Machine Learning Model Hardening

•••

For Fun and Profit

Ariel Herbert-Voss
@adversariel

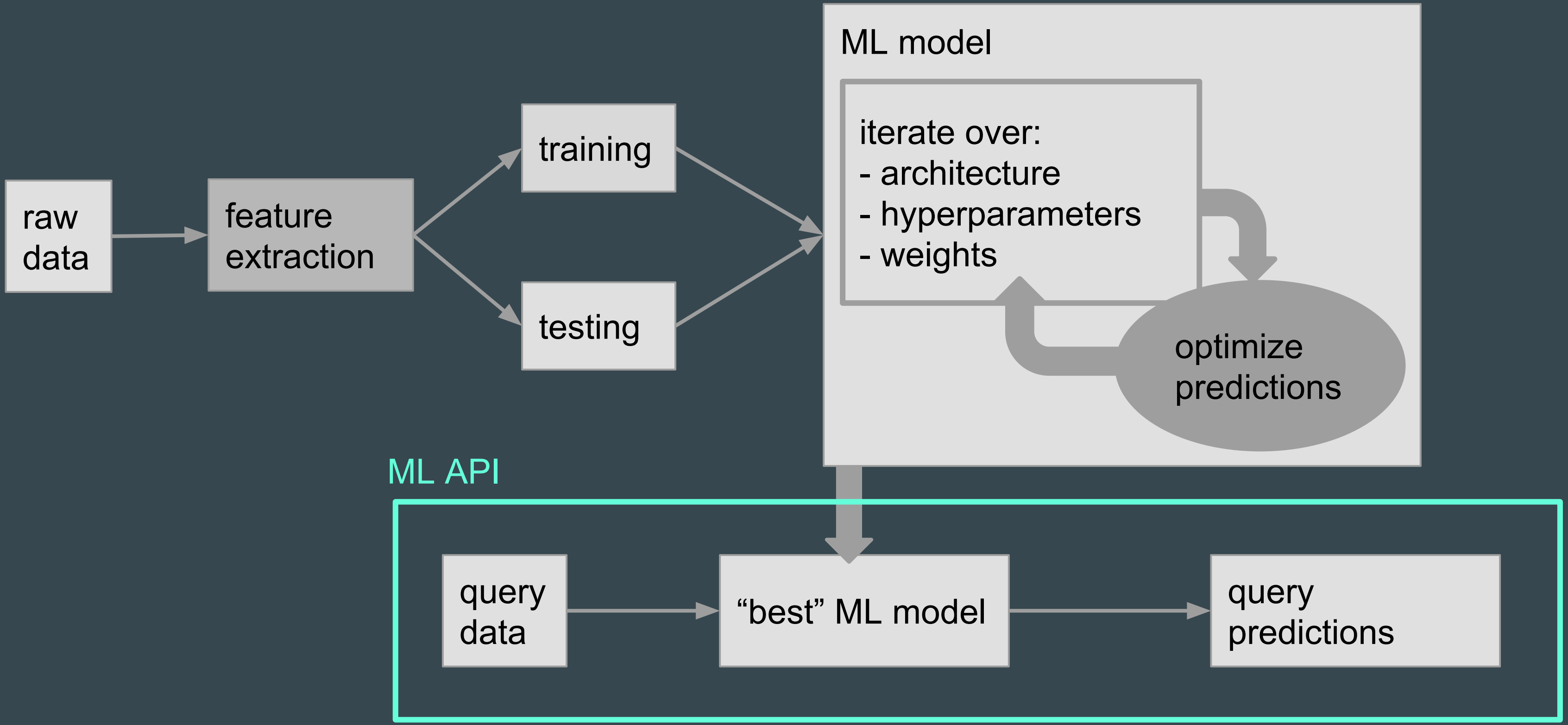
What is this about?

Most industry uses of machine learning are either deployed on site or provide API access

I will thoroughly disabuse you of the notion that it is a good idea to implement a vanilla ML model API with no model hardening

For simplicity, we’re talking about black box access to neural network-based machine learning models

Machine Learning pipeline



raw data

feature extraction

training

testing

ML model

iterate over:
- architecture
- hyperparameters
- weights

optimize predictions

ML API

query data

“best” ML model

query predictions

Threats versus Solutions

Attack	**data**	**model**	**predictions**
Adversarial examples			X
Model inversion	X	X	
Memorization	X	X	
Model theft		X	

Solutions:

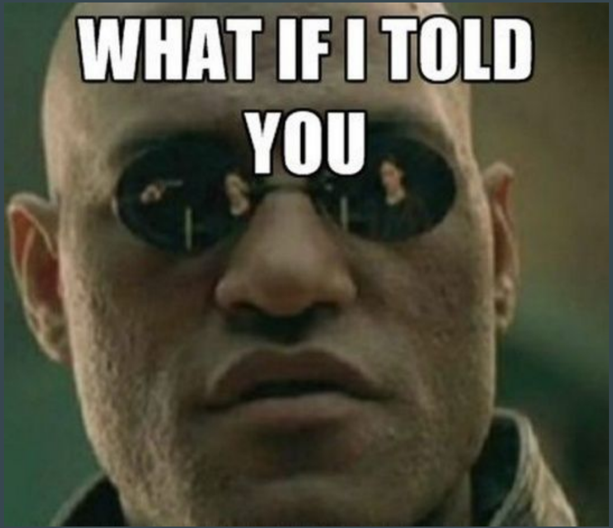
- Homomorphic encryption
- Secure multiparty encryption
- Differential privacy

Threats versus Solutions

Attack	data	model	predictions
Adversarial examples			X
Model inversion	X	X	
Memorization	X	X	
Model theft		X	

Solutions:

- Homomorphic encryption
- Secure multiparty encryption
- Differential privacy



Homomorphic encryption

Can perform computations on encrypted information

- Can’t read data but still preserves statistical structure
- Fully homomorphic encryption schemes are too slow to be practical
- Only fits needs if model is not an API

Secure multi-party computation

Multiple parties can jointly compute a function while keeping the function input private

- Cheaper than homomorphic encryption but requires more interaction between parties
- Have to redefine operators and functions
- Also slow as hell

Differential privacy

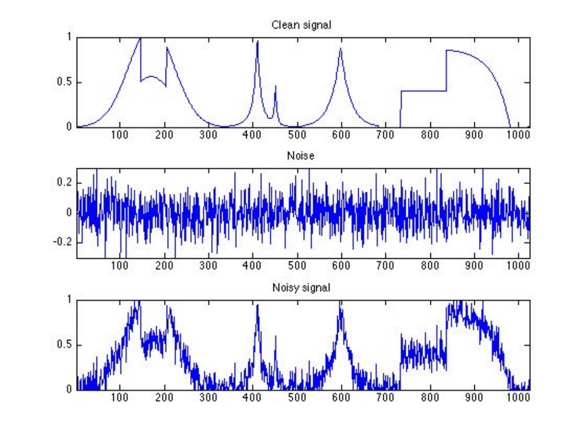
Adding or removing an element from the data doesn’t change the output distribution very much

- Also very slow, BUT
- Even works in scenarios where adversary has full knowledge of training mechanisms and access to parameters

Differential privacy

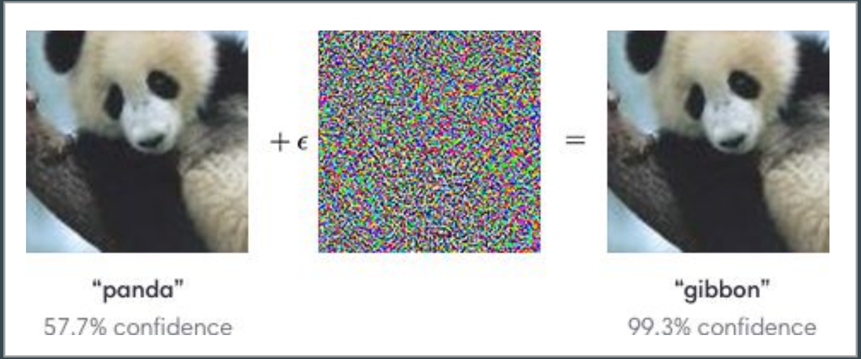
How do we do it?

1. Add noise to the output
2. Keep track of how many data access requests are granted

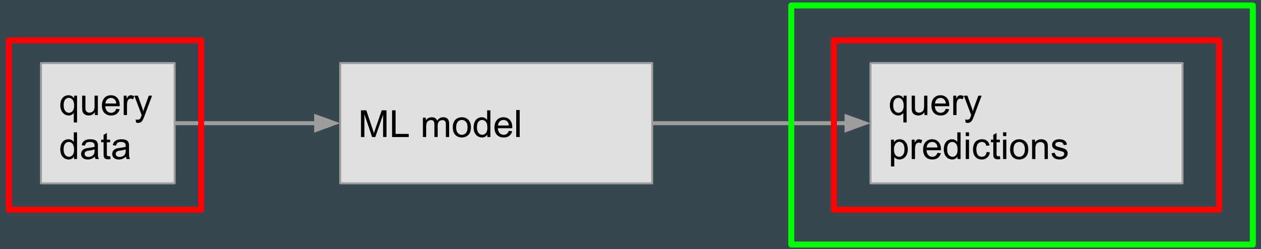


Adversarial examples

Give some slightly perturbed input to get incorrect predictions



Adversarial examples



query data

ML model

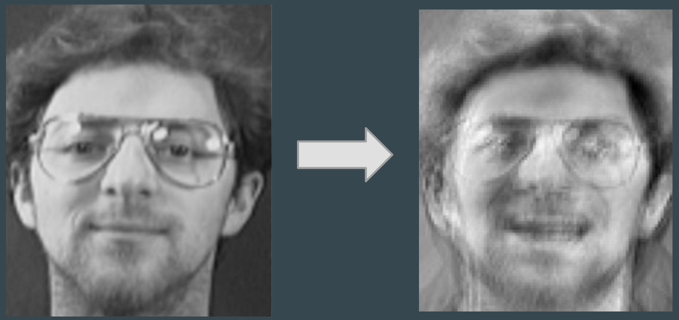
query predictions

50% dog
50% fried chicken

Model inversion

Given a categorization model/API that provides confidence values and predictions, we can recover information encoded in the model through the training data

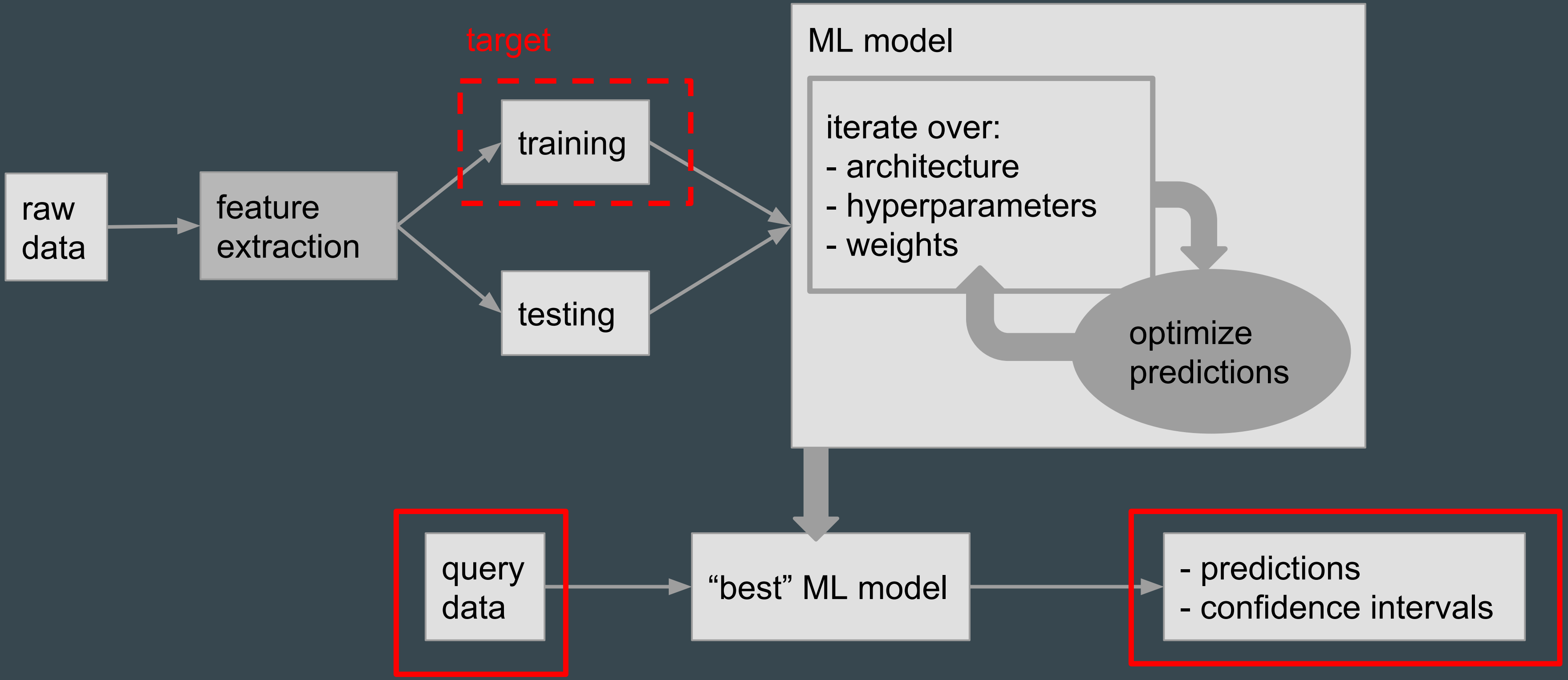
Scenario: adversary has somebody's name and wants to get an image of that person out of a facial recognition API



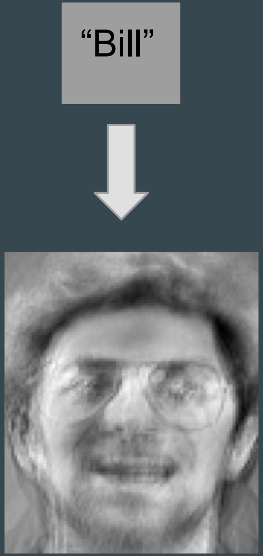
training image

recovered image

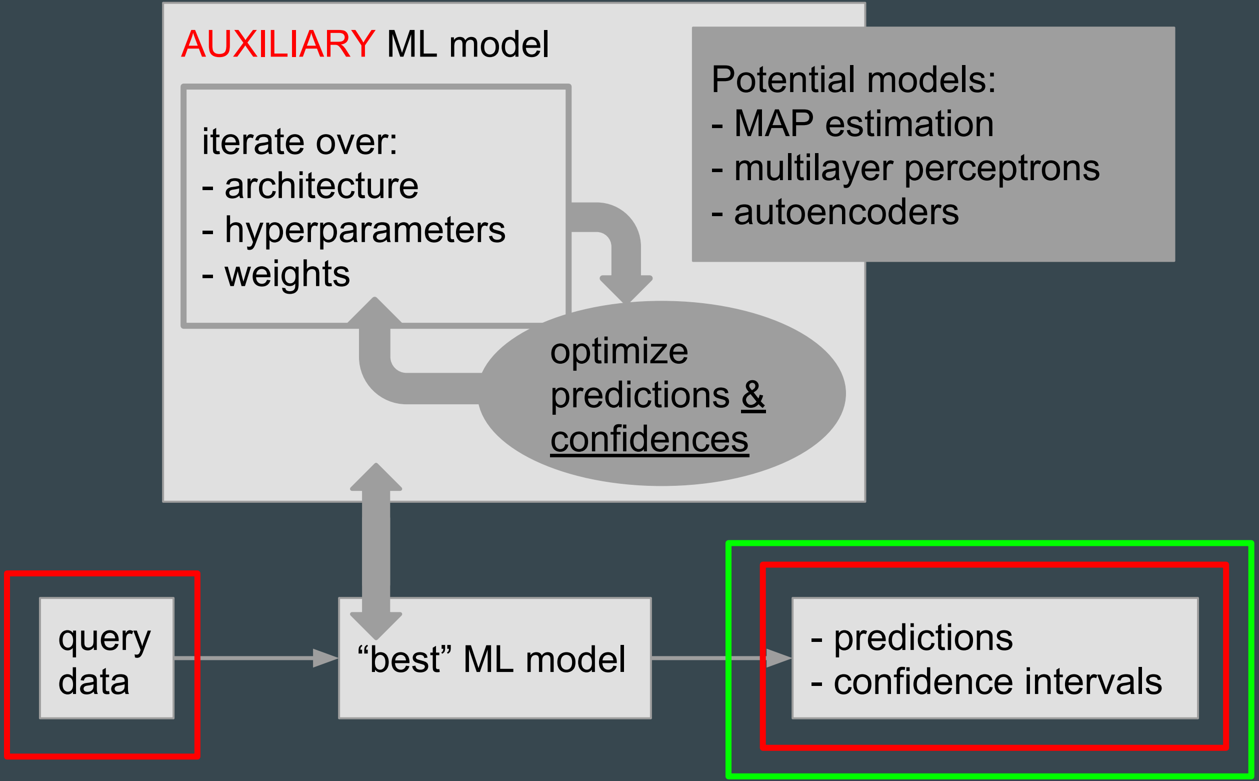
Model inversion



Model inversion



"Bill"



AUXILIARY ML model

iterate over:
- architecture
- hyperparameters
- weights

optimize predictions <u>&</u> <u>confidences</u>

Potential models:
- MAP estimation
- multilayer perceptrons
- autoencoders

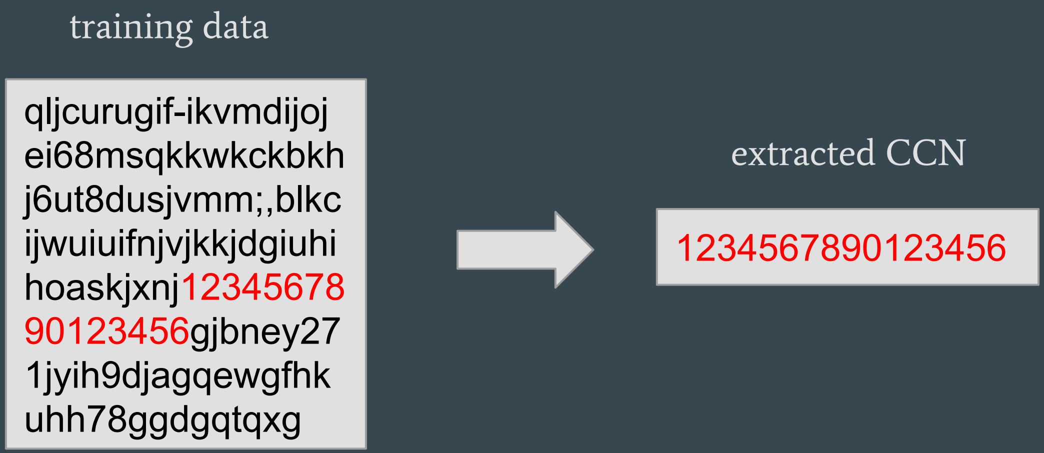
query data

"best" ML model

- predictions
- confidence intervals

Memorization

Given a known data format like a credit card number we can extract this information by using a search algorithm on the model predictions



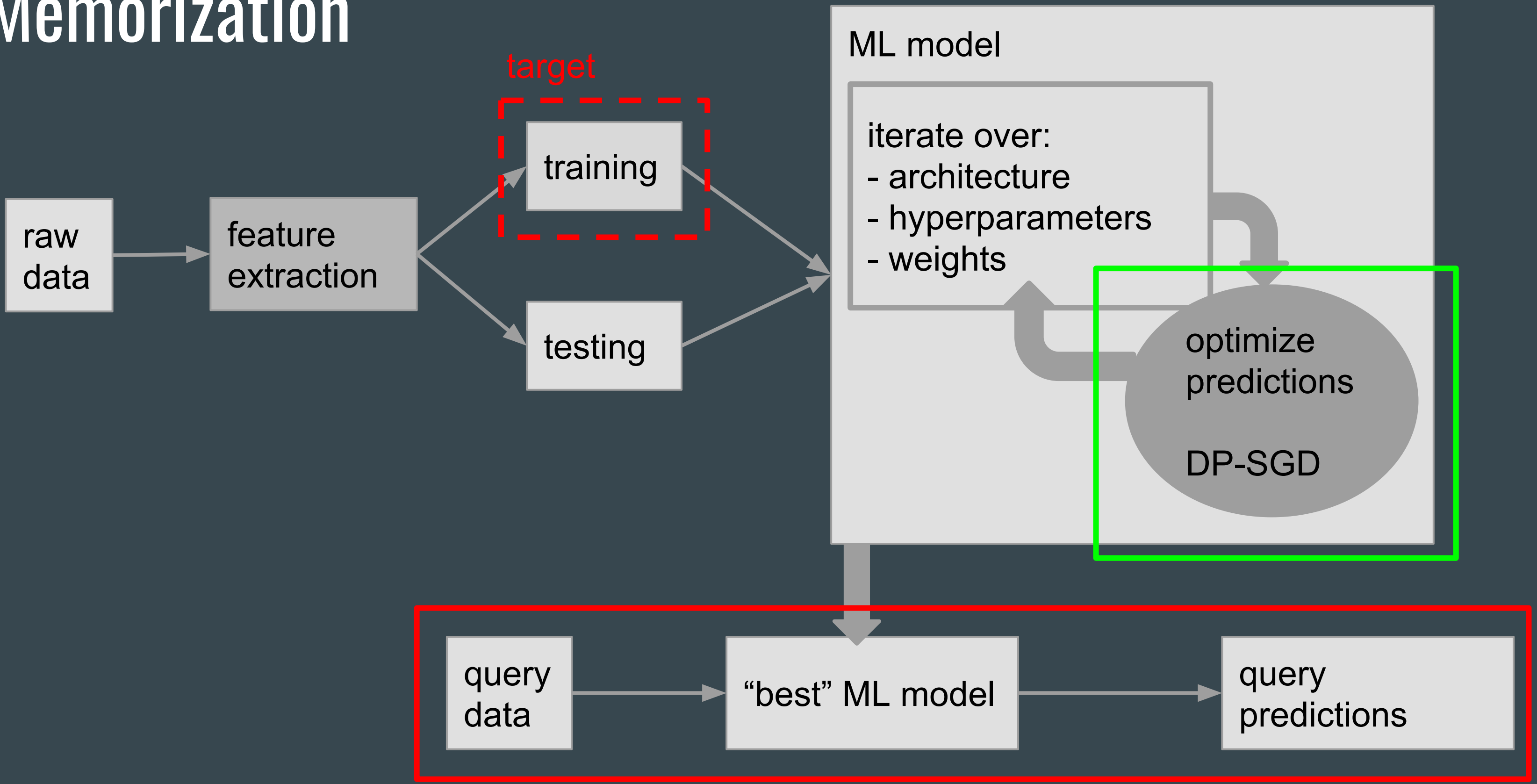
training data

qljcurugif-ikvmdijoj
ei68msqkkwkckbkh
j6ut8dusjvmm;,blkc
ijwuiuifnjvjkkjdgiuhi
hoaskjxnj12345678
90123456gjbney27
1jyih9djagqewgfhk
uhh78ggdgqtqxg

extracted CCN

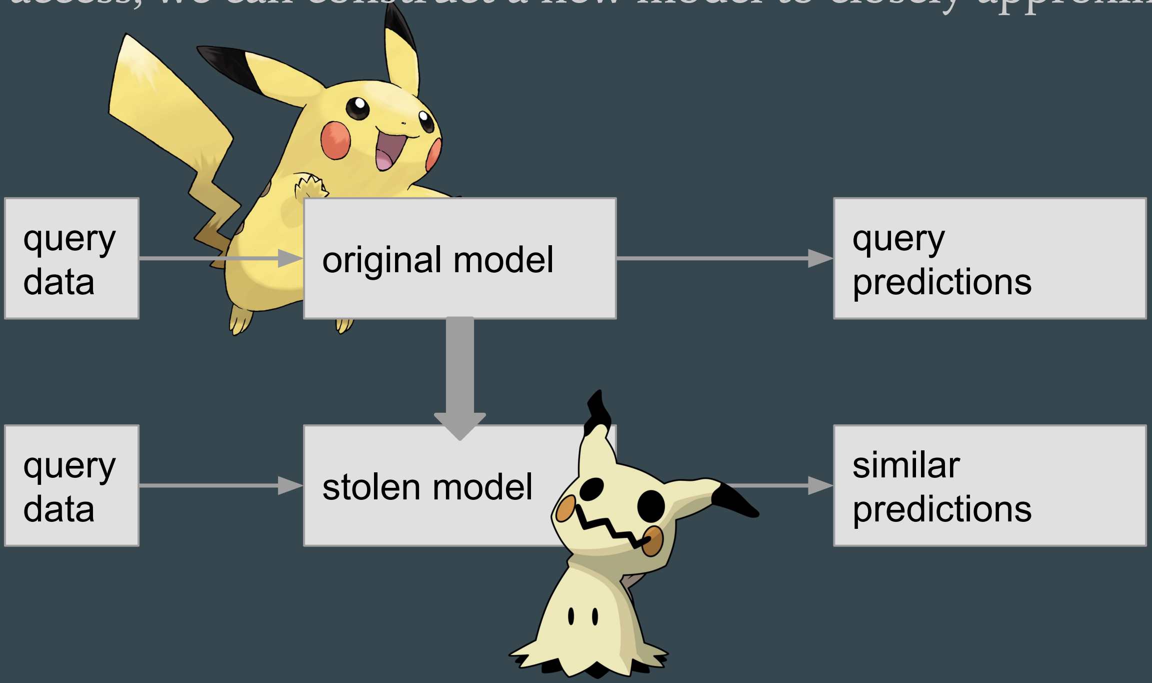
1234567890123456

Memorization



Model theft

Given black box access, we can construct a new model to closely approximate target



query data

original model

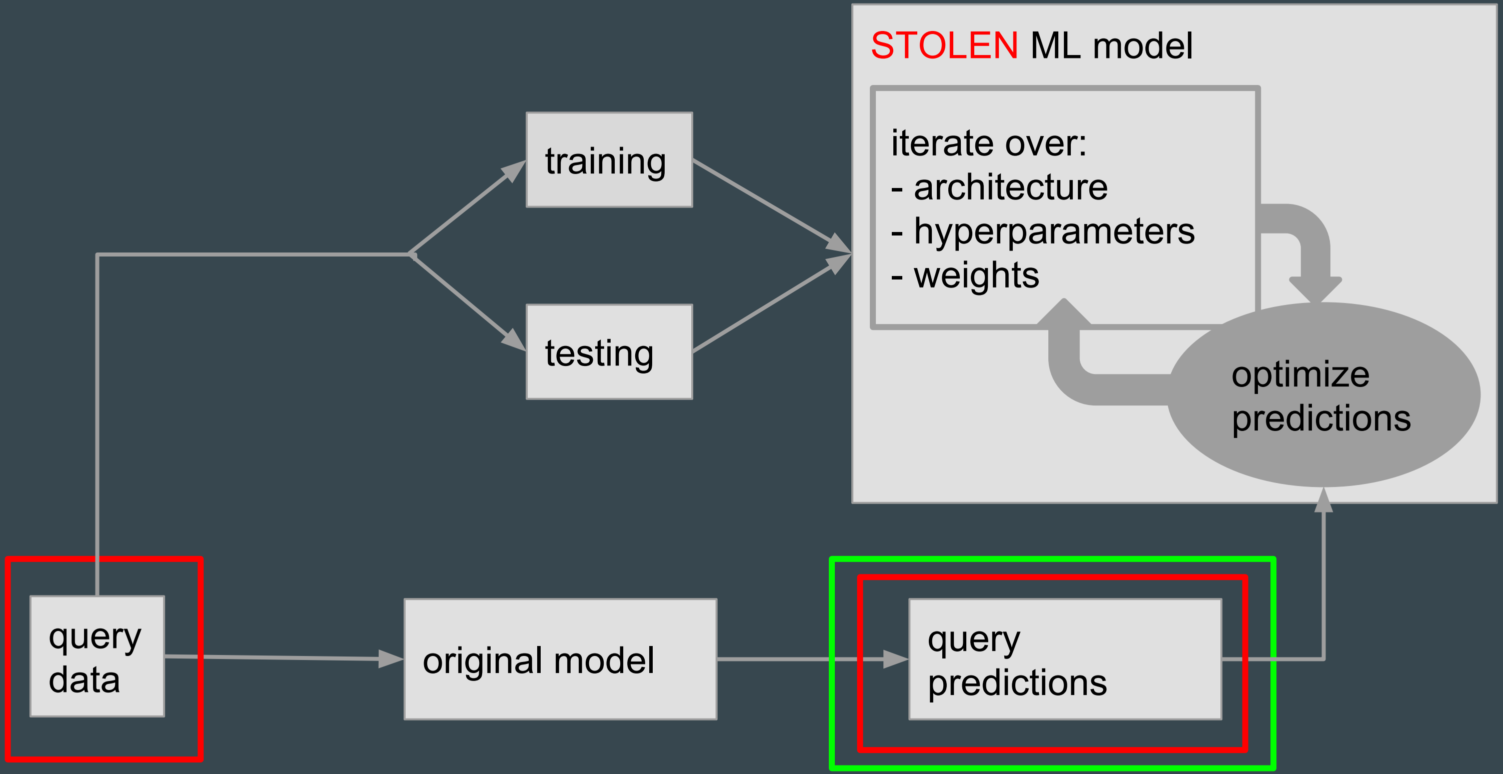
query predictions

query data

stolen model

similar predictions

Model theft


STOLEN ML model
iterate over:
- architecture
- hyperparameters
- weights
optimize predictions
training
testing
query data
original model
query predictions

General observations

Think about model hardening from the perspective of black box access

Notice how many of these attacks don’t matter if the data is encrypted or not as long as you can still get a “clear value” for the predictions

Model hardening is still more engineering than science, but much of it involves adding noise to the predictions to make it harder for adversaries to misuse information

Other hardening methods

Defensive distillation

Deep k-Nearest Neighbors

Ensemble adversarial training

Practical take-away summary slide

Hardening tips:

- Give the bare minimum amount of information
- Add some noise to output predictions
- Consider using an ensemble of models and return aggregate predictions

Most attacks are trying to get at information held in the model

So far differential privacy is the most reliable method of model hardening

For more information

Differential privacy: https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/dwork.pdf

Adversarial examples: https://arxiv.org/pdf/1605.07277.pdf

Model inversion: https://www.cs.cmu.edu/~mfredrik/papers/fjr2015ccs.pdf

Memorization: https://arxiv.org/pdf/1802.08232.pdf

Model theft: https://arxiv.org/pdf/1609.02943.pdf

Feel free to contact me via twitter or protonmail (adversariel), or grab a beer with me