونة واستقلالية. تتمثل أهداف التعلم الرئيسية في فهم التعلم الفيدرالي للتعاون الذي يحافظ على الخصوصية، وإدراك التحدي الحاسم للتعلم الآلي الخصومي، والحصول على نظرة عامة على الذكاء الاصطناعي الآمن كمياً، وكلاء الأمن المستقلين، ودور الذكاء الاصطناعي في تطوير البرمجيات الآمنة. تمثل هذه المواضيع الجبهة التالية في بناء أنظمة دفاع ذكية حقاً.

### 1.17 التعلم الفيدرالي للدفاع التعاوني والحافظ للخصوصية

أحد أهم التحديات في الأمن السيبراني هو أن المدافعين غالباً ما يعملون في عزلة، غير قادرين على مشاركة بيانات التهديدات الحساسة بسبب مخاوف تتعلق بالخصوصية والسرية. يقدم التعلم الفيدرالي (Federated Learning (FL حلاً قوياً لهذه المشكلة، مما يتيح تدريب النماذج بشكل تعاوني دون تجميع البيانات الأولية في مكان مركزي.

### 1.1.17 مفاهيم التعلم الفيدرالي

التعلم الفيدرالي (Federated Learning - FL) هو نهج تعلم آلي موزع يتم فيه تدريب نموذج عبر العديد من الأجهزة أو المؤسسات اللامركزية التي تحتفظ ببيانات محلية [52]. بدلاً من إرسال البيانات الحساسة إلى خادم مركزي، يقوم كل مشارك، أو "عميل" client، بتدريب نموذج محلياً. بعد ذلك، يرسل العملاء تحديثات النموذج فقط—مثل التغييرات في أوزان النموذج—إلى خادم مركزي. يقوم الخادم بتجميع هذه التحديثات لإنشاء نموذج عالمي محسن، ثم يرسله مرة أخرى إلى العملاء للتحولة التالية من التدريب. تتكرر هذه الدورة، مما يسمح للنموذج العالمي بالتعلم من المعرفة الجماعية لجميع المشاركين دون الكشف عن أي بيانات أولية.

تتضمن البنية النموذجية خادماً مركزياً ينسق العملية والعديد من العملاء (على سبيل المثال، شركات مختلفة، مستشفيات، أو حتى أجهزة محمولة) الذين يقومون بالحسابات المحلية. هناك تنوع لهذا النهج يسمى التعلم المنفصل Split Learning، حيث يتم تقسيم نموذج الشبكة العصبية إلى أجزاء. يعالج الجزء الموجود لدى العميل البيانات الأولية محلياً، ويتم إرسال مخرجاته (التنشيطات الوسيطة) إلى الخادم لإكمال الحساب مع الجزء الخاص بالخادم من الشبكة. هذا يقلل بشكل أكبر من المعلومات المشتركة مع الخادم المركزي.

### 2.1.17 تطبيقات في الأمن السيبراني

تطبيقات التعلم الفيدرالي FL في الأمن السيبراني تحويلية، حيث تعزز التعاون في مجالات كان ذلك مستحيلاً فيها سابقاً:

- مشاركة معلومات التهديدات Threat Intelligence: يمكن لعدة مؤسسات مالية تدريب نموذج قوي لكشف الاحتيال بشكل جماعي. تستخدم كل مؤسسة بيانات معاملاتها الخاصة لتدريب نموذج محلي، وتشارك فقط تحديثات النموذج. يستفيد النموذج العالمي الناتج من بيانات متنوعة دون أن تضطر أي من البنوك إلى الكشف عن معلومات عملائها.
- تحليل البرامج الضارة الموزع Distributed Malware Analysis: يمكن لموردي حلول الأمن بناء نموذج أكثر قوة وشمولية للكشف عن البرامج الضارة من خلال التدريب على مجموعات بياناتهم الجماعية والخاصة لعينات البرامج الضارة. وهذا يسمح للنموذج العالمي بالتعرف على مجموعة أوسع من التهديدات مما يمكن لأي مورد واحد رؤيته بمفرده.
- الكشف التعاوني عن الشذوذ Collaborative Anomaly Detection: في البيئات

التي تحتوي على العديد من أجهزة الاستشعار الموزعة، مثل شبكات إنترنت الأشياء In-ternet of Things (IoT)، يمكن استخدام التعلم الفيدرالي FL لتدريب نموذج عالمي للكشف عن الشذوذ [61]. يمكن لكل جهاز IoT أو بوابة شبكة محلية المساهمة في تحديد أنماط الهجوم الجديدة مع الحفاظ على خصوصية بيانات حركة المرور المحلية.

### 3.1.17 التجميع الآمن والمتانة

على الرغم من عدم مشاركة البيانات الأولية، إلا أن تحديثات النموذج نفسها يمكن أن تسرب أحياناً معلومات حول بيانات التدريب الأساسية. لمواجهة هذا، تم تطوير العديد من تقنيات التجميع الآمن:

- التشفير المتماثل Homomorphic Encryption: تتيح هذه التقنية للخادم إجراء عمليات رياضية، مثل جمع تحديثات النموذج، مباشرة على البيانات المشفرة دون الحاجة إلى فك تشفيرها أولاً. وهذا يضمن أن الخادم المركزي لا يرى أبداً التحديثات الفردية كنص عادي.

- الحوسبة الآمنة متعددة الأطراف (SMC) Secure Multi-Party Computation: هي مجموعة من البروتوكولات التشفيرية التي تسمح لعدة أطراف بحساب دالة بشكل مشترك على مدخلاتهم (تحديثات النموذج) مع الحفاظ على خصوصية تلك المدخلات عن بعضهم البعض وعن الخادم. \* الخصوصية التفاضلية Differential Privacy: تتضمن هذه التقنية إضافة كمية محسوبة بعناية من الضوضاء الإحصائية إلى تحديث نموذج كل عميل قبل إرساله إلى الخادم. يوفر هذا ضمانات رياضية بأن مساهمة بيانات أي عميل واحد يتم إخفاؤها، مما يحمي الخصوصية الفردية.

تحدي حاسم آخر هو الدفاع ضد العملاء الخبيثاء الذين قد يحاولون تخريب عملية التدريب. يشير التجميع المتين ضد التسميم Poisoning-Robust Aggregation إلى الأساليب المصممة لاكتشاف وتجاهل تحديثات النموذج الخبيثة أو الشاذة، مما يضمن أن عدداً قليلاً من الجهات الفاعلة السيئة لا يمكنها إفساد النموذج العالمي النهائي.

### 4.1.17 عرض مفاهيمي: الكشف الفيدرالي عن الشذوذ

لتوضيح الفكرة الأساسية للتعلم الفيدرالي، يقدم القائمة 92 محاكاة مبسطة ومفاهيمية. في هذا السيناريو، نخيل "عميلين" منفصلين، لكل منهما بياناته الخاصة. يقومان بتدريب نموذج بسيط من



نوع المشفر التلقائي Autoencoder محلياً للكشف عن الشذوذ. ثم يقوم "خادم" مركزي بتجميع معرفتهما المكتسبة عن طريق حساب متوسط أوزان نماذجهما. توضح هذه المحاكاة عملية المتوسط الفيدرالي federated averaging، التي هي جوهر العديد من أنظمة التعلم الفيدرالي. لاحظ أن هذا المثال لا يتضمن اتصالات شبكية حقيقية أو ميزات أمان مثل التشفير من أجل البساطة. في سيناريو واقعي، ستم إدارة العملية الموضحة في القائمة 92 بواسطة إطار عمل مثل PySyft أو Flower، والذي يتعامل مع خطوات الاتصال والتجميع الآمنة تلقائياً. النقطة الأساسية هي أن النموذج العالمي يتحسن من خلال التعلم من جميع بيانات العملاء دون أن تغادر البيانات الأولية لأي عميل يئته المحلية أبداً.

## 2.17 التعلم الآلي الحصوي في الأمن السيبراني

في حين أن نماذج الذكاء الاصطناعي يمكن أن تكون مدافعين أقوياء، إلا أنها ليست معصومة من الخطأ. يمكن خداعها والتلاعب بها من قبل مهاجمين ماهرين. التعلم الآلي الحصوي Ad-versarial Machine Learning هو مجال مخصص لدراسة الهجمات على نماذج التعلم الآلي وتطوير دفاعات لجعلها أكثر قوة وموثوقة، خاصة في التطبيقات ذات الأهمية الأمنية.

### 1.2.17 نظرة أعمق على الهجمات الحصوية

يمكن تصنيف الهجمات الحصوية بشكل عام بناءً على هدف المهاجم ومعرفته بالنموذج. فيما يلي بعض الأنواع الأكثر شيوعاً في الأمن السيبراني [14]:

- هجمات التهرب Evasion Attacks: هذا هو النوع الأكثر شيوعاً من الهجمات أثناء مرحلة تشغيل النموذج. ينشئ المهاجم مدخلاً خبيثاً يكون نسخة معدلة قليلاً من هجوم حقيقي. على سبيل المثال، قد يضيف المهاجم بضعة بايتات مختارة بعناية إلى ملف برامج ضارة. تم تصميم هذه التغييرات لتكون صغيرة بما يكفي لعدم التأثير على وظيفة البرامج الضارة ولكنها مهمة بما يكفي لخداع كاشف قائم على الذكاء الاصطناعي لتصنيف الملف على أنه حميد.

- هجمات التسميم Poisoning Attacks: تستهدف هذه الهجمات مرحلة تدريب نموذج الذكاء الاصطناعي. يقوم المهاجم بحقن بيانات خبيثة مصممة خصيصاً في مجموعة بيانات التدريب. هذه البيانات "المسمومة" تفسد عملية التعلم، مما يتسبب في أداء ضعيف للنموذج.

النهائي أو، بشكل أكثر خبثاً، في وجود "باب خلفي" backdoor. الباب الخلفي هو مشغل مخفي، على سبيل المثال، أي ملف برامج ضارة يحتوي على سلسلة محددة من الأحرف تبدو غير ضارة قد يتم تصنيفه تلقائياً على أنه آمن بواسطة النموذج المخترق.

- هجمات عكس النموذج Model Inversion Attacks: تهدف هذه الهجمات إلى خرق خصوصية بيانات التدريب. من خلال إرسال استعلامات متكررة إلى نموذج مدرب وتحليل تنبؤاته، قد يتمكن المهاجم من إعادة بناء أجزاء من البيانات الحساسة التي تم تدريب النموذج عليها. على سبيل المثال، يمكن للمهاجم إعادة بناء صور وجوه من نموذج التعرف على الوجه.

- هجمات استنتاج العضوية Membership Inference Attacks: هذا هجوم آخر يركز على الخصوصية حيث يكون الهدف هو تحديد ما إذا كانت قطعة معينة من البيانات قد تم تضمينها في مجموعة تدريب النموذج. على سبيل المثال، يمكن للمهاجم استخدام هذا المعرفة ما إذا كان السجل الطبي لشخص معين قد تم استخدامه لتدريب نموذج تنبؤات الرعاية الصحية.

## 2.2.17 الدفاعات ضد الهجمات الخصوصية

يبحث مجتمع الأمن بنشاط عن دفاعات لجعل نماذج الذكاء الاصطناعي أكثر مرونة:

- التدريب الخصومي Adversarial Training: هذا هو الدفاع الأكثر فعالية واستخداماً على نطاق واسع. الفكرة هي "تطعيم" النموذج عن طريق تضمين أمثلة خصومية كجزء من بيانات التدريب الخاصة به. من خلال تعلم تصنيف هذه المدخلات الخبيثة بشكل صحيح أثناء التدريب، يصبح النموذج أكثر قوة ضد هجمات مماثلة أثناء النشر.

- التحسين المتين Robust Optimization: هي طرق تدريب تهدف إلى تقليل الخطأ في أسوأ الحالات للنموذج على المدخلات التي تم تشويشها. هذا يجعل النموذج بطبيعته أقل حساسية للتغيرات الصغيرة في مدخلاته.

- التقطير الدفاعي Defensive Distillation: تتضمن هذه التقنية عملية تدريب من خطوتين. أولاً، يتم تدريب نموذج "معلم" كبير. بعد ذلك، يتم تدريب نموذج "طالب" أصغر على المخرجات الاحتمالية للنموذج المعلم. يمكن لهذه العملية أن تجعل حدود قرار النموذج أكثر سلاسة، مما يجعل من الصعب على المهاجم العثور على أمثلة خصومية.

- المتانة المعتمدة Certified Robustness: يمكن لبعض الطرق أن توفر ضماناً رياضياً بأن تنبؤ النموذج لن يتغير لأي مدخل ضمن نطاق صغير معين حول نقطة بيانات معينة. على الرغم من قوتها، غالباً ما تكون هذه التقنيات مكلفة حسابياً ويمكن أن تقلل أحياناً من الدقة الإجمالية للنموذج.

- تحويلات الإدخال Input Transformations: يتضمن هذا النهج معالجة المدخلات مسبقاً لمحاولة إزالة أي تشويشات خصومية قبل أن تصل إلى النموذج. تشمل الأمثلة تقليل عمق ألوان الصورة أو تطبيق مرشحات التنعيم.

### 3.2.17 تحليل خوارزمي: طريقة إشارة التدرج السريعة

طريقة إشارة التدرج السريعة (FGSM) Fast Gradient Sign Method هي خوارزمية كلاسيكية وبديهية لتوليد هجمات التهرب [?]. تعمل عن طريق إيجاد الاتجاه الذي يجب تغيير ميزات الإدخال فيه لزيادة خطأ النموذج إلى أقصى حد. تتم العملية على النحو التالي:

(1) التمرير الأمامي وحساب الخسارة: يتم تمرير عينة إدخال شرعية (مثل صورة لميزات ملف حميد) عبر النموذج المستهدف وحساب الخسارة loss. الخسارة هي مقياس لمدى خطأ تنبؤ النموذج.

(2) حساب التدرج: حساب تدرج gradient دالة الخسارة بالنسبة لميزات الإدخال. التدرج هو متجه يشير في اتجاه الزيادة الأكثر حدة في الخسارة. بعبارة أخرى، يخبرنا بكيفية تغيير كل ميزة إدخال لجعل خطأ النموذج أكبر ما يمكن.

(3) أخذ الإشارة: تطبيق دالة الإشارة على التدرج. هذا يبسط متجه التدرج بحيث يكون كل عنصر إما 1- أو 0 أو 1+. هذا يخبرنا فقط بالاتجاه (زيادة أو نقصان) الذي يجب تغيير كل ميزة فيه، وليس الحجم.

(4) توليد مثال خصومي: ضرب متجه الإشارة الناتج في رقم صغير، يسمى إبسيلون epsilon ( $\epsilon$ )، والذي يتحكم في الحجم الإجمالي للتشويش. إضافة هذا التشويش الصغير إلى الإدخال الأصلي. والنتيجة هي إدخال جديد معدل قليلاً يشبه إلى حد كبير الأصلي ولكن من المرجح أن يتم تصنيفه بشكل خاطئ بواسطة النموذج. الصيغة هي:

$$x_{adv} = x + \epsilon \cdot \text{sign}(\nabla_x L(\theta, x, y))$$

حيث  $x$  هو الإدخال الأصلي،  $y$  هو التصنيف الحقيقي،  $\theta$  هي أوزان النموذج، و  $L$  هي دالة الخسارة.

### 3.17 الذكاء الاصطناعي الآمن كميًا وتشفير ما بعد الكوانتم

بالنظر إلى المستقبل البعيد، يمثل صعود الحوسبة الكمومية quantum computing تحديًا أساسيًا للأمن السيبراني الحديث. في حين أن أجهزة الكمبيوتر الكمومية العملية وواسعة النطاق لا تزال على بعد سنوات، فإن تأثيرها المحتمل يتطلب استعدادًا استباقيًا. يمكن لجهاز كمبيوتر كمومي متسامح مع الأخطاء كسر العديد من خوارزميات التشفير بالمتاح العام التي نعتمد عليها اليوم، مثل RSA وتشفير المنحنى الإهليلجي Elliptic Curve Cryptography (ECC) [60]. تؤمن هذه الخوارزميات كل شيء من حركة مرور الويب (HTTPS) إلى المعاملات المالية. يمكن لخصم مدعوم بالقدرة الكمومية فك تشفير الاتصالات السابقة والحالية، وتزوير التوقيعات الرقمية، وتقويض الثقة التي يقوم عليها عالمنا الرقمي. في هذا المشهد الجديد، يظهر مجالان مرتبطان:

- **تشفير ما بعد الكوانتم (Post-Quantum Cryptography (PQC):** هو تطوير خوارزميات تشفير جديدة آمنة ضد الهجوم من قبل أجهزة الكمبيوتر الكلاسيكية والكمومية. تعتمد هذه الخوارزميات الجديدة على مسائل رياضية مختلفة يعتقد أنها صعبة الحل حتى بالنسبة لأجهزة الكمبيوتر الكمومية.
  - **الذكاء الاصطناعي الآمن كميًا Quantum-Safe AI:** هو مجال استشرافي يركز على تأمين أنظمة الذكاء الاصطناعي في عالم ما بعد الكوانتم. ويشمل ذلك استخدام PQC لحماية نماذج الذكاء الاصطناعي نفسها، ومجموعات البيانات الضخمة التي يتم تدريبها عليها، وقنوات الاتصال التي تستخدمها. قد يلعب الذكاء الاصطناعي أيضًا دورًا في المساعدة في تصميم خوارزميات PQC الجديدة والتحقق من صحتها.
- بالنسبة للذكاء الاصطناعي في الأمن السيبراني، يعني هذا أن أنظمة الدفاع الذكية المستقبلية يجب أن تُبنى على أساس تشفير مقاوم للكم quantum-resistant cryptography لضمان سلامتها وسريتها على المدى الطويل.

### 4.17 وكلاء الأمن المستقلون والتعلم المعزز

تتمثل الرؤية النهائية للذكاء الاصطناعي في الأمن السيبراني في إنشاء وكلاء أمن مستقلين تمامًا autonomous security agents. ستكون هذه أنظمة ذكاء اصطناعي قادرة على اكتشاف التهديدات السيبرانية وتحليلها والاستجابة لها في الوقت الفعلي مع تدخل بشري ضئيل أو معدوم.

إحدى التقنيات التمكينية الرئيسية لهذه الرؤية هي التعلم المعزز Reinforcement Learning (RL). على عكس التعلم الخاضع للإشراف، الذي يتعلم من البيانات المصنفة، يتعلم وكلاء RL من خلال التفاعل مع بيئة وتلقي مكافآت أو عقوبات على أفعالهم [?]. تشمل التطبيقات المحتملة ما يلي:

- الاستجابة الآلية للحوادث Automated Incident Response: يمكن لوكيل RL تعلم التسلسل الأمثل للإجراءات التي يجب اتخاذها عند اكتشاف اختراق، مثل عزل الجهاز المخترق عن الشبكة، وحظر عنوان IP للمهاجم في جدار الحماية، وإنهاء العمليات الخبيثة.

- أنظمة الدفاع التكيفية Adaptive Defense Systems: يمكن لوكيل مستقل تعديل قواعد جدار الحماية أو سياسات نظام كشف التسلل باستمرار بناءً على مشهد التهديدات الحالي، وتعلم مواجهة تقنيات الهجوم الجديدة عند ظهورها.

- الصيد الذكي للتهديدات Intelligent Threat Hunting: يمكن لوكيل ذكاء اصطناعي البحث بشكل استباقي في شبكة عن علامات الاختراق، وتعلم مع مرور الوقت أين يبحث عن أكثر المؤشرات دقة لنشاط المهاجم، تماماً مثل المحلل البشري المتمرس.

في حين أن الأنظمة المستقلة بالكامل لا تزال في مرحلة البحث إلى حد كبير، إلا أنها تمثل هدفاً طويل الأجل لإنشاء أنظمة دفاع يمكنها العمل بسرعة الآلة.

## 5.17 الذكاء الاصطناعي لتطوير البرمجيات الآمنة

تجذر العديد من الحوادث الأمنية في الثغرات التي تم إدخالها أثناء تطوير البرمجيات. بدأ الذكاء الاصطناعي في لعب دور حاسم في تحويل الأمن "إلى اليسار"، ودعجه مباشرة في دورة حياة التطوير للعثور على العيوب وإصلاحها قبل نشر البرنامج على الإطلاق. غالباً ما يطلق على هذا المجال اسم DevSecOps.

قبل مناقشة دور الذكاء الاصطناعي، من المهم فهم طريقتين تقليديتين:

- الاختبار الثابت لأمن التطبيقات Static Application Security Testing (SAST): هذه طريقة "الصندوق الأبيض" white-box التي تحلل الكود المصدري للتطبيق أو الكود الثنائي دون تشغيله. إنها جيدة في العثور على أنواع معينة من الأخطاء مثل حقن SQL injection أو فيض الدائري buffer overflows.

## • الاختبار الديناميكي لأمن التطبيقات Dynamic Application Security Testing

(DAST): هذه طريقة "الصندوق الأسود" black-box التي تختبر تطبيقاً قيد التشغيل من الخارج، وترسل طلبات تبدو ضارة للعثور على ثغرات مثل البرمجة النصية عبر المواقع cross-site scripting أو التكوينات الأمنية الخاطئة.

يمكن للذكاء الاصطناعي أن يعزز بشكل كبير هذه الأساليب التقليدية ويقدم قدرات جديدة:

• تحسين كشف الثغرات: يمكن لنماذج الذكاء الاصطناعي، خاصة تلك التي يمكنها فهم بنية الكود (مثل شبكات الرسم البياني العصبية Graph Neural Networks)، تحسين SAST وDAST عن طريق اكتشاف الثغرات المعقدة أو المنطقية التي قد تفوتها المساحات الضوئية البسيطة القائمة على القواعد.

## • توليد الكود الآمن: يمكن لنماذج اللغة الكبيرة Large Language Models (LLMs)

المدرّبة على كميات هائلة من الكود الآمن أن تعمل كمساعدين أقوى للمطورين. يمكنها اقتراح مقتطفات من الكود الآمن، والتحذير من الأنماط غير الآمنة في الوقت الفعلي، ومساعدة المطورين على كتابة كود أكثر أماناً من البداية.

## • توليد تصحيحات الثغرات: مجال بحثي مثير للاهتمام يتضمن استخدام الذكاء الاصطناعي

لتوليد تصحيحات برمجية تلقائياً لإصلاح الثغرات المحددة. يمكن أن يقلل هذا بشكل كبير من الوقت المستغرق لتأمين تطبيق بعد اكتشاف عيب.

## • أتمتة مراجعة الكود: يمكن للذكاء الاصطناعي مساعدة مراجعي الكود البشريين عن

طريق الإبلاغ التلقائي عن أقسام الكود المعقدة أو المشبوهة التي تتطلب فحصاً دقيقاً، مما يجعل عملية المراجعة أسرع وأكثر فعالية.

## خاتمة

قدم هذا الفصل لمحة عن مستقبل الذكاء الاصطناعي في الأمن السيبراني. لقد رأينا اتجاهات واضحة نحو أنظمة أكثر تعاونية، وحافطة للخصوصية، وقوية، ومستقلة. يمكن التعلم الفيدرالي Federated Learning المؤسسات من العمل معاً ضد التهديدات المشتركة دون التضحية بالخصوصية. تعد دراسة التعلم الآلي الخصومي Adversarial ML ضرورية لبناء دفاعات ذكاء اصطناعي يمكنها الصمود أمام المهاجمين المصممين. بالنظر إلى المستقبل، سيتطلب صعود الحوسبة الكمومية تحولاً أساسياً إلى تشفير ما بعد الكوانتم لحماية أنظمة الذكاء الاصطناعي والبيانات الخاصة بنا. أخيراً، يعد دمج الذكاء الاصطناعي في دورة حياة تطوير البرمجيات ورؤية وكلاء الأمن

المستقلين بإنشاء أنظمة ليست ذكية في الكشف فحسب، بل أيضًا استباقية وذاتية الشفاء في دفاعها. بينما تواصل رحلتك، نشجعك على استكشاف الأبحاث المذكورة في هذه المجالات. يتطور المجال بسرعة، والتعلم المستمر هو مفتاح البقاء في صدارة تهديدات الغد.``

---

```

1 import torch
2 import torch.nn as nn
3 from torch.utils.data import TensorDataset, DataLoader
4 import numpy as np
5
6 # 1. Define a simple Autoencoder model for anomaly detection
7 class Autoencoder(nn.Module):
8 def __init__(self):
9 super(Autoencoder, self).__init__()
10 self.encoder = nn.Sequential(nn.Linear(10, 5), nn.ReLU())
11 self.decoder = nn.Sequential(nn.Linear(5, 10), nn.Sigmoid())
12
13 def forward(self, x):
14 x = self.encoder(x)
15 x = self.decoder(x)
16 return x
17
18 # 2. Simulate two clients with their own private data
19 np.random.seed(42)
20 client1_data = torch.tensor(np.random.rand(100, 10), dtype=torch.float32)
21 client2_data = torch.tensor(np.random.rand(120, 10), dtype=torch.float32)
22 client1_loader = DataLoader(TensorDataset(client1_data), batch_size=10)
23 client2_loader = DataLoader(TensorDataset(client2_data), batch_size=10)
24
25 clients = [{'model': Autoencoder(), 'data_loader': client1_loader},
26 {'model': Autoencoder(), 'data_loader': client2_loader}]
27
28 # 3. Simulate local training on each client
29 def train_local_model(model, data_loader, epochs=1):
30 criterion = nn.MSELoss()
31 optimizer = torch.optim.Adam(model.parameters(), lr=0.01)
32 model.train()
33 for epoch in range(epochs):
34 for data in data_loader:
35 inputs, = data # Unpack the tuple
36 outputs = model(inputs)
37 loss = criterion(outputs, inputs)
38 optimizer.zero_grad()
39 loss.backward()
40 optimizer.step()
41 return model.state_dict() # Return model weights (the "update")
42
43 print("--- Simulating Federated Learning Rounds ---")
44 global_model = Autoencoder()
45
46 # Simulate 2 rounds of federated training
47 for round_num in range(2):
48 print(f"\n--- Round {round_num + 1} ---")
49 local_updates = []
50
51 # Clients train locally and send updates
52 for i, client in enumerate(clients):
53 print(f"Client {i+1} is training locally...")
54 client['model'].load_state_dict(global_model.state_dict())
55 local_weights = train_local_model(client['model'], client['data_loader'])
56 local_updates.append(local_weights)
57
58 # 4. Server aggregates the updates (Federated Averaging)
59 print("Server is aggregating model updates...")
60 global_weights = global_model.state_dict()
61 for key in global_weights.keys():
62 # Simple average of the weights from all clients
63 global_weights[key] = torch.stack(
64 [local_updates[i][key] for i in range(len(local_updates))], 0
65).mean(0)
66
67 # 5. Server distributes the new global model
68 global_model.load_state_dict(global_weights)
69 print("New global model distributed to clients.")
70
71 print("\nFederated Learning simulation complete.")

```

---



## المختصرات والمصطلحات

FL (التعلم الفيدرالي Federated Learning) نهج تعلم آلي machine learning موزع يمكن عدة مشاركين من تدريب نموذج مشترك بشكل تعاوني دون تجميع بياناتهم الأولية الخاصة في مكان مركزي. بدلاً من ذلك، يتم فقط مشاركة تحديثات النموذج وتجميعها.

التعلم المقسم Split Learning هو أحد أشكال التعلم الموزع distributed learning حيث يتم تقسيم نموذج الشبكة العصبية neural network إلى أجزاء. يعالج العميل البيانات الأولية محلياً، ويتم إرسال ناتجه الوسيط إلى خادم لإكمال الحساب، مما يقلل من كشف البيانات بشكل أكبر.

IoT (إنترنت الأشياء Internet of Things) شبكة من الأجهزة المادية والمركبات والمباني وغيرها من العناصر المترابطة والمزودة بأجهزة استشعار وبرامج تسمح لهذه الأشياء بالاتصال وتبادل البيانات.

SMC (الحوسبة الآمنة متعددة الأطراف Secure Multi-Party Computation) مجموعة من البروتوكولات التشفيرية cryptographic protocols التي تسمح لعدة أطراف بحساب دالة بشكل مشترك على مدخلاتهم مع الحفاظ على خصوصية تلك المدخلات عن بعضهم البعض وعن الخادم المركزي.

التشفير المتماثل Homomorphic Encryption شكل من أشكال التشفير encryption يسمح بإجراء الحسابات مباشرة على البيانات المشفرة دون الحاجة إلى فك تشفيرها أولاً، مما يضمن خصوصية البيانات حتى أثناء المعالجة.

الخصوصية التفاضلية Differential Privacy تقنية توفر ضمانات خصوصية رياضية عن طريق إضافة كمية محسوبة بعناية من الضوضاء الإحصائية إلى البيانات أو تحديثات النماذج، مما يخفي مساهمة أي فرد بعينه.

التجميع المقاوم للتسميم Poisoning-Robust Aggregation في التعلم الفيدرالي Federated Learning، هي طرق مصممة لاكتشاف وتجاهل تحديثات النماذج الخبيثة المقدمة من جهات فاعلة سيئة، مما يمنعها من إفساد النموذج العالمي المشترك.

الرمز التلقائي Autoencoder نوع من الشبكات العصبية neural network يستخدم بشكل شائع لكشف الشذوذ anomaly detection، حيث يتعلم ضغط البيانات إلى تمثيل مضغوط (تشفير encode) ثم إعادة بنائها إلى شكلها الأصلي (فك التشفير decode).

تعلم الآلة التنافسي Adversarial Machine Learning مجال في أمن الذكاء الاصطناعي  
AI security يركز على دراسة الهجمات ضد نماذج التعلم الآلي machine learning  
وتطوير الدفاعات المقابلة لجعلها أكثر قوة وموثوقية.

هجمات التهرب Evasion Attacks النوع الأكثر شيوعاً من الهجمات التنافسية -adversarial attack، حيث يقوم المهاجم بإجراء تعديلات صغيرة ودقيقة على مدخل خبيث (مثل ملف برمجية خبيثة malware) لخداع كاشف الذكاء الاصطناعي لتصنيفه على أنه حميد.

هجمات التسميم Poisoning Attacks هجمات تستهدف مرحلة تدريب نموذج الذكاء الاصطناعي، حيث يقوم المهاجم بحقن بيانات خبيثة معدة خصيصاً في مجموعة التدريب لإفساد النموذج النهائي أو إنشاء باب خلفي backdoor مخفي.

هجمات عكس النموذج Model Inversion Attacks نوع من هجمات الخصوصية privacy attack حيث يحاول الخصم إعادة بناء بيانات تدريب حساسة عن طريق الاستعلام المتكرر عن نموذج مدرب وتحليل مخرجاته.

هجمات استنتاج العضوية Membership Inference Attacks هجمة خصوصية privacy attack يكون الهدف منها تحديد ما إذا كان سجل بيانات معين جزءاً من مجموعة البيانات المستخدمة لتدريب نموذج تعلم آلي machine learning.

التدريب التنافسي Adversarial Training تقنية دفاع شائعة حيث يتم تدريب النموذج عمداً على مزيج من الأمثلة العادية والتنافسية adversarial examples، مما يجعله أكثر صموداً أمام الهجمات المستقبلية.

التقطير الدفاعي Defensive Distillation طريقة دفاع يتم فيها تدريب نموذج "طالب" أصغر على المخرجات الاحتمالية لنموذج "معلم" أكبر، مما يمكن أن يُسهّل حدود القرار ويجعل نموذج الطالب أكثر قوة.

المتانة المعتمدة Certified Robustness طرق دفاع توفر ضماناً رياضياً بأن تنبؤ النموذج سيبقى دون تغيير لأي مدخل ضمن نطاق صغير معين حول نقطة بيانات محددة.

FGSM (طريقة الإشارة المتدرجة السريعة Fast Gradient Sign Method) خوارزمية كلاسيكية لتوليد الأمثلة التنافسية adversarial examples. تعمل عن طريق حساب تدرج gradient خطأ النموذج بالنسبة للمدخلات ثم تعديل المدخلات بشكل طفيف في الاتجاه الذي يزيد هذا الخطأ إلى أقصى حد.

ECC (تشفير المنحنى الإهليلجي Elliptic Curve Cryptography) نوع من تشفير المفتاح العام public-key cryptography يستخدم على نطاق واسع لتأمين الاتصالات الرقمية ولكنه عرضة للكسر بواسطة الحواسيب الكمومية quantum computers الكبيرة في المستقبل.

PQC (التشفير ما بعد الكمي Post-Quantum Cryptography) مجال في علم التشفير-cryp-tography يركز على تطوير خوارزميات تشفير cryptographic algorithms جديدة تكون آمنة ضد الهجمات من كل من الحواسيب الكلاسيكية والكمومية classical and quantum computers.

الذكاء الاصطناعي الآمن كموياً Quantum-Safe AI مجال ناشئ يركز على تأمين أنظمة الذكاء الاصطناعي في عالم ما بعد الكم post-quantum، وذلك بشكل أساسي عن طريق استخدام التشفير ما بعد الكمي PQC لحماية نماذج الذكاء الاصطناعي وبيانات تدريبها وقنوات الاتصال التي تستخدمها.

RL (التعلم المعزز Reinforcement Learning) نوع من تعلم الآلة machine learning حيث يتعلم وكيل ذكاء اصطناعي اتخاذ القرارات المثلى من خلال التفاعل مع بيئة وتلقي مكافآت أو عقوبات على أفعاله.

ممارسات التطوير والأمان والعمليات DevSecOps ممارسة دمج اختبار الأمان وإدارة الثغرات مباشرة في كل مرحلة من مراحل دورة حياة تطوير البرمجيات والعمليات (التطوير والعمليات DevOps).

SAST (اختبار أمان التطبيقات الثابت Static Application Security Testing) طريقة اختبار "الصندوق الأبيض" white-box تقوم بتحليل الكود المصدري أو الملف الثنائي للتطبيق من الداخل للعثور على ثغرات أمنية دون تشغيل البرنامج فعلياً.

DAST (اختبار أمان التطبيقات الديناميكي Dynamic Application Security Testing) طريقة اختبار "الصندوق الأسود" black-box تقوم بفحص تطبيق قيد التشغيل من الخارج، عن طريق إرسال طلبات متنوعة للعثور على ثغرات أمنية وأخطاء في التكوين.

LLMs (النماذج اللغوية الكبيرة Large Language Models) نماذج ذكاء اصطناعي مدربة على كميات هائلة من النصوص والتعليمات البرمجية، قادرة على فهم وإنشاء والمساعدة في كتابة التعليمات البرمجية لتحسين أمان البرمجيات أثناء التطوير.

# Bibliography

- [1] VirusShare: Malware sample repository.
- [2] Internet crime report 2022. Technical report, Federal Bureau of Investigation, Internet Crime Complaint Center (IC3), 2022.
- [3] abuse.ch. MalwareBazaar: Malware sample exchange.
- [4] Zeeshan Ahmad, A Shahid Khan, C Wai Shiang, J Abdullah, and F Ahmad. Network intrusion detection system: A systematic study of machine learning and deep learning approaches. *Transactions on Emerging Telecommunications Technologies*, 32(1):e4150, 2021.
- [5] Hyrum S. Anderson and Phil Roth. EMBER: An open dataset for training static PE malware machine learning models, 2018.
- [6] Ankur Ankan and Abinash Panda. pgmpy: Probabilistic graphical models using python. In *Proceedings of the 14th Python in Science Conference (SciPy 2015)*, pages 6–11, 2015.
- [7] Battista Biggio, Blaine Nelson, and Pavel Laskov. Poisoning attacks against support vector machines. In *Proceedings of the 29th International Conference on Machine Learning (ICML)*, pages 1807–1814, 2012.
- [8] Philippe Biondi. Scapy. <http://www.secdev.org/projects/scapy>, 2010.
- [9] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.
- [10] Markus M. Breunig, Hans-Peter Kriegel, Raymond T. Ng, and Jörg Sander. LOF: Identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, pages 93–104, 2000.
- [11] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuScenes: A multimodal dataset for autonomous driving. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11621–11631, 2020.

- [12] Dawn M. Cappelli, Andrew P. Moore, and Randall F. Trzeciak. The CERT guide to insider threats: How to prevent, detect, and respond to information technology crimes. 2012.
- [13] Eoghan Casey. *Digital Evidence and Computer Crime: Forensic Science, Computers, and the Internet*. Academic Press, 3rd edition, 2011.
- [14] Anirban Chakraborty, Manaar Alam, Vishal Dey, Anupam Chattopadhyay, and Debdeep Mukhopadhyay. Adversarial attacks and defences: A survey, 2018.
- [15] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. SMOTE: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.
- [16] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 785–794, 2016.
- [17] Dave Crocker, Tony Hansen, and Murray Kucherawy. Domainkeys identified mail (dkim) signatures. Rfc 6376, Internet Engineering Task Force, September 2011.
- [18] CrowdStrike. Hybrid analysis: Free automated malware analysis service.
- [19] Cybersecurity and Infrastructure Security Agency. Known exploited vulnerabilities catalog.
- [20] Databricks. What is a feature store?
- [21] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4171–4186, 2019.
- [22] Docker, Inc. Docker overview.
- [23] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96)*, pages 226–231, 1996.
- [24] European Commission. Proposal for a regulation laying down harmonised rules on artificial intelligence (artificial intelligence act), 2021.
- [25] European Union. Regulation (EU) 2016/679 of the european parliament and of the council (general data protection regulation), 2016.
- [26] European Union. Directive (EU) 2022/2555 on measures for a high common level of cybersecurity across the union (NIS2), 2022.

- [27] João Gama, Indrė Žliobaitė, Albert Bifet, Mykola Pechenizkiy, and Abdelhamid Bouchachia. A survey on concept drift adaptation. *ACM Computing Surveys*, 46(4):1–37, 2014.
- [28] Sebastian Garcia, Martin Grill, Jan Stiborek, and Alejandro Zunino. An empirical comparison of botnet detection methods. *Computers & Security*, 45:100–123, 2014.
- [29] Timnit Gebru, Jamie Morgenstern, Briana Vecchione, Jennifer Wortman Vaughan, Hanna Wallach, Hal Daumé III, and Kate Crawford. Datasheets for datasets. *Communications of the ACM*, 64(12):86–92, 2021.
- [30] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the KITTI vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3354–3361, 2012.
- [31] Richard Harang and Ethan M. Rudd. SOREL-20M: A large scale benchmark dataset for malicious PE detection, 2020.
- [32] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.
- [33] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.
- [34] Iterative, Inc. DVC: Data version control documentation.
- [35] Daniel Jurafsky and James H. Martin. *Speech and Language Processing*. 3rd (draft) edition, 2023. Draft available at <https://web.stanford.edu/~jurafsky/slp3/>.
- [36] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82(1):35–45, 1960.
- [37] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. LightGBM: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems*, volume 30, 2017.
- [38] Yoon Kim. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, 2014.
- [39] Scott Kitterman. Sender policy framework (spf) for authorizing use of domains in return paths. Rfc 7208, Internet Engineering Task Force, April 2014.
- [40] Bryan Klimt and Yiming Yang. The enron corpus: A new dataset for email classification research. In *Machine learning: ECML 2004*, pages 217–226. Springer, 2004.

- [41] Daphne Koller and Nir Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.
- [42] Dominik Kreuzberger, Niklas Kühl, and Sebastian Hirschl. Machine learning operations (MLOps): Overview, definition, and architecture, 2022.
- [43] Murray Kucherawy and Elizabeth Zwicky. Domain-based message authentication, reporting, and conformance (dmarc). Rfc 7489, Internet Engineering Task Force, March 2015.
- [44] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.
- [45] Edward A. Lee. Cyber physical systems: Design challenges. In *2008 11th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing (ISORC)*, pages 363–369, 2008.
- [46] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation forest. In *2008 Eighth IEEE International Conference on Data Mining*, pages 413–422, 2008.
- [47] Manuel Lopez-Martin, Belen Carro, Antonio Sanchez-Esguevillas, and Jaime Lloret. Conditional variational autoencoder for prediction and feature recovery applied to intrusion detection in IoT. *Sensors*, 17(9):1967, 2017.
- [48] Scott M. Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems 30 (NIPS 2017)*, pages 4765–4774, 2017.
- [49] Will Maddern, Geoffrey Pascoe, Chris Linegar, and Paul Newman. 1 year, 1000 km: The Oxford RobotCar dataset. *The International Journal of Robotics Research*, 36(1):3–15, 2017.
- [50] Prasanta Chandra Mahalanobis. On the generalised distance in statistics. *Proceedings of the National Institute of Sciences of India*, 2(1):49–55, 1936.
- [51] Aditya P. Mathur and Nils Ole Tippenhauer. SWaT: A water treatment testbed for research and training on ICS security. In *2016 International Workshop on Cyber-physical Systems for Smart Water Networks (CySWater)*, pages 31–36, 2016.
- [52] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 1273–1282, 2017.



- [53] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems (NIPS)*, volume 26, 2013.
- [54] Charlie Miller and Chris Valasek. Remote exploitation of an unaltered passenger vehicle. Black Hat USA, 2015.
- [55] Margaret Mitchell, Simone Wu, Andrew Zaldivar, Parker Barnes, Lucy Vasserman, Ben Hutchinson, Elena Spitzer, Inioluwa Deborah Raji, and Timnit Gebru. Model cards for model reporting. In *Proceedings of the Conference on Fairness, Accountability, and Transparency (FAT\*)*, pages 220–229, 2019.
- [56] Rami Mohammad and Lee McCluskey. Phishing websites [dataset], 2012.
- [57] Nour Moustafa and Jill Slay. Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set). In *2015 military communications and information systems conference (MilCIS)*, pages 1–6. IEEE, 2015.
- [58] Lakshmanan Nataraj, S. Karthikeyan, Gregoire Jacob, and B. S. Manjunath. Malware images: Visualization and automatic classification. In *Proceedings of the 8th International Symposium on Visualization for Cyber Security (VizSec)*, 2011.
- [59] National Institute of Standards and Technology. National vulnerability database (NVD).
- [60] National Institute of Standards and Technology. Post-quantum cryptography project, 2016.
- [61] Dinh C. Nguyen, Ming Ding, Pubudu N. Pathirana, Aruna Seneviratne, Jun Li, and H. Vincent Poor. Federated learning for internet of things: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, 23(3):1622–1658, 2021.
- [62] OffSec Services Limited. Exploit database.
- [63] Vern Paxson. Bro: a system for detecting network intruders in real-time. *Computer networks*, 31(23-24):2435–2463, 1999.
- [64] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.
- [65] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

- [66] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should I trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1135–1144, 2016.
- [67] Mayu Sakurada and Takehisa Yairi. Anomaly detection using autoencoders with nonlinear dimensionality reduction. In *Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis*, pages 4–11, 2014.
- [68] Gerard Salton and Christopher Buckley. Term-weighting approaches in automatic text retrieval. *Information processing & management*, 24(5):513–523, 1988.
- [69] John Salvatier, Thomas V. Wiecki, and Christopher Fonnesbeck. Probabilistic programming in python using PyMC3. *PeerJ Computer Science*, 2:e55, 2016.
- [70] Bernhard Schölkopf, John C. Platt, John Shawe-Taylor, Alex J. Smola, and Robert C. Williamson. Estimating the support of a high-dimensional distribution. *Neural Computation*, 13(7):1443–1471, 2001.
- [71] Iman Sharafaldin, Arash Habibi Lashkari, and Ali A Ghorbani. Toward generating a new intrusion detection dataset and intrusion traffic characterization. In *ICISSP*, pages 108–116, 2018.
- [72] State of California. California consumer privacy act of 2018 (CCPA), 2018.
- [73] Mahbod Tavallaei, Ebrahim Bagheri, Wei Lu, and Ali A Ghorbani. A detailed analysis of the kdd cup 99 data set. In *2009 IEEE symposium on computational intelligence for security and defense applications*, pages 1–6. IEEE, 2009.
- [74] The Kubernetes Authors. What is Kubernetes?
- [75] The MITRE Corporation. CVE program: Common vulnerabilities and exposures.
- [76] The MLflow Project. MLflow documentation.
- [77] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.
- [78] Rakesh Verma, Narasimha Shashidhar, and Nabil Hossain. Detecting phishing emails the natural language way. In *Computer Security – ESORICS 2012*, pages 824–841. Springer, 2012.
- [79] VirusTotal. YARA: The pattern matching swiss knife for malware researchers.

- [80] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S. Yu. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 32(1):4–24, 2021.