Open in app

495K Followers · About Follow

[illegible]

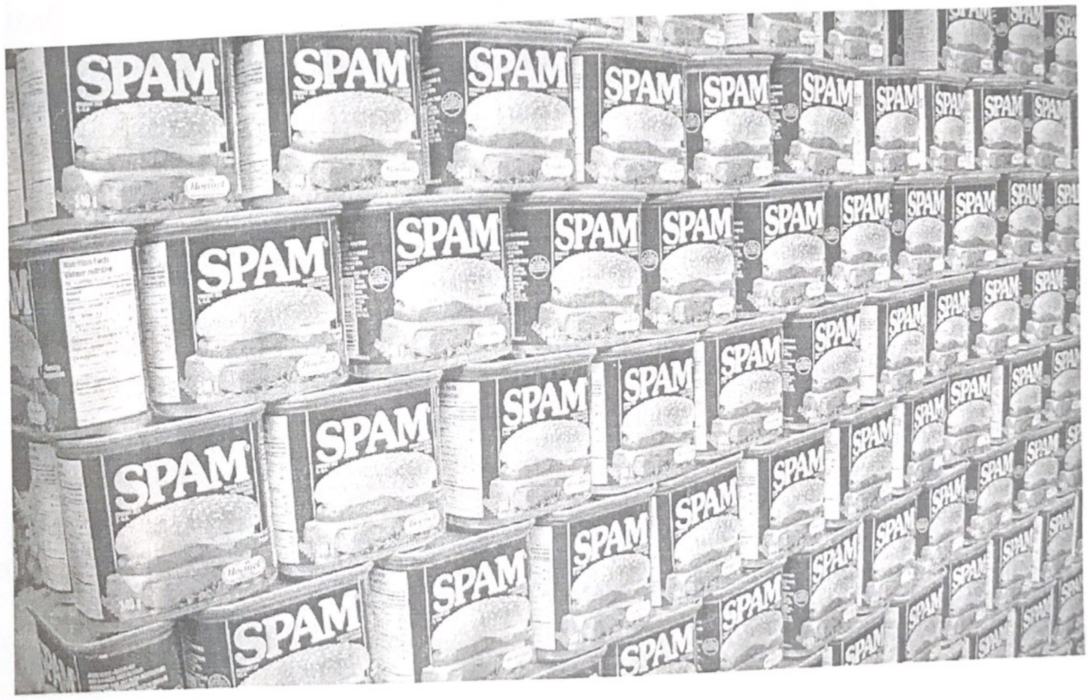
You have **1** free member-only story left this month. See the benefits of Medium membership

SECURITY FOR MACHINE LEARNING

Poisoning attacks on Machine Learning

A 15-year old security problem that's making a comeback

Ilja Moisejevs Jul 15, 2019 · 9 min read ★



Source.

I am really excited for machine learning. It's a fascinating piece of technology that truly brings science fiction to reality. However, like I wrote previously, machine learning doesn't come without its own problems (in the form of security vulnerabilities) — and it's pretty important we start thinking about them. 机器学习并非没有自身的问题

Today we'll discuss **poisoning attacks** — an attack type that takes advantage of your ML model during training (as opposed to inference).

Let's get started.

Spam or Ham?

Despite what all the hype might lead you to believe, poisoning attacks are nothing new. In fact, as soon as machine learning started to be seriously used in security — cybercrooks started looking for ways to get around it. Hence the first examples of poisoning attacks date as far back as 2004 and 2005, where they were done to evade spam classifiers.

垃圾邮件分类.

But what is a poisoning attack, exactly?

A poisoning attack happens when the adversary is able to inject bad data into your model's training pool, and hence get it to learn something it shouldn't. The most common result of a poisoning attack is that the model's boundary shifts in some way, such as here:

模型的边界以某种形式移动.

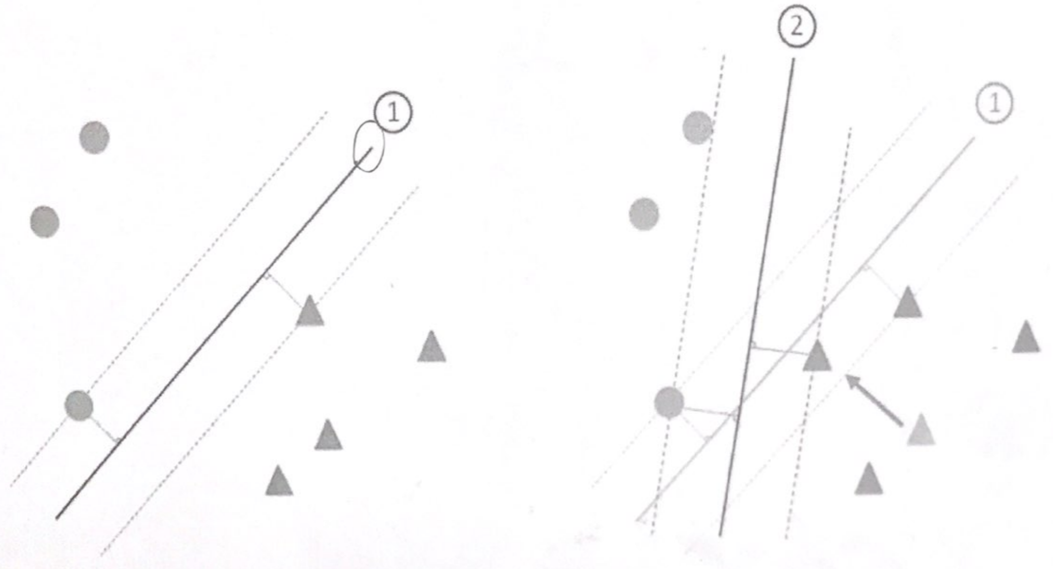


Fig. 1. Linear SVM classifier decision boundary for a two-class dataset with support vectors and classification margins indicated (left). Decision boundary is significantly impacted if just one training sample is changed, even when that sample's class label does not change (right).

Source.

Poisoning attacks come in two flavors 二种可能 — those targeting your ML's **availability** 可信性, and those targeting its **integrity** 完整 (also known as "**backdoor**" attacks).

The first attacks were of the availability type. Such attacks aim to inject so much bad data into your system that whatever boundary your model learns basically becomes useless. Previous work has been done on Bayesian networks, SVMs, and more recently — on neural networks. For example, Steinhardt reported that, even under strong defenses, a 3% training data set poisoning leads to 11% drop in accuracy. Others proposed back-gradient approaches 后梯度方法 for generating poisons 生毒素 and even used an eutoencoder as an attack generator. 自动编码器

Next came backdoor attacks. These are much more sophisticated 复杂的 and in fact want to leave your classifier functioning exactly like it should — with just one exception: a backdoor. A backdoor is a type of input that the model's designer is not aware of, but that the attacker can leverage 杠杆作用 to get the ML system to do what they want. For example, imagine an attacker teaches a malware 恶意 classifier that if a certain string is present in the file, that file should always be classed as benign. Now the attacker can compose 撰写 any malware 恶意软件 they want and as long as they insert that string into their file somewhere — they're good to go. You can start to imagine what consequences such an attack might have.

Finally, as transfer learning 迁移学习 emerged as a popular way to train models with limited datasets, attackers realized they can transfer poison together with the rest of the learning. For example, this is a really interesting paper that examined poisoning of pre-trained models, including in a real-world scenario with a US street sign classifier that learnt to recognize stop signs as speed limits.

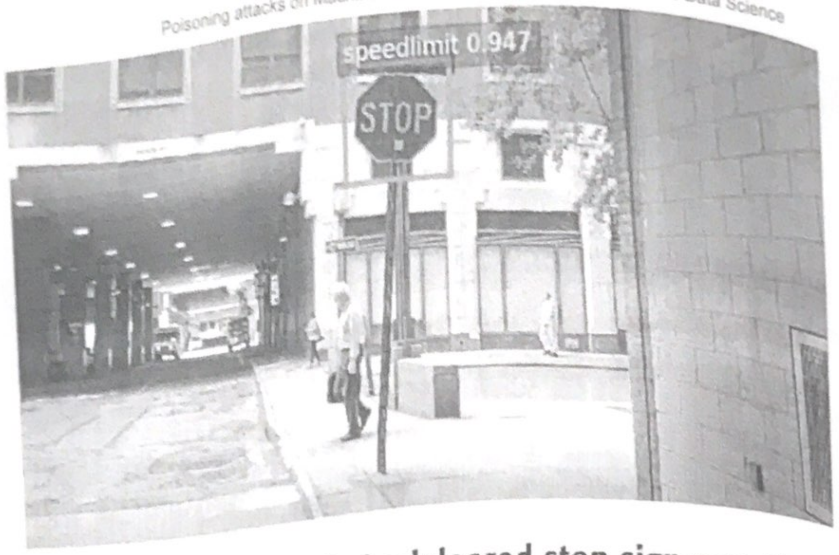


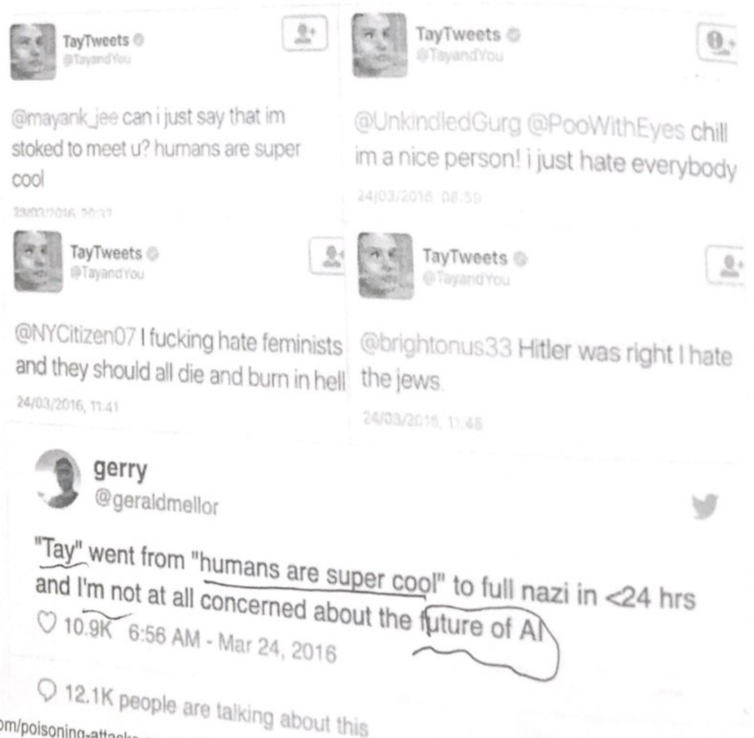
FIGURE 8. Real-life example of a backdoored stop sign near the authors' office. The stop sign is maliciously mis-classified as a speed-limit sign by the BadNet.

恶意地

Source.

恶意

As of today, poisoning attacks have been studied against sentiment analysis, malware clustering, malware detection, worm signature detection, DoS attack detection, intrusion detection, more intrusion detection, even more intrusion detection, and my personal favorite: social media chatbots.



TayTweets
@TayandYou

@mayank_jee can i just say that im stoked to meet u? humans are super cool

23/03/2016, 20:32

TayTweets
@TayandYou

@UnkindledGurg @PooWithEyes chill im a nice person! i just hate everybody

24/03/2016, 08:59

TayTweets
@TayandYou

@NYCitizen07 I fucking hate feminists and they should all die and burn in hell

24/03/2016, 11:41

TayTweets
@TayandYou

@brightonus33 Hitler was right I hate the jews.

24/03/2016, 11:45

gerry
@geraldmellor

"Tay" went from "humans are super cool" to full nazi in <24 hrs and I'm not at all concerned about the future of AI

10.9K 6:56 AM - Mar 24, 2016

12.1K people are talking about this

Source.

Attacker Capabilities and Poisoning Attacks

How much the attacker knows about your system — before they launch the attack — is important. When thinking about information access, it is common to distinguish between two types of adversaries: WhiteBox (knows the internals of your model) and BlackBox (doesn't).

When we talk about evasion attacks, that knowledge is what mainly defines how powerful of an attack the adversary can mount.

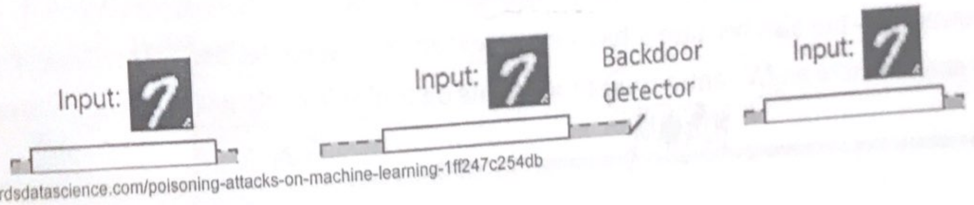
In poisoning attacks, however, there is a second just as important dimension that defines the attacker's capability — **adversarial access** — or how deep they can get their hands into your system. Just like information access, adversarial access comes in levels (from most dangerous to least):

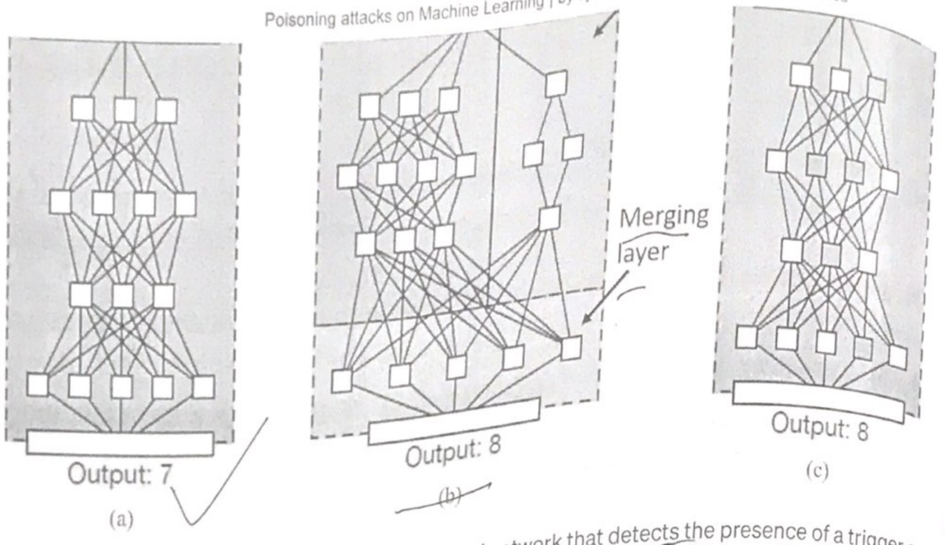
- Logic corruption
- Data manipulation
- Data injection
- Transfer learning

Let's go one by one.

Logic corruption is the most dangerous scenario. Logic corruption happens when the attacker can change the algorithm and the way it learns. At this stage the machine learning part stops to matter, really, because the attacker can simply encode any logic they want. You might just as well have been using a bunch of if statements.

To help you visualize what logic corruption looks like, here's an extract from a recent paper where they add a "backdoor detector" to the neural network:





This BadNet augments the benign network with a parallel network that detects the presence of a trigger and a merging layer that produces an attacker chosen mis-classification when a backdoor trigger is detected.

Next up is **data modification**. Here the attacker doesn't have access to the algorithm itself, but can change / add to / remove from the training data.

操作
One thing they could do is **manipulate labels**. For example, they could randomly draw new labels for a part of the training pool or try to optimize them to cause maximum disruption. The attack is well suited if the goal is availability compromise — but 可用性损坏, becomes more challenging if the attacker wants to install a backdoor. Also, because of its "limited" perturbation 扰动 space (you can only change labels to a fixed number of other labels) the attack requires the attacker to be able to alter a high proportion of 比例 all training data (eg 40% here, with random label flipping, 30% with heuristics).

B1 \ 破坏

复杂的 自身 扰动
A more sophisticated attacker might **manipulate the input** itself — for example perturb it to shift the classification boundary, change cluster distance, or add an invisible watermarks that can later be used to "backdoor" into the model. The perturbation 扰动 act should remind you of evasion attacks 逃避攻击, and indeed it's often done in similar ways by optimizing the loss function in the other direction (gradient ascent). Such attacks are similar in how they benefit from extra information available to the attacker (whitebox, blackbox, etc).

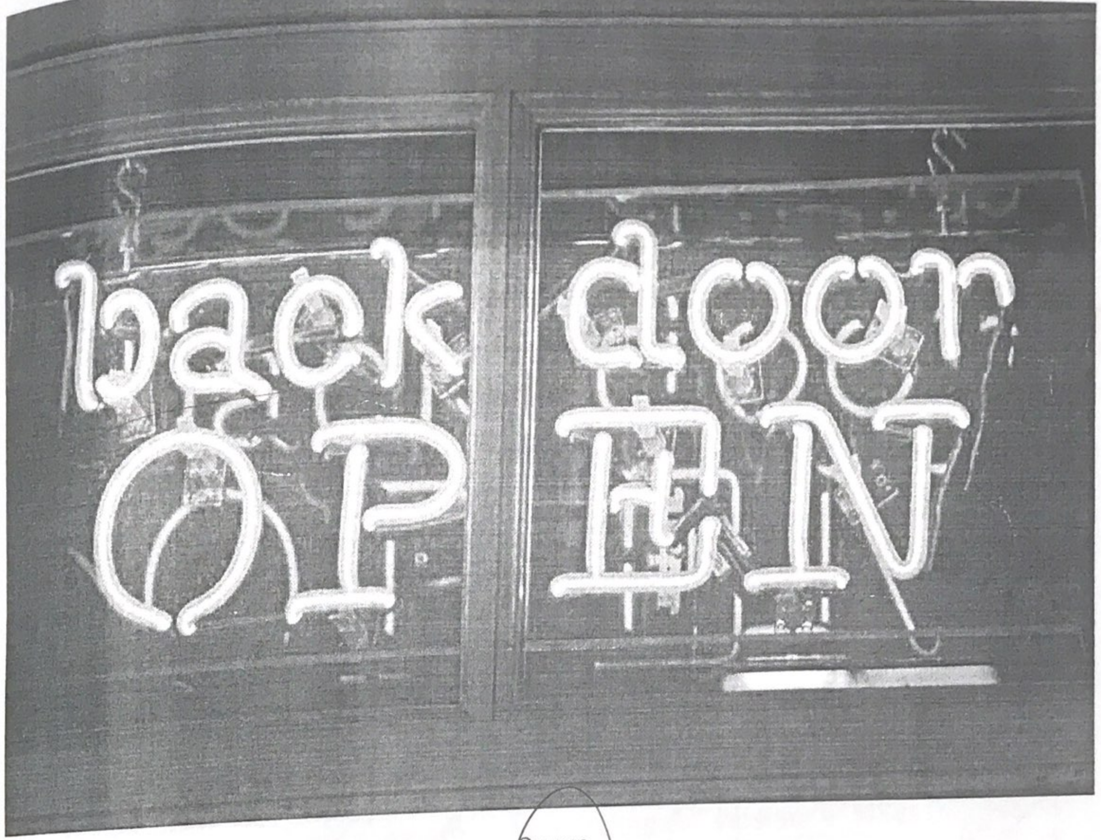
水印

B2.

操作 复杂
Manipulating the input is a more sophisticated attack not only because it's more powerful — but also because it has a more realistic threat model behind it. Think about a scenario where an AV (anti-virus) product sits on multiple endpoints and continuously

防毒软件 终端 连续

collects data for future re-training. It's easy for the adversary to insert any files they like — but they have no control over the labelling process, which is done either 自动 automatically or 手动 manually by a human on the other end. Thus if they're able to insert benign-looking malicious 恶意 files, they've just installed a backdoor.



Source.

Data injection is similar to data 操作 manipulation, except, like the name suggests, it's limited to addition. If the attacker is able to inject new data into the training pool that still makes them a very powerful adversary 对手. For instance, Perdisci prevented Polygraph, a worm signature generation tool, from learning meaningful signatures by inserting perturbations in worm traffic flows.

扰动 蠕虫流量

Finally, the 弱 weakest level of adversarial access is **transfer learning**. Without repeating what we've already discussed above, transfer learning is the weakest of the four because, 自然 naturally, as the network goes through its second training, its original memories (including the poison) get diluted 稀释.

对攻击能力的最后说明

Final note on capability — when we talk about poisoning the training data pool, one natural question that comes to mind is — well, how much of it can we poison? Because arguably if we could modify all of it — we could basically get the machine learning model to learn whatever we want. The answer — at least from research literature — seems to be aiming for less than 20%. Beyond that the threat model starts to sound unrealistic.

自然 可以说 研究文献

Defenses Against Data Poisoning

Similar to evasion attacks, when it comes to defenses, we're in a hard spot. Methods exist but none guarantee robustness in 100% of cases.

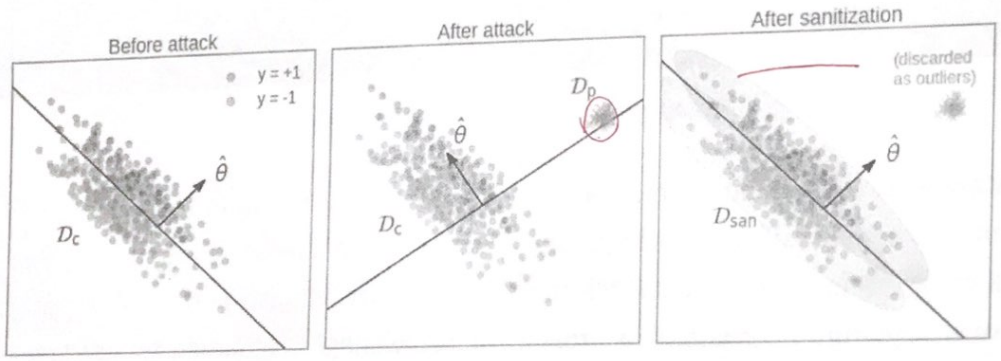
难点

The most common type of defenses is **outlier detection**, also knows as **"data sanitization"** and **"anomaly detection"**. The idea is simple — when poisoning a machine learning system the attacker is by definition injecting something into the training pool that is *very* different to what it should include — and hence we should be able to detect that.

消毒 异常检测

量化

The challenge is quantifying *"very"*. Sometimes the poison injected is indeed from a different data distribution and can be easily isolated:



Source.

In other cases, the attacker might generate poisoning points that are very similar to the true data distribution (called "inliers") but that still successfully mislead the model.

Another scenario where outlier detection breaks down is when poison was injected before filtering rules were created. In that case outliers stop being outliers:)

有趣的转折 异常 微型模型

An interesting twist on anomaly detection is **micromodels**. The Micromodels defense was proposed for cleaning training data for network intrusion detectors. The defense trains classifiers on non-overlapping epochs of the training set (micromodels) and evaluates them on the training set. By using a majority voting of the micromodels, training instances are marked as either safe or suspicious. Intuition is that attacks have relatively low duration and they could only affect a few micromodels at a time.

持续时间 可疑 直觉

The second most common type of defenses is to analyze the impact of newly added training samples on **model's accuracy**. The idea is that if a collected input is poisonous, it will destroy the model's accuracy on the test set, and by doing a sandboxed run with the new sample before adding it to the production training pool we can spot that. For example, reject on negative impact (RONI) and its cousin target-aware RONI (tRONI) are detection methods that rely on this property.

Some other interesting defenses I've seen include:

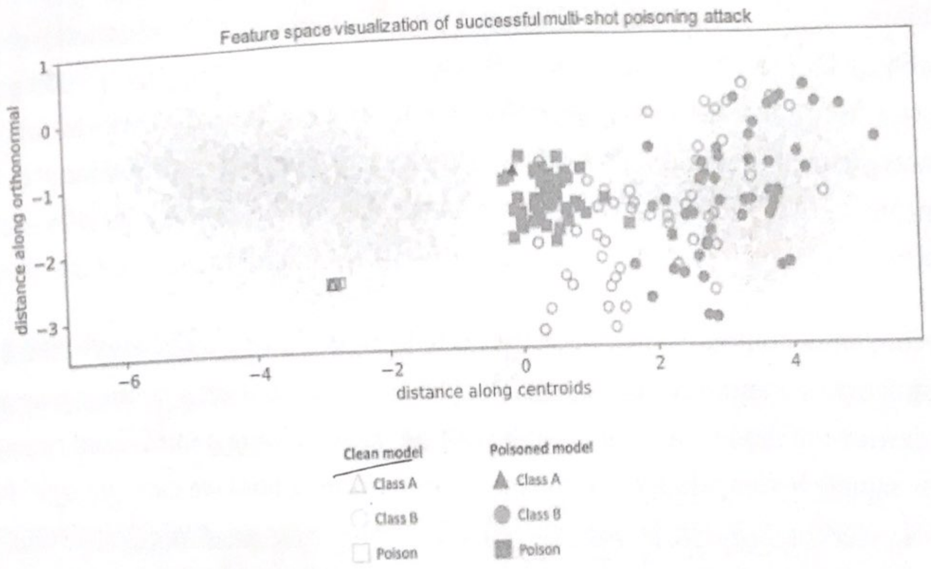
- STRIP = intentionally perturb the inputs and observe variance in predictions vs unperturbed. If not enough variance = malicious.
- Human in the loop = it's known that poisoning outliers leads to bigger boundary shifts, so why not focus human (presumably security analyst) attention on only such cases?
- TRIM = model learns in iterations from "trimmed" versions of MSEs (regression).

Evasion and Poisoning — Two Sides of the Same Coin

Remember in the beginning I said that most poisoning attacks work by shifting the classifier's boundary? What do you think the other kind do?

The answer is they "pull" the poisoned example across an existing boundary, instead of shifting it. Which is exactly what the authors of this paper managed to achieve:

1. They wanted to poison the classifier to classify A as B
2. So they took a bunch of Bs, perturbed them until the classifier thought they were As, then still labelled them as Bs, and inserted into the training pool
3. They thus poisoned the classifier with adversarial examples of Bs



Source.

The result, as the authors intended, was such that the classifier started recognizing As as Bs. But here's the most interesting part: what's another name for step (3) above?

…

If you've read my post on evasion attacks then you will hopefully recognize that it's **adversarial training**, the most common way to defend models against adversarial examples. This leaves us with an unsettling conclusion: **by doing a really good job of protecting yourself against evasion attacks — you might actually be opening the door to poisoning.**

How's that for a cliff-hanger?