# Machine Learning Security and Privacy
# Part II

Asst. Prof. Sinem Sav

Slides adapted from Carmela Troncoso, Reza Shokri, Bryan Ford, Jamie Hayes, Bogdan Kulynych, Nicolas Papernot, Nicholas Carlini, Rebekah Overdorf, Florian Tramer, Jean-Pierre Hubaux…

# Machine Learning – Security and Privacy (II)

- *Basics*
- *Model stealing*
- *Privacy issues*

- Altering the output

- Biases and fallacies

- Federated learning

# Machine Learning – Security and Privacy (II)

- *Basics*
- *Model stealing*
- *Privacy issues*

- **Altering the output**

- Biases and fallacies

- Federated learning

# “Adversarial Examples”: inputs that will make ML fail

There is no widely accepted definition of the term “Adversarial example”.

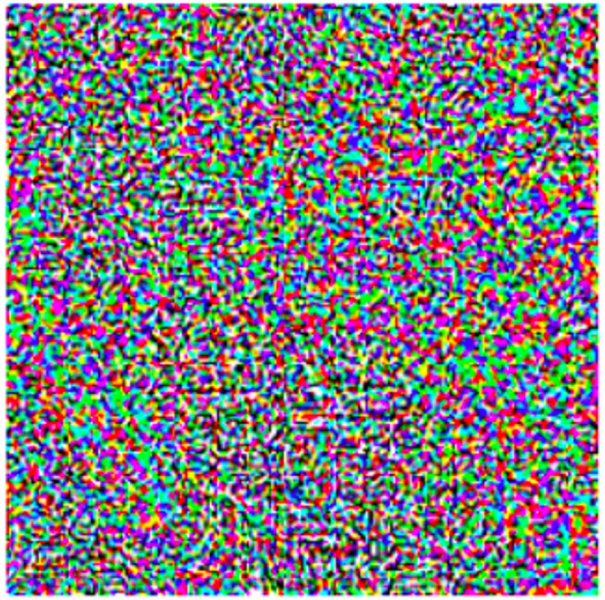
> *The phrase was coined by Szegedy et al. in 2014 [1] with respect to computer vision problems but goes back further to work on spam and malware evasion done by Srndic & Laskov [2], Biggio et al. [3], and Dalvi et al. [4].*

Working definition:

***Inputs to a model that an attacker has designed to cause the model to make a mistake.***

[1] Szegedy et al. Intriguing properties of neural networks, 2014
[2] Srndic & Laskov. Detection of malicious pdf files based on hierarchical document structure, NDSS 2013
[3] Biggio et al. Is data clustering in adversarial settings secure?, 2013
[4] Dalvi et al. Adversarial classification, 2013

$+ .007 \times$



$=$

Panda

Adversarial Perturbation

Gibbon

**IID assumptions no longer hold.**

(1) *Identical*: inputs are intentionally manipulated to not belong to the training distribution.
(2) *Independence*: inputs are no longer drawn independently; the attacker may sample from a single input repeatedly.

Goodfellow et al. Explaining and Harnessing Adversarial Examples, 2014
https://arxiv.org/abs/1412.6572

# Machine Learning 101: Quick refresher

Objective: find model parameters that *minimize* empirical loss

$$w^* = \arg\min_w \mathbb{E}_{x,y\sim D}[L(x, y; w)]$$

$$= \arg\min_w \sum_{x,y\in X} L(x, y; w)$$

Data distribution

Loss at (x, y) if model parameters are w

Training data

# Machine Learning 101: Quick refresher

***(simplified)**

How do we find those weights?
Often, the answer is to use a flavor of gradient descent*:

$$w := w - \lambda \nabla_w L(x, y; w)$$

Learning rate

Learning step

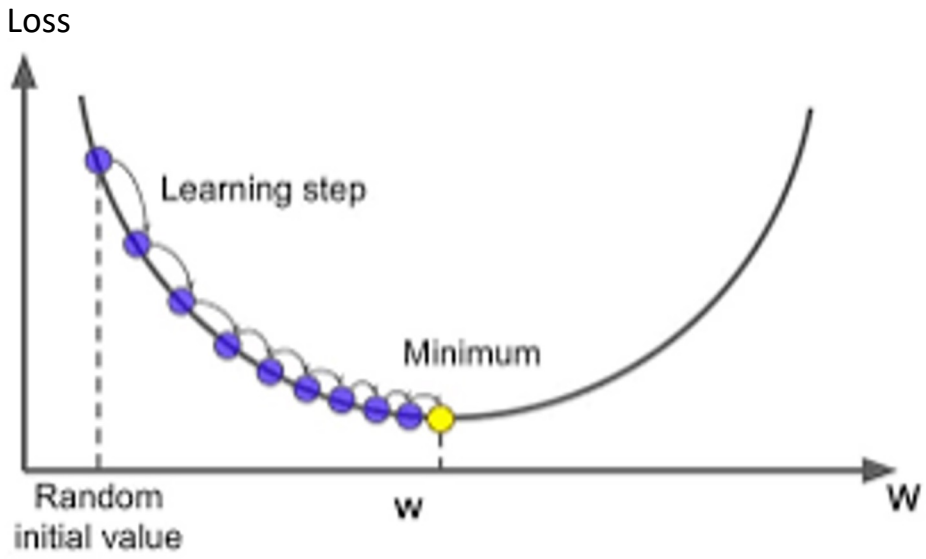



Image: Saugat Bhattarai

# Adversarial example problem

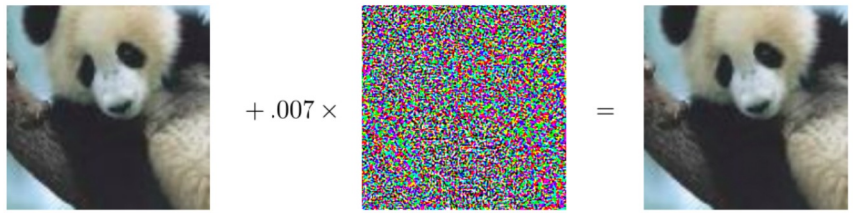
**Goal**: find a perturbation that *maximizes* loss

$$\delta^* = \arg\max_{\delta} L(x + \delta, y; w)$$

$$\text{s.t. } x + \delta \approx x$$

(x, y) is the initial example

Some similarity relation that usually encodes "imperceptible change"



# How to define similarity?

The similarity relation is often represented as *adversarial cost* constraint.

If the goal is to be imperceptible, the common cost is a norm or perturbation:

$$\delta^* = \arg\max_{\delta} L(x+\delta, y; w)$$

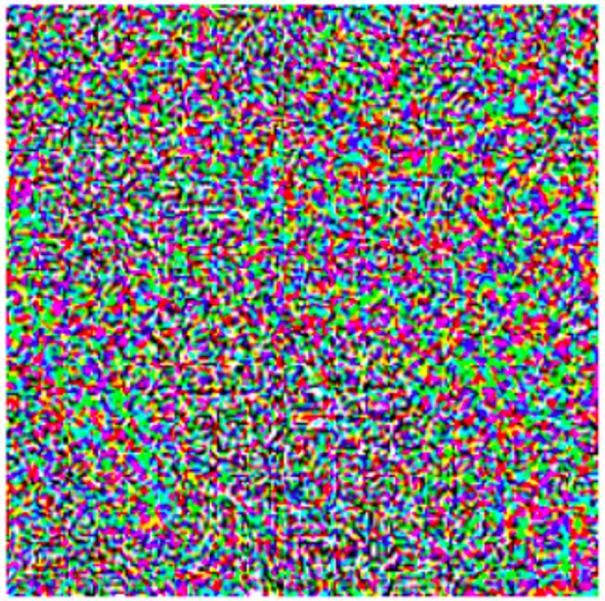
$$\text{s.t.} \quad \| \delta \|_p < \varepsilon$$

Budget: max allowed norm of the perturbation. E.g., inter-class distance

Cost: norm of the perturbation

# Is this a good constraint for imperceptible adversarial examples?

$+ .007 \times$



$=$

Panda

$\delta$

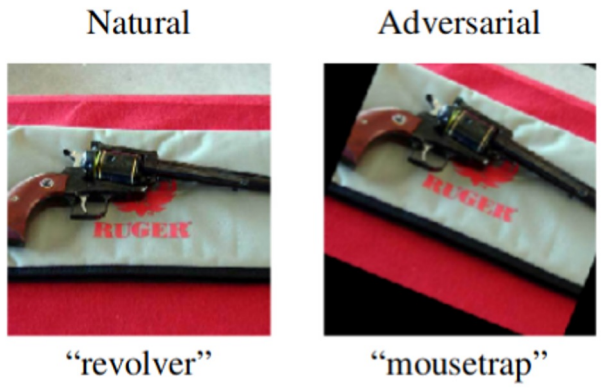
Gibbon

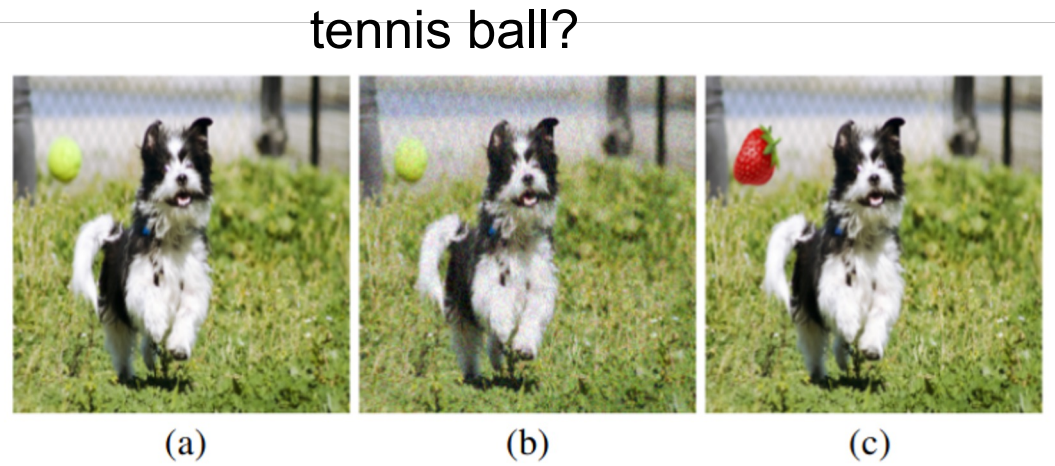
Norm (“size”) of the perturbation must be within epsilon

# Other similarity relations

Constraint on perturbation norm assumes that similar images are close in, e.g., Euclidean distance. This is not always true! [1]

For example, similarity could be defined as small affine transformations [2] :




Original

Imperceptible perturbation of l2 norm = 24.3

Semantic change l2 norm = 23.2

[1] Jacobsen et al., Exploiting Excessive Invariance caused by Norm-Bounded Adversarial Robustness, 2019, https://arxiv.org/pdf/1903.10484.pdf
[2] Engstrom et al., Exploring the Landscape of Spatial Robustness, 2019, https://arxiv.org/pdf/1712.02779.pdf

# Attacks do not have to be imperceptible!

Figure 3: An impersonation using frames. Left: Actress Reese Witherspoon (by Eva Rinaldi / CC BY-SA / cropped from https://goo.gl/a2sCdc). Image classified correctly with probability 1. Middle: Perturbing frames to impersonate (actor) Russel Crowe. Right: The target (by Eva Rinaldi / CC BY-SA / cropped from https://goo.gl/AO7QYu).

Sharif et al. Accessorize to a Crime: Real and Stealthy Attacks on State-of-the-Art Face Recognition. CCS 2016
https://www.cs.cmu.edu/~sbhagava/papers/face-rec-ccs16.pdf

# How to solve the optimization problem?

Recall the adversarial example problem:

$$\delta^* = \arg\max_{\delta} L(x+\delta, y; w)$$

$$\text{s.t.} \quad \| \delta \|_p < \varepsilon$$

# Projected gradient descent attack

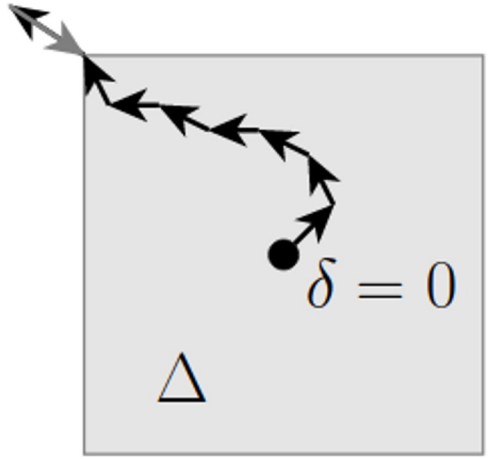
Take a step in the direction of maximum loss, project to meet constraint. Run for several dozens iterations from a random starting delta:

$$\delta := \text{proj}[\delta + \lambda \ \text{sign}\ (\nabla_{\delta} L(x + \delta, y; w))]$$

If outside of the **similarity** region, project back inside

Direction of the gradient

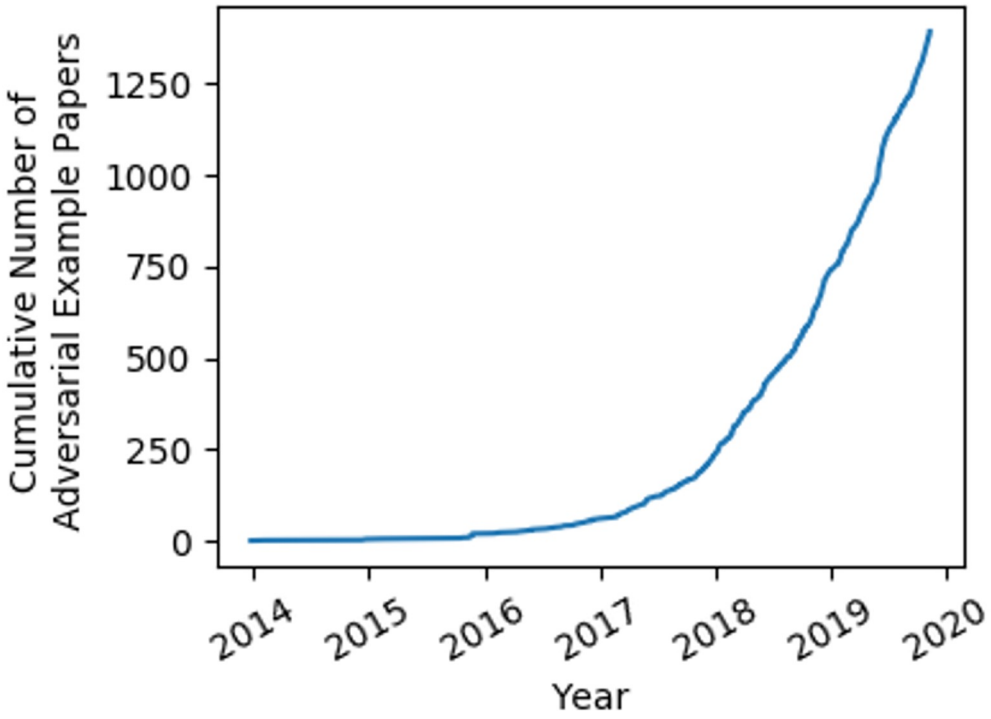
Gradient w.r.t. **input**, not parameters!




[1] Mądry et al., Towards Deep Learning Models Resistant to Adversarial Attacks, 2019

Image: Zico Kolter

# This field evolves very fast…



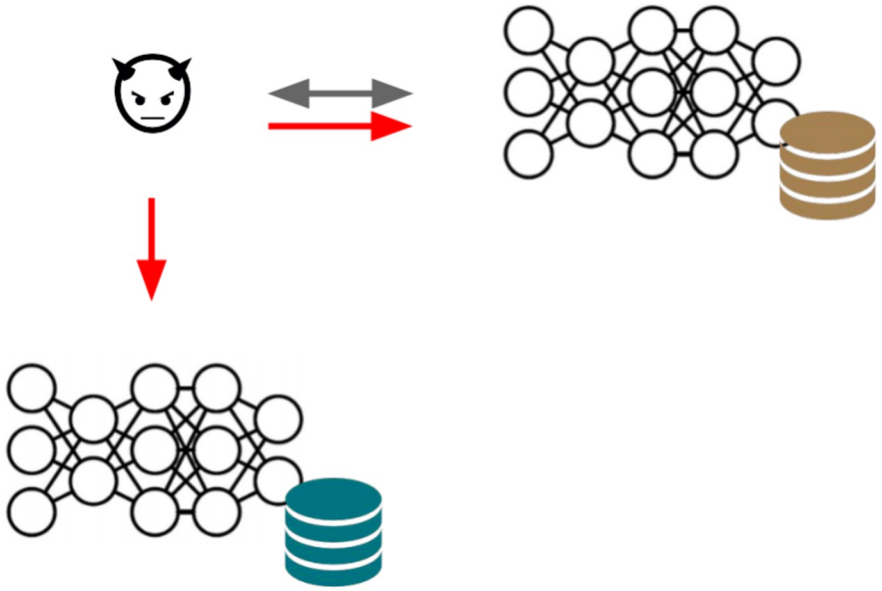

https://nicholas.carlini.com/writing/2019/all-adversarial-example-papers.html

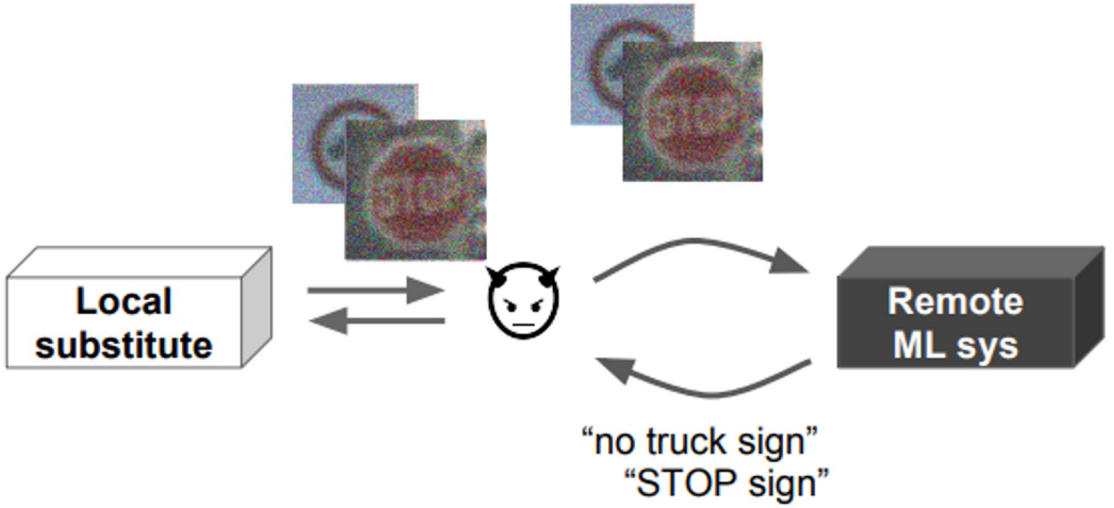
N. Carlini: On Evaluating Adversarial Robustness *CAMLIS (keynote), 2019*.
https://www.youtube.com/watch?v=-p2il-V-0fk

# Transferability Property

Adversarial examples have a **transferability** property:

*samples crafted to mislead a model A are likely to mislead a model B*





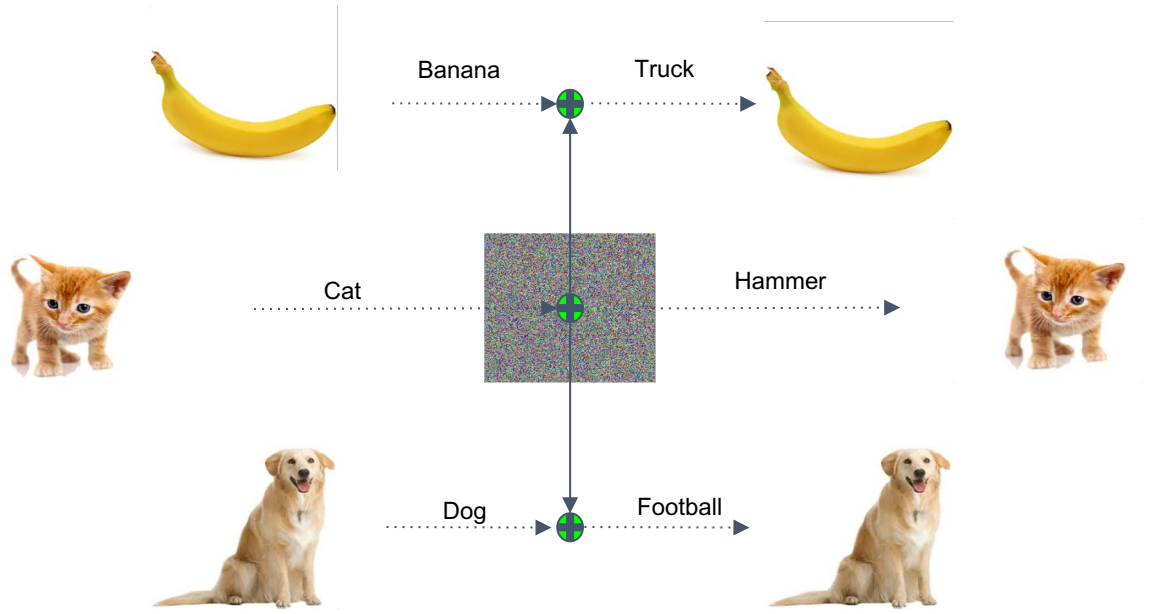
# Why do adversarial examples exist and why do they transfer?

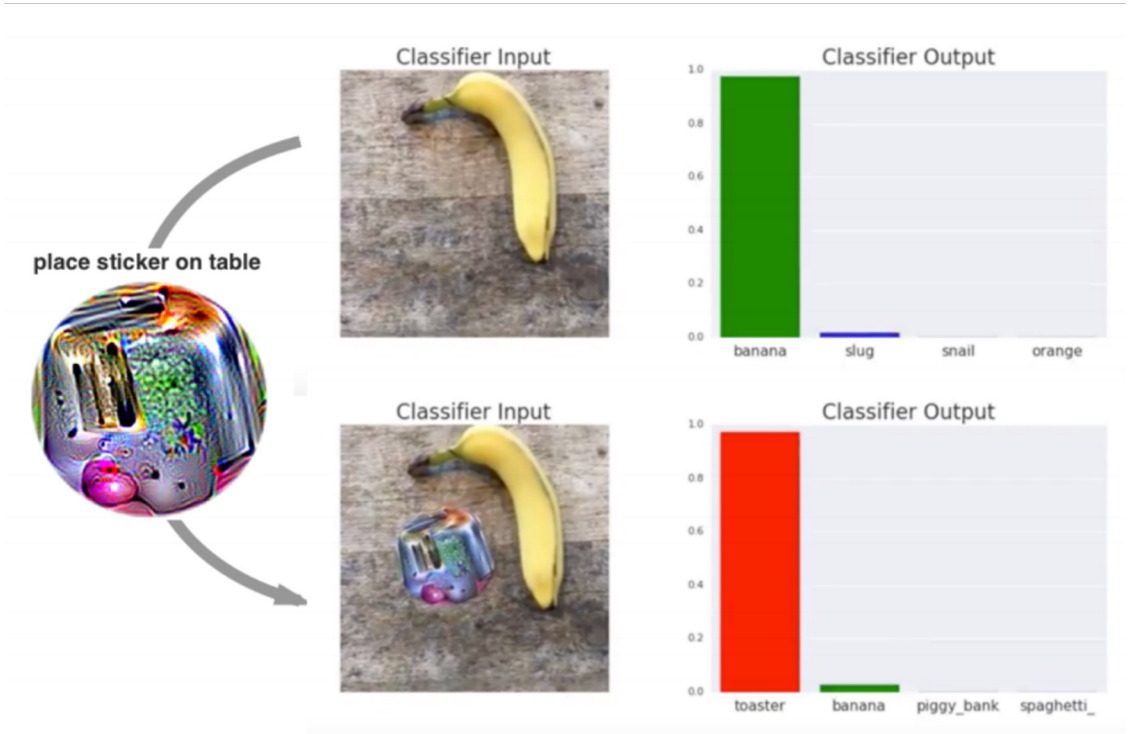
- Linear behavior in high dimensional spaces is sufficient to produce adversarial examples in neural networks.
- Too much freedom! Impossible to handle all directions
- And many more….?

[GSS15] Goodfellow et al. Explaining and Harnessing Adversarial Examples
[LCL17] Liu et al. Delving into Transferable Adversarial Examples and Black-Box Attacks

In the most extreme case, it is possible to construct a single perturbation that will fool a model when added to any image.

Attackers need minimal resources to attack the system!




[GSS15] Goodfellow et al. Explaining and Harnessing Adversarial Examples
[MFF16] Moosavi-Dezfooli et al. Universal adversarial perturbations.

# Defending against adversarial examples?

Recall the machine learning training objective:

$$w^* = \arg\min_w \mathbb{E}_{x,y\sim D}[L(x, y; w)]$$
$$= \arg\min_w \sum_{x,y\in X} L(x, y; w)$$

# Defending against adversarial examples?

Defending in general is very hard. Can only defend against a particular threat model (e.g., perturbations up to epsilon norm), and normally no guarantees.

Standard way is *adversarial training* (based on *robust optimization)*. It means training on simulated adversarial examples:

$$w^* = \min_w \sum_{x,y \in X} \underbrace{\max_\delta L(x+\delta, y; w)}_{\text{Adversarial example}}$$

e.g. 20 random restarts, 100 steps

"threat model"

$$\text{s.t.} \quad \| \delta \|_p < \varepsilon$$

# Preventing adversarial examples

Certified defenses [1]

Guarantee robustness to norm bounded attacks by ensuring that no example can exist inside a ball with radius the norm used for the perturbation

Detect suspicious queries [2]

Identify deviations from expected on distribution of successive queries from a client

 Unclear whether they are really effective

[1] Lecuyer et al. Certified robustness to adversarial examples with differential privacy. 2019
[2] Chen et al. Stateful detection of black-box adversarial attacks. 2019

# Attacks are not restricted to computer vision

**Twitter Bot Detection [1].**

Detection tools can easily be fooled by tweaking the number of replies or retweets.

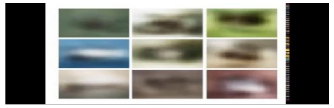
**Text Classification [2].**

| Original Text Prediction = **Negative**. (Confidence = 78.0%) |
| --- |
| *This movie had **terrible** acting, **terrible** plot, and **terrible** choice of actors. (Leslie Nielsen ...come on!!!) the one part I **considered** slightly funny was the battling FBI/CIA agents, but because the audience was mainly **kids** they didn't understand that theme.* |
| Adversarial Text Prediction = **Positive**. (Confidence = 59.8%) |
| *This movie had **horrific** acting, **horrific** plot, and **horrifying** choice of actors. (Leslie Nielsen ...come on!!!) the one part I **regarded** slightly funny was the battling FBI/CIA agents, but because the audience was mainly **youngsters** they didn't understand that theme.* |

**Audio [3].**

"without the dataset the article is useless"

" "



"okay google browse to evil dot com"

"speech can be embedded in music "

[1] Kulynych et al. Evading classifiers in discrete domains with provable optimality guarantees.
[2] Alzantot et al. Generating Natural Language Adversarial Examples.
[3] Carlini et al. Audio Adversarial Examples.

# Takeaways on adversarial examples

Protecting ML against output alteration (by adversarial examples) is **HARD**

If the adversary controls the inputs

- in deployed models, the adversary can always win!

Unclear whether there will ever be effective defenses

High-dimensional spaces difficult to characterize

Adversarial training is the most promising protection technique

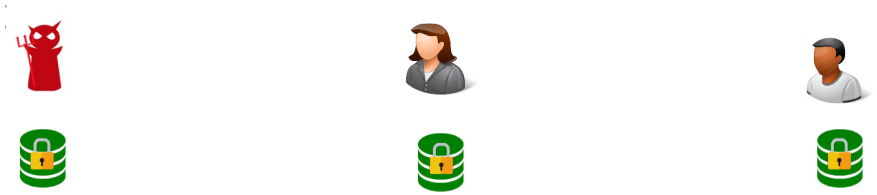
Conclusion: If adversarial examples can occur for an ML usage you are considering, be very careful because this can create a severe vulnerability

# Backdoor and Poisoning attacks

# Assumptions

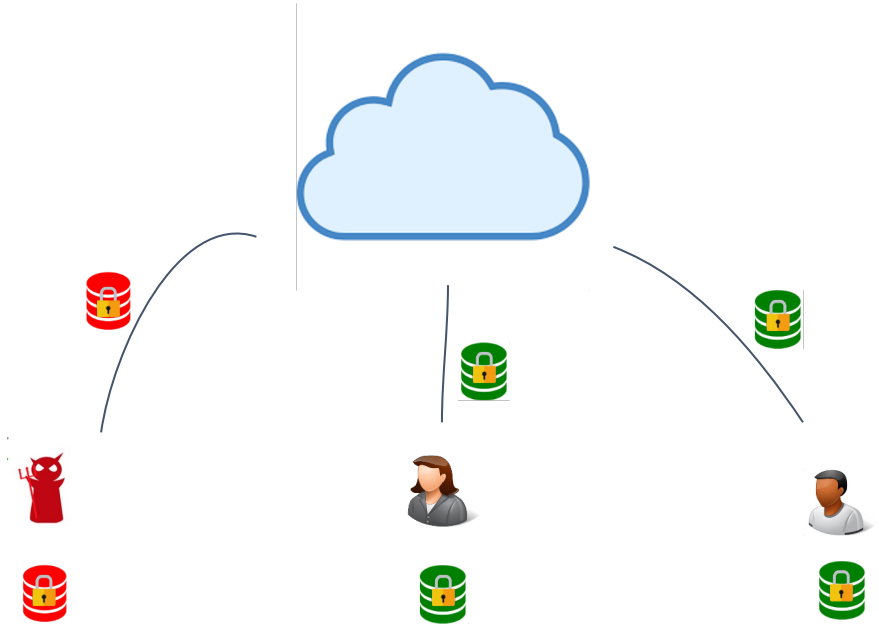
# Assumptions

- Adversary controls a subset of inputs.



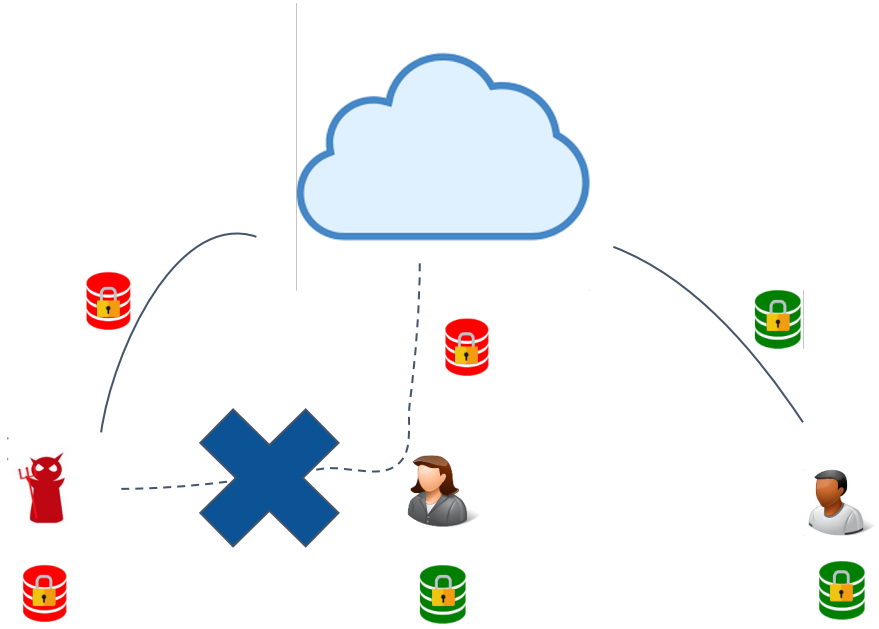
# Assumptions

- Adversary controls a subset of inputs.
- Adversary can modify the inputs they send to central server.



# Assumptions

- Adversary controls a subset of inputs.
- Adversary can modify the inputs they send to central server.
- Adversary **cannot** modify the training code or inputs from other parties.



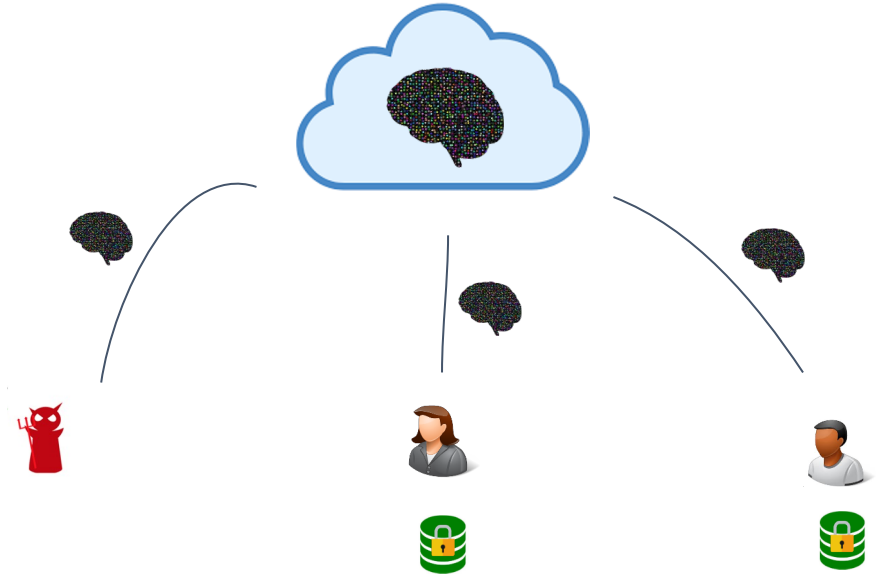
# Assumptions

- Adversary controls a subset of inputs.
- Adversary can modify the inputs they send to central server.
- Adversary **cannot** modify the training code or inputs from other parties.

# Potential Attacks

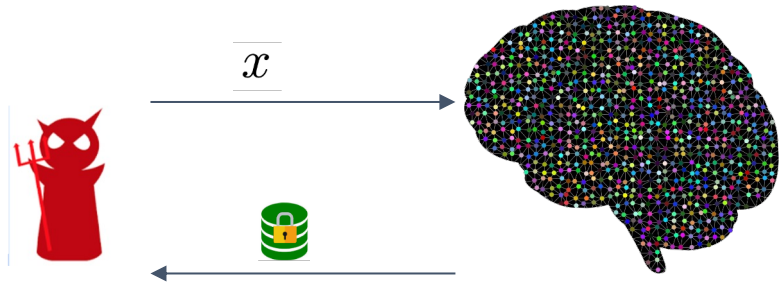
# Assumptions

- Adversary controls a subset of inputs.
- Adversary can modify the inputs they send to central server.
- Adversary **cannot** modify the training code or inputs from other parties.



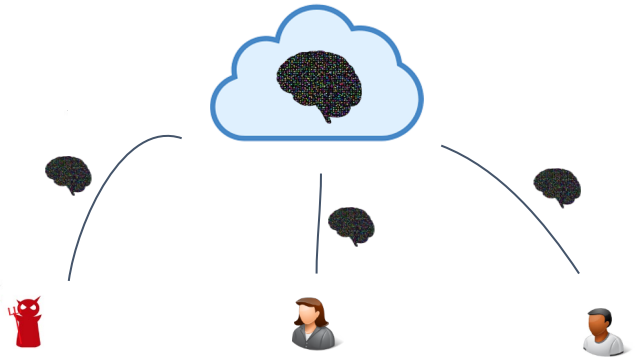
# Potential Attacks

- Create a model that reveals information about other parties data.



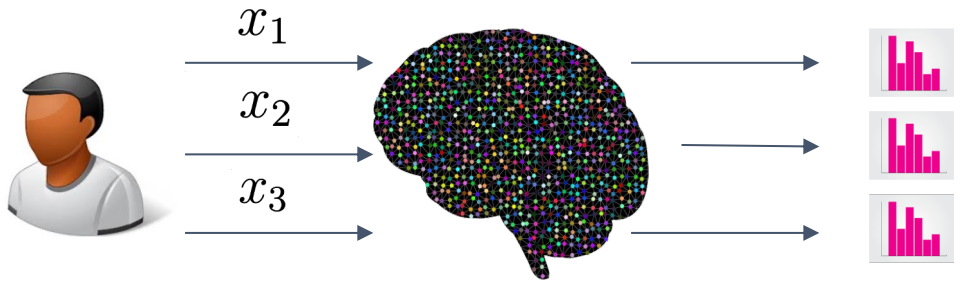
# Assumptions

- Adversary controls a subset of inputs.
- Adversary can modify the inputs they send to central server.
- Adversary **cannot** modify the training code or inputs from other parties.



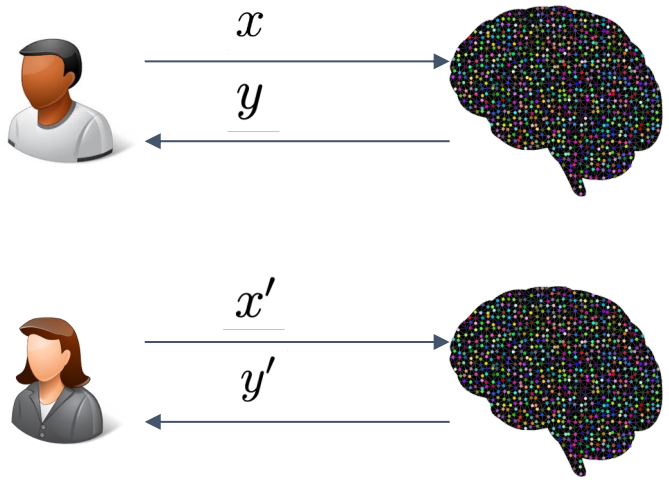
# Potential Attacks

- Create a model that reveals information about other parties data.
- Reduce the utility of the model (*poisoning*).



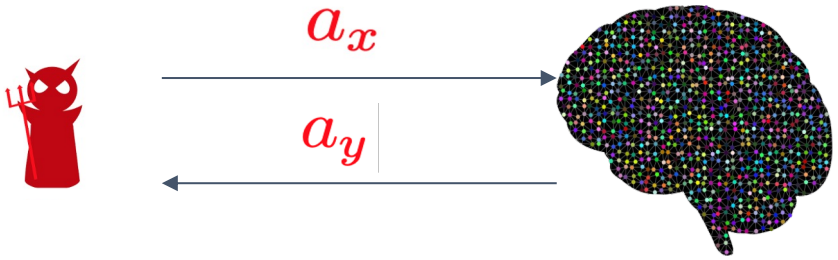
# Assumptions

- Adversary controls a subset of inputs.
- Adversary can modify the inputs they send to central server.
- Adversary **cannot** modify the training code or inputs from other parties.



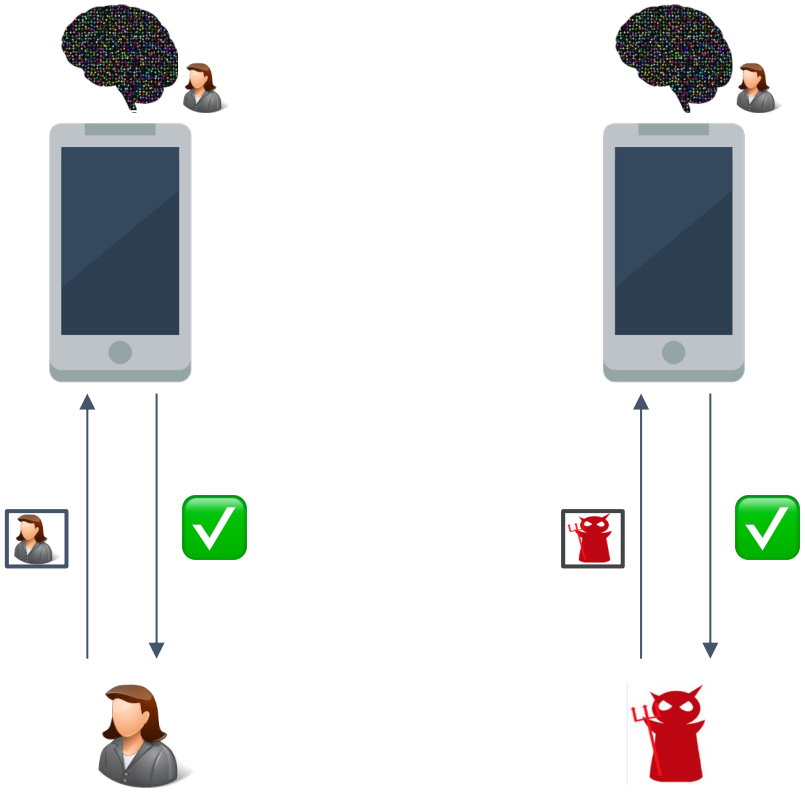

# Potential Attacks

- Create a model that reveals information about other parties data.
- Reduce the utility of the model (*poisoning*).
- Backdoor the model.

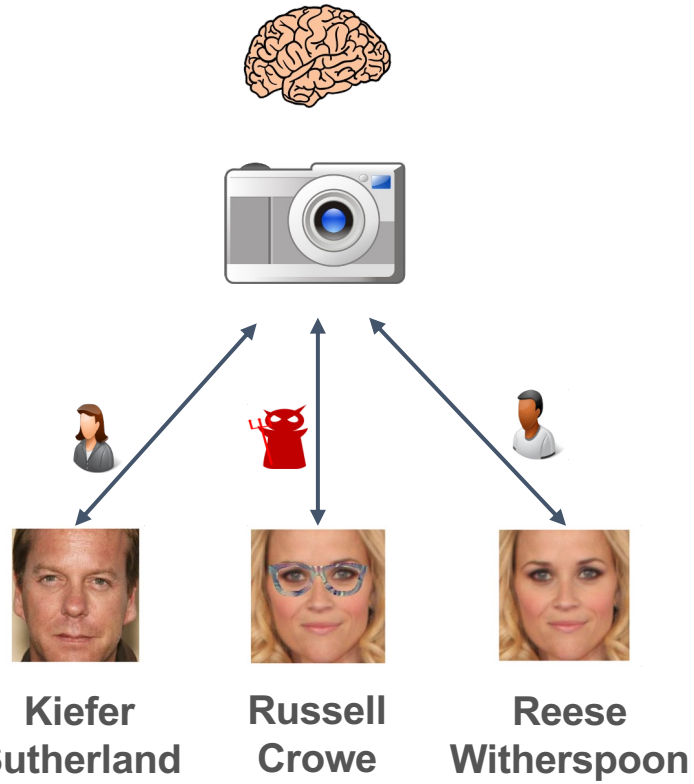


# Backdoor attack

*Given a trained model **f** that performs well on test data, there exists attacker chosen inputs **x** with attacker chosen class **t**, such that **f(x) = t**.*







# How to mount a poisoning attack?

## Quick ML refresher I

Objective: find model parameters that *minimize* empirical loss

$$w^* = \arg\min_w \mathbb{E}_{x,y\sim D}[L(x, y; w)]$$

$$= \arg\min_w \sum_{x,y\in X} L(x, y; w)$$

Data distribution

Loss at (x, y) if model parameters are w

Training data

# How to mount a poisoning attack?

One way is to solve a two-level optimization problem:

Whatever the adversary wants to achieve, e.g., high loss, mistakes on certain inputs (backdooring)

$$\delta^* = \arg\max_{\delta} U(w^*)$$

$$\text{where } w^* = \arg\min_{w} \sum_{x,y \in X \cup \{x' + \delta\}} L(x, y; w)$$

Simulated training on normal data *X* and a poisoned example x' + delta

# How to mount a poisoning attack?

$$\delta^* = \arg\max_{\delta} U(w^*)$$

$$\text{where } w^* = \arg\min_{w} \sum_{x,y \in X \cup \{x' + \delta\}} L(x, y; w)$$

Usually computationally expensive (have to simulate training inside) and requires many poisoned inputs—but powerful.

# How to mount a poisoning attack?

**Very similar to adversarial examples but should be in the training time + you should be to poison!!**

$$\delta^* = \arg\max_{\delta} U(w^*)$$

$$\text{where } w^* = \arg\min_{w} \sum_{x,y \in X \cup \{x' + \delta\}} L(x, y; w)$$

Usually computationally expensive (have to simulate training inside) and requires many poisoned inputs—but powerful.

# Machine Learning – Security and Privacy (II)

- *Basics*
- *Model stealing*
- *Privacy issues*

- Altering the output

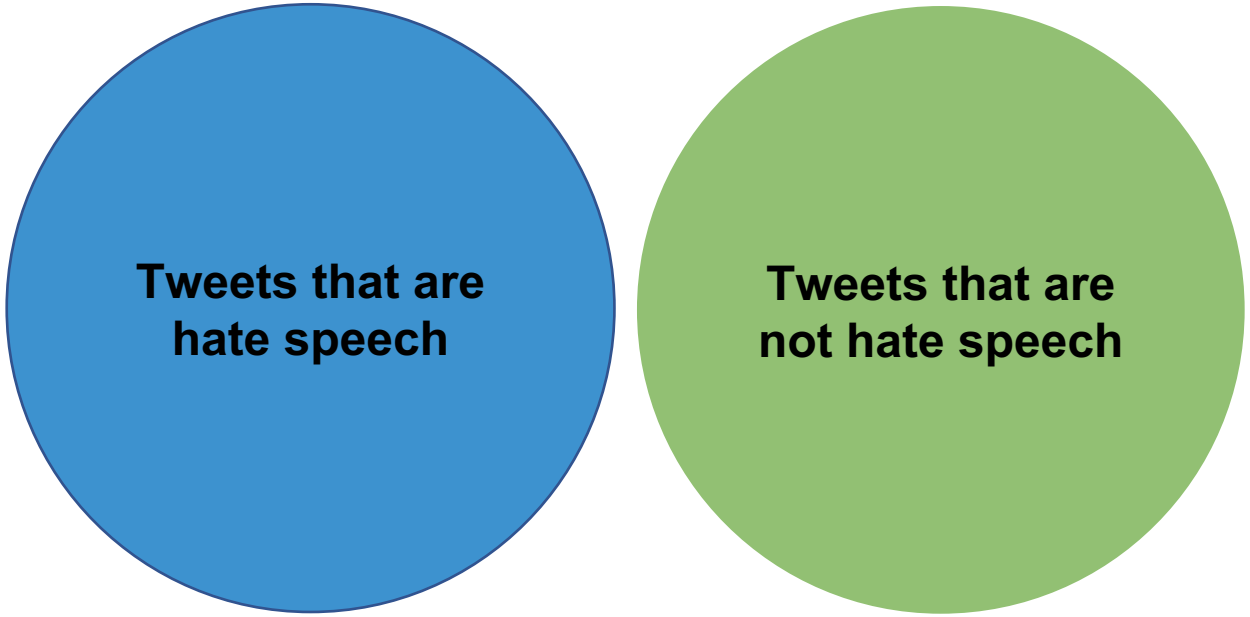
- **Biases and fallacies**

- Federated learning

# Base Rate Fallacy / Prosecutor's fallacy

**Quick reminder: AI performance metrics**

**Example**: Hate speech detection



# Base Rate Fallacy / Prosecutor's fallacy

**Quick reminder: AI performance metrics**

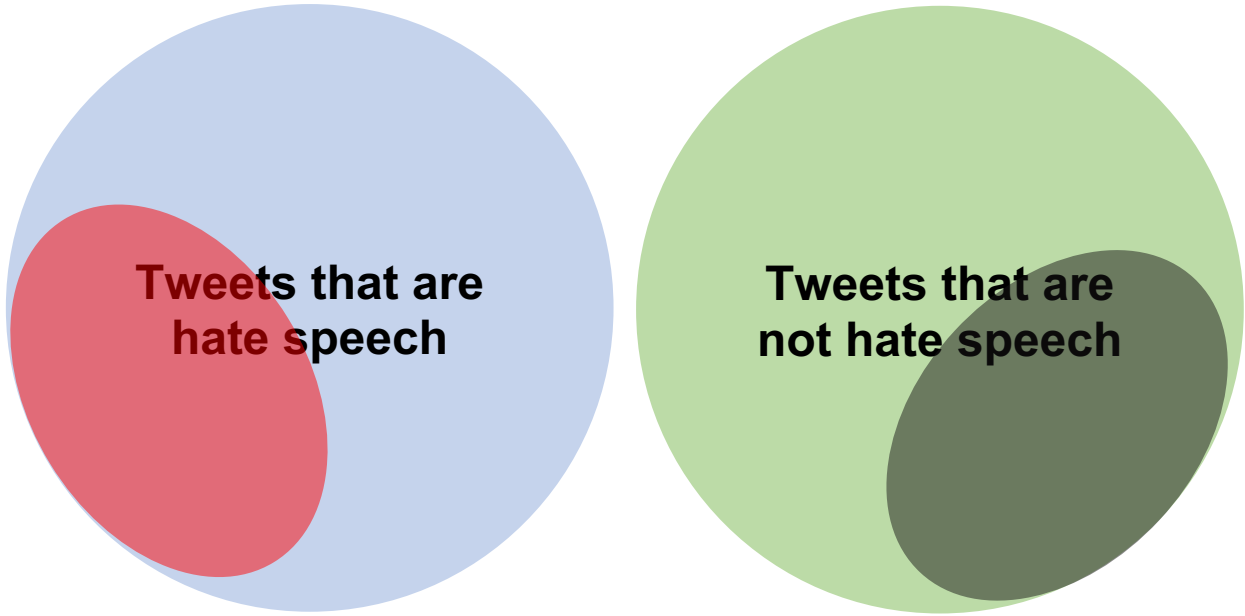
**Example**: Hate speech detection

**True Positives** — **Prediction: hate speech**

**False Negative** — **Prediction: not hate speech**

**True Negatives** — **Prediction: not hate speech**

**False Positives** — **Prediction: hate speech**



# Base Rate Fallacy / Prosecutor's fallacy

**Quick reminder: AI performance metrics**

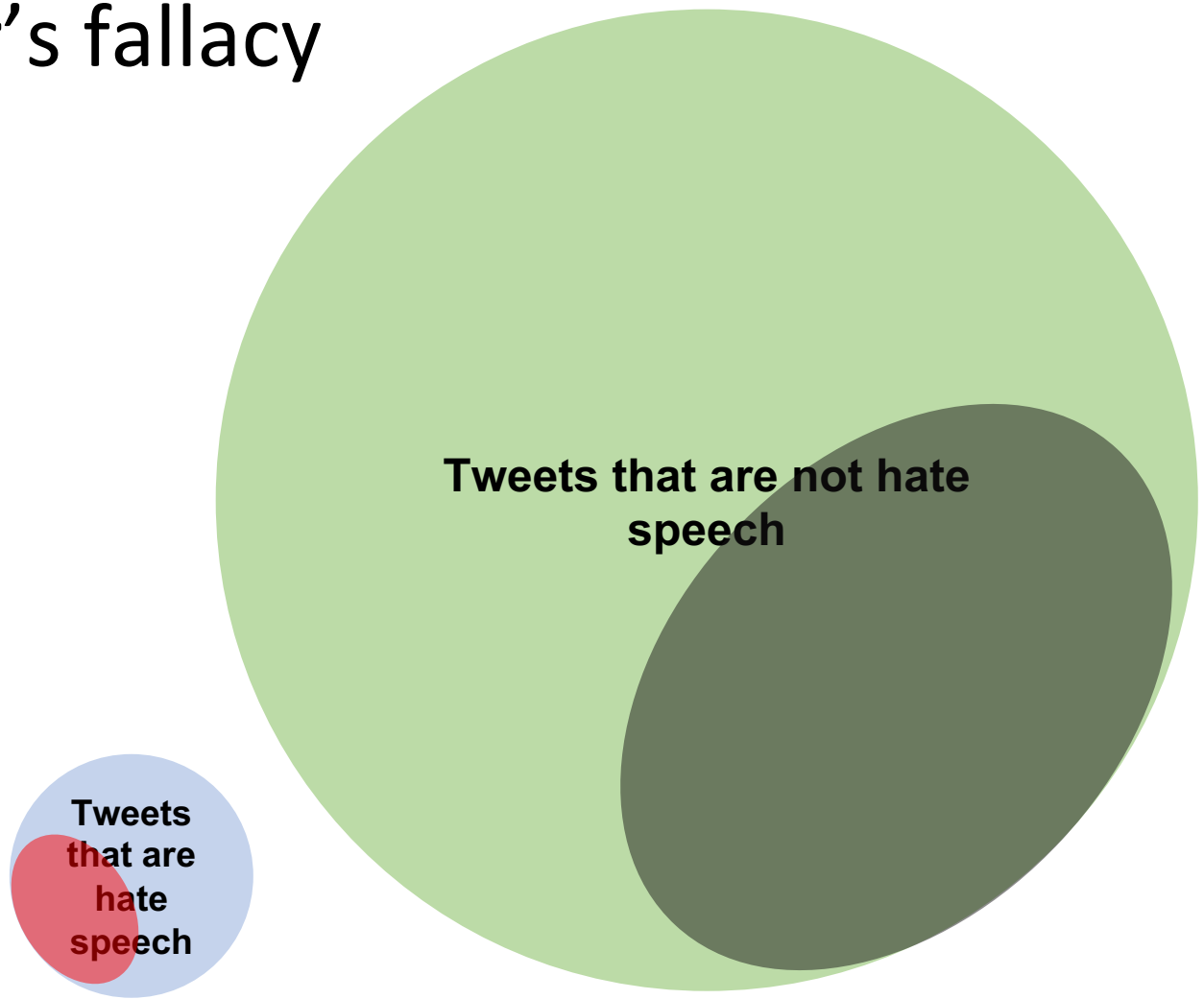
**Example**: Hate speech detection

**True Positives** — **Prediction: hate speech**

**False Negative** — **Prediction: not hate speech**

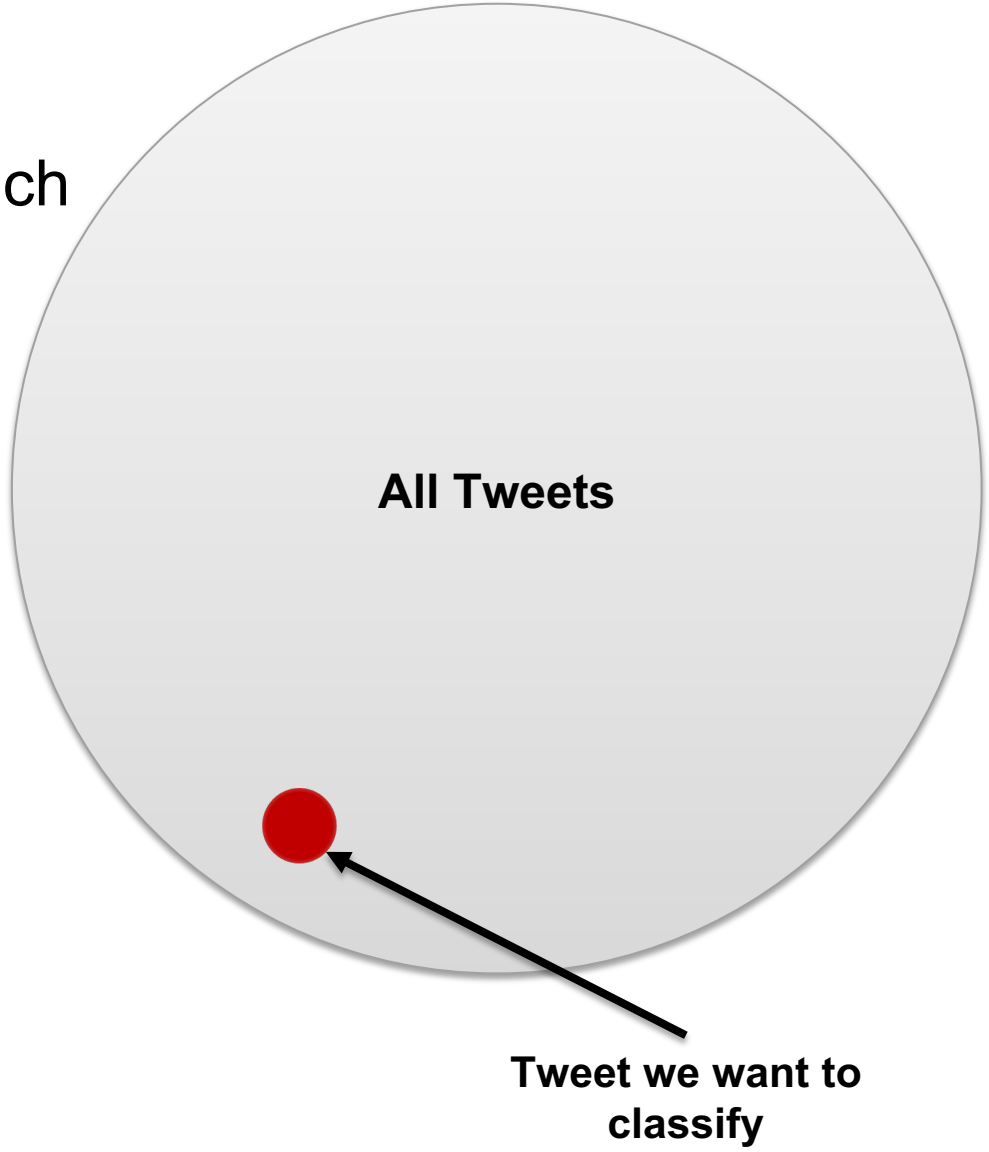
**True Negatives** — **Prediction: not hate speech**

**False Positives** — **Prediction: hate speech**



# Base Rate Fallacy / Prosecutor's fallacy

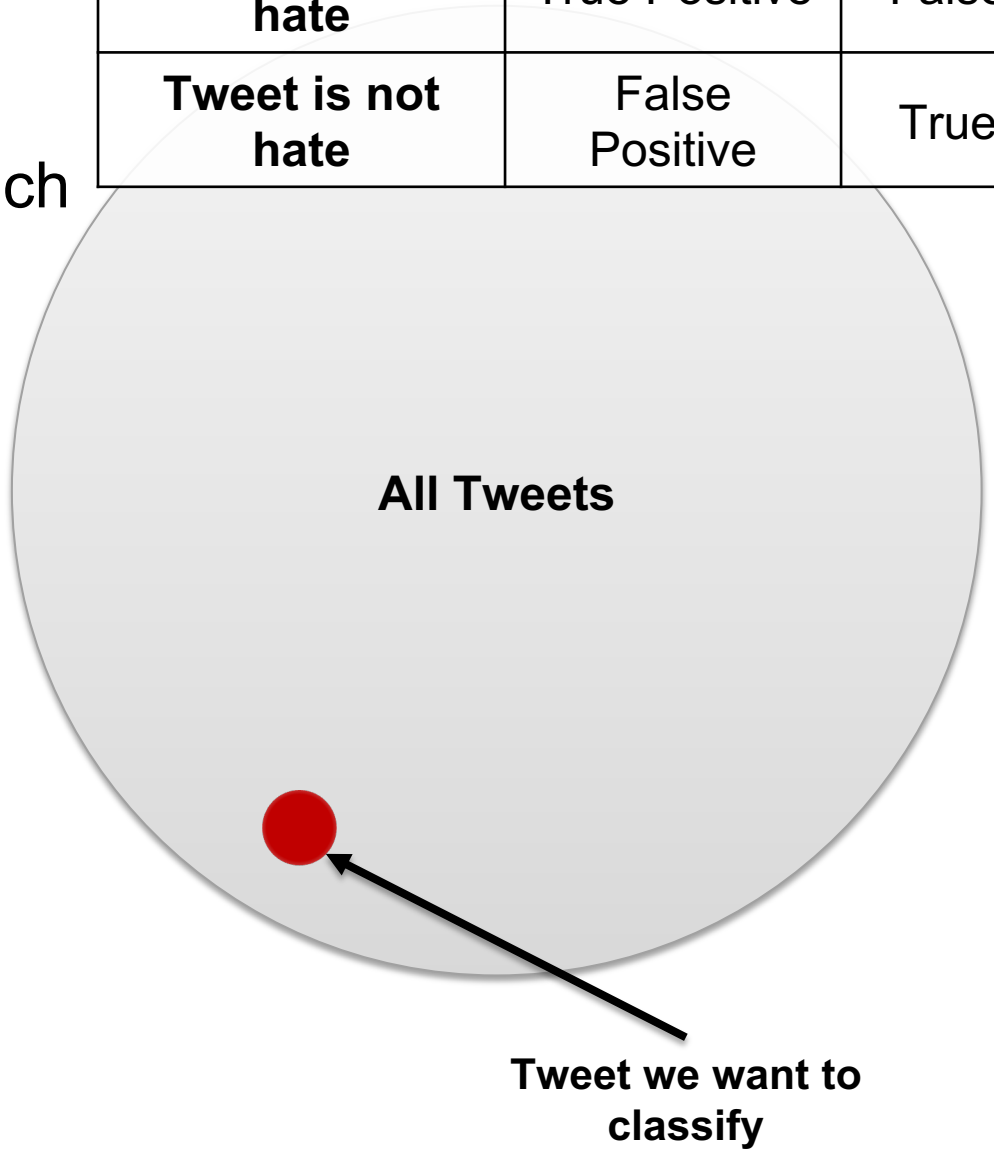
- **Example**: Predicting hate speech
- **"Reality"**: 1 tweet out of 1000 is hate speech



# Base Rate Fallacy / Prosecutor's fallacy

- **Example**: Predicting hate speech
- **"Reality"**: 1 tweet out of 1000 is hate speech

| | Classifier Says Hate | Classifier Says Not Hate |
|---|---|---|
| **Tweet actually hate** | True Positive | False Negative |
| **Tweet is not hate** | False Positive | True Negative |

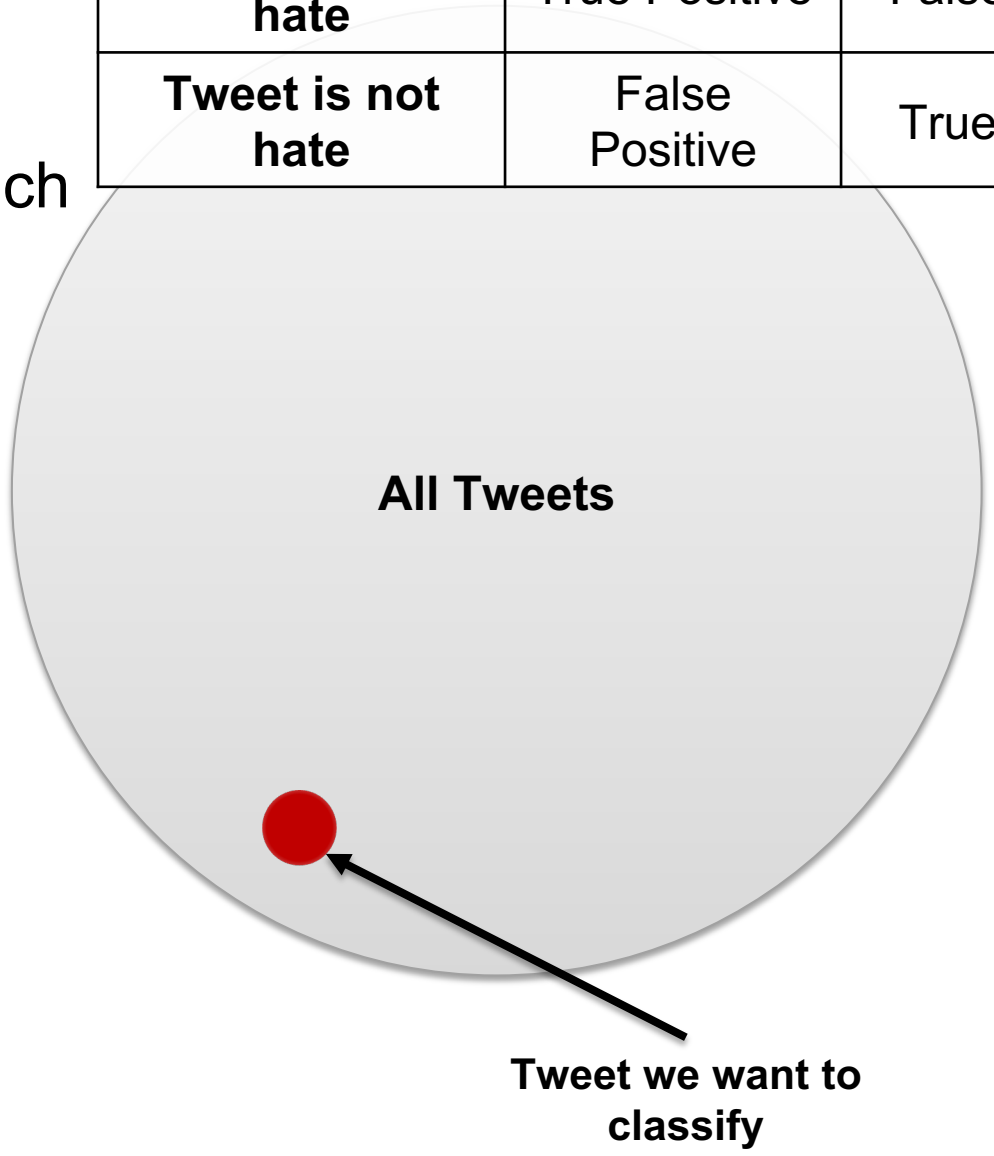


# Base Rate Fallacy / Prosecutor's fallacy

- **Example**: Predicting hate speech
- **"Reality"**: 1 tweet out of 1000 is hate speech

- **Our classifier**:
  - False Positive Rate of 5%
    - 5 out 100 times a non-hate tweet is said to be hate speech
  - True Positive Rate of 100%
    - **Zero** false negatives, never misses a hate speech tweet
    - If tweet is hate, the classifier will say YES

| | **Classifier Says Hate** | **Classifier Says Not Hate** |
|---|---|---|
| **Tweet actually hate** | True Positive | False Negative |
| **Tweet is not hate** | False Positive | True Negative |

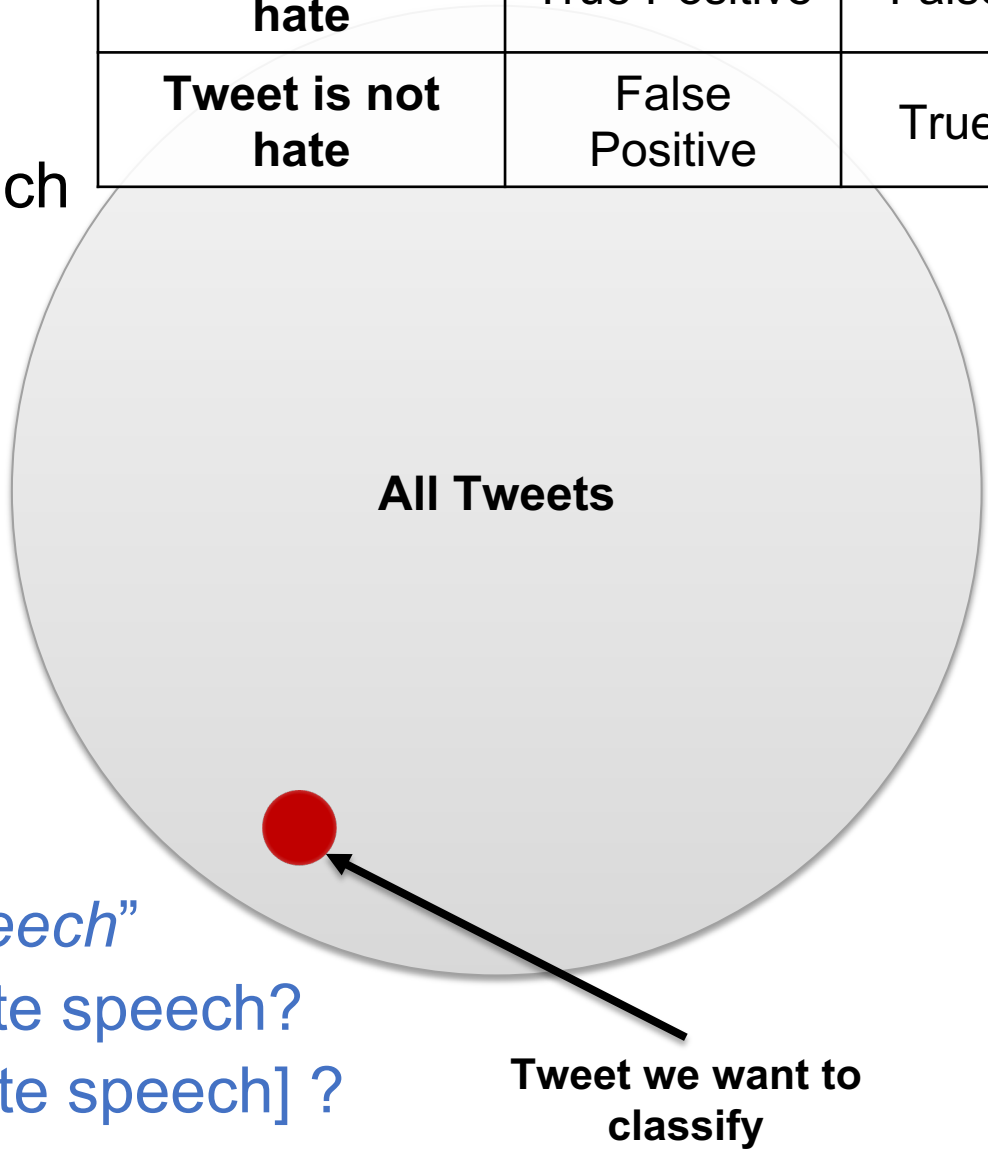


# Base Rate Fallacy / Prosecutor's fallacy

| | Classifier Says Hate | Classifier Says Not Hate |
|---|---|---|
| **Tweet actually hate** | True Positive | False Negative |
| **Tweet is not hate** | False Positive | True Negative |

- **Example**: Predicting hate speech
- **"Reality"**: 1 tweet out of 1000 is hate speech

- **Our classifier**:
  - False Positive Rate of 5%
    - 5 out 100 times a non-hate tweet is said to be hate speech
  - True Positive Rate of 100%
    - **Zero** false negatives, never misses a hate speech tweet
    - If tweet is hate, the classifier will say YES




Given a tweet, the classifier says "*hate speech*"

What is the probability that it is actually hate speech?

Pr[Hate speech | Classifier Says hate speech] ?

# Base Rate Fallacy / Prosecutor's fallacy

$$P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)}$$

**Given a tweet, the classifier says "*hate speech*"**
**What is the probability that it is actually hate speech?**

- **Example**: Predicting hate speech
- **"Reality"**: 1 tweet out of 1000 is hate speech
- **Our classifier**:
  - False Positive Rate of 5%
    - 5 out 100 times a non-hate tweet is said to be hate speech
  - True Positive Rate of 100%
    - Zero false negatives, never misses a hate speech tweet
    - If the tweet is hate, the classifier will say YES

Pr[Hate | Classifier Says Hate] =

= Pr[Classifier Says Hate | Hate] Pr[Hate] / Pr[Classifier Says Hate]

Pr[Classifier Says Hate | Hate] Pr[Hate]

+

Pr[Classifier Says Hate | not Hate] Pr[Not Hate]

(1- Pr[Hate])

= 1 * 0.001 / (1 * 0.001 + 0.05 * 0.999) = 0.001 / 0.05095 = **2%**

# Base Rate Fallacy / Prosecutor's fallacy

**Given a tweet, the classifier says "*hate speech*"**
**What is the probability that they actually will?**

- **Example**: Predicting hate speech
- **"Reality"**: 1 tweet out of 1000 is hate speech
- **Our classifier**:
  - False P
    - 5 out
      spee
  - True Po
    - Zero
      twee
    - If the

Pr[Hate |

= Pr[Clas

= 1 * 0.00

Hate]

Pr[Hate])

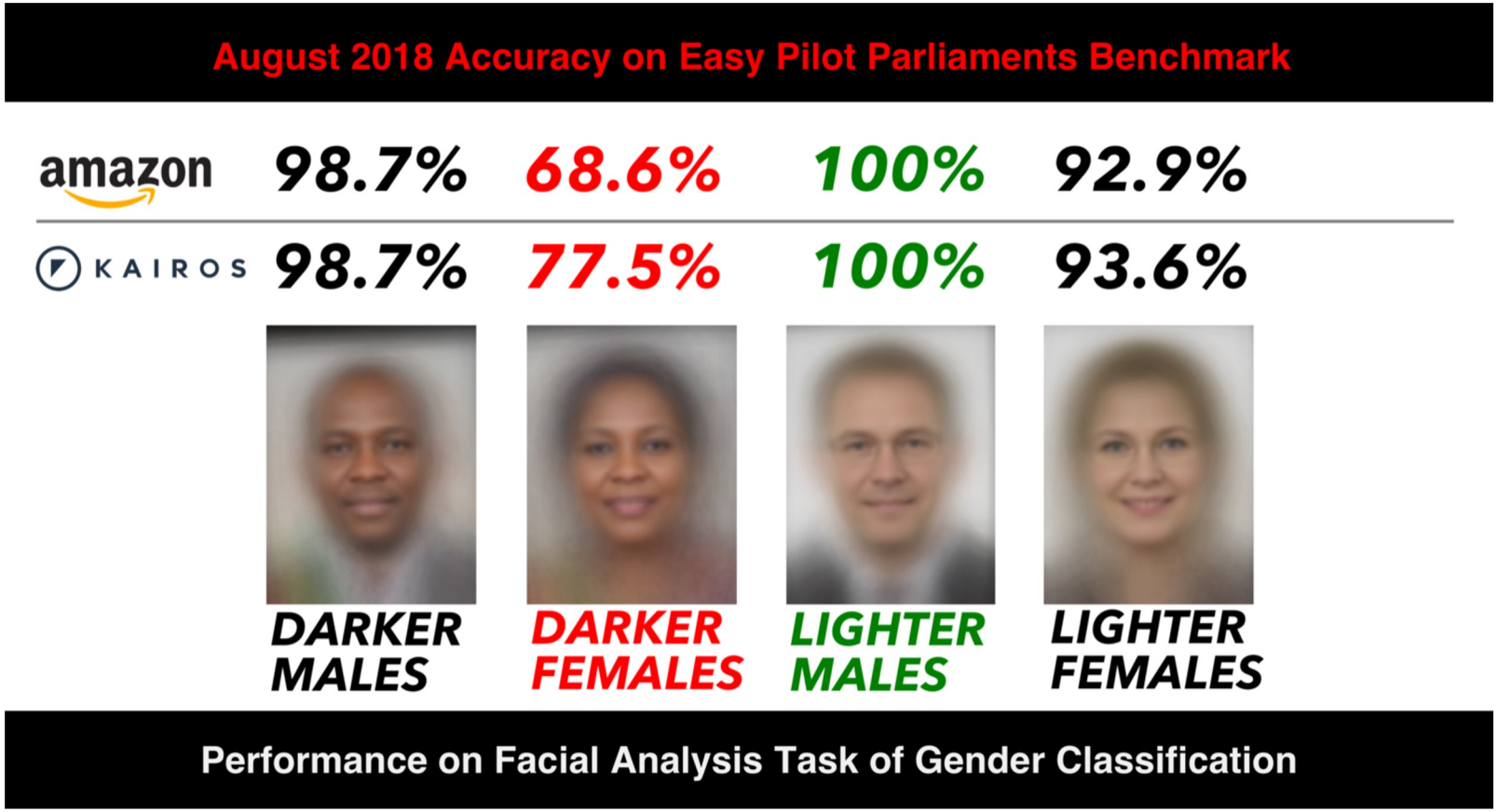
**Take Away**

*The result of the classifier must be taken with a grain of salt, where the salt is the base rate for the event we are trying to predict/classify*

*Events whose frequency is average may be fine, but rare events (i.e., very weak signals) are hard to find*

*See also https://en.wikipedia.org/wiki/Prosecutor%27s_fallacy*

# Distributional Shift

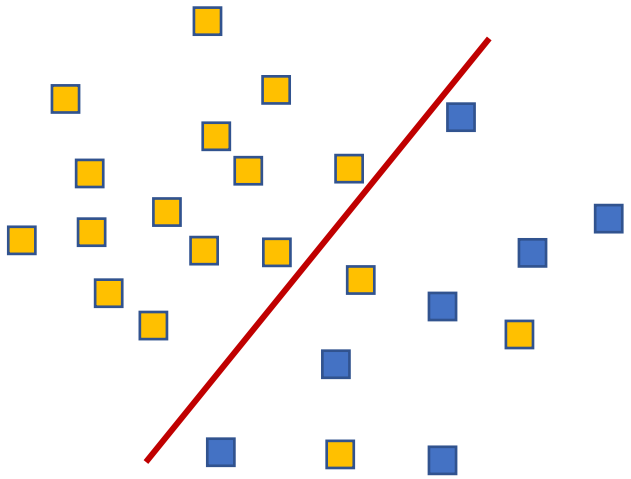



On Recent Research Auditing Commercial Facial Analysis Technology
https://medium.com/@bu64dcjrytwitb8/on-recent-research-auditing-commercial-facial-analysis-technology-19148bda1832
Response: Racial and Gender bias in Amazon Rekognition — Commercial AI System for Analyzing Faces.
https://medium.com/@Joy.Buolamwini/response-racial-and-gender-bias-in-amazon-rekognition-commercial-ai-system-for-analyzing-faces-a289222eeced

# Distributional Shift

**Wide Definition**
*Face recognition*: race / gender
*Migration prediction*: countries
*Behavior prediction*: social network
...

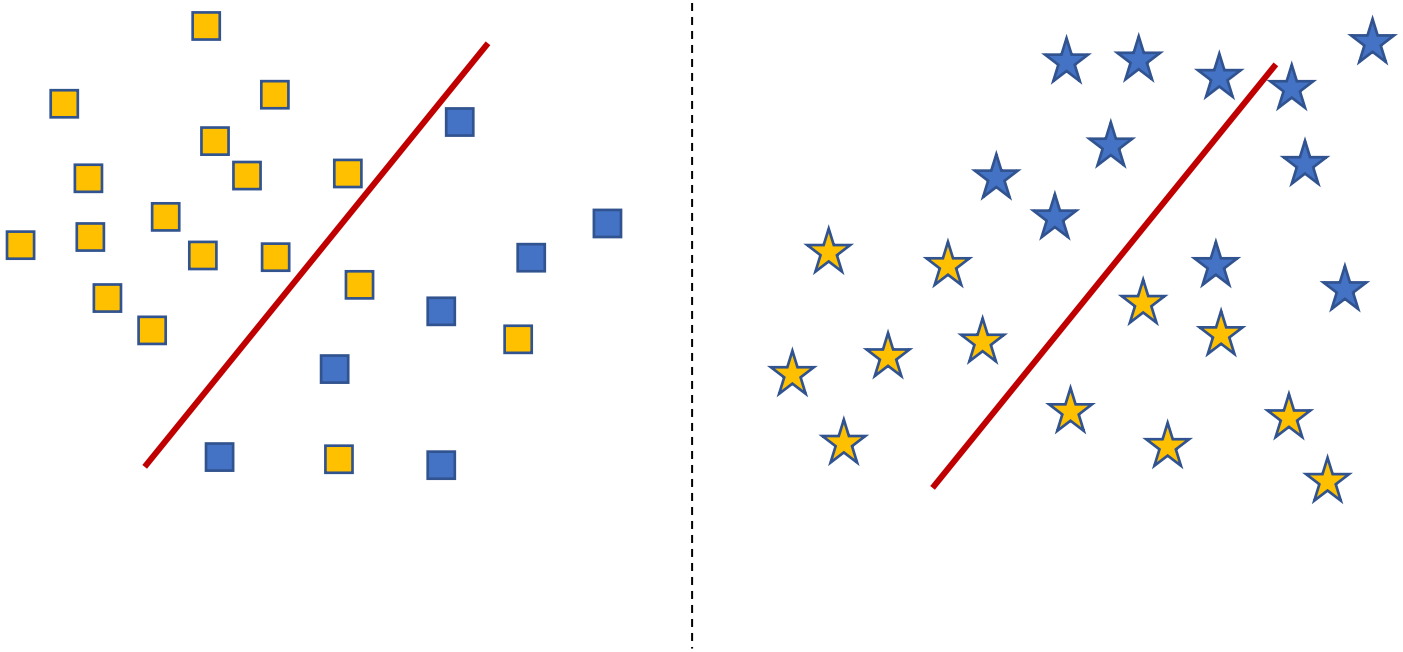
Occurs when a classifier is trained in one area and deployed in another.



- Example: Hate speech prediction
- Training/Test in United States
- Yellow = Tweets that are hate speech
- Blue = Tweets that are not hate speech

# Distributional Shift

Occurs when a classifier is trained in one area and deployed in another.



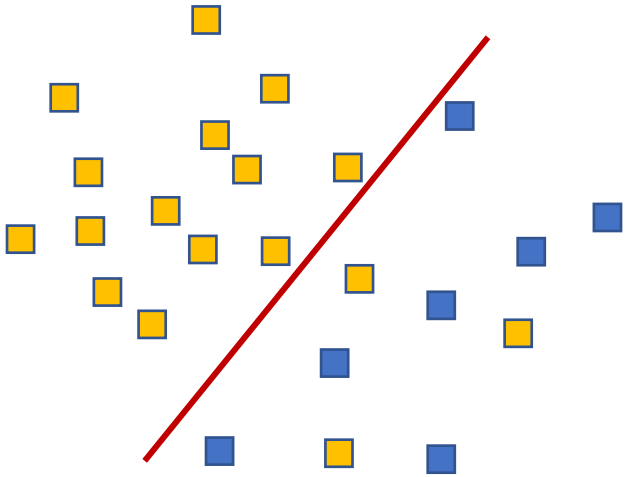
- Example: Hate speech prediction
- Yellow = Tweets that are hate speech
- Blue = Tweets that are not hate speech

Training/Test in US

Training/Test in UK

# Distribution of Errors

Occurs when most mistakes of the classifier are concentrated in a **subpopulation/group**



- Example: Hate speech prediction
- Training/Test in United States
- Yellow = Tweets that are hate speech
- Blue = Tweets that are not hate speech

# Transparency: correlation or causation?

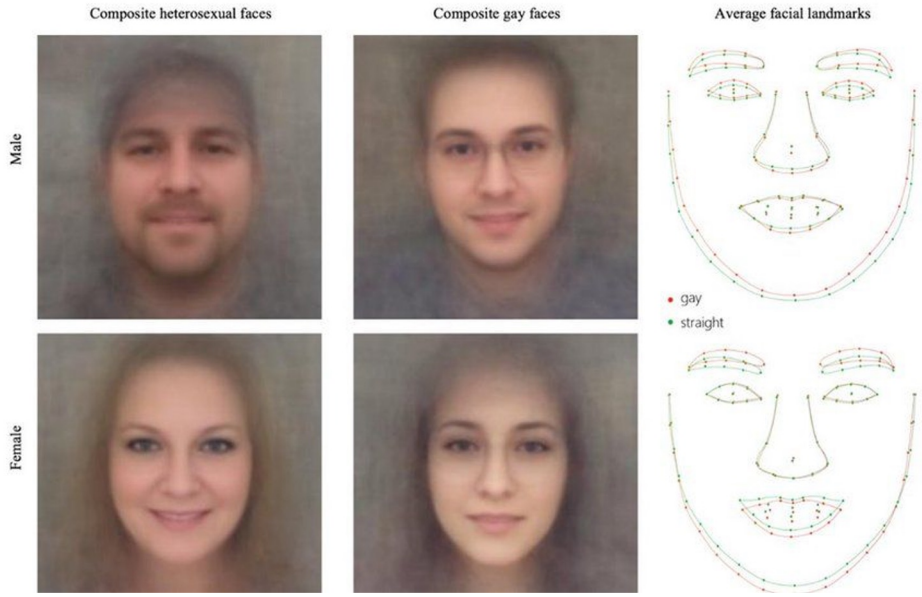
- Classification Task: Should we send home a patient with Bronchitis?
- Rule-Based Learning
  - If x, then y
  - Human readable rules: causation is intrinsic
- Machine Learning
  - Better accuracy
  - Not always possible to understand why a decision is made -> **Explainability**


https://www.umutozel.com/explainable-ai

# Controversial ML research

One example of “what is it learning?”

**When the algorithm was presented with two photos where one picture was definitely of a gay man and the other heterosexual, it was able to determine which was which 81% of the time. With women, the figure was 71%**




**What did the algorithm learn?**

**Faces or stereotypical poses/gestures in the Dating site & Facebook pictures used for training?**

**Would it work in other social networks?**

**Does it work evenly for different races?**

**And for different social groups?**

**“A facial recognition experiment that claims to be able to distinguish between gay and heterosexual people has sparked a row between its creators and two leading LGBT rights groups.”**

https://www.bbc.co.uk/news/amp/technology-41188560
https://medium.com/@blaisea/do-algorithms-reveal-sexual-orientation-or-just-expose-our-stereotypes-d998fafdf477

# Yet another problem… Bias!

# What is Bias?

**Statistical bias**: difference between an estimator expected value and the true value

Very limited! Nothing about errors, nothing about distributional shift

**Group fairness**: outcome should not differ between demographic groups

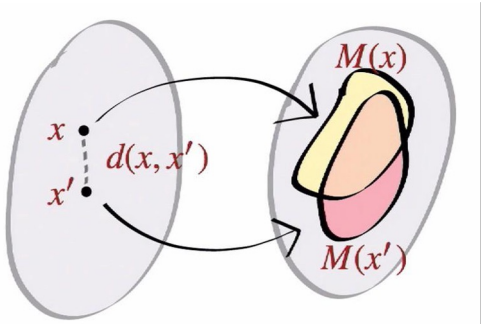
*Predictive parity*: same prediction regardless of group (aka, Calibration)

*Equal false positive* (rates)

*Equal false negative* (rates)

…

**Individual fairness**: **similar?** individuals should be treated **similarly?**




Arvind Narayanan: **Tutorial: 21 fairness definitions and their politics**
https://www.youtube.com/watch?v=jIXIuYdnyyk

# Bias detection and mitigation

**What-if approach**: play with the values until something changes, associate with bias
Google What-If

**Explainability**: try to understand why the prediction happen, associate with bias
FairML: https://github.com/adebayoj/fairml
Lime: https://github.com/marcotcr/lime

**Mitigation** IBM Bias Assessment Toolkit (metrics, some detectors, nice references and tutorials)
https://github.com/Trusted-AI/AIF360

Key questions:

How do you know you explored the full space?

What about “proxies” (attributes associated to sensitive attributes)?

What about biases outside of your system?

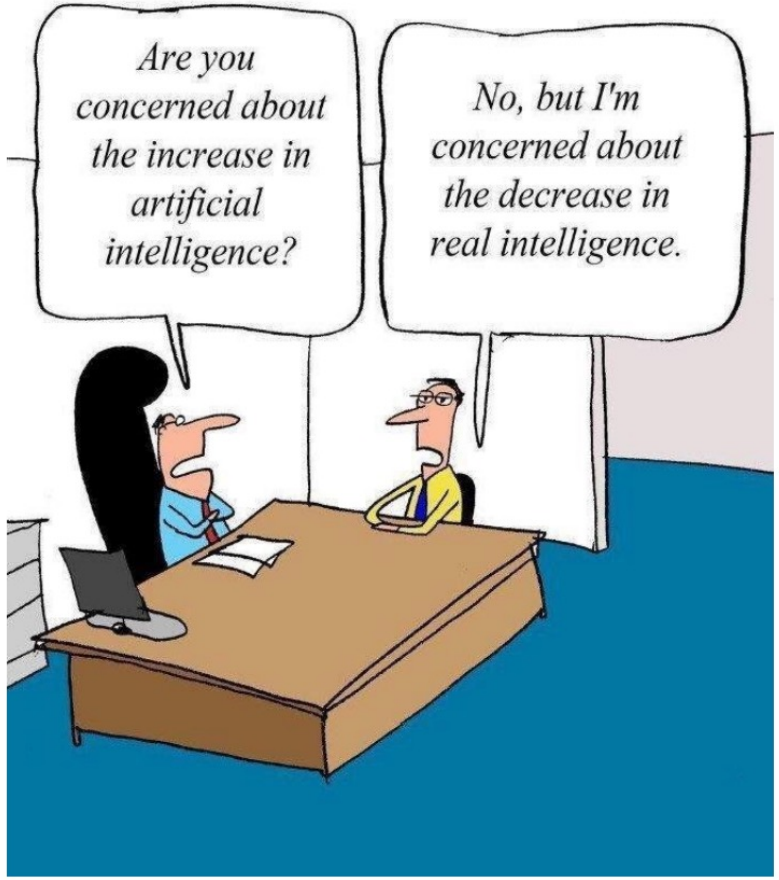
What about inconsistencies/contradictions between metrics?

# Takeaways on biases and fallacies

Deploying machine learning is hard: reality is far from lab conditions

Base rate matters: it is always hard to get good results on weak signals

Biases come in many flavors, it is hard to remove




https://cdn.bdigital.org/PDF/BDC18/BDC18_ExplainableAI.pdf

# Machine Learning – Security and Privacy (II)

- *Basics*
- *Model stealing*
- *Privacy issues*

- Altering the output

- Biases and fallacies

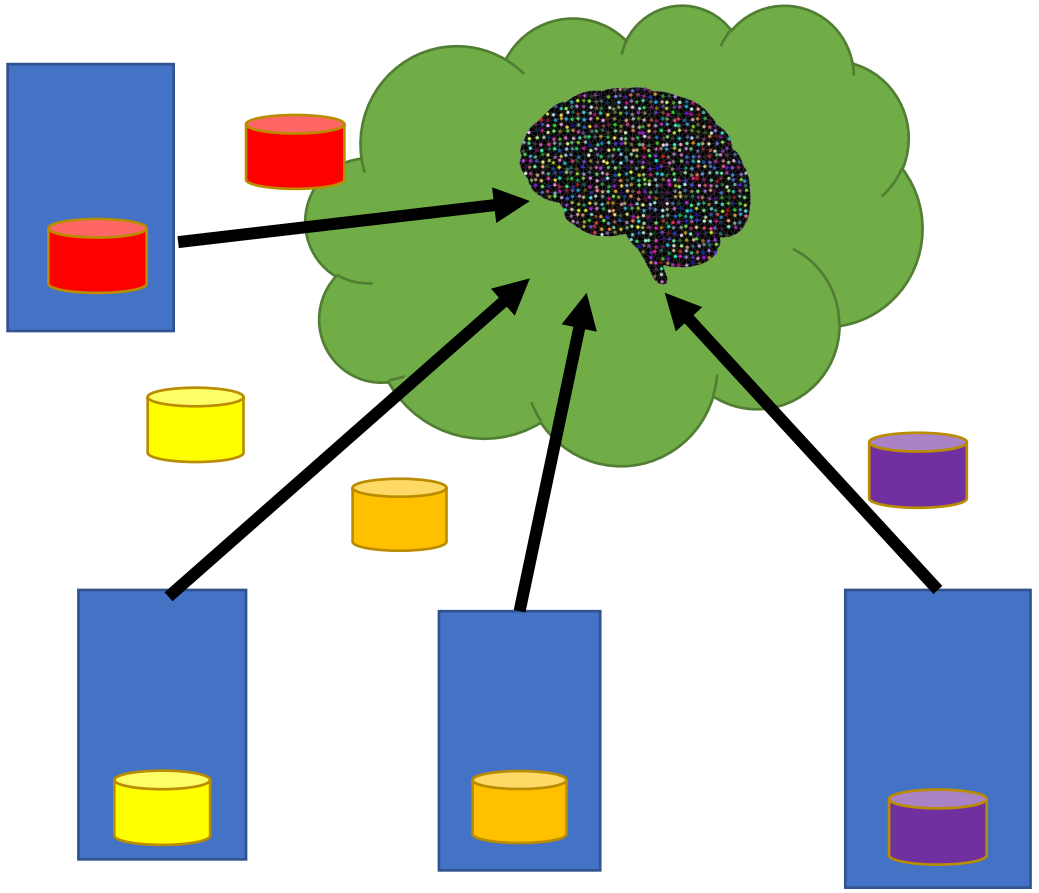
- **Federated learning**

**Collaborative learning:** multiple entities, such as individuals, organizations, or study centers, work together to build the machine learning models.

E.g., hospitals

Motivation for collaborative learning…

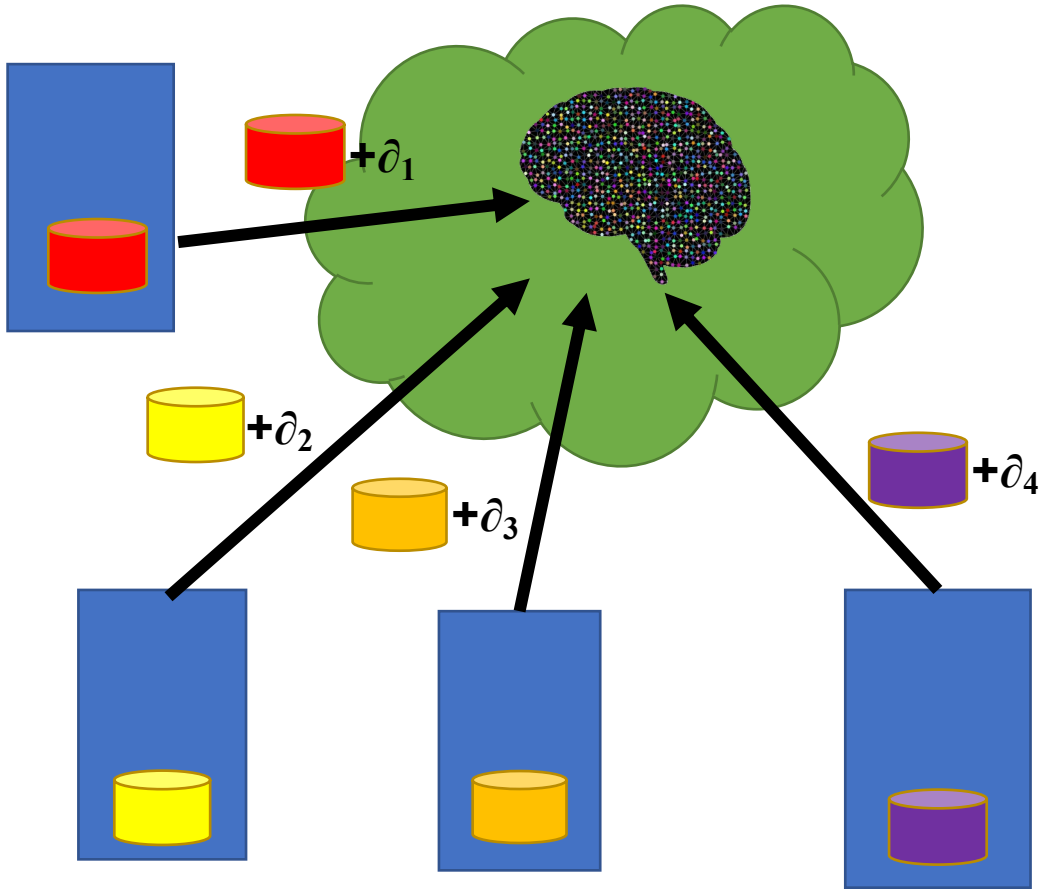
# Learning without privacy (traditional centralized approach)



**SHORTCOMINGS**

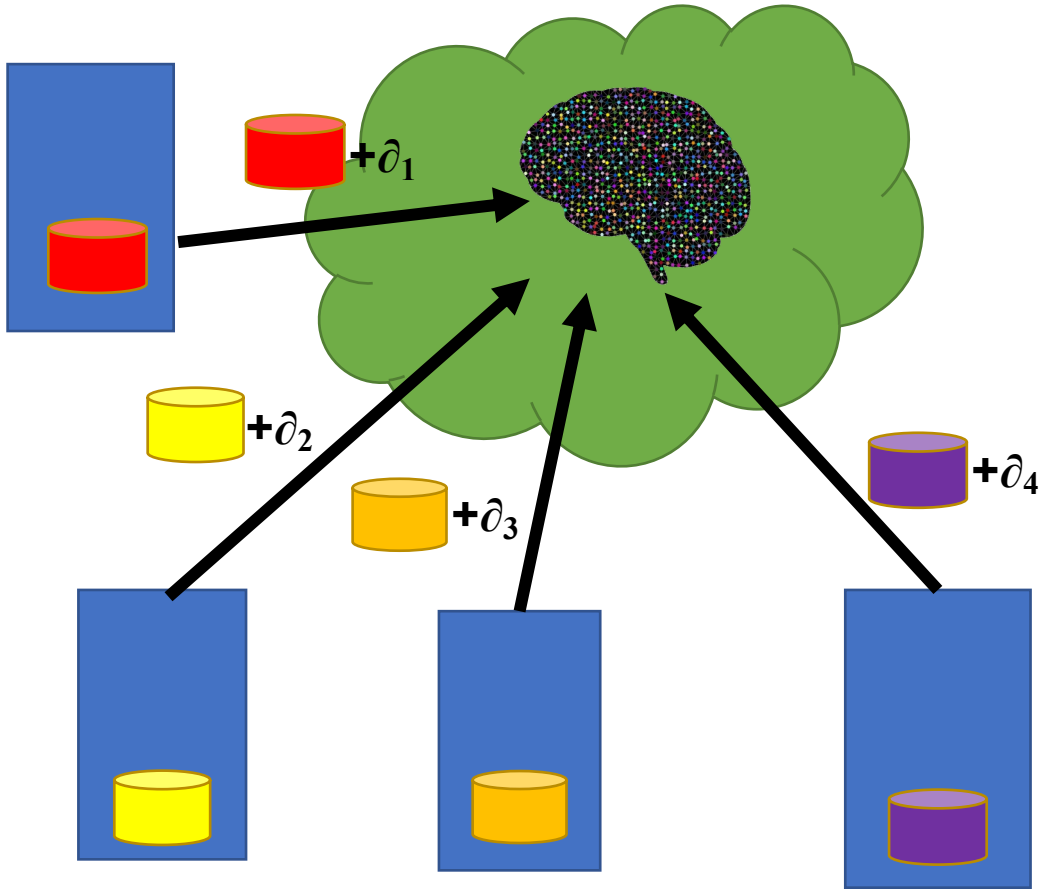
- **Privacy protection/regulations**
  - **GDPR, HIPAA…**
- **Data protection**
  - **Security of the central database**
  - **Need to trust the central server**
- Need to interface **several** ethics committees

# Learning with privacy




- Instead of sending their data directly, clients send data with **Differentially private** data
  - Given the sample, one cannot learn the value
- Challenge: add enough noise to hide the data but still provide a good model

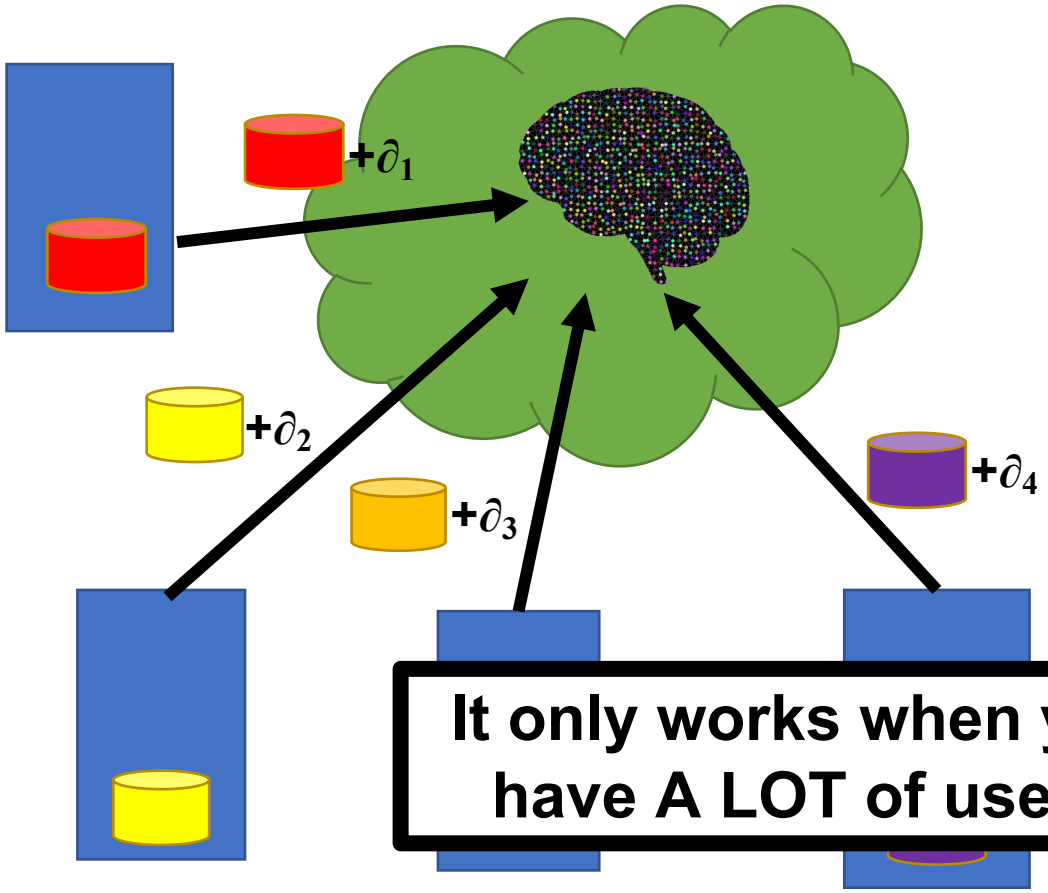
# Learning with privacy




- Instead of sending their data directly, clients send data with Differentially private noise
  - Given the sample, one cannot learn the value
- Challenge: add enough noise to hide the data but still provide a good model

Google RAPPOR - Collect data from phones
Apple - Collect data from phones
Federated learning - Share models
Smart energy - Collect measurements

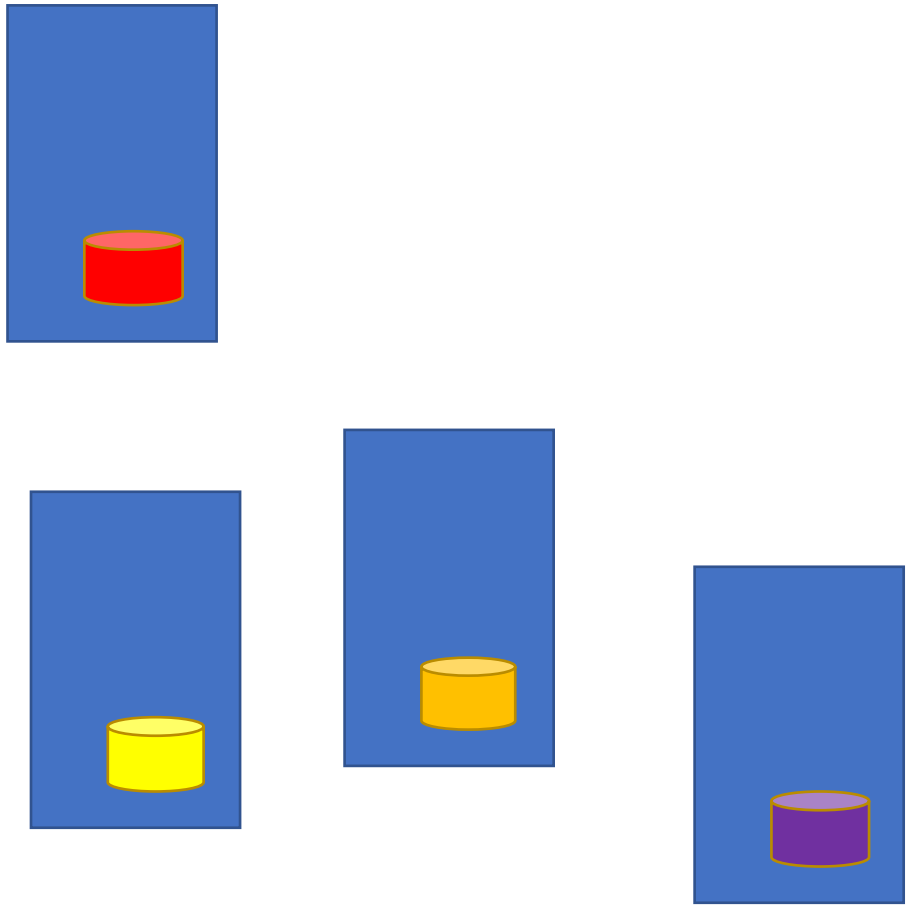
# Learning with privacy




- Instead of sending their data directly, clients send data with Differentially private noise
  - Given the sample, one cannot learn the value
- Enough noise to hide the data but still provide a good model

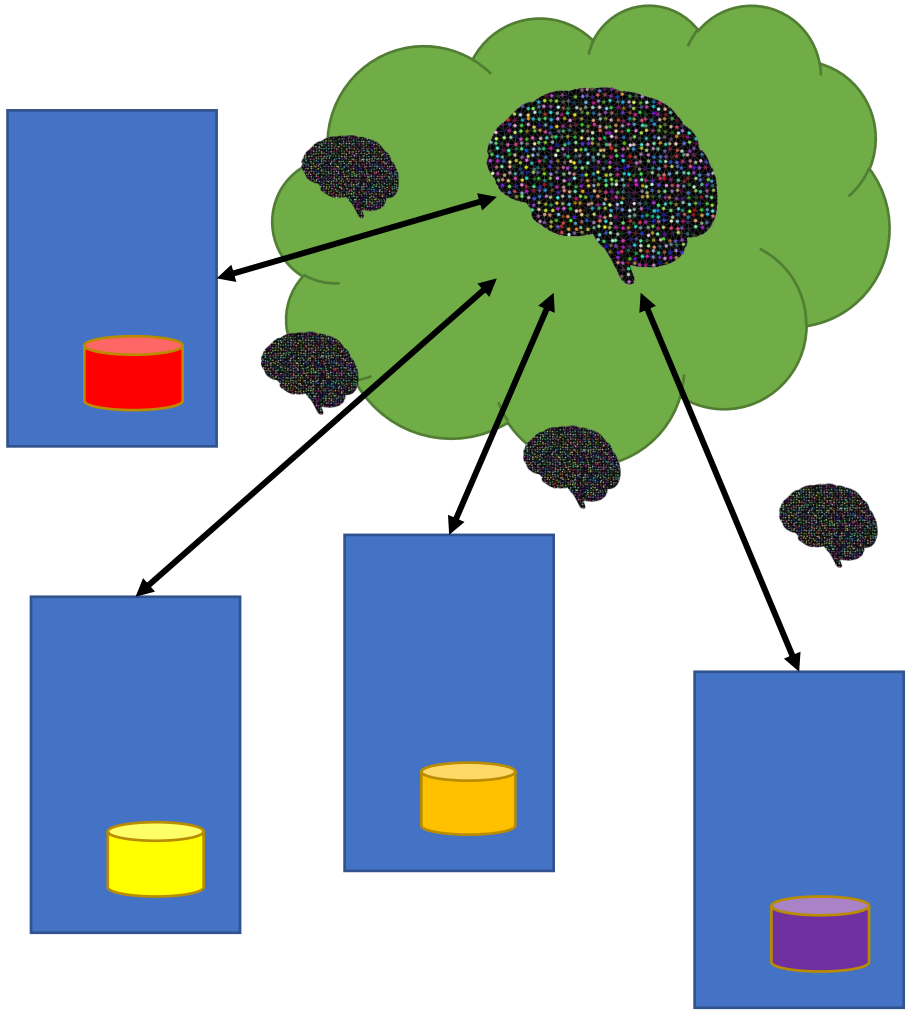
Google RAPPOR - Collect data from phones
Apple - Collect data from phones
Federated learning - Share models
Smart energy - Collect measurements

# Another option is to learn locally…



- For ML to be effective models need to be trained on a significant amount of data
- What if you do not have the data??

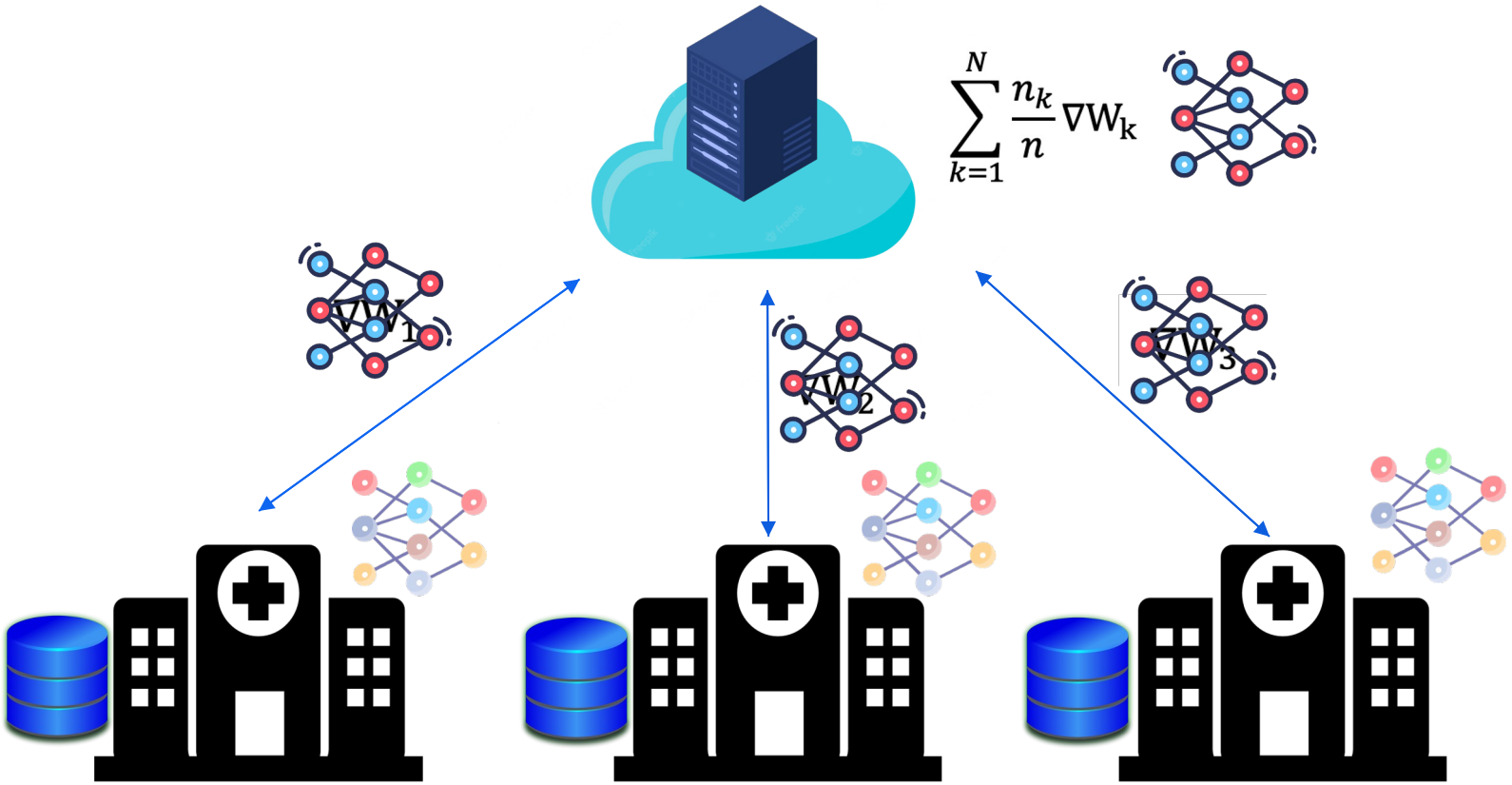
# Federated Learning – Learn as a team



**Idea**: combine small models from many small datasets to obtain the model

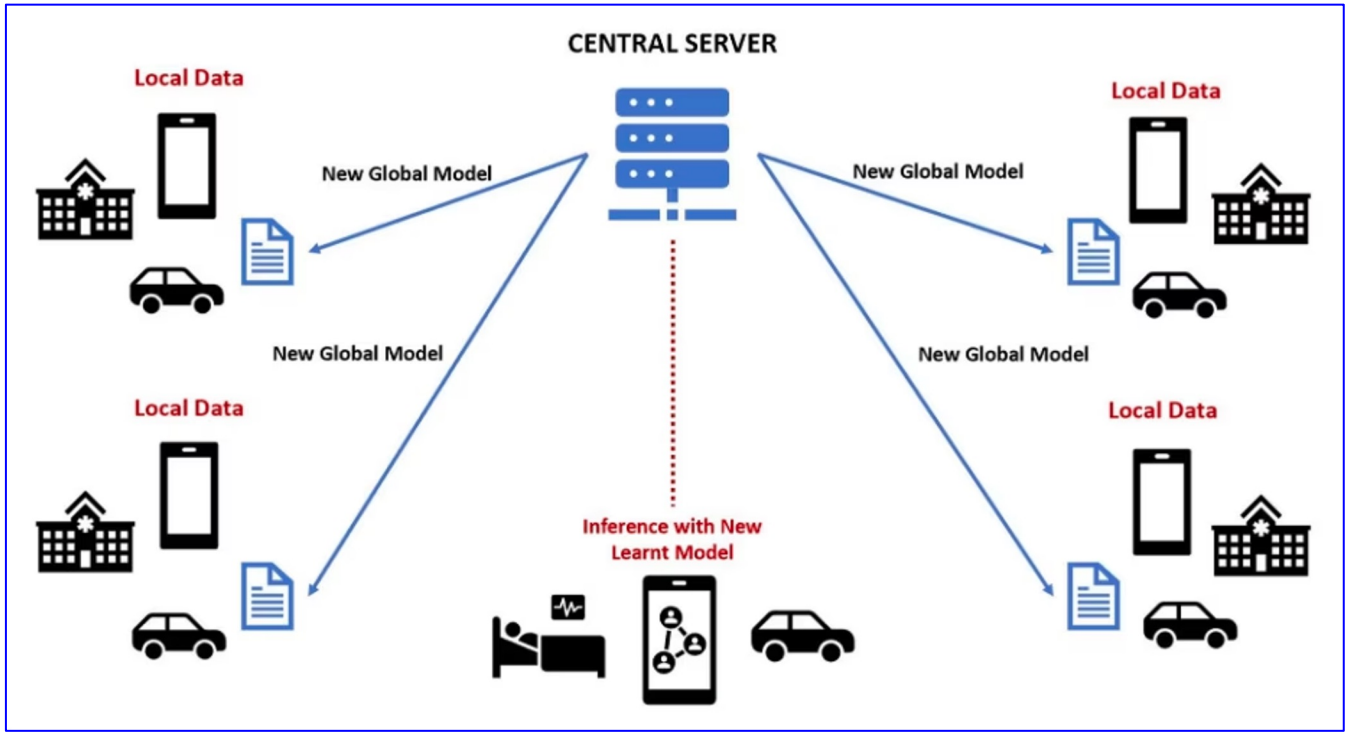
- Construct some model with weights w.
- Download this model to each client.
- Take some subset of your clients that are online, compute a updated, personalized set of weights *locally*
- Update model with the weighted average of all selected clients

# Federated Learning [1,2]




[1] J. Konecny, H. B. McMahan, D. Ramage, and P. Richtárik. Federated optimization: Distributed machine learning for on-device intelligence. CoRR, abs:1610.02527, 2016.
[2] J. Konecny, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon. Federated learning: Strategies for improving communication efficiency. CoRR, abs/1610.05492, 2016.

# Federated Learning [1,2]



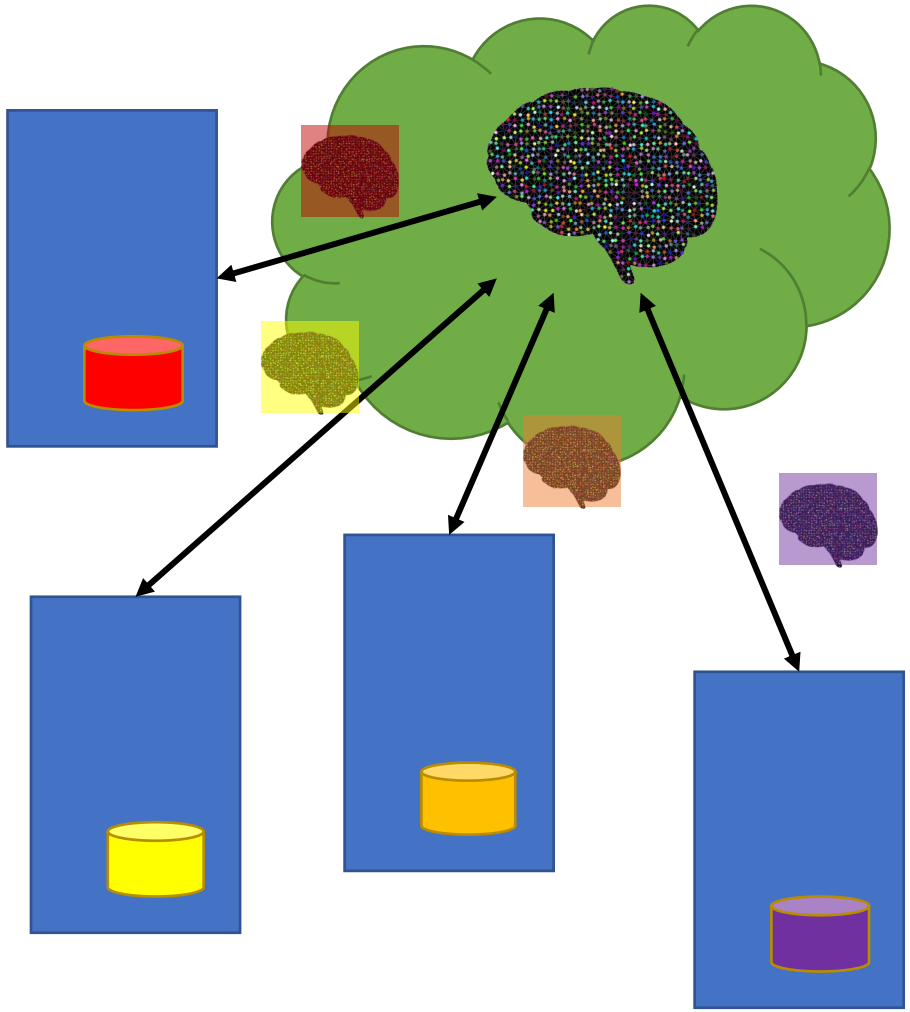

https://www.netapp.com/blog/future-of-AI-federated-learning/

1. Cross-device setting
2. Cross-silo setting

**Data protection: the output leaks nothing about the original data, right?**

Hmm… Are we sure?

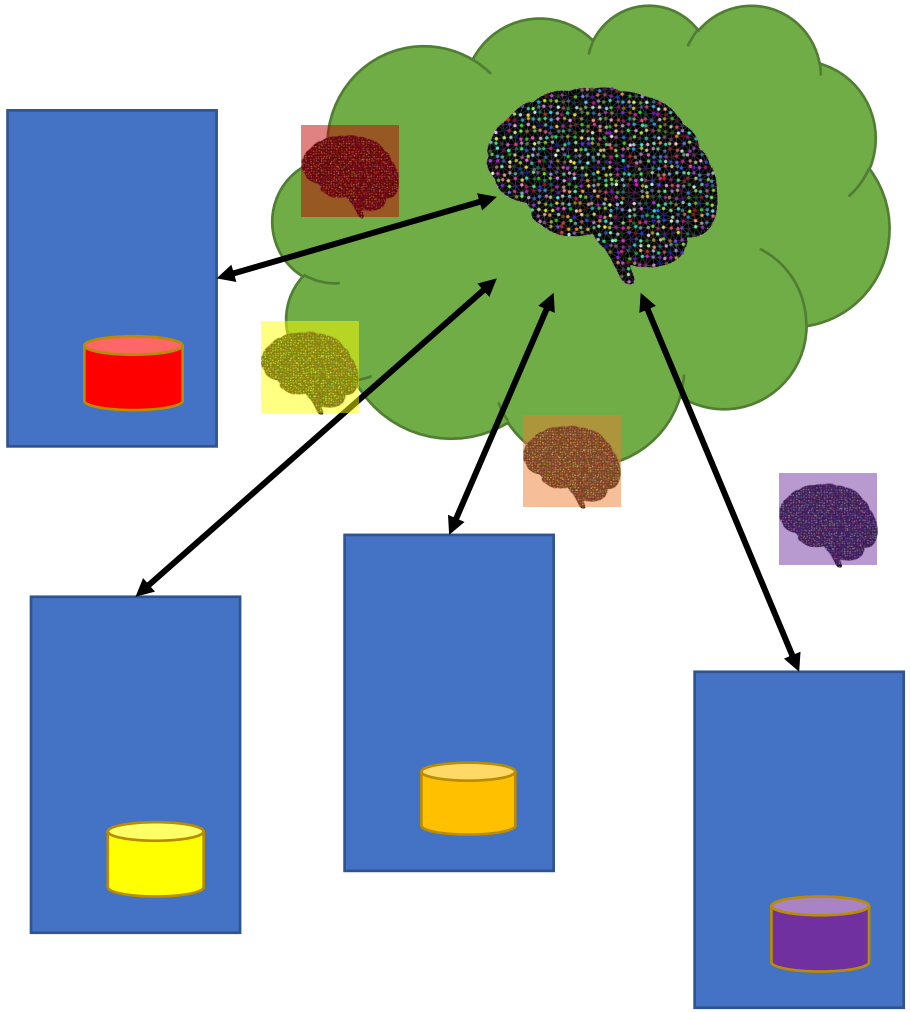
# Federated Learning – **Privacy problem?**



**Idea**: combine many small datasets to get a large dataset

- Construct some model with weights w.
- Download this model to each client.
- Take some subset of your clients that are online, compute an updated, personalized set of weights *locally*
- Update model with the weighted average of all selected clients

# Federated Learning – **Privacy problem?**



**Idea**: combine many small datasets to get a large dataset

- Construct some model with weights w.
- Download this model to each client.
- Take some subset of your clients that are online, compute an updated, personalized set of weights *locally*
- Update model with the weighted average of all selected clients

**Clients need to reveal their models!**
**That "reveals" their data!**

# Federated Learning Leaks

RESEARCH-ARTICLE

mation Leakage from

Beyond Inferring Class Representatives: User-Level Privacy Leakage From Federated Learning

April 2019

DOI: 10.1109/INFOCOM.20

Conference: IEEE INFOCOM

Zhibo Wang · Song Men

Deep L

Part of Ac

AuthorFe

Author

*Ligeng Zhu, Zhijian Liu, Song Han*

Cruz Authors Info & Affiliations

y • October 2017

"Parameters can also be obfuscated via differential privacy (DP) to make information extraction even more challenging, as proposed by Shokri and Shmatikov at CCS'15.
Unfortunately, we show that any privacy-preserving collaborative deep learning is susceptible to a powerful attack that we devise in this paper."

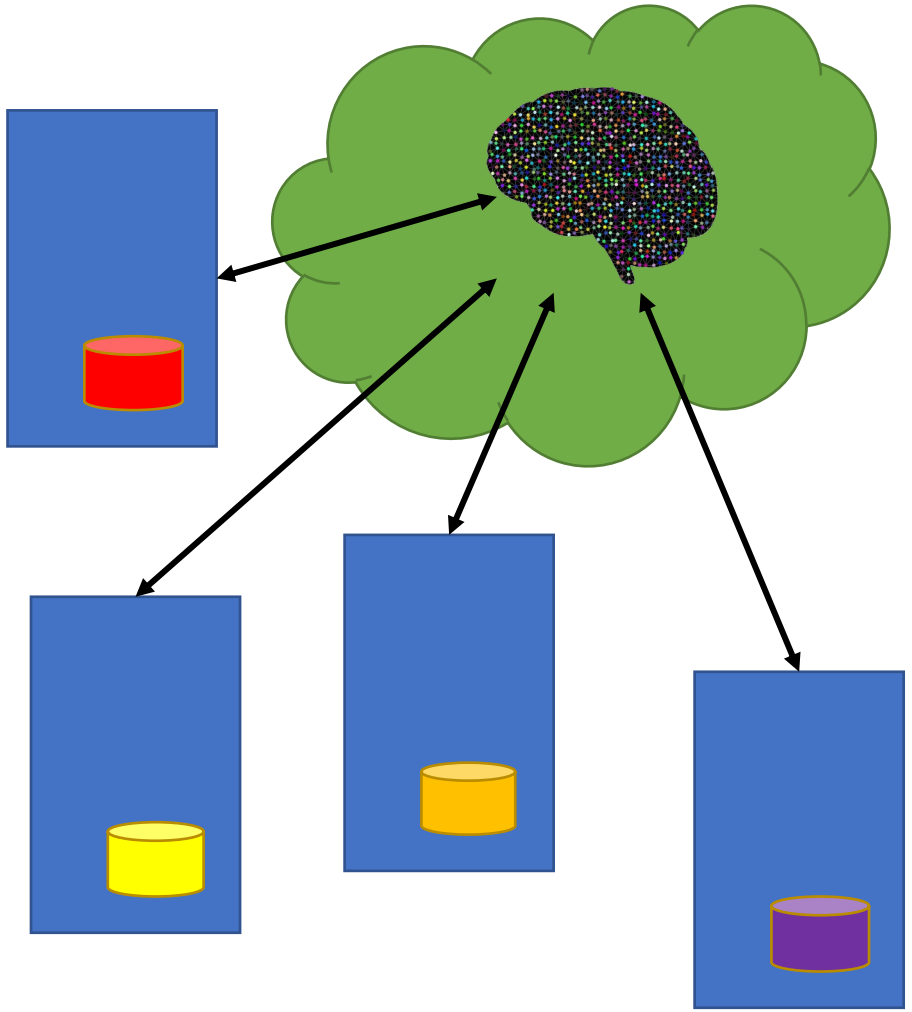
## Abstract

Passing gradient is a widely used scheme in
to share: i.e, the training set will not be leake
only takes few gradient steps to process and
validate the effectiveness of our algorithm o
and thereby and raise people's awareness to

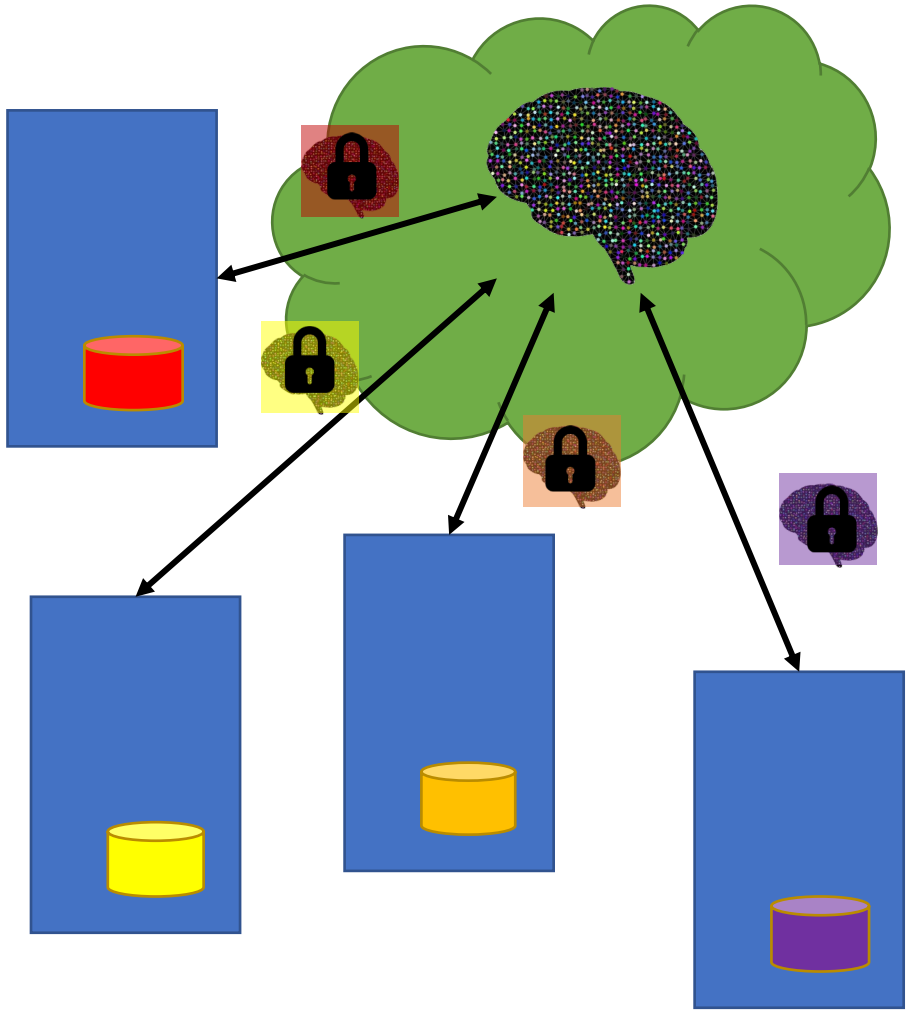
# Information Leaks in Federated Learning

Anastasia Pustozerova* and Rudolf Mayer*
{apustozerova,rmayer}@sba-research.org
*SBA Research, Vienna, Austria

# Federated Learning – **Solutions?**



# Federated Learning – Solutions?



**Option 1**: Encrypt!

**1.A:** Homomorphic encryption

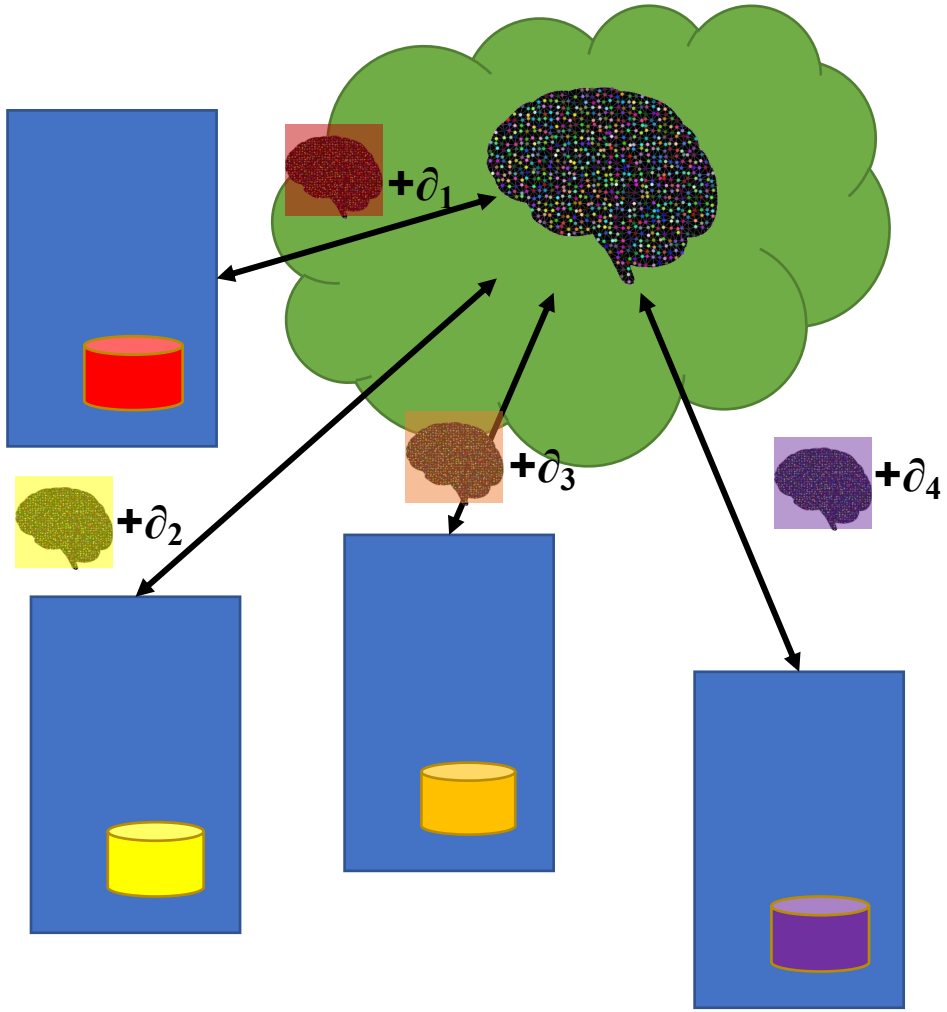
**1.B**: Multi party computation

All parties blind their input

Blinding factors cancel

Bonawitz, K., Ivanov, V., Kreuter, B., Marcedone, A., McMahan, H. B., Patel, S., ... & Seth, K. Practical secure aggregation for privacy-preserving machine learning. CCS 2017.

# Federated Learning – Solutions?




**Option 2**: Differential privacy

Before sending models, add noise

Actually, add noise to the gradient

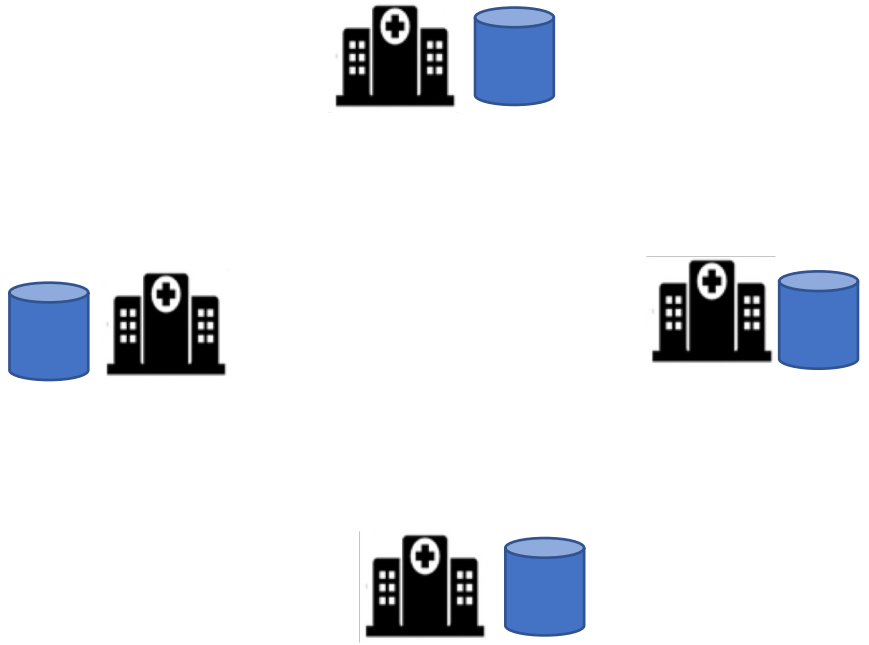
Tradeoff privacy vs functionality

Some attacks still possible

Shokri, R., & Shmatikov, V. Privacy-preserving deep learning. CCS 2015
L.Melis, C.Song, E. De Cristofaro, V.Shmatikov. Exploiting Unintended Feature Leakage in Collaborative Learning S&P 2019

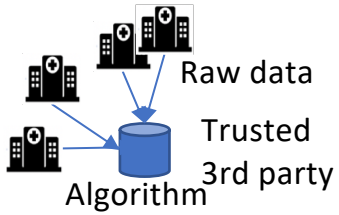
# Multi-site studies (cross-silo collaborative learning)

- Benefit: increase the amount of available data and thus the statistical significance of findings



# Multi-site clinical studies – Current approaches

## (a) Fully centralized



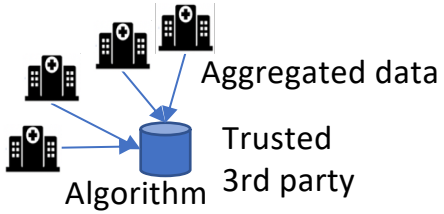

All of Us
EGA
Genomics England

**Data Access Agreement**

Lengthy, Costly

- Transfer raw data to a central database
- Data protection: security of the central database
- Need to trust the central server

## (b) Meta-analysis




https://covidclinical.net/

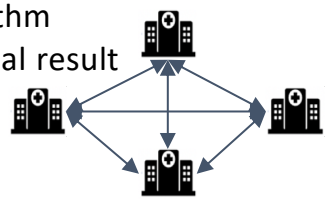
**Data Access Agreement**

Lengthy, Costly

- For each study, aggregated data provided by each site
- Still need to trust the central server

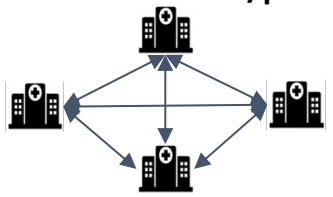
## (c) Decentralized

Algorithm
+ partial result



## (d) Secure Decentralized

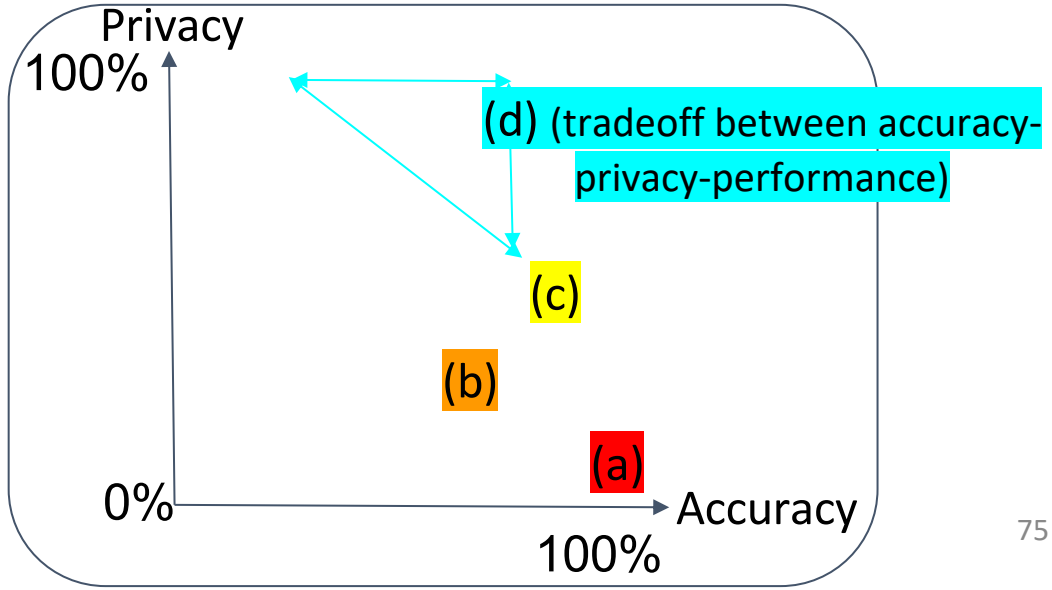
Algorithm + **obfuscated/protected** partial result



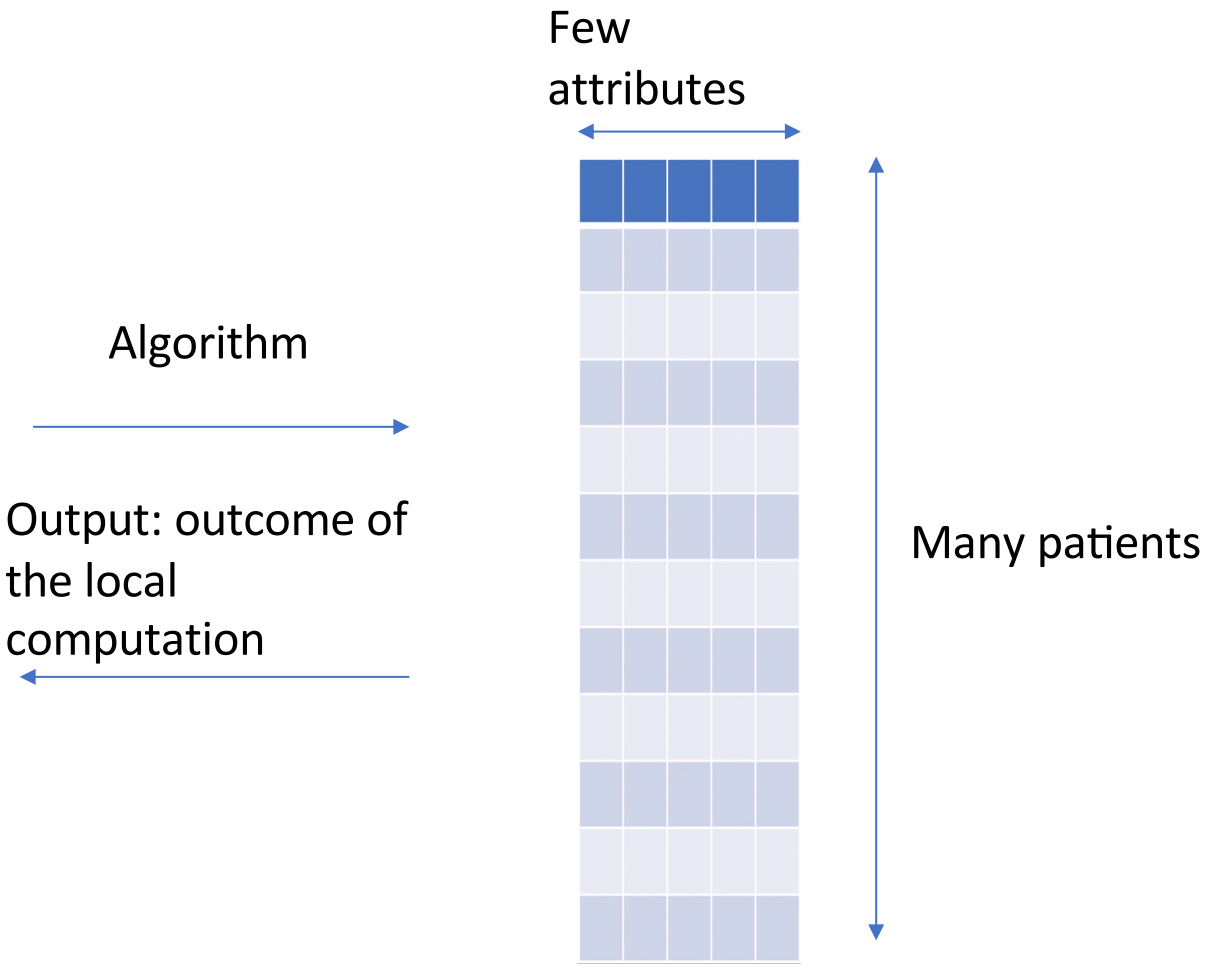
- "Send the algorithm to the data"

http://www.datashield.ac.uk
Personalized Health Train (PHT)

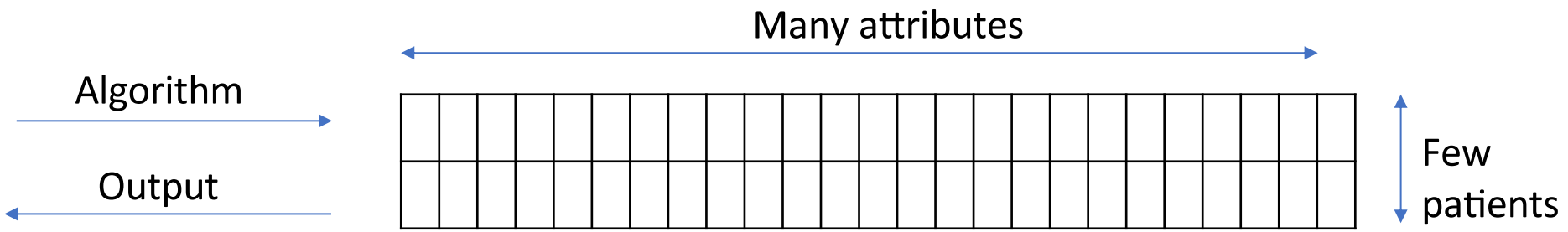
- M. Kim et al. "Secure and Differentially Private Logistic Regression for Horizontally Distributed Data," TIFS 2019
- SCOR (https://academic.oup.com/jamia/advance-article/doi/10.1093/jamia/ocaa172/5869802)



# “Send the algorithm to the data”



# “Send the algorithm to the data”




Technological evolution and medical progress will lead to more and more situations of this kind

→ **Patient data will not be protected anymore from the recipient of the output**
**Communicating aggregates is not privacy-preserving**

Typical examples:

- Rare diseases
- Stratification
- Presence of genetic data

Even if sites store their data with cloud providers, the issue remains

# State-of-the-art PPML Solutions

- **Multiparty Computation (MPC) - based solutions [1][2][3][4][5][6][7]**
  - Operate on the 2, 3, or 4 server models based on secret-sharing techniques
  - Limit the number of parties among which the trust is split
  - Assume an honest majority among the computing servers
  - Require parties to communicate (i.e., secret share) their data outside their premises
- **Differential Privacy (DP) - based solutions [8][9][10]**
  - Exchange differentially private intermediate values in between data owners.
  - Utility degradation
  - High privacy budgets
  - The practical level of privacy achieved in practice remains unclear
- **Federated Learning [11][12]**
  - Instead of bringing the data to the model, bring the model to the data for local model updates
  - The model is also sensitive
  - Sharing intermediate model updates leads to various privacy attacks, such as extracting parties' inputs [13][14][15] or membership inference [16][17].

# State-of-the-art PPML Solutions

- Questionable data protection guarantees
- Reliance on the manual work of ethics committees
- Slow, ad hoc procedures
- Not scalable with the number of parties
- The trust is split to a limited number of computing servers

# Example solution…

# Problem Definition

Attacker's model:

- Honest but curious data providers
- Malicious queriers (not addressed here)

Enable the training and evaluation of ML in a **federated** setting and provide end-to-end protection of:

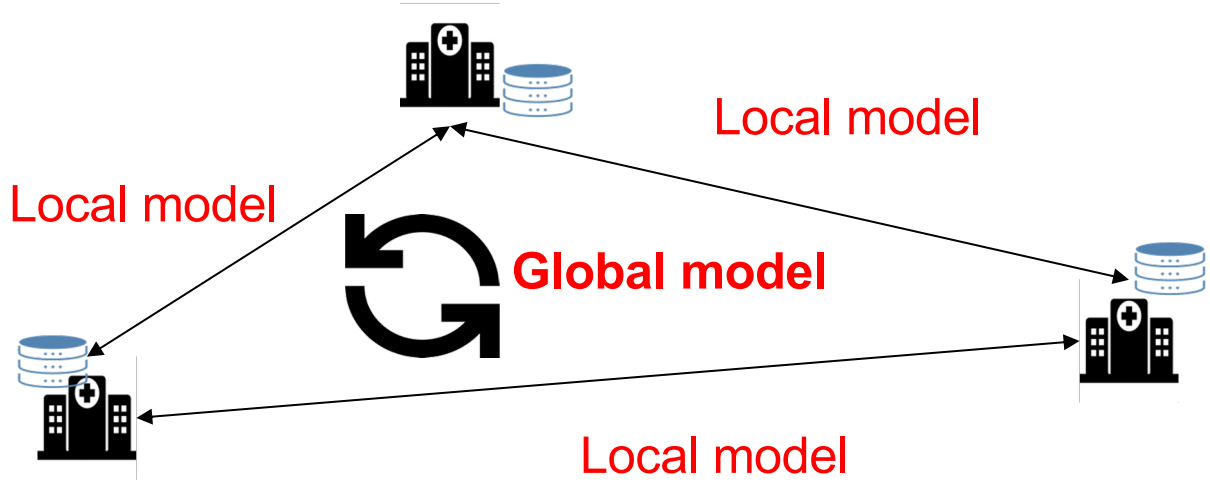
- the parties' **training data**,
- the **resulting model**, and
- the **querier's evaluation data**

# **Example Solution [1]** (homomorphic encryption based)

Relies on **federated learning** and **multiparty homomorphic encryption**

**- This solution (slides 58-85) is also an example for privacy-preserving system design**

- Local data is always kept in the premises
- The ML model weights are always kept encrypted!



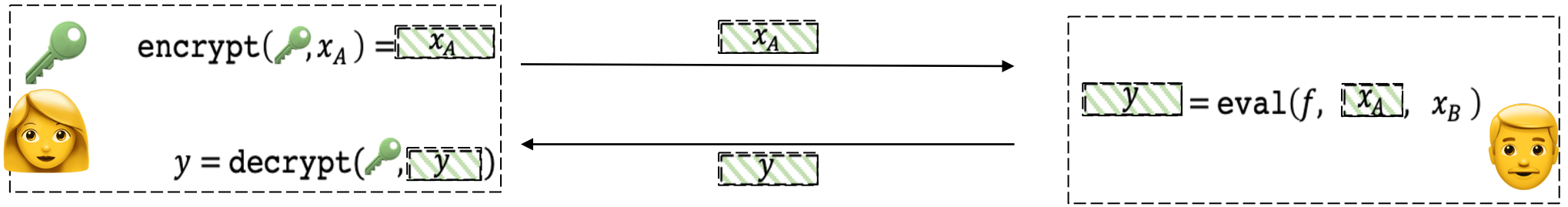

**[1] POSEIDON: Privacy-Preserving Federated Neural Network Learning,** S.Sav, A. Pyrgelis, J.R. Troncoso-Pastoriza, D. Froelicher, J.P. Bossuat, J.S. Sousa and JP Hubaux, NDSS 2021

# Building Blocks

| | | |
|---|---|---|
| 01 | **Multiparty Homomorphic Encryption (MHE)** | • Based on Ring learning with errors (RLWE)<br>• Cheon-Kim-Kim-Song (CKKS) variant<br>• Distributed bootstrapping |
| 02 | **Machine Learning** | • Gradient Descent<br>• Logistic/Linear Regression<br>• Neural Networks |
| 03 | **Distributed Machine Learning** | • Distributed Gradient Descent<br>• Privacy-Preserving Distributed Machine Learning |

# Building Blocks: **Homomorphic Encryption**

- Computations on ***encrypted*** data
- Secure **2-party** computation



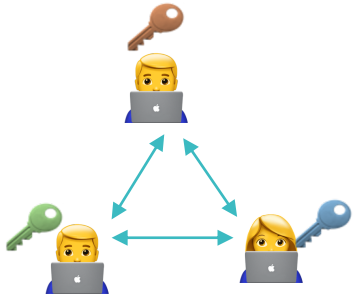

**Limitation**: What if there are more than 2 parties?

# Building Blocks: **Multiparty Homomorphic Encryption (MHE)**

- A **public collective key** known by all parties
- Corresponding **secret key** is distributed among parties

**Protocol: PubKeyGen**

PubKeyGen(🔑, 🔑, 🔑) = 🔑



**Protocol: Decrypt**

decrypt(🔑, 🔑, 🔑, $y$) = $y$

- **Decryption** requires collaboration between parties
- Enables *efficient* **collaborative bootstrapping** for refreshing ciphertexts

This solution relies on a MHE scheme that is CKKS variant, RLWE-based ->

RLWE: Ring Learning With Errors: computation problem that supports homomorphic encryption and resistance against security against post-quantum attacks



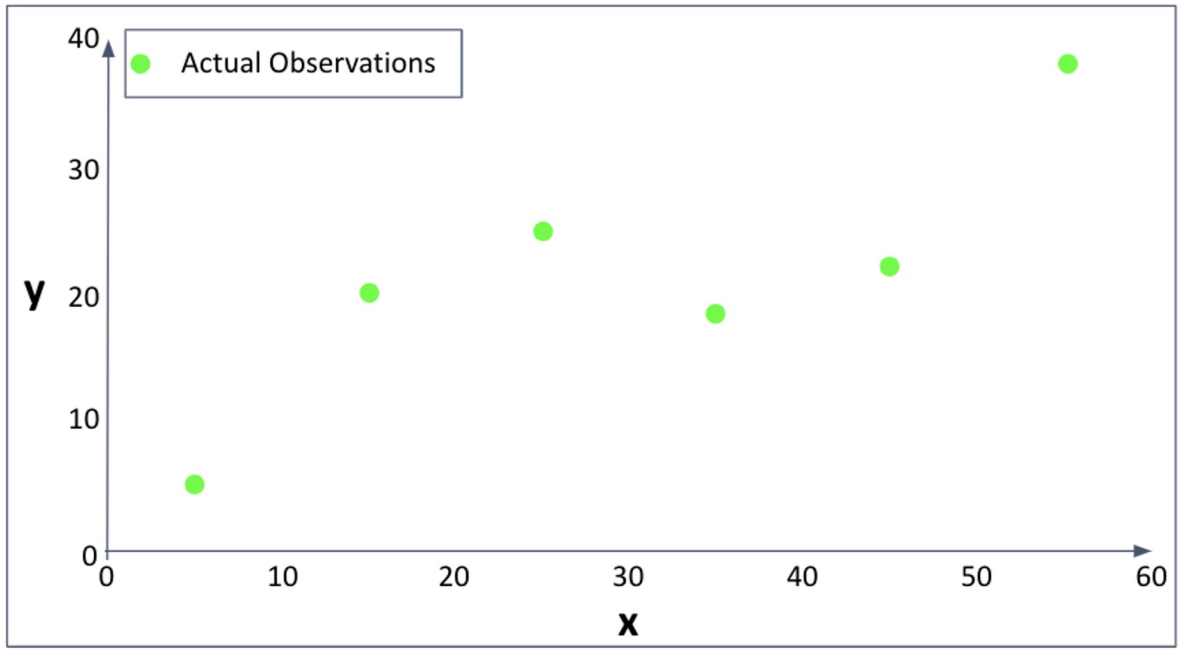
C. Mouchet, J. R. Troncoso-pastoriza, J.-P. Bossuat, and J. P. Hubaux. Multiparty homomorphic encryption: From theory to practice, 2020. https://eprint.iacr.org/2020/304, 2019.

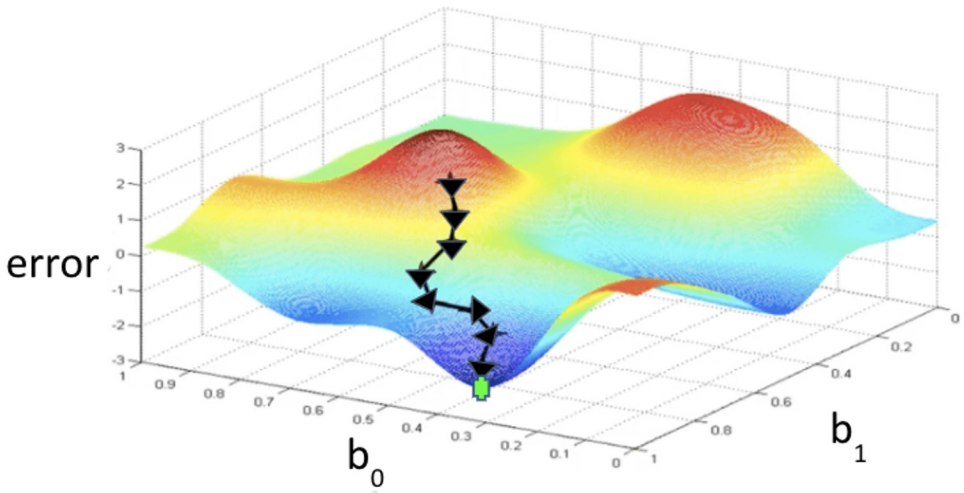
Building Blocks: **Gradient Descent for Linear/Logistic Regression**

## Background:

**Goal**: Find the line (defined by $\mathbf{b_0}$ and $\mathbf{b_1}$) that best fits the dots ($x_i$,$y_i$).

**Generic Method to find the best $\mathbf{b_0}$ and $\mathbf{b_1}$:** gradient descent is used to find the $\mathbf{b_0}$, $\mathbf{b_1}$ that give the minimum error.



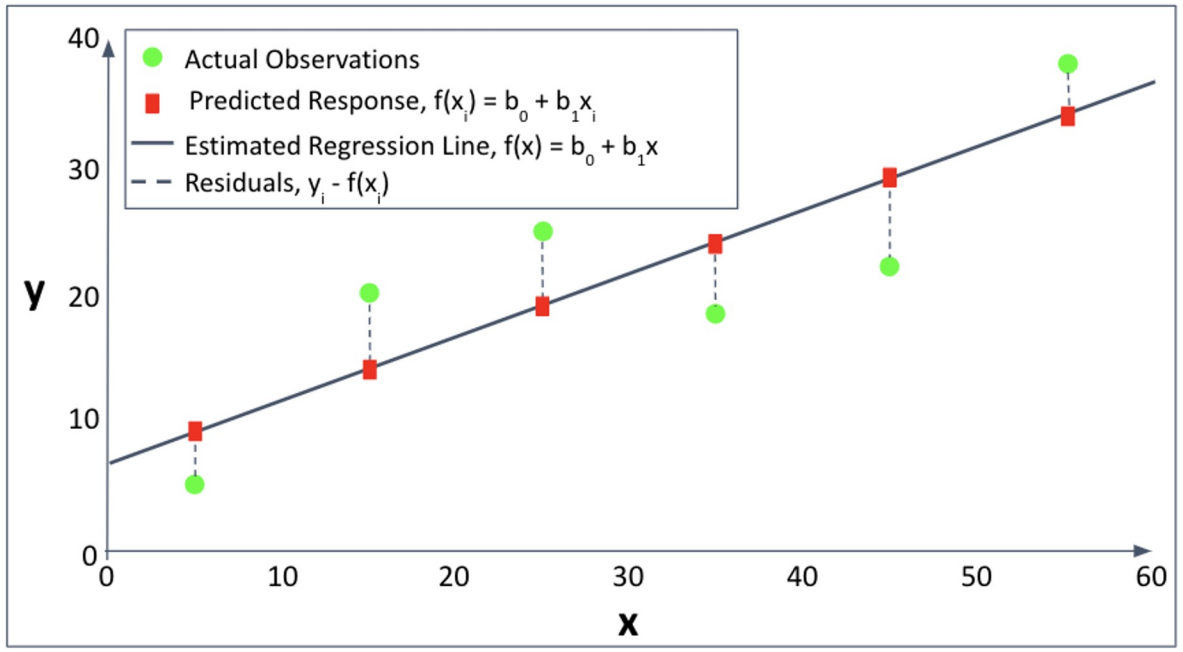



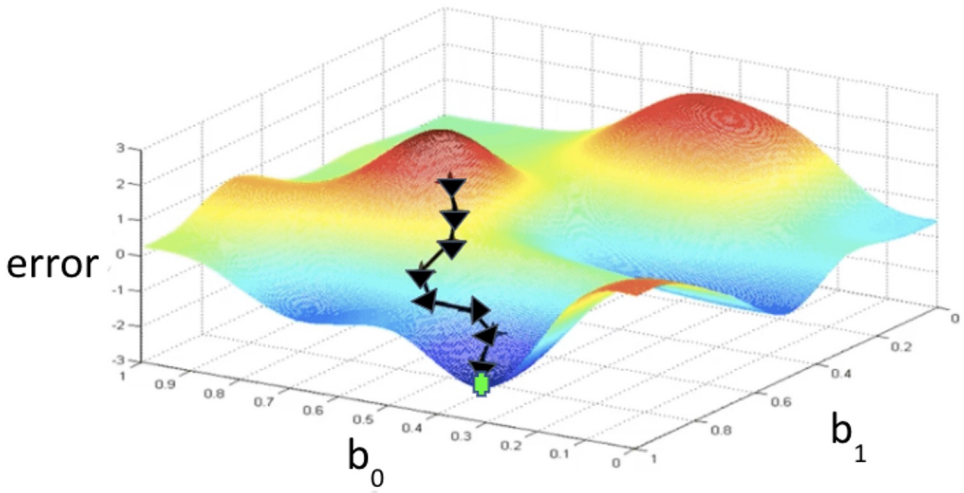
Building Blocks: **Gradient Descent for Linear/Logistic Regression**

## Background:

**Goal**: Find the line (defined by $\mathbf{b_0}$ and $\mathbf{b_1}$) that best fits the dots ($x_i$,$y_i$).

**Generic Method to find the best $b_0$ and $b_1$:** gradient descent is used to find the $\mathbf{b_0}$, $\mathbf{b_1}$ that give the minimum error.

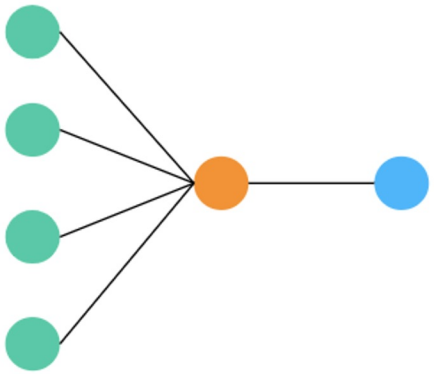


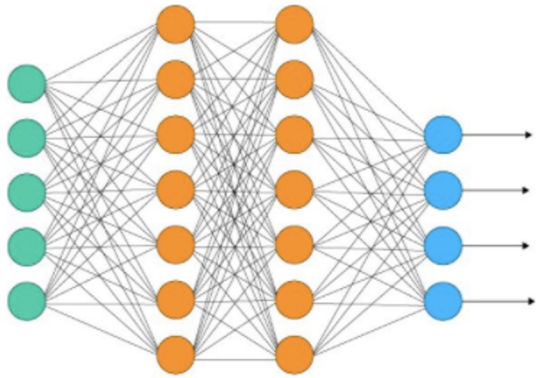
# Building Blocks: **Gradient Descent for Complex Models**

- Gradient descent: widely used technique for the training of machine learning models, even for more complex models such as neural networks.

Generalized Linear Models
(Represented as a simple neural network)



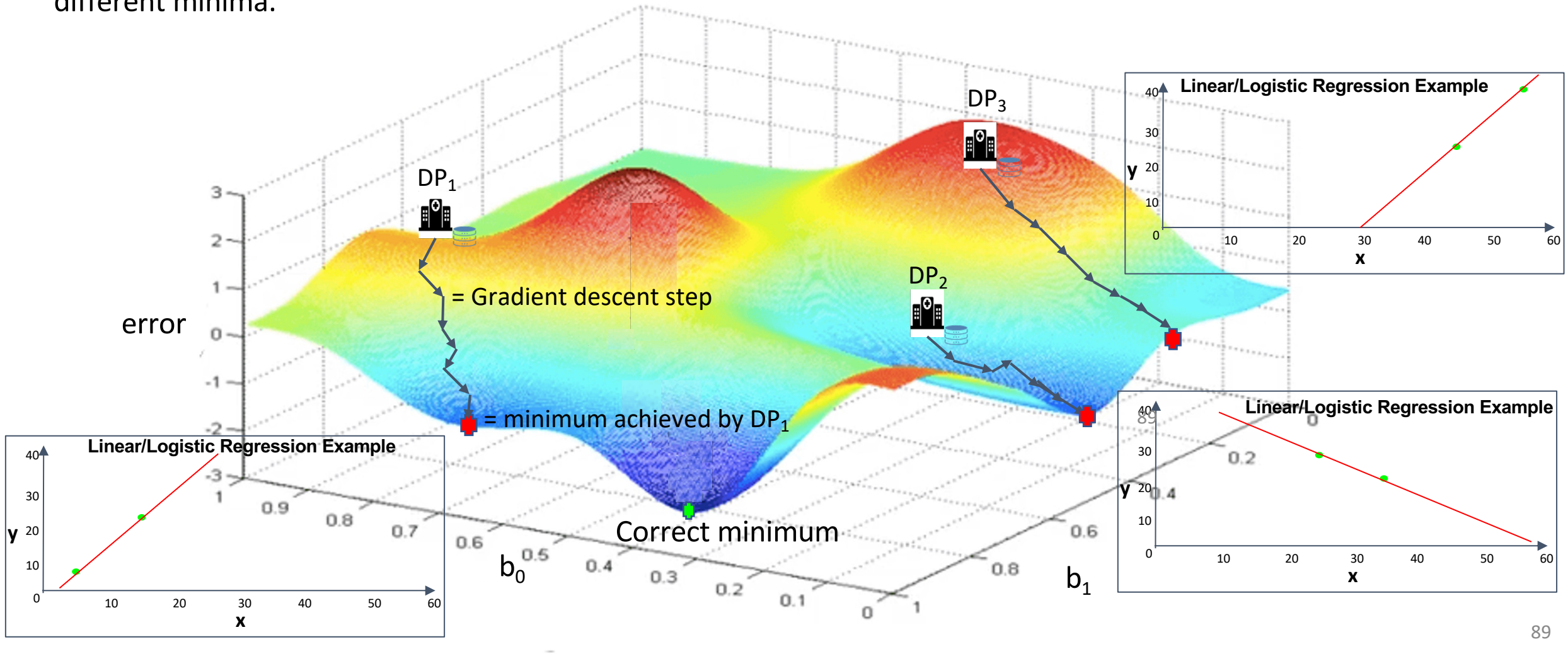
Deeper Neural Network



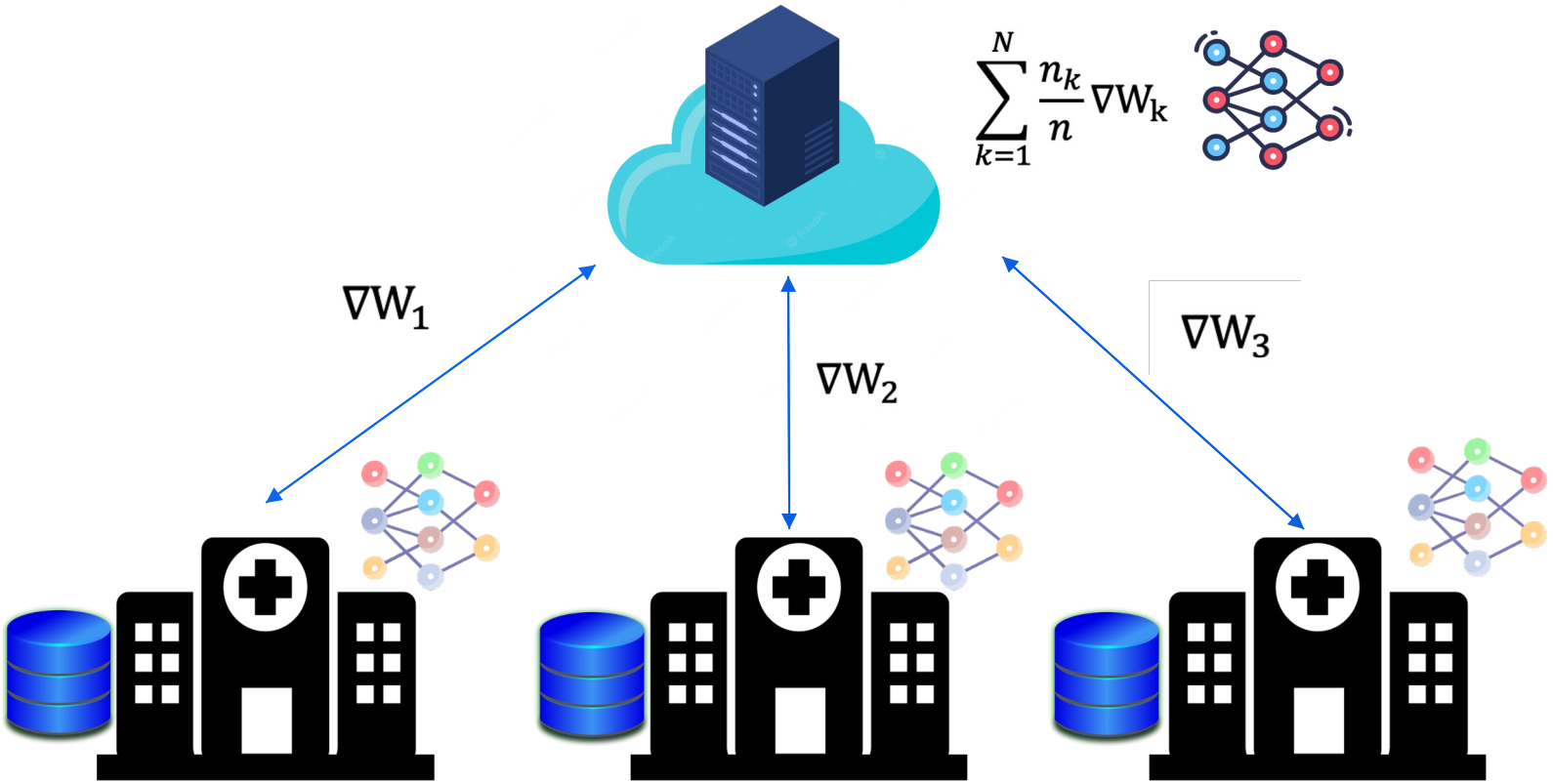
Input Layer  Hidden Layer  Output Layer

# Building Blocks: Distributed **Gradient Descent**

**Problem:** the data providers have to collaborate during the gradient descent, otherwise they can find different minima.



# Building Blocks: **Federated Learning**



# System and Threat Model

**System Model**

- N parties, each locally holding its own data train a ML model.
- At the end of the training process, a querier queries the model and obtains prediction results on its evaluation data
- The parties are interconnected and organized in a tree-network structure for communication efficiency


**Threat Model**

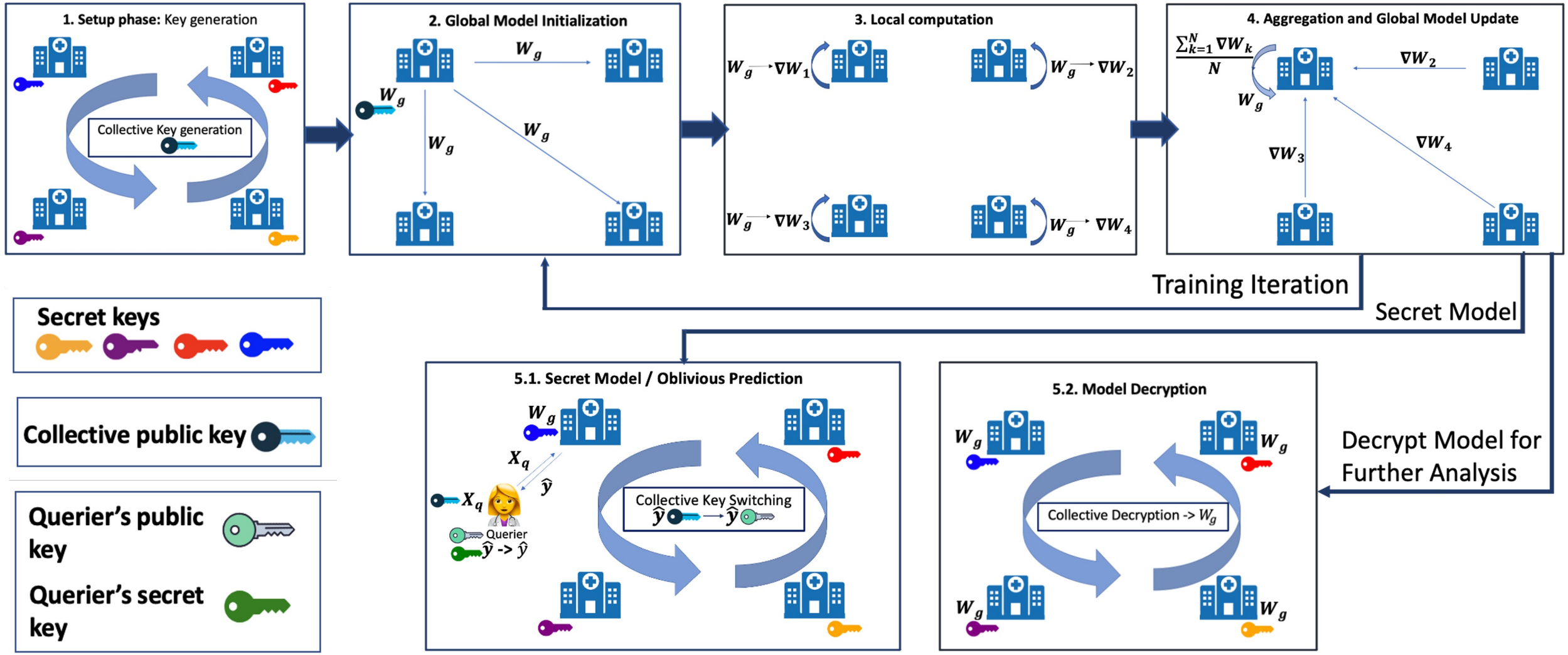
- A passive-adversary model with collusion of up to N-1 parties
- i.e., the parties follow the protocol but up to N −1 parties might share among them their inputs and observations during the training phase of the protocol, to extract information about the other parties' inputs through membership inference or federated learning attacks.

# Objectives


**Data Confidentiality:** During training and prediction, no party (including the querier) should learn more information about the input data of any other honest party, other than what can be deduced from its own input data.

**Model Confidentiality:** During training and prediction, no party (including the querier) should gain more information about the trained model weights, other than what can be deduced from its own input data.

# Solution Overview



# Main Challenges

MapReduce p.-p. Distributed Learning

PREPARE:
DKeyGen( ) →
Collective standardizatin of data
Each $DP_i$ pre-process his data

*Iterate*

MAP:
Each $DP_i$ locally trains on its data and uses the general model $\langle \mathbf{W_G} \rangle$ to output its local model $\langle \mathbf{W_i} \rangle$

COMBINE:
All $\langle \mathbf{W_i} \rangle$ are combined

REDUCE:
$DP_1$ updates the general model $\langle \mathbf{W_G} \rangle$

PREDICTION:
$DP_1$ uses $\langle \mathbf{W_G} \rangle$ to predict on $\langle$**new data**$\rangle$

**Gradient Descent**:

- Non-polynomial activation functions, i.e., Sigmoid, Softmax
- Heavy homomorphic operations, i.e., rotations, bootstrapping
- Model-specific functions, i.e., pooling in convolutional neural networks (CNNs)

**Bootstrapping:**
The model is persistent among multiple iterations → large multiplicative depth → ciphertext needs to be bootstrapped

**Parametrization:**
Tight links between learning parameters and cryptographic parameters

# Main Challenges ⟷ Solution [1]

**Gradient Descent**:

- Non-polynomial activation functions, i.e., Sigmoid, Softmax
- Heavy homomorphic operations, i.e., rotations, bootstrapping
- Model-specific functions, i.e., pooling in convolutional neural networks (CNNs)

⟷

- Least-squares approximation of activation functions
- Problem-specific packing schemes to enable Single Instruction, Multiple Data (SIMD)
- Introduce several functions to distributed bootstrapping

**Bootstrapping:**

The model is persistent among multiple iterations → large multiplicative depth → ciphertext needs to be bootstrapped

⟷

- Efficient and collaborative distributed bootstrapping
- Minimizing the number of bootstraps via parametrization

**Parametrization:**

Tight links between learning parameters and cryptographic parameters

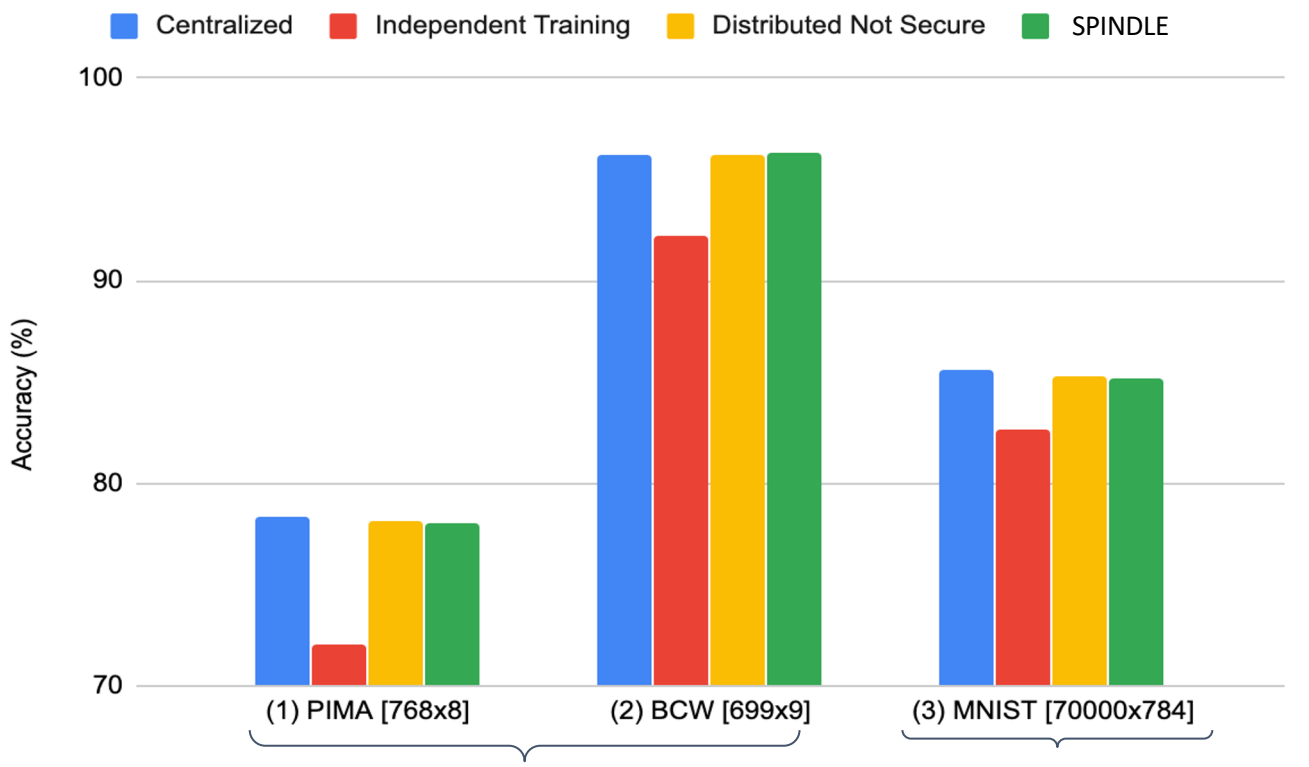
⟷

- A constrained optimization problem for choosing the cryptographic parameters

**[1] POSEIDON: Privacy-Preserving Federated Neural Network Learning,** S.Sav, A. Pyrgelis, J.R. Troncoso-Pastoriza, D. Froelicher, J.P. Bossuat, J.S. Sousa and JP Hubaux, NDSS 2021

# Evaluation for Logistic Regression: Accuracy

Accuracy close to centralized solution and **(almost) same accuracy as non-secure distributed solutions**




Evaluation Parameters
10 Data providers (DPs)
128-bit security level

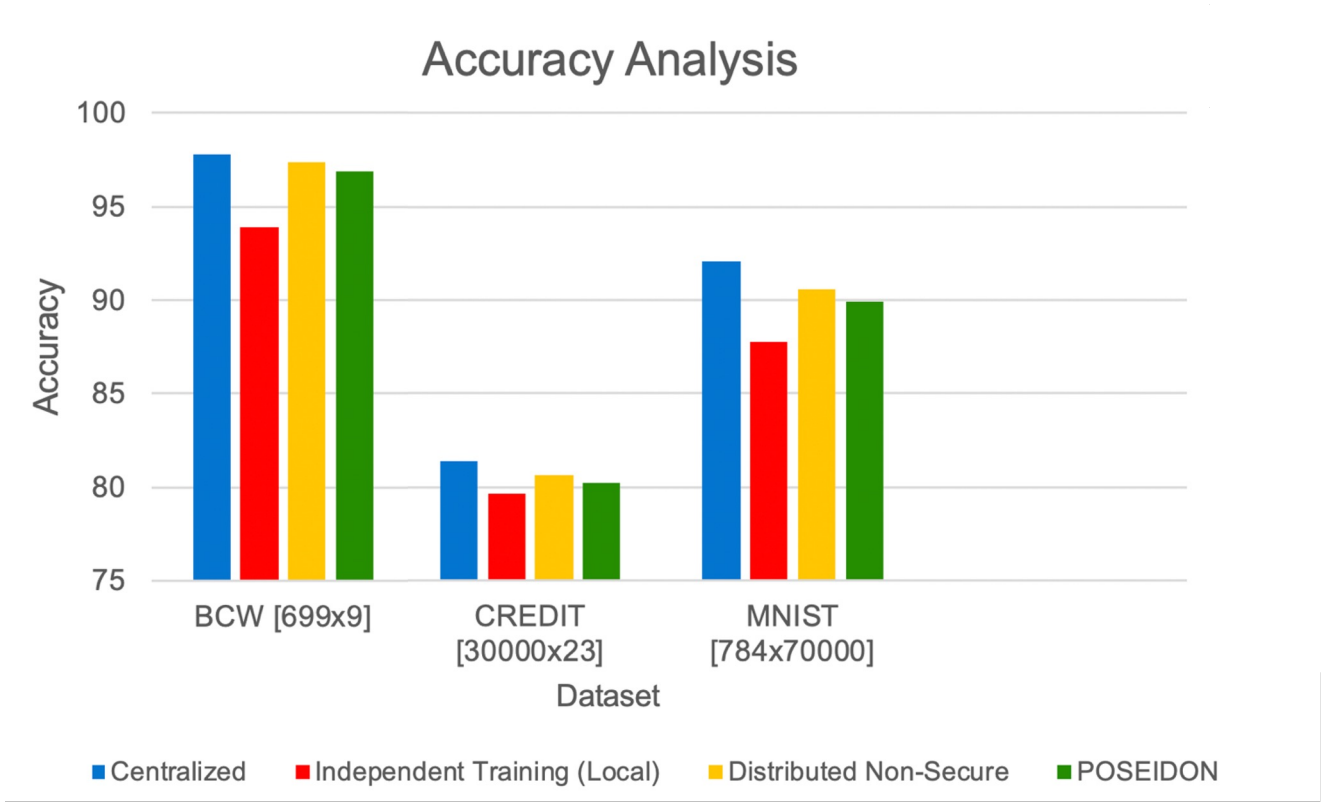
Legend
Dataset: *Name [#samples x #features]*

(1) Pima = Pima Indians Diabetes
https://www.kaggle.com/uciml/pima-indians-diabetes-database
(2) BCW = Breast cancer Wisconsin (original)
https://archive.ics.uci.edu/ml/datasets/breast+cancer+wisconsin+(original)
(3) MNIST
Y. LeCun and C. Cortes. Handwritten digit database. 2010.

SPINDLE: Scalable Privacy-Preserving Distributed Learning
D. Froelicher, J.R. Troncoso-Pastoriza, A. Pyrgelis, S.Sav, J.S. Sousa, J.P. Bossuat and J.P. Hubaux, under submission, 2020

# Evaluation for Neural Networks: Accuracy

Accuracy close to centralized solution and **(almost) same accuracy as non-secure distributed solutions**




Evaluation Parameters

10 Data providers

128-bit security level

Dataset <-> Neural Network Structure:

- BCW, CREDIT <-> 2-layer, 64 neurons per layer
- MNIST <-> 3-layer, 64 neurons per layer

**[nxd] -> Dataset size with n samples and d features**

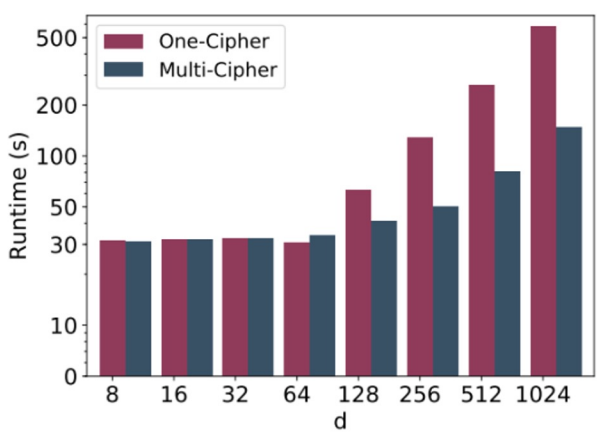
CREDIT: predictive accuracy of probability of default of credit card clients – Yeh and Lien, 2009

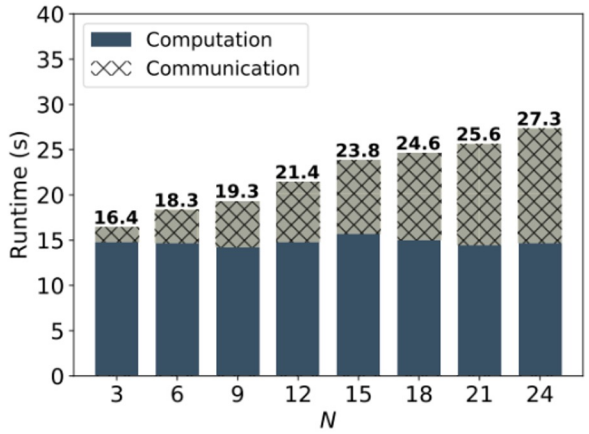
Accuracy can be further improved by additional learning rounds.

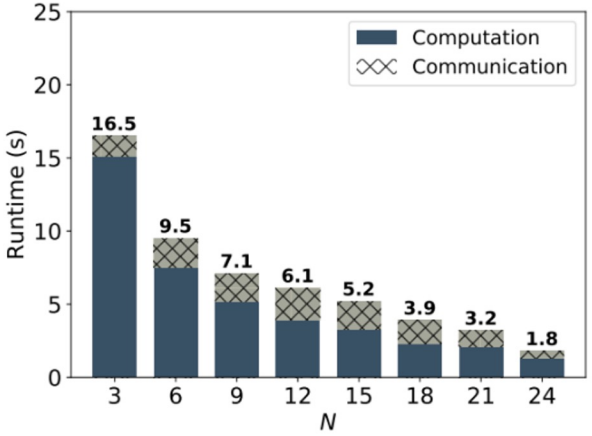
POSEIDON: Privacy-Preserving Federated Neural Network Learning
S.Sav, A. Pyrgelis, J.R. Troncoso-Pastoriza, D. Froelicher, J.P. Bossuat , J.S. Sousa and J.P. Hubaux, NDSS 2021

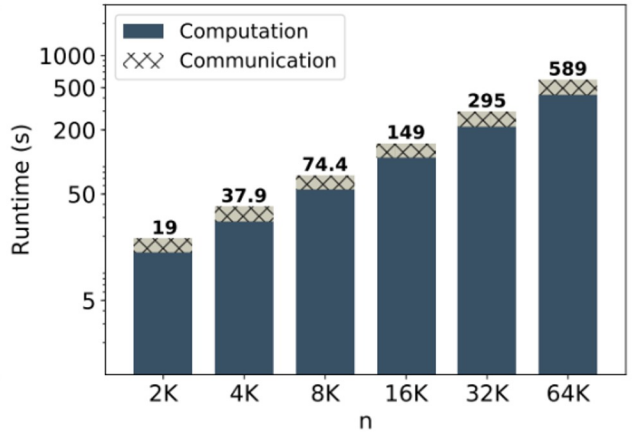
# Evaluation : Performance

Parameters:
- 2-layers neural network with 64 hidden neurons
- N=10 data providers, 1 epoch times (processing all data from all DPs once)
- 128-bit security level




**(a)** Increasing number of features ($d$), $N = 10$, and $n = 2{,}000 * N$. **(b)** Increasing number of parties ($N$), each having 200 samples ($n$). **(c)** Increasing number of parties ($N$) when $n$ (number of data samples) is fixed to 600. **(d)** Increasing number of data samples ($n$) when $N$ (number of parties) is fixed to 10.

**Parallelization** at multiple levels:
- data encryption
- data providers local threading
- among multiple data providers

**Multi-Cipher** shows the performance with optimized cryptographic parameters and multiple ciphertexts per weight matrix

Each data provider: 2 Intel Xeon E5-2680 v3 CPUs, 2.5GHz frequency, 24 threads on 12 cores, 256GB RAM. Bandwidth: 1Gbps

# Conclusion on evaluation

- Accuracy close to **non-secure distributed solutions**
- Scalability analysis shows that our computation and communication overhead scales
  - **linearly** with the number of parties
  - **logarithmically** with the number of features or the number of neurons in each layer

# Features and Guarantees

Functionalities

- Descriptive statistics
- Set operations
- Cosine similarity

- Linear regression
- Logistic regression
- Generalized linear models

- Multilayer perceptrons
- Convolutional Neural Networks
- Recurrent Neural Networks

Data Security

- The data never leave the data providers' premises
- Intermediate computation results are always protected
- Quantum-resistant analysis protocols

Privacy

- The querier only sees the final result

- Data providers cannot see the local data of the other participants or the querier's data

# Conclusion on Poseidon (solution example)

- Problem: Privacy-preserving *distributed* machine learning
- Solution: Multi-party homomorphic encryption (MHE)
  - Perform computations without “seeing” the data
  - Rely on decentralized trust
  - No need to transfer the data
- The number of data providers can be unbounded
- No need to use differential privacy to protect partial aggregate results

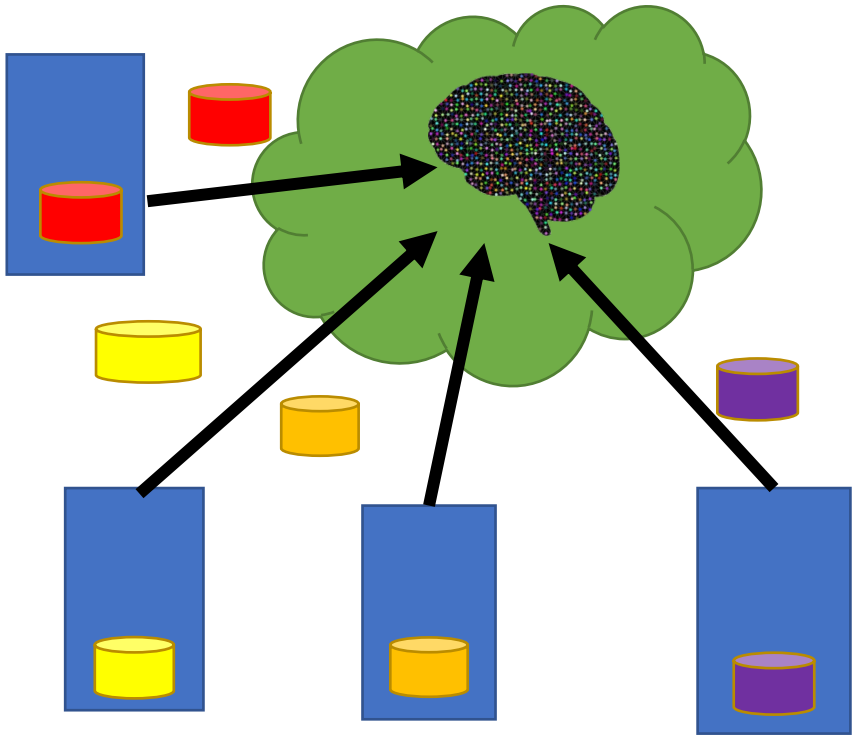

lds.epfl.ch

**SPINDLE: Scalable Privacy-Preserving Distributed Learning.** D. Froelicher, J.R. Troncoso-Pastoriza, A. Pyrgelis, S.Sav, J.S. Sousa, J.P. Bossuat and J.P. Hubaux, PoPETS 2021 https://arxiv.org/abs/2005.09532

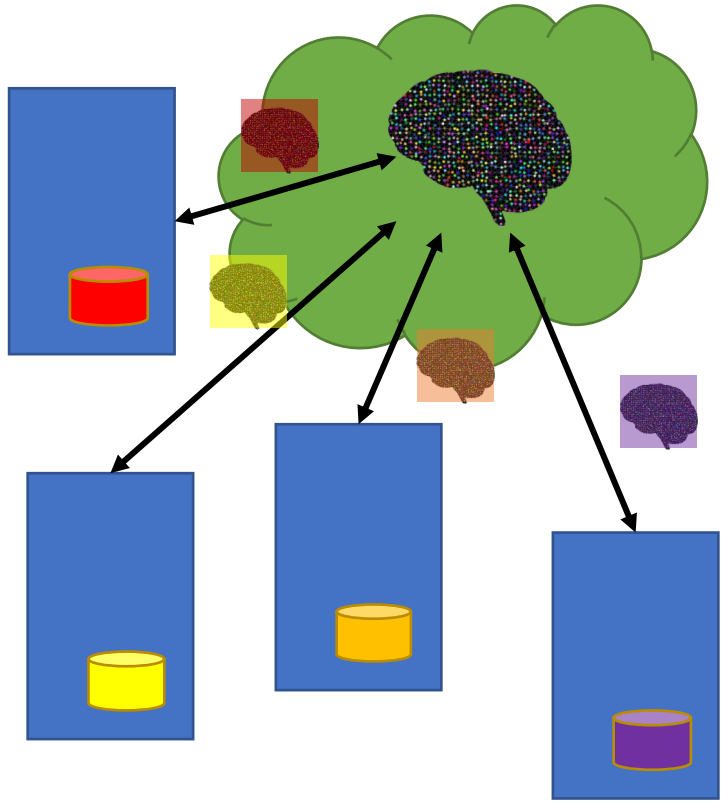
**POSEIDON: Privacy-Preserving Federated Neural Network Learning**
S.Sav, A. Pyrgelis, J.R. Troncoso-Pastoriza, D. Froelicher, J.P. Bossuat , J.S. Sousa and J.P. Hubaux, NDSS 2021

# Adversarial examples against federated learning (challenging!)

Centralized

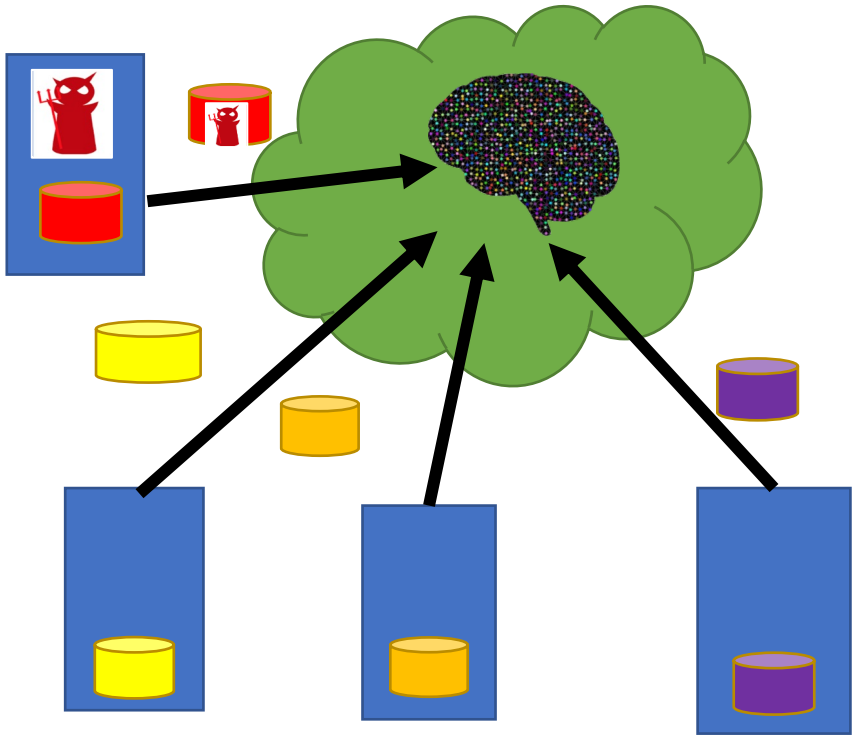


Federated

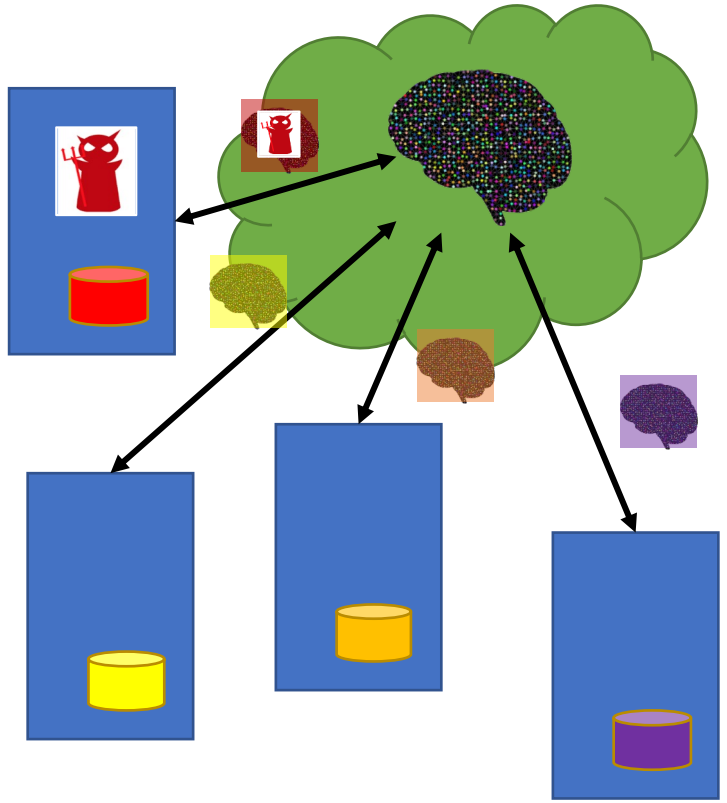


# Adversarial examples against federated learning (challenging!)

Centralized



Federated

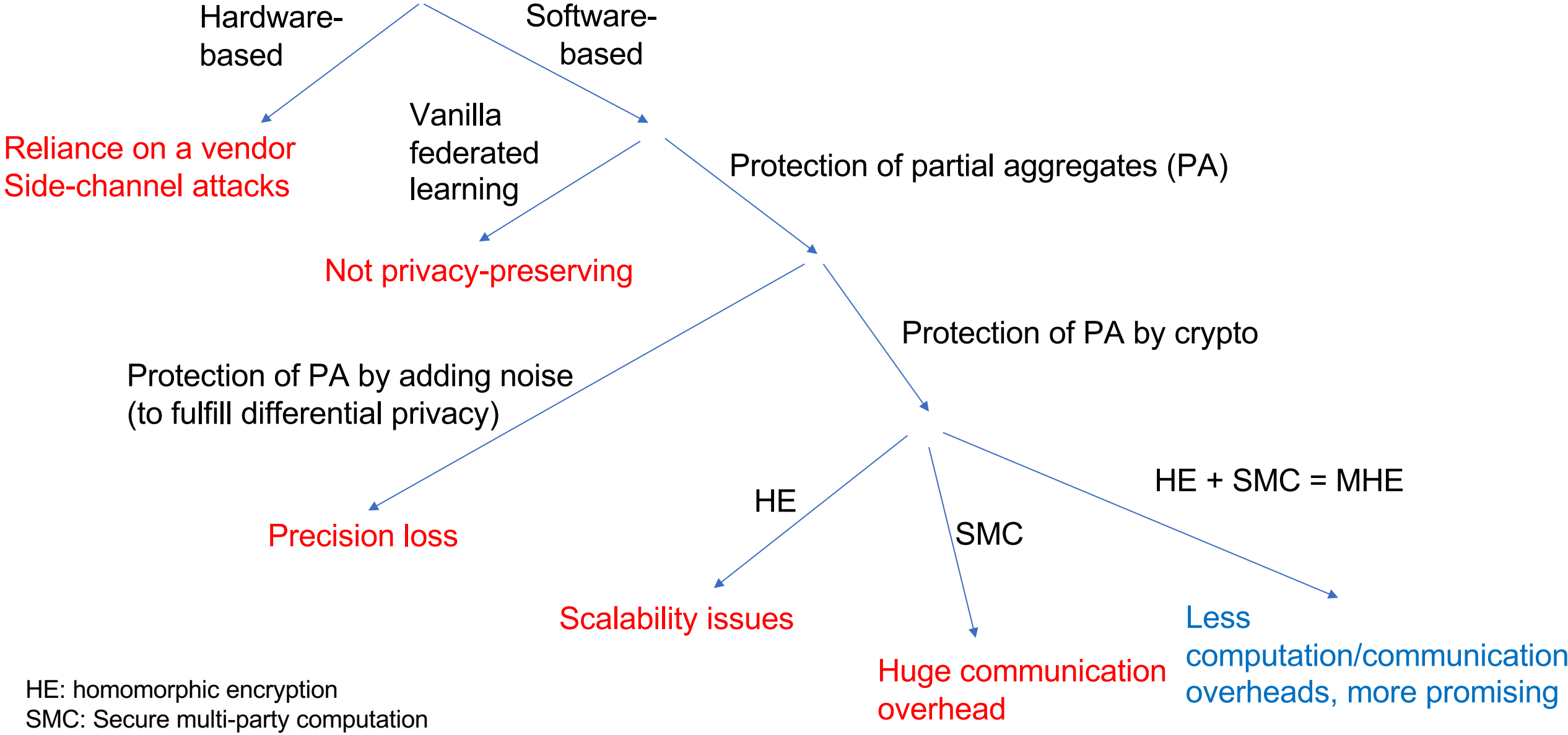


Difference with the previous section on adversarial examples: here the adversary controls only a subset of the inputs

# References for prev. slides


[1] T. Chen and S. Zhong. Privacy-preserving backpropagation neural network learning. IEEE Transactions on Neural Networks, 20(10):1554–1564, Oct 2009.
[2] H. Chaudhari, R. Rachuri, and A. Suresh. Trident: Efficient 4pc framework for privacy preserving machine learning. In 27th Annual Network and Distributed System Security Symposium, NDSS, pages 23–26, 2020.
[3] M. Byali, H. Chaudhari, A. Patra, and A. Suresh. FLASH: Fast and robust framework for privacy-preserving machine learning. Proceedings on Privacy Enhancing Technologies, 2020:459–480, 2020.
[4] P. Mohassel and P. Rindal. Aby 3: a mixed protocol framework for machine learning. In ACM Conference on Computer and Communications Security (CCS), 2018.
[5] P. Mohassel and Y. Zhang. Secureml: A system for scalable privacy-preserving machine learning. In 2017 IEEE Symposium on Security and Privacy (SP), pages 19–38, May 2017.
[6] Wagh, D. Gupta, and N. Chandran. Securenn: 3-party secure computation for neural network training.Privacy Enhancing Technologies (PETS), 2019.
[7] S. Wagh, S. Tople, F. Benhamouda, E. Kushilevitz, P. Mittal, and T. Rabin.FALCON: Honest-majority maliciously secure framework for private deep learning.arXiv preprint arXiv:2004.02229, 2020.
[8] W. Li, F. Milletarì, D. Xu, N. Rieke, J. Hancox, W. Zhu, M. Baust, Y. Cheng, S. Ourselin, M. J. Cardoso, and A. Feng. Privacy-preserving federated brain tumour segmentation. In H.-I. Suk, M. Liu, P. Yan, and C. Lian, editors, International Workshop in Machine Learning in Medical Imaging (MLMI). Springer, 2019.
[9] H. B. McMahan, D. Ramage, K. Talwar, and L. Zhang. Learning differentially private recurrent language models. In International Conference on Learning Representations, 2018.
[10] H. B. McMahan, D. Ramage, K. Talwar, and L. Zhang. Learning differentially private recurrent language models. In International Conference on Learning Representations, 2018.
[11] J. Konecný, H. McMahan, D. Ramage, and P. Richtárik. Federated optimization: ˇ Distributed machine learning for on-device intelligence. 10 2016.
[12] H. B. McMahan, E. Moore, D. Ramage, and B. A. y Arcas. Federated learning of deep networks using model averaging. CoRR, abs/1602.05629, 2016.
[13] B. Hitaj, G. Ateniese, and F. Perez-Cruz. Deep models under the GAN: Information leakage from collaborative deep learning. In Proceedings of the 2017 14 ACM SIGSAC Conference on Computer and Communications Security, CCS '17, page 603–618, New York, NY, USA, 2017. Association for Computing Machinery.
[14] Z. Wang, M. Song, Z. Zhang, Y. Song, Q. Wang, and H. Qi. Beyond inferring class representatives: User-level privacy leakage from federated learning. In IEEE INFOCOM 2019 - IEEE Conference on Computer Communications, pages 2512–2520, 2019.
[15] L. Zhu, Z. Liu, and S. Han. Deep leakage from gradients. In H. Wallach, H. Larochelle, A. Beygelzimer, F. dÁlché-Buc, E. Fox, and R. Garnett, editors, Advances in Neural Information Processing Systems 32, pages 14774–14784. Curran Associates, Inc., 2019.
[16] L. Melis, C. Song, E. De Cristofaro, and V. Shmatikov. Exploiting unintended feature leakage in collaborative learning. In 2019 IEEE Symposium on Security and Privacy (SP), pages 691–706, 2019.
[17] M. Nasr, R. Shokri, and A. Houmansadr. Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning. In 2019 IEEE Symposium on Security and Privacy (SP), pages 739–753, 2019.

# Technology selection for privacy-conscious data sharing




HE: homomorphic encryption
SMC: Secure multi-party computation
MHE: Multi-party homomorphic encryption

# Takeaways on federated learning

Combining decentralized data sets to train a ML model while protecting privacy is doable in various ways:

- By adding noise to partial aggregates
- By sending partial aggregates to a trusted execution environment
- By a combination of homomorphic encryption and secure multiparty computation

Additional resource:
https://www.openmined.org

Start-up example: https://tuneinsight.com