Machine Learning Models as an Attack Surface

©Patrick Hall*

H_2O.ai

July 15, 2019

Contents

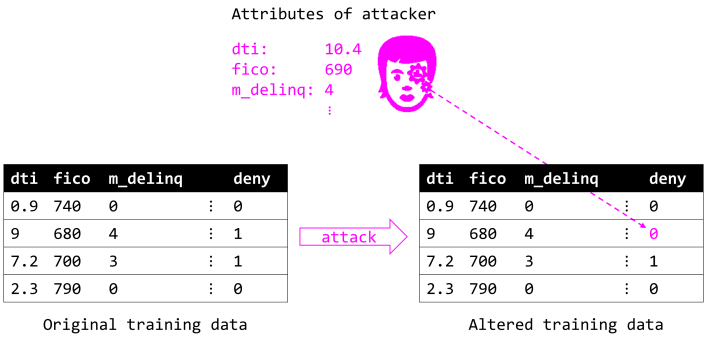
Why Attack Machine Learning Models?

- A hacker or a malicious or extorted insider (along with their criminal associates or organized extortionists) desires loans, insurance policies, jobs, favorable criminal risk assessments, or other beneficial outcomes associated with predictive or pattern recognition models; OR they desire negative outcomes for others.
- Corporate espionage.
- Intellectual property theft.

Data Poisoning Attacks: **How?**

- A hacker gains unauthorized access to training data and alters it before model training or retraining.
- A malicious or extorted data science or IT insider does the same, working at a ...
  - small disorganized firm where the same person is allowed to manipulate training data, train models, and deploy models.
  - massive firm, and slowly requesting or accumulating the kind of permissions needed to manipulate training data, train models, and deploy models.

Data Poisoning Attacks: **How?**



Original training data

Altered training data

Attacker alters model training data to ensure favorable outcomes

Data Poisoning Attacks: **Defenses**

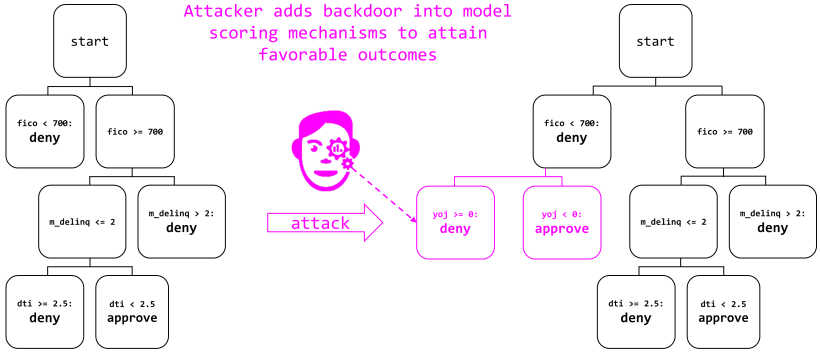
- **Disparate impact analysis**: Use tools like aequitas, AIF360, or your own fair lending tools, to look for unintentional (or intentional) discrimination in your model's predictions.
- **Fair or private models**: Like learning fair representations (LFR) or private aggregation of teacher ensembles (PATE) [5], [9].
- **Reject on negative impact (RONI) analysis**: See: *The Security of Machine Learning*.
- **Residual analysis**: especially for large positive deviance residuals.
- **Self-reflection**: Score your models on your employees, consultants, and contractors and look for anomalously beneficial predictions.

Backdoors and Watermarks: **How?**

- A hacker gains unauthorized access to your production scoring code OR ...
- A malicious or extorted data science or IT insider changes your production scoring code ...

... adding a backdoor that can be exploited using water-marked data.

Backdoors and Watermarks: **How?**



Backdoors and Watermarks: **Defenses**

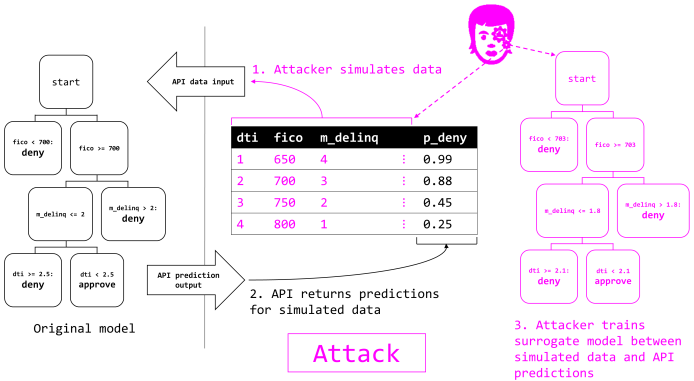
- **Anomaly detection**: Screen your production scoring queue with an autoencoder, a type of machine learning (ML) model that can detect anomalous data.
- **Data integrity constraints**: Don't allow impossible or unrealistic combinations of data into your production scoring queue.
- **Disparate impact analysis**: See Slide 6.
- **Version control**: Track your production model scoring code just like any other piece of enterprise software.

Surrogate Model Inversion Attacks: **How?**

Due to lax security or a distributed attack on your model API or other model endpoint, a hacker or competitor simulates data, submits it, receives predictions, and trains a surrogate model between their simulated data and your model predictions. This surrogate can ...

- expose your proprietary business logic, i.e. "model stealing" [8].
- reveal sensitive aspects of your training data.
- be the first stage of a membership inference attack (see Slide 14).
- be a test-bed for adversarial example attacks (see Slide 17).

Surrogate Model Inversion Attacks: **How?**



Surrogate Model Inversion Attacks: **Defenses**

- **Authentication**: Always 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.
- **Throttling**: Consider artificially slowing down your prediction response times, especially after anomalous behavior is detected.
- **White-hat surrogate models**: Try to build your own surrogate models as a white-hat hacking exercise to see what an attacker could learn.

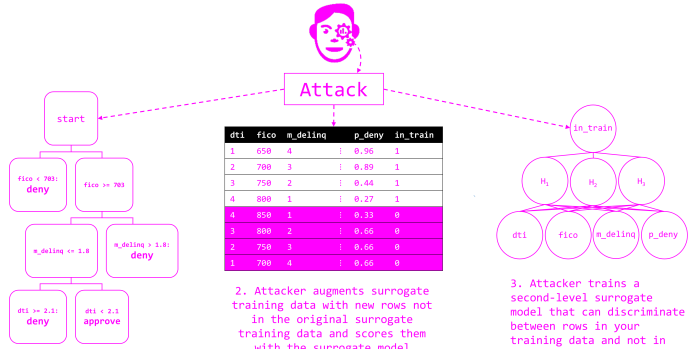
Membership Inference Attacks: **How?**

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

- this sophisticated, damaging, two-stage attack begins with a surrogate model inversion attack (see Slide: 11).
- A second-level surrogate is then trained to discriminate between rows of data in, and not in, the first-level surrogate's training data.
- When applied to your model API or other model endpoint, the second-level surrogate can dependably reveal whether a row of data was in, or not in, your training data [7].

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

Membership Inference Attacks: **How?**



1. Attack begins with trained surrogate model

2. Attacker augments surrogate training data with new rows not in the original surrogate training data and scores them with the surrogate model

3. Attacker trains a second-level surrogate model that can discriminate between rows in your training data and not in your training data

Membership Inference Attacks: **Defenses**

- See Slide 12.
- **Monitor for training data**: Monitor your production scoring queue for data that is within a certain range of any individual used to train the 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.

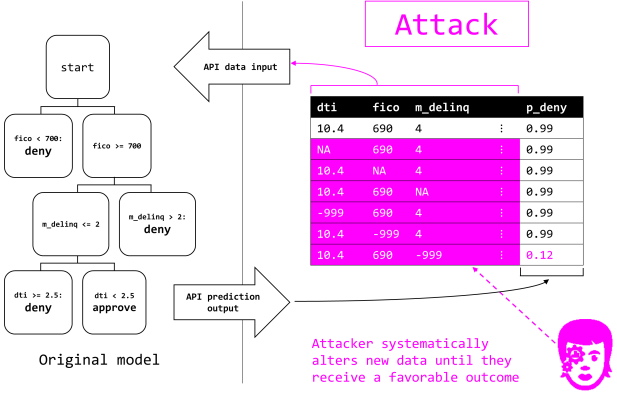
Adversarial Example Attacks: **How?**

Due to lax security or a distributed attack on your model API or other model endpoint, a hacker or competitor simulates data, submits it, receives predictions, and learns by systematic trial-and-error ...

- about 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 11).

Adversarial Example Attacks: **How?**



Adversarial Example Attacks: **Defenses**

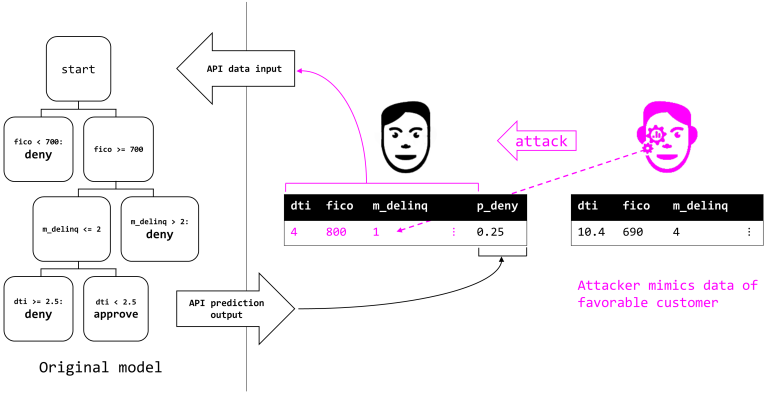
- **Anomaly detection**: See Slide 9.
- **Authentication**: See Slide 12.
- **Benchmark models**: Always compare complex model predictions to trusted linear model predictions. If the two model's predictions diverge beyond some acceptable threshold, review the prediction before you issue it.
- **Throttling**: See Slide 12.
- **Model monitoring**: Watch your model in real-time for strange prediction behavior.
- **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 12.

Impersonation Attacks: **How?**

- A hacker learns, by adversarial example attacks (see Slide 17), the attributes favored by your model and then impersonates them.
- A hacker learns, by disparate impact analysis (see Slide 6), that your model is discriminatory (e.g. Propublica and COMPAS, Gendershades and Rekognition), and impersonates your model's privileged class to receive a favorable outcome.[†]

[†]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 [1], [3].

Impersonation Attacks: **How?**



Impersonation Attacks: **Defenses**

- **Authentication**: See Slide 12.
- **Disparate impact analysis**: See Slide 6.
- **Model monitoring**: Watch for too many similar predictions in real-time. Watch for too many similar input rows in real-time.

General concerns

Some problems aren't associated with any one kind of attack, but could be potentially worrisome for many reasons. These might include:

- **Black-box models**: It's possible that over time a motivated, malicious actor could learn more about your own black-box model than you know and use this knowledge imbalance to carry out the attacks described in this presentation [4].
- **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 [6].
- **Distributed-denial-of-service (DDOS) attacks**: Like any other public-facing service, your model could be attacked with a DDOS attack that has nothing to do with machine learning.

General concerns

- **Distributed systems and models**: Data and code spread over many machines provides a larger, more complex attack surface for a malicious actor.
- **Package dependencies**: Any package your modeling pipeline is dependent on could potentially be hacked to conceal an attack payload.

General Solutions

There are many best practices that can defend your models in general (and that are probably beneficial for model life-cycle management). Some of these include:

- **Authenticated access and prediction throttling**: for prediction APIs and other model endpoints.
- **Benchmark models**: Always compare complex model predictions to less complex (and hopefully less hackable) model predictions. For traditional, low signal-to-noise data mining problems, predictions should probably not be too different. If they are, investigate them.
- **Encrypted, differentially private, or federated training data**: Properly implemented, encrypted, differentially private, or federated training data can thwart many types of attacks. (Improperly implemented, these technologies simply create a broader attack surface or hinder forensic efforts.)

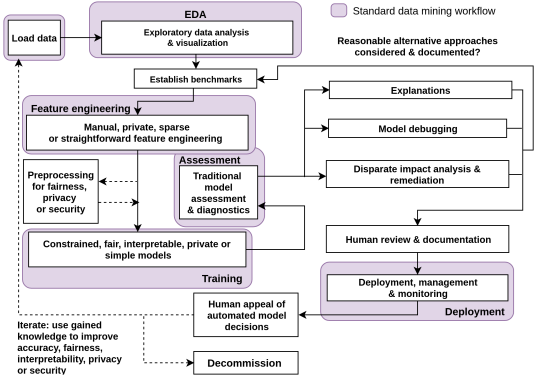
General Solutions

- **Interpretable, fair, or private models**: Some types of nonlinear models are sometimes designed to be directly interpretable, less discriminatory, or harder to hack. Consider using them. In addition to models like LFR and PATE, also checkout monotonic GBMs and Rulefit.
- **Model documentation, management, and monitoring**: Take an inventory of your predictive models. All production models should be documented well-enough that a new employee could diagnose whether its current behavior is notably different from its intended or original behavior. Also keep details about who trained what model and on what data. Additionally, analyze the inputs and predictions of deployed models on live data. If they seem strange, investigate the problem.

General Solutions

- **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 a hacker could see.
- **System monitoring and profiling**: Train an autoencoder–based anomaly detection metamodel on your entire predictive modeling system's operating statistics—the number of predictions in some time period, latency, CPU, memory, and disk loads, the number of concurrent users, and everything else you can get your hands on—and then closely monitor this metamodel for anomalies

A Blueprint for Low-Risk Machine Learning ‡



‡See: https://github.com/jphall663/hc_ml for more information.

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 for machine learning hacks, and especially organized crime extortion of insiders.
- Open, public prediction APIs are 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?
- Following best practices around IT security, model management, and model monitoring are good defenses.

References

This presentation:

https://github.com/jphall663/secure_ML_ideas

"Proposals for model vulnerability and security":

https://www.oreilly.com/ideas/proposals-for-model-vulnerability-and-security

"Can Your Machine Learning Model Be Hacked?!":

https://www.h2o.ai/blog/can-your-machine-learning-model-be-hacked/

References

[1] 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.

[2] 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.

[3] 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.

[4] 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.

References

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

[6] 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.

[7] 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.

[8] 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.

[9] 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.