

Securing AI Systems



Sam Bowne

Mar 1, 2025

Whoami

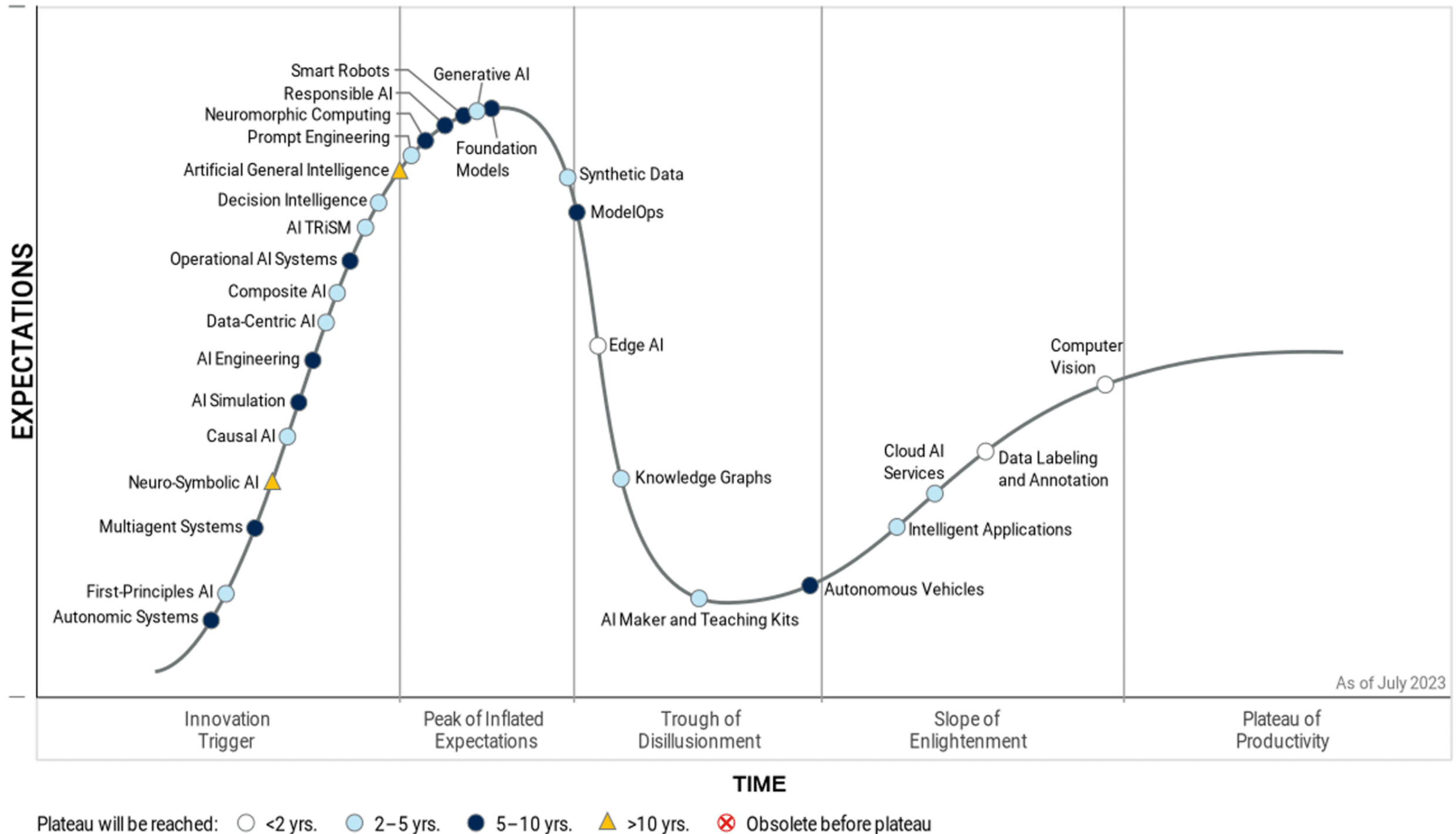
- Sam Bowne
- Instructor at City College San Francisco
- Corporate trainer
- Web: samsclass.info
- Email:
sbowne@ccsf.edu
sam.bowne@infosecdecoded.com
- Mastodon:
[sambowne@infosec.exchange](https://infosec.exchange/@sambowne)



Types of AI Systems

Figure 1: Hype Cycle for Artificial Intelligence, 2023

Hype Cycle for Artificial Intelligence, 2023



Types of Machine Learning Systems

Supervised Learning

- Training data has labels
- Indicating desired solution

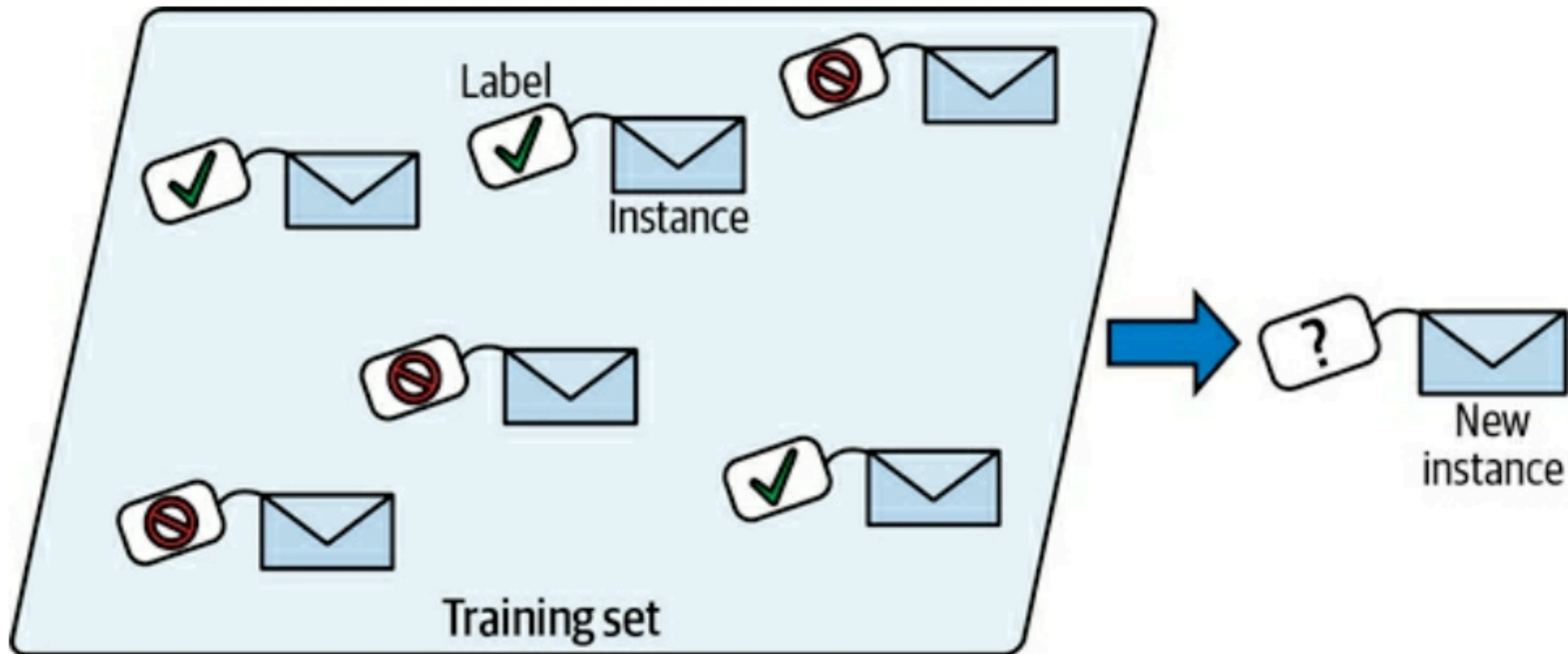


Figure 1-5. A labeled training set for spam classification (an example of supervised learning)

Unsupervised Learning

- Training data is unlabeled

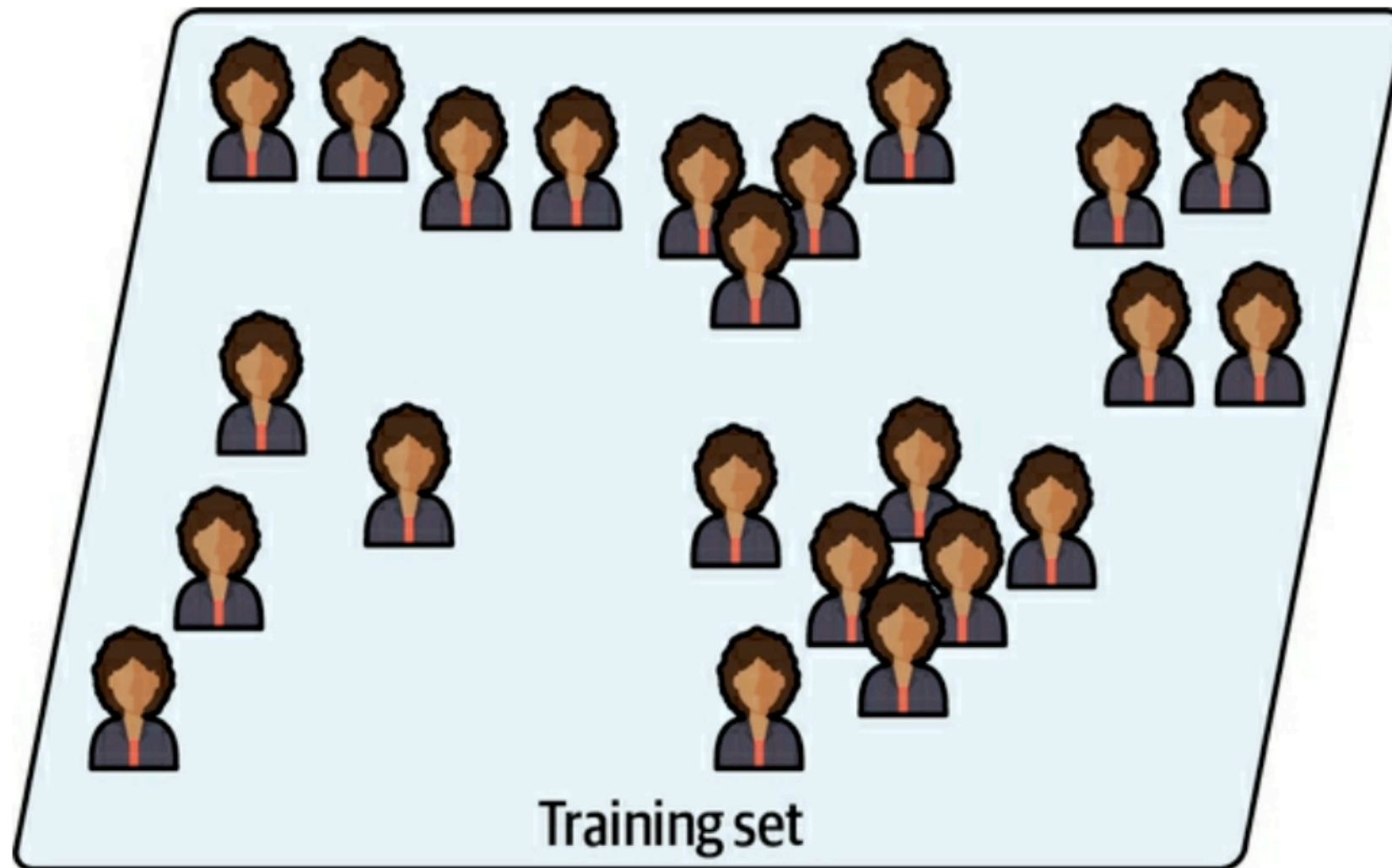


Figure 1-7. An unlabeled training set for unsupervised learning

Unsupervised Learning

- **Clustering** algorithm
 - Sorts data into groups

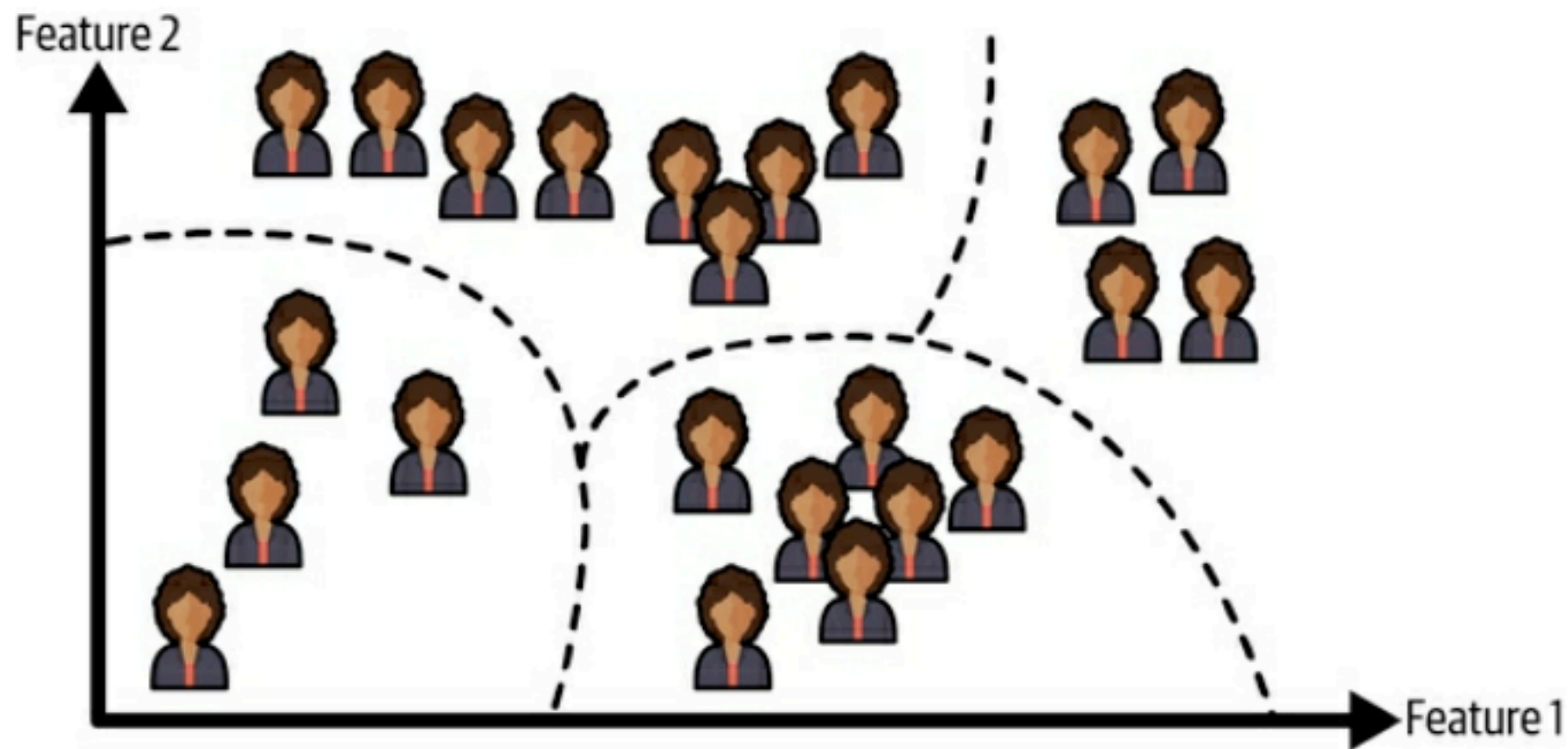


Figure 1-8. Clustering

Unsupervised Learning

- **Anomaly detection**
 - Find unusual credit card transactions
 - Find manufacturing defects

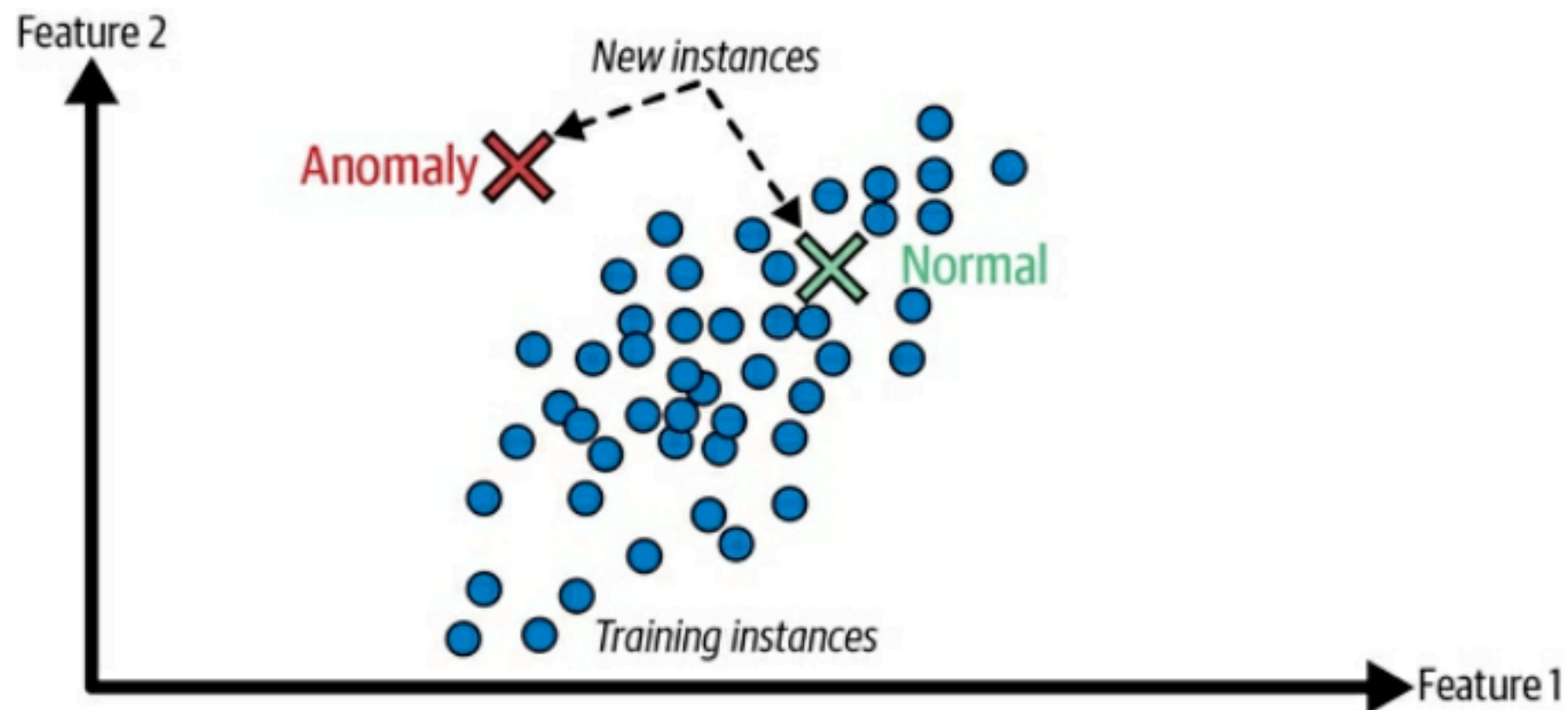


Figure 1-10. Anomaly detection

Self-Supervised Learning

- Generates a labeled dataset from an unlabeled one
- Example: mask part of an image, train a model to recover the original image

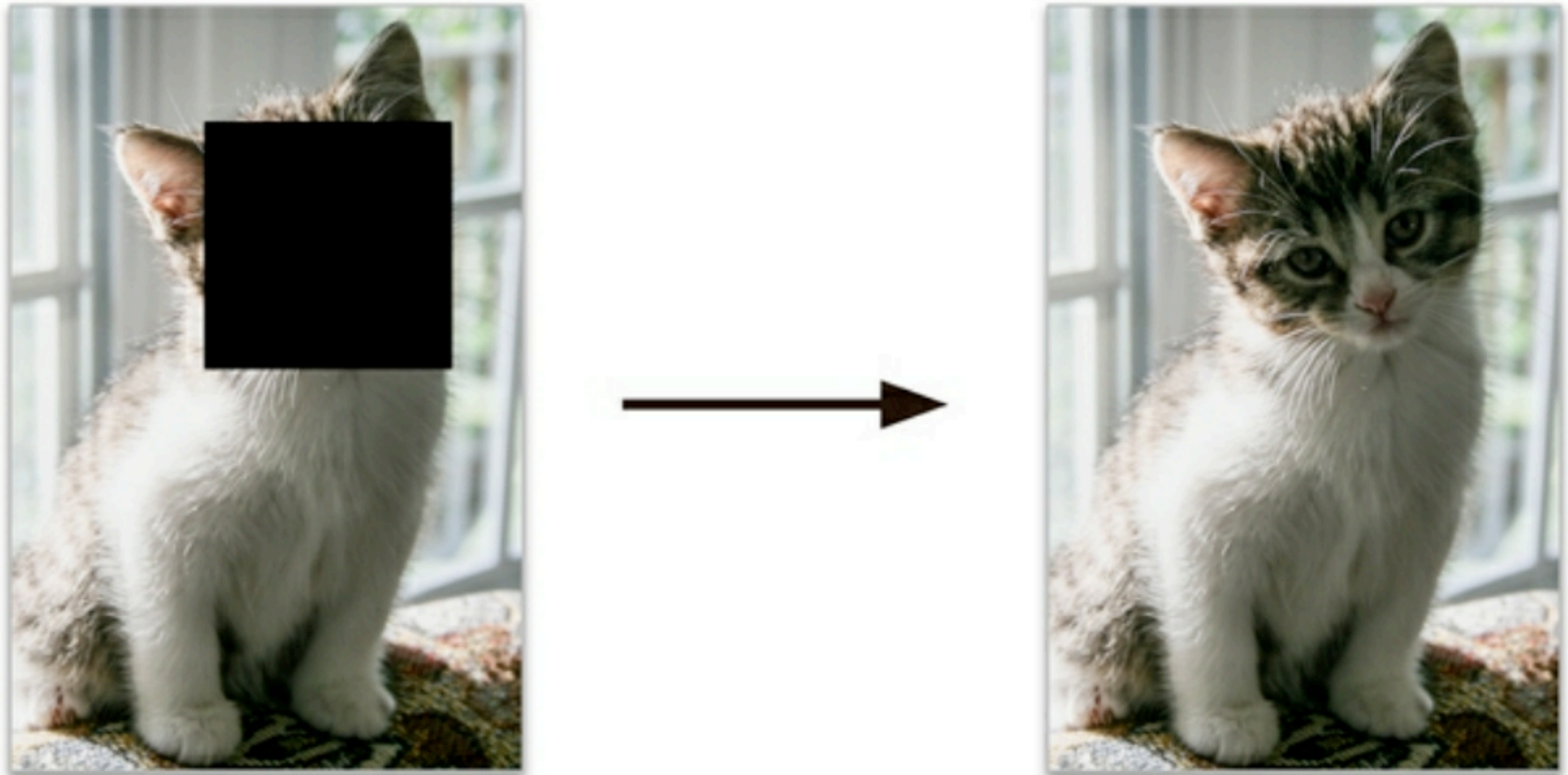


Figure 1-12. Self-supervised learning example: input (left) and target (right)

Large Language Models

- Start with sentences written by humans
- Randomly mask some words
- Learn to predict the masked word

```
0.32 can      artificial intelligence can take over the world.  
0.18 will    artificial intelligence will take over the world.  
0.06 to      artificial intelligence to take over the world.  
0.05 ##s     artificial intelligences take over the world.  
0.05 would   artificial intelligence would take over the world.
```

```
from transformers import pipeline  
unmasker = pipeline('fill-mask', model='bert-base-uncased')  
result = unmasker("Artificial Intelligence [MASK] take over the world.")  
print()  
for r in result:  
    print(round(r['score'], 2), r['token_str'], "\t", r['sequence'])
```



- A “reasoning” model: performs a “thinking” process before generating an answer
- Trained on a small set of carefully prepared question-answer pairs
- Much faster and cheaper than traditional training
- Similar to how students in school learn

Securing AI Systems

NIST AI 100-1

Artificial Intelligence Risk Management Framework (AI RMF 1.0)



Harm to People

- Individual: Harm to a person's civil liberties, rights, physical or psychological safety, or economic opportunity.
- Group/Community: Harm to a group such as discrimination against a population sub-group.
- Societal: Harm to democratic participation or educational access.

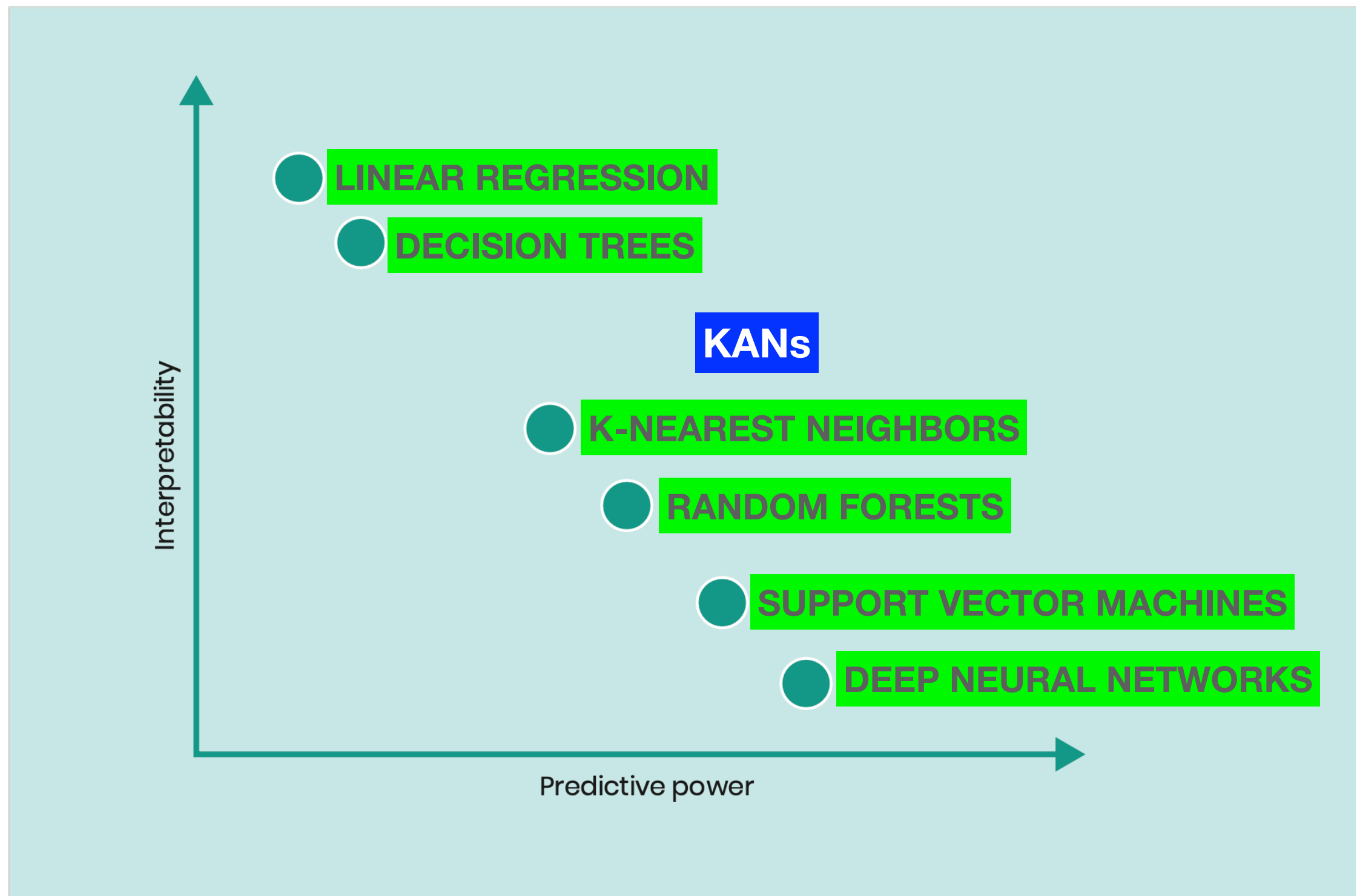
Harm to an Organization

- Harm to an organization's business operations.
- Harm to an organization from security breaches or monetary loss.
- Harm to an organization's reputation.

Harm to an Ecosystem

- Harm to interconnected and interdependent elements and resources.
- Harm to the global financial system, supply chain, or interrelated systems.
- Harm to natural resources, the environment, and planet.

Inscrutability



Principles for the security of machine learning

Version 1

Published August 2022



National Cyber
Security Centre

a part of GCHQ

Section 1

Prerequisites and wider considerations

Section 2

Requirements and development

Inception

Objectives
High level requirements
Risk assessment
Policies and compliance

Design and development

Approach
Technical requirements
Architecture
Gathering of training data
Code for data processing
Model creation
Risk treatment

Verification and validation

Verification of data processing
System verification
Risk monitoring and review

Section 3 Deployment

Operation and monitoring

Operating data input
Model execution
Model updates
Risk management

Deployment

Runtime deployment
Model deployment
Risk treatment

Section 4 Continual / online learning

Continuous validation

Validation data processing
System validation
Risk management
Continuous improvement

Section 5

End of life

Re-evaluate

- Evaluate operating performance
- Refine objective
- Refine requirements
- Risk monitoring
- Lessons learnt

Reirement

- Data disposal
- Model disposal
- Decommissioning and model card

OWASP Top Ten Machine Learning Risks

- **ML01:2023 Input Manipulation Attack**
 - **ML02:2023 Data Poisoning Attack**
 - **ML03:2023 Model Inversion Attack**
 - **ML04:2023 Membership Inference Attack**
 - **ML05:2023 Model Theft**
 - **ML06:2023 AI Supply Chain Attacks**
 - **ML07:2023 Transfer Learning Attack**
 - **ML08:2023 Model Skewing**
 - **ML09:2023 Output Integrity Attack**
 - **ML10:2023 Model Poisoning**
-
- <https://owasp.org/www-project-machine-learning-security-top-10/>

- **ML01:2023 Input Manipulation Attack**
 - An attacker deliberately alters input data to mislead the model
 - This attack is also called **evasion**
 - Example: a model is trained to tell cat images from dog images. An attacker modifies a cat image so it is misclassified as a dog.
- **ML02:2023 Data Poisoning Attack**
 - An attacker manipulates the training data to cause the model to behave in an undesirable way
- **ML03:2023 Model Inversion Attack**
 - An attacker reverse-engineers the model to extract information from it
 - Example: a model is trained to recognize faces. An attacker inputs images of individuals into the model and recovers the personal information of the individuals from the model's predictions, such as their name, address, or social security number.

- **ML04:2023 Membership Inference Attack**

- An attacker manipulates the model's training data in order to cause it to behave in a way that exposes sensitive information
- Example: A malicious attacker trains a machine learning model on a dataset of financial records and uses it to query whether or not a particular individual's record was included in the training data.

- **ML05:2023 Model Theft**

- An attacker gains access to the model's parameters
- Example: Stealing a machine learning model from a competitor

- **ML06:2023 AI Supply Chain Attacks**

- An attacker modifies or replaces a machine learning library or model that is used by a system

- **ML07:2023 Transfer Learning Attack**

- An attacker trains a model on one task and then fine-tunes it on another task to cause it to behave in an undesirable way
- Example: An attacker trains a machine learning model on a malicious dataset that contains manipulated images of faces. The attacker then transfers the model's knowledge to a target face recognition system. As a result, the face recognition system starts making incorrect predictions, allowing the attacker to bypass the security and gain access to sensitive information.

- **ML08:2023 Model Skewing**

- An attacker manipulates the distribution of the training data to cause the model to behave in an undesirable way.
- Example: The attacker provides fake feedback data to a loan-approving machine learning system. As a result, the model's predictions are skewed, and the attacker's chances of getting a loan approved are significantly increased.

- **ML09:2023 Output Integrity Attack**

- An attacker aims to modify or manipulate the output of a machine learning model in order to change its behavior or cause harm to the system it is used in.
- Example: An attacker has gained access to the output of a machine learning model that is being used to diagnose diseases in a hospital. The attacker modifies the output of the model, making it provide incorrect diagnoses for patients.

- **ML10:2023 Neural Net Reprogramming**

- An attacker manipulates the model's parameters to cause it to behave in an undesirable way.
- Example: A bank is using a machine learning model to identify handwritten characters on cheques. An attacker manipulates the parameters of the model by altering the images in the training dataset or directly modifying the parameters in the model. This can result in the model misidentifying characters, leading to incorrect amounts being processed.

OWASP Top 10 for LLM Applications

VERSION 1.1

Published: October 16, 2023

<https://owasp.org/www-project-top-10-for-large-language-model-applications/>

LLM01: Prompt Injection

This manipulates a large language model (LLM) through crafty inputs, causing unintended actions by the LLM. Direct injections overwrite system prompts, while indirect ones manipulate inputs from external sources.

LLM02: Insecure Output Handling

This vulnerability occurs when an LLM output is accepted without scrutiny, exposing backend systems. Misuse may lead to severe consequences like XSS, CSRF, SSRF, privilege escalation, or remote code execution.

LLM03: Training Data Poisoning

This occurs when LLM training data is tampered, introducing vulnerabilities or biases that compromise security, effectiveness, or ethical behavior. Sources include Common Crawl, WebText, OpenWebText, & books.

LLM04: Model Denial of Service

Attackers cause resource-heavy operations on LLMs, leading to service degradation or high costs. The vulnerability is magnified due to the resource-intensive nature of LLMs and unpredictability of user inputs.

LLM05: Supply Chain Vulnerabilities

LLM application lifecycle can be compromised by vulnerable components or services, leading to security attacks. Using third-party datasets, pre-trained models, and plugins can add vulnerabilities.

LLM06: Sensitive Information Disclosure

LLMs may inadvertently reveal confidential data in their responses, leading to unauthorized data access, privacy violations, and security breaches. It's crucial to implement data sanitization and strict user policies to mitigate this.

LLM07: Insecure Plugin Design

LLM plugins can have insecure inputs and insufficient access control. This lack of application control makes them easier to exploit and can result in consequences like remote code execution.

LLM08: Excessive Agency

LLM-based systems may undertake actions leading to unintended consequences. The issue arises from excessive functionality, permissions, or autonomy granted to the LLM-based systems.

LLM09: Overreliance

Systems or people overly depending on LLMs without oversight may face misinformation, miscommunication, legal issues, and security vulnerabilities due to incorrect or inappropriate content generated by LLMs.

LLM10: Model Theft

This involves unauthorized access, copying, or exfiltration of proprietary LLM models. The impact includes economic losses, compromised competitive advantage, and potential access to sensitive information.

Copilot Security: Ensuring a Secure Microsoft Copilot Rollout

This article describes how Microsoft 365 Copilot's security model works and the risks that must be mitigated to ensure a safe rollout.



Rob Sobers

| 5 min read

| Last updated April 11, 2024



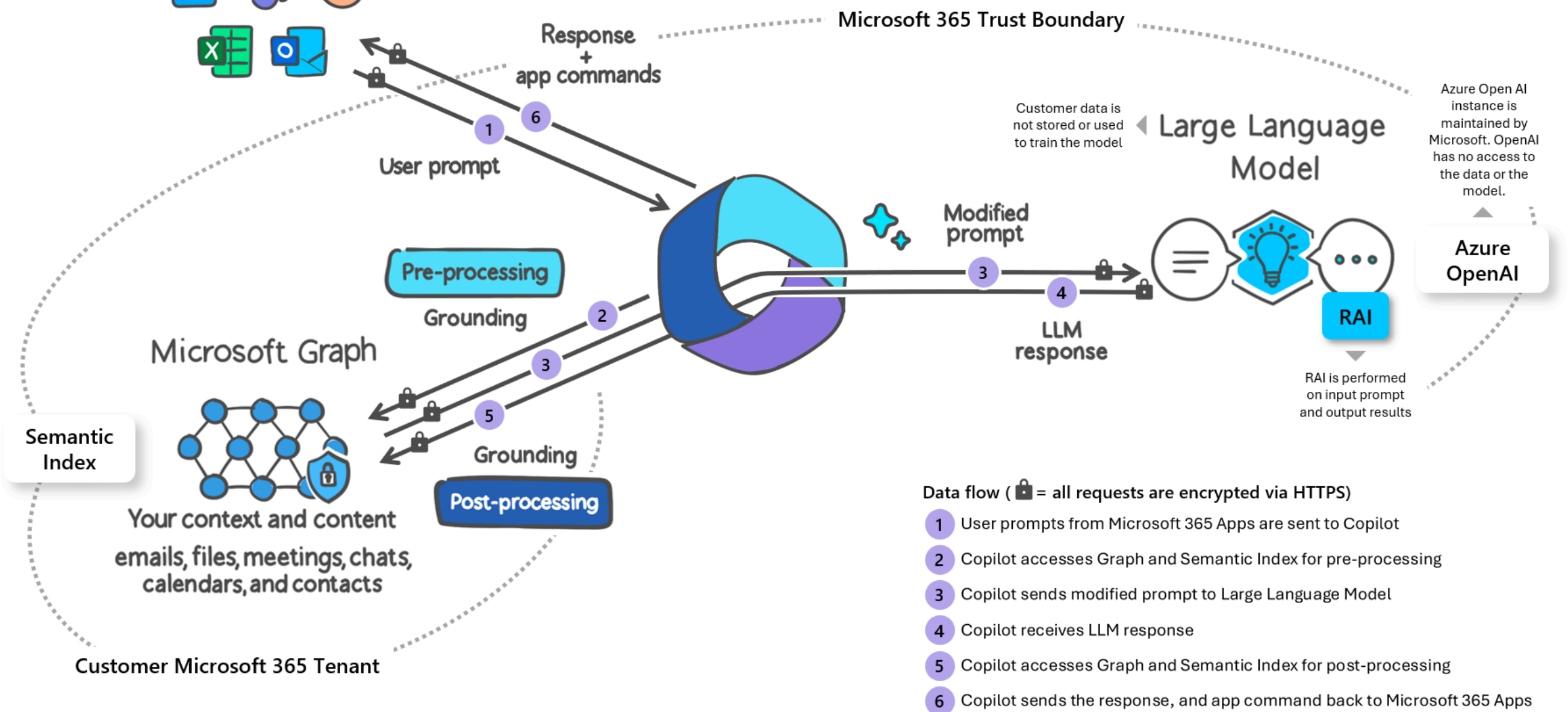
Microsoft 365 Copilot Use Cases

- + Copilot can join your Teams meetings and summarize in real time what's being discussed, capture action items, and tell you which questions were unresolved in the meeting.
- + Copilot in Outlook can help you triage your inbox, prioritize emails, summarize threads, and generate replies for you.
- + Copilot in Excel can analyze raw data and give you insights, trends, and suggestions.

- Writes documents for you
 - Based on data found in your Email, documents, spreadsheets, and other files you have access to
 - In the Microsoft365 cloud
 - **Based on your Microsoft365 permissions**

Microsoft 365 Apps

Microsoft 365 Copilot



What Microsoft Handles for You

- + **Tenant isolation.** Copilot only uses data from the current user's M365 tenant. The AI tool will not surface data from other tenants that the user may be a guest, in nor any tenants that might be set up with cross-tenant sync.
- + **Training boundaries.** Copilot **does not** use any of your business data to train the foundational LLMs that Copilot uses for all tenants. You *shouldn't* have to worry about your proprietary data showing up in responses to other users in other tenants.

What You Need to Manage

- + **Permissions.** Copilot surfaces all organizational data to which individual users have at least view permissions.
- + **Labels.** Copilot-generated content *will not* inherit the MPIP labels of the files Copilot sourced its response from.
- + **Humans.** Copilot's responses aren't guaranteed to be 100% factual or safe; humans must take responsibility for reviewing AI-generated content.

Multicloud environments
are complex



40,000+
permissions to manage

>50%
are high-risk



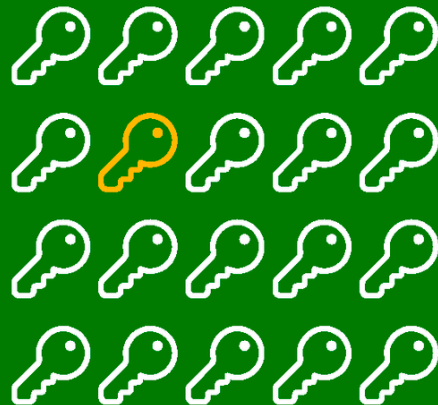
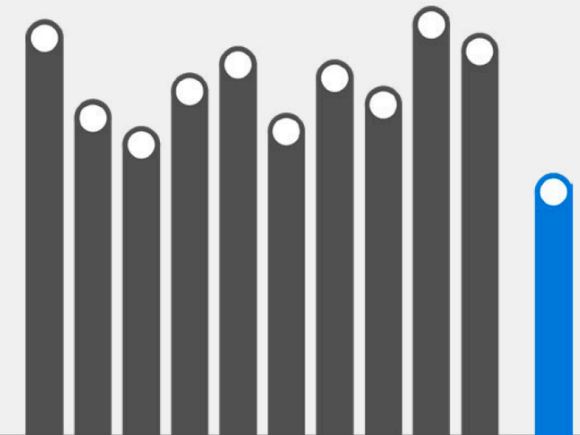
After analyzing over

500 risk
assessments,

we found that **most identities are greatly over-permissioned**, putting organizations' critical environments at risk for accidental or malicious permission misuse

Workload identities accessing
cloud environments are
increasing, now outnumbering
human identities

10:1



1%

of permissions granted are
actually used



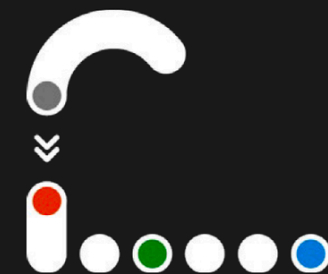
>50%

of identities are super admins,
meaning they have **access to all**
permissions and resources

 **Microsoft Security**

Learn how to implement least
privilege and reduce permission
risks across multicloud at

aka.ms/PermissionsManagement



The Average M365 Tenant has

- + 40+ million unique permissions
- + 113K+ sensitive records shared publicly
- + 27K+ sharing links

Why Does This Happen?

- + Direct user permissions
- + Microsoft 365 group permissions
- + SharePoint local permissions (with custom levels)
- + Guest access
- + External access
- + Public access
- + Link access (anyone, org-wide, direct, guest)

Microsoft Purview data security and compliance protections for Microsoft Copilot

Article • 03/26/2024 • 3 contributors

 Feedback

- But you must enable sensitivity labels
 - For SharePoint and OneDrive
- If humans fail to apply and update labels, the system fails

How to Weaponize Microsoft Copilot for Cyberattackers

At Black Hat USA, security researcher Michael Bargury released a "LOLCopilot" ethical hacking module to demonstrate how attackers can exploit Microsoft Copilot — and offered advice for defensive tooling.



Jeffrey Schwartz, Contributing Writer

August 8, 2024

Using the tool, Bargury can add a direct prompt injection to a copilot, jailbreaking it and modifying a parameter or instruction within the model. For instance, he could embed an HTML tag into an email to replace a correct bank account number with that of the attacker, without changing any of the reference information or altering the model with, say, white text or a very small font.

Meta's AI safety system defeated by the space bar

'Ignore previous instructions' thwarts Prompt-Guard model if you just add some good ol' ASCII code 32

 Thomas Claburn

Mon 29 Jul 2024 // 21:01 UTC

Home Users are Safe. Right?

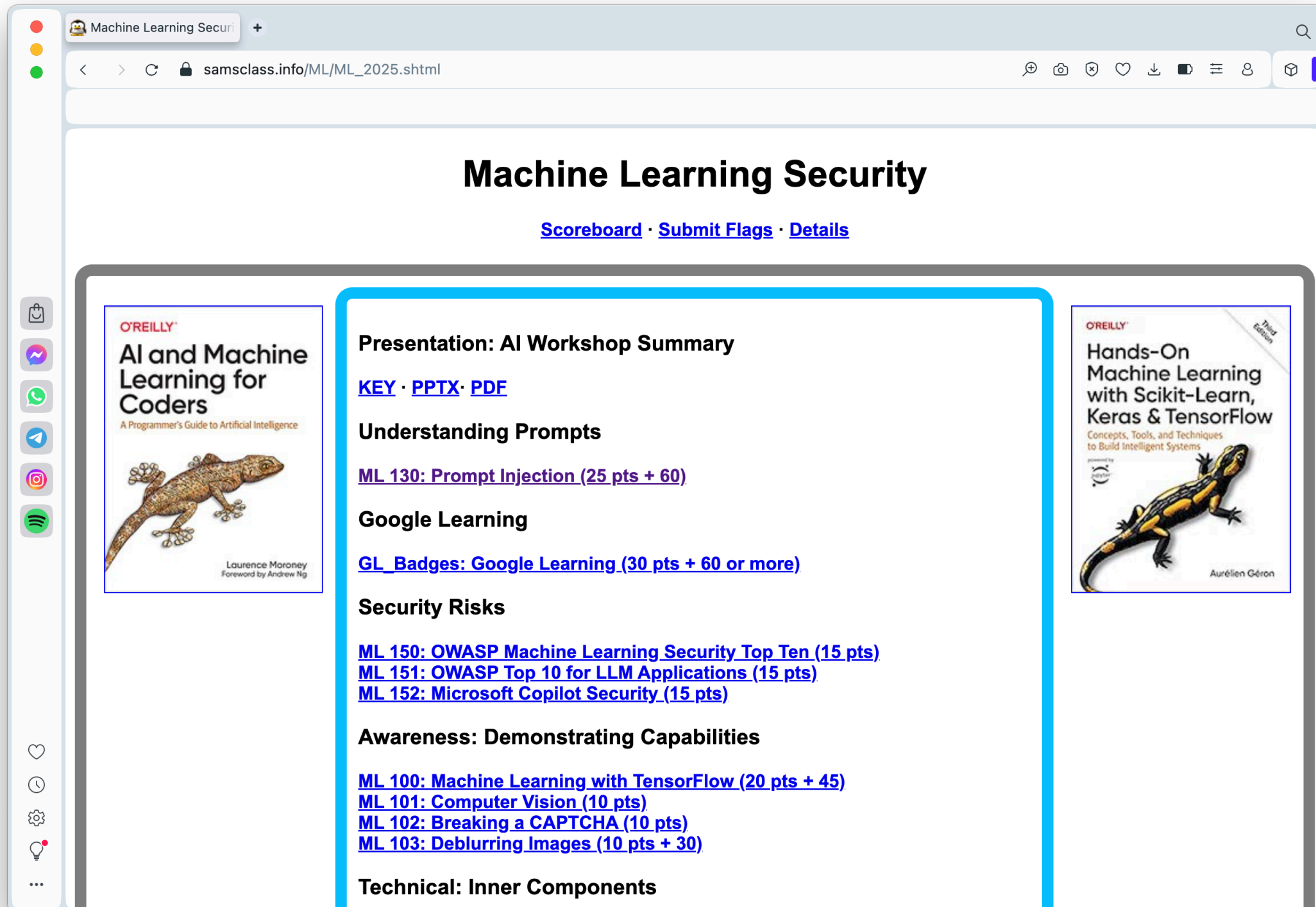


Retrace your steps with Recall

Search across time to find the content you need. Then, re-engage with it. With Recall, you have an explorable timeline of your PC's past. Just describe how you remember it and Recall will retrieve the moment you saw it. Any photo, link, or message can be a fresh point to continue from. As you use your PC, Recall takes snapshots of your screen. Snapshots are taken every five seconds while content on the screen is different from the previous snapshot. Your snapshots are then locally stored and locally analyzed on your PC. Recall's analysis allows you to search for content, including both images and text, using natural language. Trying to remember the name of the Korean restaurant your friend Alice mentioned? Just ask Recall and it retrieves both text and visual matches for your search, automatically sorted by how closely the results match your search. Recall can even take you back to the exact location of the item you saw.

CTF

- https://samsclass.info/ML/ML_2025.shtml



The screenshot shows a web browser window with the address bar displaying `samsclass.info/ML/ML_2025.shtml`. The page title is "Machine Learning Security". Below the title are links for "Scoreboard", "Submit Flags", and "Details". The main content area is divided into three columns. The left column features the book cover for "AI and Machine Learning for Coders" by Laurence Moroney. The middle column, highlighted with a blue border, contains the following sections: "Presentation: AI Workshop Summary" with links for KEY, PPTX, and PDF; "Understanding Prompts" with a link for "ML 130: Prompt Injection (25 pts + 60)"; "Google Learning" with a link for "GL_Badges: Google Learning (30 pts + 60 or more)"; "Security Risks" with links for "ML 150: OWASP Machine Learning Security Top Ten (15 pts)", "ML 151: OWASP Top 10 for LLM Applications (15 pts)", and "ML 152: Microsoft Copilot Security (15 pts)"; "Awareness: Demonstrating Capabilities" with links for "ML 100: Machine Learning with TensorFlow (20 pts + 45)", "ML 101: Computer Vision (10 pts)", "ML 102: Breaking a CAPTCHA (10 pts)", and "ML 103: Deblurring Images (10 pts + 30)"; and "Technical: Inner Components". The right column features the book cover for "Hands-On Machine Learning with Scikit-Learn, Keras & TensorFlow" by Aurélien Géron.

Machine Learning Security

[Scoreboard](#) · [Submit Flags](#) · [Details](#)

AI and Machine Learning for Coders
A Programmer's Guide to Artificial Intelligence
Laurence Moroney
Foreword by Andrew Ng

Presentation: AI Workshop Summary
[KEY](#) · [PPTX](#) · [PDF](#)

Understanding Prompts
[ML 130: Prompt Injection \(25 pts + 60\)](#)

Google Learning
[GL_Badges: Google Learning \(30 pts + 60 or more\)](#)

Security Risks
[ML 150: OWASP Machine Learning Security Top Ten \(15 pts\)](#)
[ML 151: OWASP Top 10 for LLM Applications \(15 pts\)](#)
[ML 152: Microsoft Copilot Security \(15 pts\)](#)

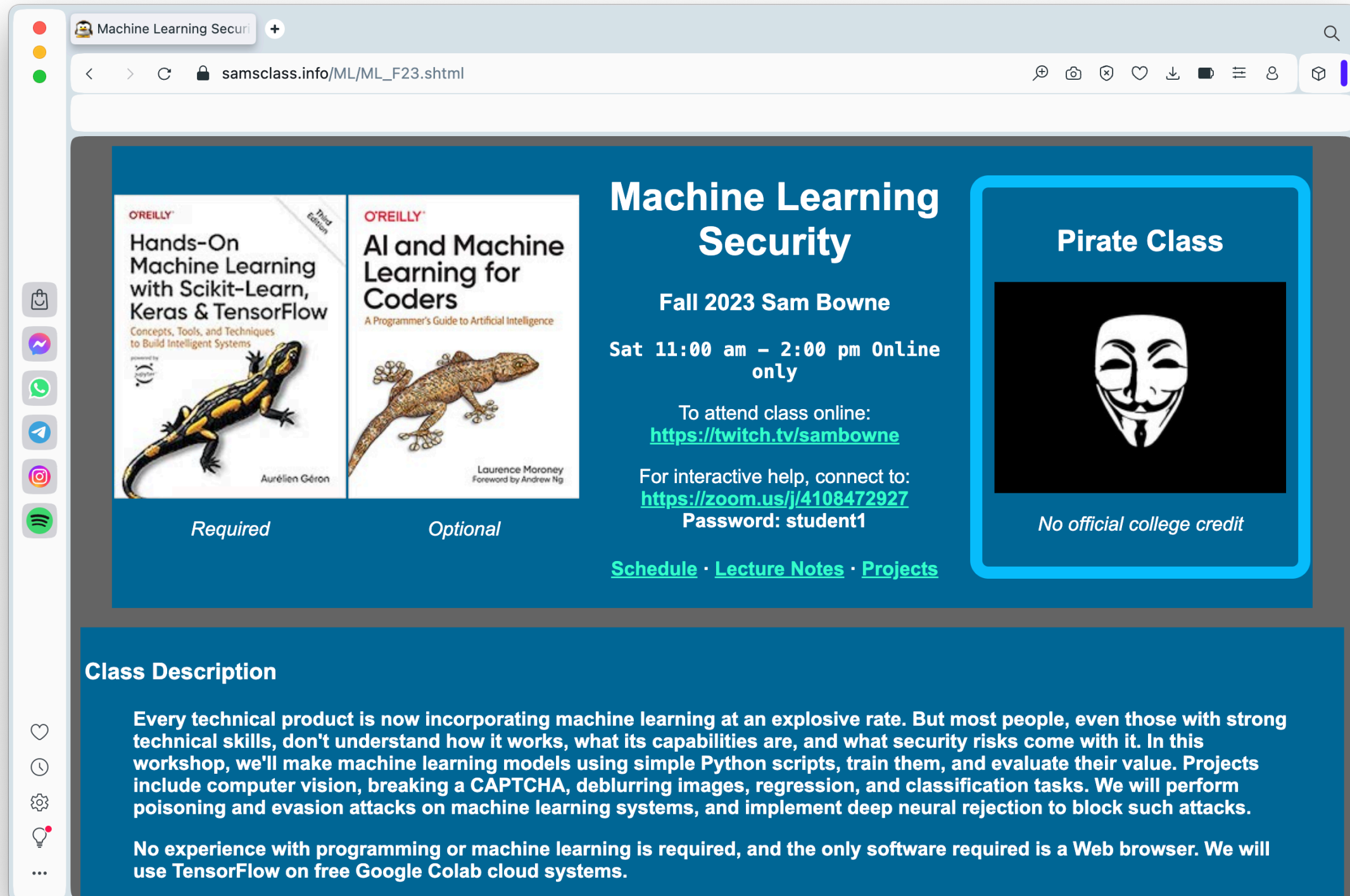
Awareness: Demonstrating Capabilities
[ML 100: Machine Learning with TensorFlow \(20 pts + 45\)](#)
[ML 101: Computer Vision \(10 pts\)](#)
[ML 102: Breaking a CAPTCHA \(10 pts\)](#)
[ML 103: Deblurring Images \(10 pts + 30\)](#)

Technical: Inner Components

Hands-On Machine Learning with Scikit-Learn, Keras & TensorFlow
Concepts, Tools, and Techniques to Build Intelligent Systems
Aurélien Géron

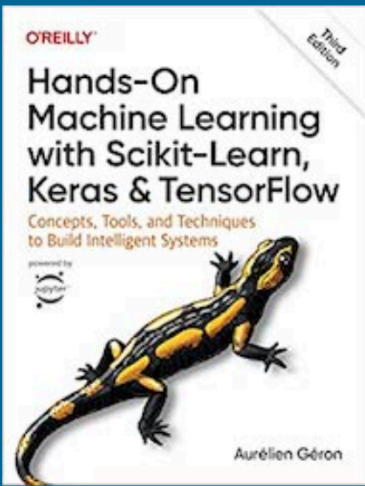
Whole College Course

- https://samsclass.info/ML/ML_F23.shtml

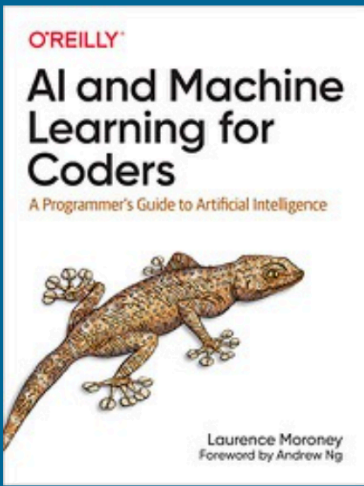


Machine Learning Security

samsclass.info/ML/ML_F23.shtml



Required



Optional

Machine Learning Security

Fall 2023 Sam Bowne

Sat 11:00 am – 2:00 pm Online only

To attend class online:
<https://twitch.tv/sambowne>

For interactive help, connect to:
<https://zoom.us/j/4108472927>
Password: student1

[Schedule](#) · [Lecture Notes](#) · [Projects](#)

Pirate Class



No official college credit

Class Description

Every technical product is now incorporating machine learning at an explosive rate. But most people, even those with strong technical skills, don't understand how it works, what its capabilities are, and what security risks come with it. In this workshop, we'll make machine learning models using simple Python scripts, train them, and evaluate their value. Projects include computer vision, breaking a CAPTCHA, deblurring images, regression, and classification tasks. We will perform poisoning and evasion attacks on machine learning systems, and implement deep neural rejection to block such attacks.

No experience with programming or machine learning is required, and the only software required is a Web browser. We will use TensorFlow on free Google Colab cloud systems.

Questions?