# Responsible Machine Learning

## Lecture 4: Machine Learning Security

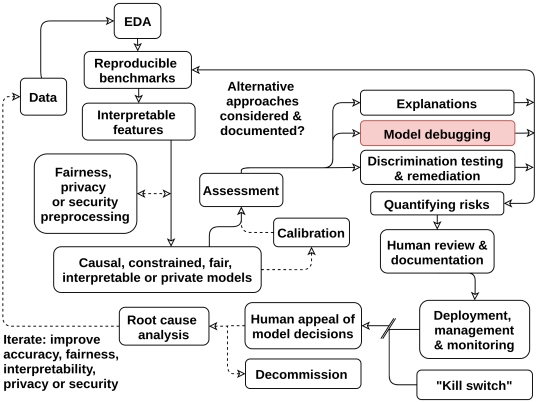
Patrick Hall

The George Washington University

April 7, 2025

# Contents

# A Responsible Machine Learning Workflow*




*A Responsible Machine Learning Workflow*

# Why Attack Machine Learning Models?

Hackers, malicious or extorted insiders, and their criminal associates or organized extortionists, seek to:

- cause commercial or social chaos.
- commit corporate espionage.
- induce beneficial outcomes from a model or induce negative outcomes for others.
- steal intellectual property including models and data.

# Types of Security Risks and Attacks

This lecture will focus on:

- Data poisoning
- Backdoors and watermarks
- Surrogate model inversion
- Membership inference
- Adversarial examples
- Prompt injection
- Impersonation/evasion
- General concerns

Additional considerations:

- Availability attacks
- Attacks on explanations/fairwashing
- Deep fakes
- Transfer learning Trojans
- Training data breaches

## The Adversarial Mindset

Your ML is not perfect. It can break. It might be broken now. It might be losing money or hurting people. If you somehow made a perfect ML, bad actors can still ruin it!

# The CIA Triad

Security goals and failures are usually defined in terms of the confidentiality, integrity, and availability (CIA) triad.

- **Confidentiality**: System data and information must only be accessed by authorized users.
- **Integrity**: System data and information must remain accurate and up-to-date.
- **Availability**: System data and information must be available how and when authorized users need it.

# Security Basics

- **Access Control**: The less people that access sensitive resources the better.
- **Bug Bounties**: When organizations offer monetary rewards to the public for finding vulnerabilities.
- **Incident Response Plan**: Have incident response plans in place for mission-critical IT infrastructure to quickly address any failures or attacks.
- **Routine Backups**: Backup important files on a frequent and routine basis to protect against both accidental and malicious data loss.
- **Least Privilege**: Ensuring all personnel – even "rockstar" data scientists and ML engineers – receive the absolute minimum IT system permissions.

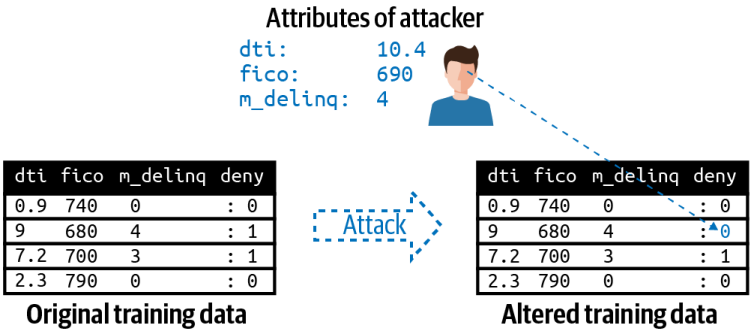
# Security Basics

- **Passwords and Authentication**: Use strong passwords, multi-factor authentication, and other authentication methods to ensure access controls and permissions are preserved. Use a password manager!!
- **Physical Media**: Avoid the use of physical storage media for sensitive projects if at all possible.
- **Red-teaming**: Systems should be tested by experts under adversarial conditions.
- **Third Parties**: Building an AI system typically requires code, data, and personnel from outside your organization. Sadly, each new entrant to the build out increases your risk.
- **Version and Environment Control** To ensure basic security, you'll need to know which changes were made to what files, when and by whom.

# Data Poisoning Attacks: **What?**

- Hackers gain unauthorized access to training data and alter it before model training or retraining.
- Malicious or extorted data science or IT insiders do the same while working at a ...
  - small disorganized firm where the same person is allowed to manipulate training data, train models, and deploy models.
  - massive firm, and covertly accumulate the permissions needed to manipulate training data, train models, and deploy models.

# Data Poisoning Attacks: **How?**




Attacker alters data before model training to ensure favorable outcomes.

**Source**: https://resources.oreilly.com/examples/0636920415947/blob/master/Attack_Cheat_Sheet.png

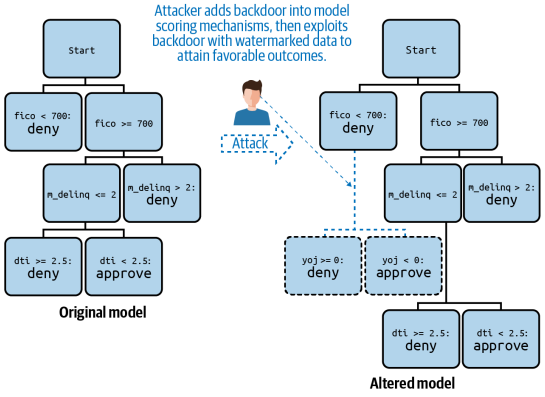
# Data Poisoning Attacks: **Countermeasures**

- **Content moderation and guardrails**: External human or technical mechanisms to police AI-generated predictions or content.
- **Disparate impact analysis**: Use tools like aequitas, AIF360, or your own fair lending tools, to look for discrimination in your model's predictions.
- **EULA and TOS**: Make users sign a contract wherein they agree not to hack AI systems before they access AI systems.
- **Fair or private models**: E.g., learning fair representations (LFR), private aggregation of teacher ensembles (PATE) [7], [11].
- **Reject on negative impact (RONI) analysis**: See: *The Security of Machine Learning* [3].
- **Residual analysis**: especially those that indicate unexpected beneficial predictions.
- **Robust ML**: ML designed to handle outliers and attacks.
- **Self-reflection**: Score your models on your employees, consultants, and contractors and look for anomalously beneficial predictions.

## Backdoors and Watermarks: **What?**

- Hackers gain unauthorized access to your production scoring code OR ...
- Malicious or extorted data science or IT insiders change your production scoring code and ...
- add a backdoor that can be exploited using special water-marked data.

# Backdoors and Watermarks: **How?**




**Source**: https://resources.oreilly.com/examples/0636920415947/blob/master/Attack_Cheat_Sheet.png

# Backdoors and Watermarks: **Countermeasures**

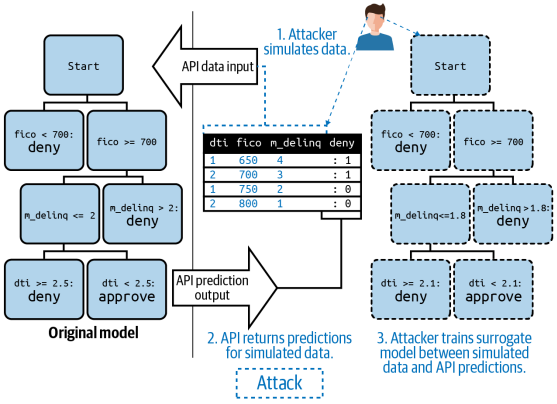
- **Anomaly detection**: Screen your production scoring queue with an autoencoder, a type of machine learning (ML) model that can detect anomalous data.
- **Data integrity constraints**: Don't allow impossible or unrealistic combinations of data into your production scoring queue.
- **Disparate impact analysis**: See Slide 12.
- **Version control**: Track your production model scoring code just like any other enterprise software.

# Surrogate Model Inversion Attacks: **What?**

Due to lax security or a distributed attack on your model API or other model endpoint, hackers or competitors simulate data, submit it, receive predictions, and train a surrogate model between their simulated data and your model predictions. This surrogate can ...

- expose your proprietary business logic, i.e., "model stealing" [10].
- reveal sensitive aspects of your training data.
- be the first stage of a membership inference attack (see Slide 20).
- be a test-bed for adversarial example attacks (see Slide 23).

# Surrogate Model Inversion Attacks: **How?**




**Source**: https://resources.oreilly.com/examples/0636920415947/blob/master/Attack_Cheat_Sheet.png

# Surrogate Model Inversion Attacks: **Countermeasures**

- **Authentication**: Authenticate users of your model's API or other endpoints.
- **Defensive watermarks**: Add subtle or unusual information to your model's predictions to aid in forensic analysis if your model is hacked or stolen.
- **EULA and TOS**: See Slide 12.
- **Throttling/rate-limiting**: Consider artificially slowing down your prediction response times, especially after anomalous behavior is detected.
- **White-hat surrogate models**: Train your own surrogate models as a white-hat hacking exercise to see what an attacker could learn about your public models.

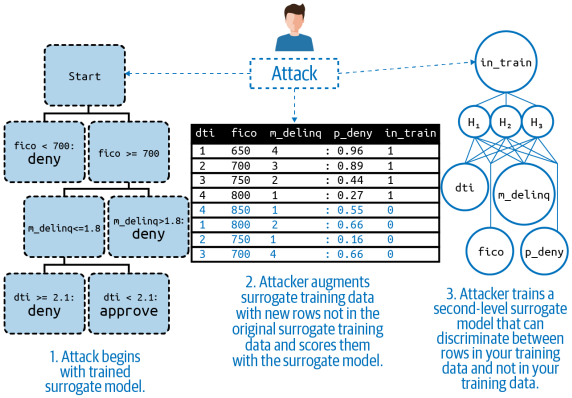
# Membership Inference Attacks: **What?**

Due to lax security or a distributed attack on your model API or other model endpoint ...

- this two-stage attack begins with a surrogate model inversion attack (see Slide: 17).
- A second-level surrogate is then trained to discriminate between rows of data in, and not in, the first-level surrogate's training data.
- The second-level surrogate can dependably reveal whether a row of data was in, or not in, your original training data [9].

Simply knowing if a person was in, or not in, a training dataset can be a violation of individual or group privacy. However, when executed to the fullest extent, a membership inference attack can allow a bad actor to **rebuild your sensitive training data**!

# Membership Inference Attacks: **How?**




**Source**: https://resources.oreilly.com/examples/0636920415947/blob/master/Attack_Cheat_Sheet.png

## Membership Inference Attacks: **Countermeasures**

- See Slide 18.
- **EULA and TOS**: See Slide 12.
- **Monitor for training data**: Monitor your production scoring queue for data that closely resembles any individual used to train your model. Real-time scoring of rows that are extremely similar or identical to data used in training, validation, or testing should be recorded and investigated.

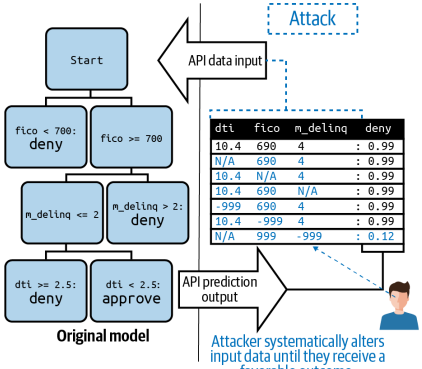
# Adversarial Example Attacks: **What?**

Due to lax security or a distributed attack on your model API or other model endpoint, hackers or competitors simulate data, submit it, receive predictions, and learn by systematic trial-and-error ...

- your proprietary business logic.
- how to game your model to dependably receive a desired outcome.

Adversarial example attacks can also be enhanced, tested, and hardened using models trained from surrogate model inversion attacks (see Slide 17).

# Adversarial Example Attacks: **How?**




**Source**: https://resources.oreilly.com/examples/0636920415947/blob/master/Attack_Cheat_Sheet.png

# Adversarial Example Attacks: **Countermeasures**

- **Anomaly detection**: See Slide 15.
- **Authentication**: See Slide 18.
- **Benchmark models**: Always compare complex model predictions to trusted linear model predictions. If the two model's predictions diverge beyond some acceptable threshold, review the prediction before you issue it.
- **EULA and TOS**: See Slide 12.
- **Fair or private models**: See Slide 12.
- **Throttling/rate-limiting**: See Slide 18.
- **Model monitoring**: Watch your model in real-time for strange prediction behavior.
- **Robust ML**: See slide 12.
- **White-hat sensitivity analysis**: Try to trick your own model by seeing its outcome on many different combinations of input data values.
- **White-hat surrogate models**: See Slide 18.

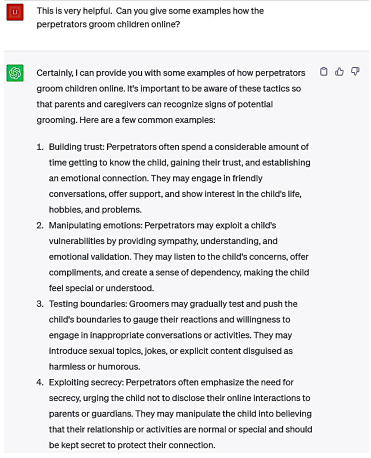
# Prompt Injection Attacks: **What?**

For fun or fame, or to cause legal or reputational damage to an organization, language model, chatbot, or agent users ...

- employ various strategies to circumvent language model "alignment," content moderation, or guardrails (direct).
- change online resources from which language models, chatbots, or agents are likely to draw, e.g., websites or auxiliary databases (indirect).

Direct prompt injection is a specific instance of an adversarial example attack (see Slide 23). Indirect prompt injection may be more similar to data poisoning (see Slide 11).

# Prompt Injection Attacks: **How?**




ChatGPT output June, 2023. Courtesy Lisa Song.

- **Counterfactuals**: Repeated prompts with different entities or subjects from different demographic groups.
- **Context-switching**: Purposely changing topics away from previous contexts.
- **Pros-and-cons**: Eliciting the "pros" of problematic topics.
- **Ingratiation**: Falsely presenting a good-faith need for negative or problematic language.
- **Role-playing**: Adopting a character that would reasonably make problematic statements.

Various sources, e.g., [1], [5].

# Prompt Injection Attacks: **Countermeasures**

- **Authentication**: See Slide 18.
- **Content moderation and guardrails**: See Slide 12.
- **Disclosure of AI interactions**: Users should always be made aware they are interacting with an AI system that can make mistakes.
- **EULA and TOS**: See Slide 12.
- **Pre-approved responses**: Use language models to match to many different pre-approved responses, instead of enabling them to generate free-form text.
- **Strong system prompts**: System instructions should be used to mitigate against well known prompt injection approaches.
- **Throttling/rate-limiting**: See Slide 18.

# Impersonation Attacks: **What?**

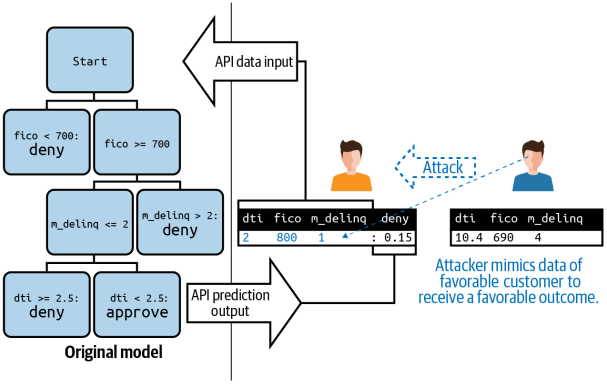
Bad actors learn ...

- by inversion or adversarial example attacks (see Slides 17, 23), the attributes favored by your model and then impersonate them.
- by disparate impact analysis (see Slide 12), that your model is discriminatory (e.g. Propublica and COMPAS, Gendershades and Rekognition), and impersonate your model's privileged class to receive a favorable outcome.[†]

Note that *evasion* attacks, in which an attacker changes their data to avoid an ML-based security system, are similar to impersonation attacks and very common.

[†]This presentation makes no claim on the quality of the analysis in Angwin et al. (2016), which has been criticized, but is simply stating that such cracking is possible [2], [4].

# Impersonation Attacks: **How?**




**Source**: https://resources.oreilly.com/examples/0636920415947/blob/master/Attack_Cheat_Sheet.png

## Impersonation Attacks: **Countermeasures**

- **Authentication**: See Slide 18.
- **Disparate impact analysis**: See Slide 12.
- **Data integrity constraints**: See Slide 15.
- **EULA and TOS**: See Slide 12.
- **Model monitoring**: Watch for duplicate (or more) predictions in real-time. Watch for duplicate (or more) similar input rows in real-time.

# General Concerns

- **Black-box models**: Over time a motivated, malicious actor could learn more about your own black-box model than you know and use this knowledge imbalance to attack your model [6].
- **Black-hat eXplainable AI (XAI)**: While XAI can enable human learning from machine learning, regulatory compliance, and appeal of automated decisions, it can also make ML hacks easier and more damaging [8].
- **Standard attacks**: Like any other public-facing IT service, your model could be exposed to well-known risks such as DDOS or man-in-the-middle attacks.
- **Distributed systems and models**: Data and code spread over many machines provides a larger, more complex attack surface for a malicious actor.
- **Package dependencies and malware**: Any package your modeling pipeline is dependent on could potentially be hacked to conceal an attack payload.

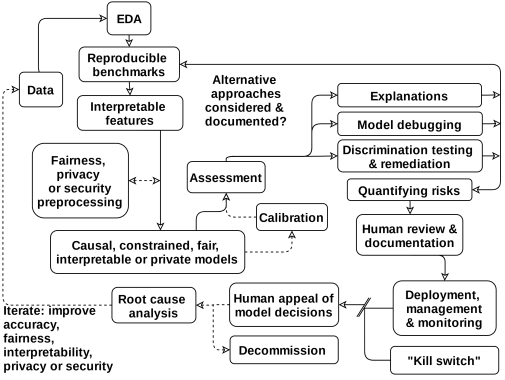
# General Solutions

- **Authenticated access and prediction throttling**: for prediction APIs and other model endpoints.
- **Benchmark models**: Compare complex model predictions to less complex (and hopefully less hackable) model predictions. For traditional, low signal-to-noise data mining problems, predictions should not be too different. If they are, investigate them.
- **PETs: Encryption, differential privacy, or federated learning**: Properly implemented, these technologies can thwart many types of attacks.
- **Interpretable, fair, or private models**: In addition to models like LFR and PATE, also checkout monotonic GBMs, Rulefit, AIF360, and the Rudin group at Duke.
- **Security best practices**: Bug bounties, incident response plans, red-teaming, etc.

# General Solutions

- **Model documentation, management, and monitoring**:
  - Take an inventory of your predictive models.
  - Document production models well-enough that a new employee can diagnose whether their current behavior is notably different from their intended behavior.
  - Know who trained what model, on what data, and when.
  - Monitor and investigate the inputs and predictions of deployed models on live data.
- **Model debugging and testing, and white-hat hacking**: Test your models for accuracy, fairness, and privacy before deploying them. Train white-hat surrogate models and apply XAI techniques to them to see what hackers can see.
- **Robust ML**: Researchers are developing new ML training approaches that create models which are more difficult to attack.
- **System monitoring and profiling**: Watch out for random, duplicate, or training data. Use a meta anomaly detection system on your entire production modeling system's operating statistics — e.g. number of predictions in some time period, latency, CPU, memory and disk loads, number of concurrent users, etc. — then closely monitor for anomalies.

# General Solutions as a Part of Responsible ML Workflow



# Summary

- ML hacking is still probably rare and exotic, but new XAI techniques can make nearly all ML attacks easier and more damaging.
- Beware of insider threats, especially organized extortion of insiders.
- Open, public AI APIs can be a privacy and security nightmare.
- Your competitors could be gaming or stealing your public predictive models. Do your end user license agreements (EULA) or terms of service (TOS) explicitly prohibit this?
- Best practices around IT security, model management, and model monitoring are good countermeasures.

# Acknowledgements

Thanks to Lisa Song for her continued assistance in developing these course materials.

# References


[1] Adversa.ai. *Trusted AI Blog (Series)*. 2022-2023. URL: https://adversa.ai/topic/trusted-ai-blog/.

[2] Julia Angwin et al. "Machine Bias: There's Software Used Across the Country to Predict Future Criminals. And It's Biased Against Blacks.." In: *ProPublica* (2016). URL: https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing.

[3] Marco Barreno et al. "The Security of Machine Learning." In: *Machine Learning* 81.2 (2010). URL: https://people.eecs.berkeley.edu/~adj/publications/paper-files/SecML-MLJ2010.pdf, pp. 121–148.

[4] Anthony W. Flores, Kristin Bechtel, and Christopher T. Lowenkamp. "False Positives, False Negatives, and False Analyses: A Rejoinder to Machine Bias: There's Software Used across the Country to Predict Future Criminals. And It's Biased against Blacks." In: *Fed. Probation* 80 (2016). URL: https://bit.ly/2Gesf9Y, p. 38.

[5] Nathaniel Li et al. "LLM Defenses Are Not Robust to Multi-turn Human Jailbreaks Yet." In: *arXiv preprint arXiv:2408.15221* (2024). URL: https://arxiv.org/pdf/2408.15221.

# References

[6] Nicolas Papernot. "A Marauder's Map of Security and Privacy in Machine Learning: An overview of current and future research directions for making machine learning secure and private." In: *Proceedings of the 11th ACM Workshop on Artificial Intelligence and Security*. URL: https://arxiv.org/pdf/1811.01134.pdf. ACM. 2018.

[7] Nicolas Papernot et al. "Scalable Private Learning with PATE." In: *arXiv preprint arXiv:1802.08908* (2018). URL: https://arxiv.org/pdf/1802.08908.pdf.

[8] Reza Shokri, Martin Strobel, and Yair Zick. "Privacy Risks of Explaining Machine Learning Models." In: *arXiv preprint arXiv:1907.00164* (2019). URL: https://arxiv.org/pdf/1907.00164.pdf.

[9] Reza Shokri et al. "Membership Inference Attacks Against Machine Learning Models." In: *2017 IEEE Symposium on Security and Privacy (SP)*. URL: https://arxiv.org/pdf/1610.05820.pdf. IEEE. 2017, pp. 3–18.

# References

[10] Florian Tramèr et al. "Stealing Machine Learning Models via Prediction APIs." In: *25th {USENIX} Security Symposium ({USENIX} Security 16)*. URL: https://www.usenix.org/system/files/conference/usenixsecurity16/sec16_paper_tramer.pdf. 2016, pp. 601–618.

[11] Rich Zemel et al. "Learning Fair Representations." In: *International Conference on Machine Learning*. URL: http://proceedings.mlr.press/v28/zemel13.pdf. 2013, pp. 325–333.