Machine Learning Classification over Encrypted Data

Raphaël Bost
Université Rennes 1
MIT

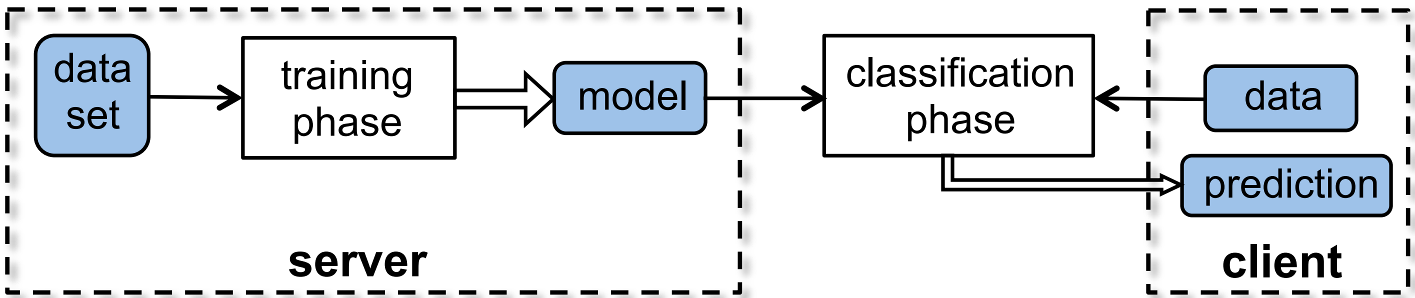
Raluca Ada Popa,
ETH Zürich
MIT

Stephen Tu
MIT

Shafi Goldwasser
MIT

Classification
(Machine Learning)

- Supervised learning (training)
- Classification



Problem

- The provider's model is sensitive

  financial model, genetic sequences, …

- Client's private data

  medical records, credit history, …

Problem

- The provider's model is sensitive

  financial model, genetic sequences, …

- Client's private data

  medical records, credit history, …

MPC / 2PC

Using General 2PC ?

+ Works for every circuit

+ Constant number of interactions

- Have to build circuits

- Hard to ‘compose’

- Not easily reusable

Using General 2PC ?

+ Works for every circuit

+ Constant number of interactions

- Have to build circuits

- Hard to ‘compose’

- Not easily reusable

➡ *Ad Hoc* protocols

Goal

- Enable classification without sacrificing privacy
- Secure classification, no learning
  the model is already known
- Practical performance

Approach

- Classifiers as specialized 2PC
- Identify and construct reusable building blocks
- Threat model: passive (honest-but-curious) adversary

Insight

ML Algorithm	Classifier
Perceptron	Linear
Least squares	Linear
Fischer linear discriminant	Linear
Support vector machine	Linear
Naïve Bayes	Naïve Bayes
ID3/C4.5	Decision trees

Insight

- Identify core operations
- Construct reusable/composable building blocks
- Choose the best fitted primitives

  Homomorphic Encryption, FHE, Garbled Circuits, …

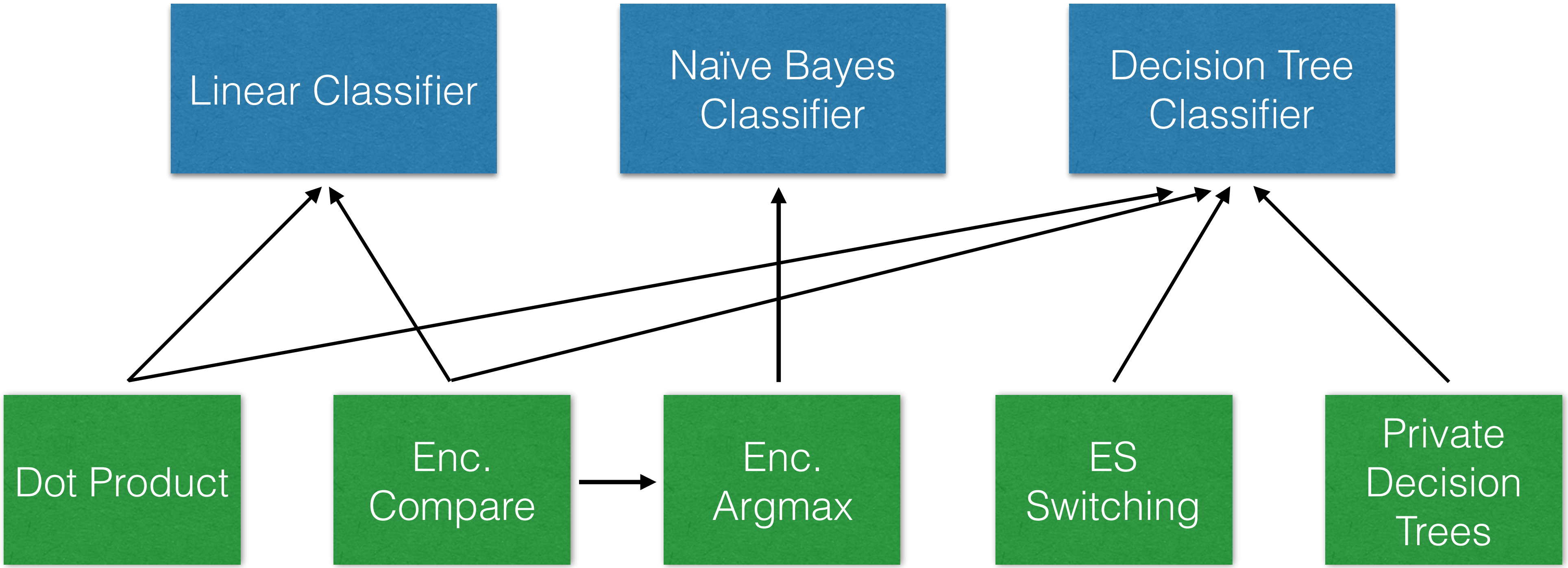
Related Work

- Privacy-preserving training
  - Using FHE, linear means classifier [GLN12]
  - Specific techniques for Naïve Bayes [VKC08], decision trees [BDMN05,LP00], linear discriminant [DHC04], kernel methods [LLM06]
- Privacy-preserving classification
  - Using FHE, outsource computation [BLN13]
  - Secure branching programs [BFK+09, BFL+09]
  - Specific classifiers (face recognition/detection) [SSW09, AB07]

Building Blocks

- Dot product
- Encrypted Comparison
- Encrypted (arg)max
- Decision trees
- Encryption scheme switching

Classifiers from blocks



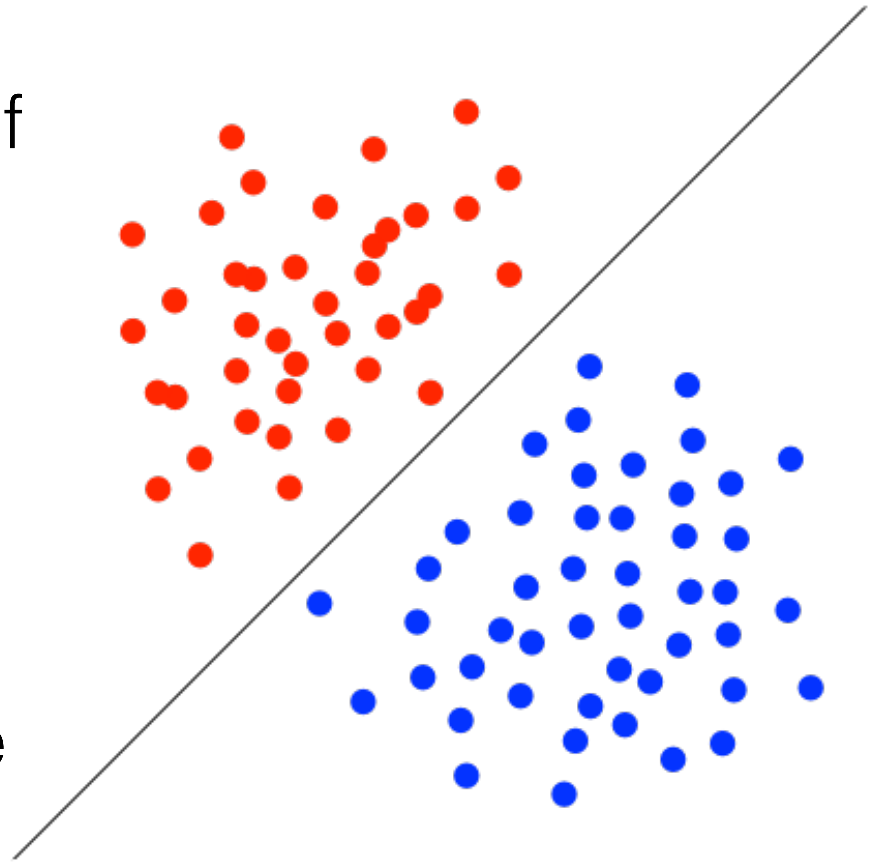
Classifiers

In Practice

- Linear Classifier
- Naïve Bayes Classifier
- Decision Trees

Linear Classifier

- Separate two sets of points
- Very common classifier
- Dot product + Encrypted compare



Linear Classifier

Model Size	Time / protocol		Total	Comm.	Inter.
	Dot Product	Enc. Comp.			
30	<0.01s	0.194 s	0.204 s	35.84 kB	7
47	0.024 s	0.194 s	0.217 s	40.19 kB	7

Evaluation on UC Irvine ML databases
40 ms network latency
2,66 GHz Intel Core i7

Naïve Bayes Classifier

$$\operatorname*{argmax}_{i \in [k]} p(C = c_i) \prod_{j=1}^{d} p(X_j = x_j | C = c_i)$$

Naïve Bayes Classifier

$$\operatorname*{argmax}_{i \in [k]} \underline{p(C = c_i)} \prod_{j=1}^{d} \underline{p(X_j = x_j | C = c_i)}$$

Naïve Bayes Classifier

$$\underset{i \in [k]}{\operatorname{argmax}}\, \underline{p(C = c_i)} \prod_{j=1}^{d} \underline{p(X_j = x_j | C = c_i)}$$

$$\underset{i \in [k]}{\operatorname{argmax}} \quad \log p(C = c_i) \sum_{j=1}^{d} \log p(X_j = x_j | C = c_i)$$

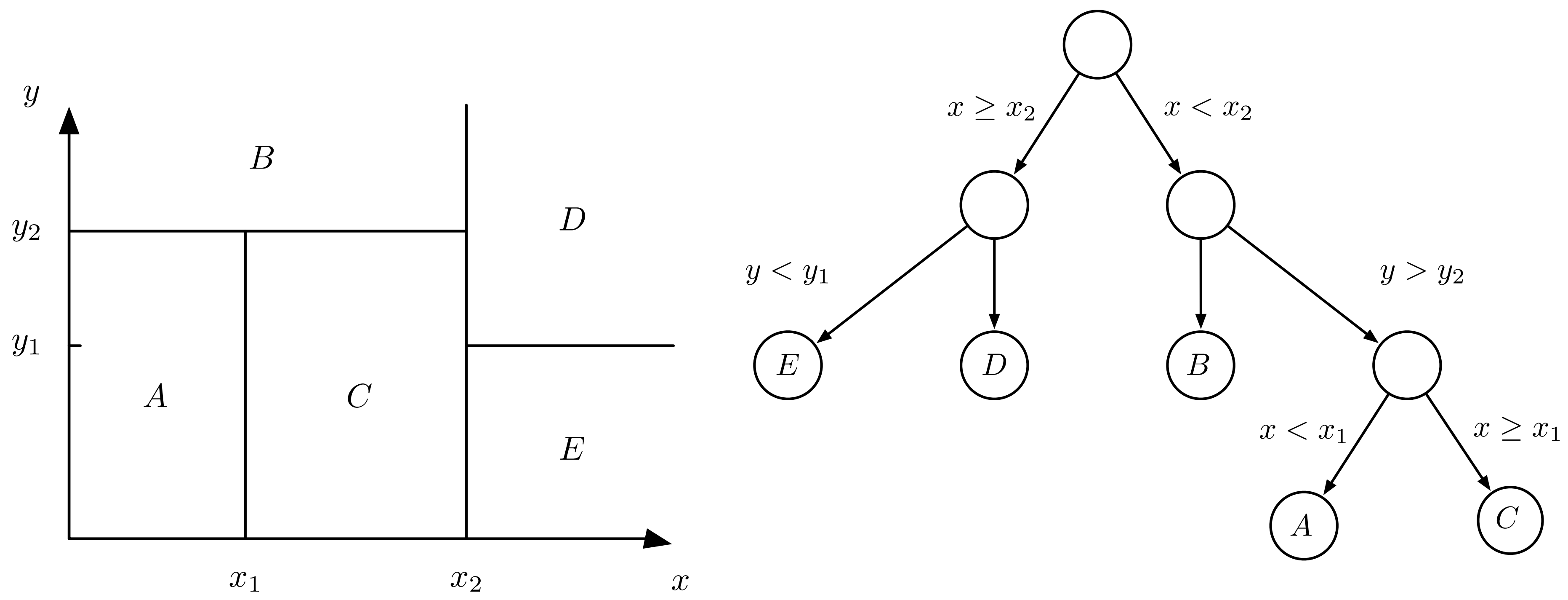
- Additive homomorphism + Encrypted argmax

Naïve Bayes Classifier

# Cat.	# Features	Argmax	Total Time	Comm.	Inter.
2	9	0.40 s	0.48 s	72.47 kB	14
5	9	1.33 s	1.42 s	150.7 kB	42
24	70	3.38 s	3.81 s	1911 kB	166

Evaluation on UC Irvine ML databases
40 ms network latency
2,66 GHz Intel Core i7

Decision Trees



Decision Tree

- Combination of other classifiers
- In this example, linear classifiers
- Linear classifier + ES Switching + Decision Trees

Decision Tree

Tree Specs.		Time / Protocol			Total	Comm.	Inter.
Nodes	Depth	Lin. Class.	ES Switch	Decision Tree (FHE)			
4	4	0.45 s	1.64 s	0.27 s	2.3 s	2639 kB	30
6	4	1.41 s	7.41 s	0.93 s	9.8 s	3555 kB	44

Evaluation on UC Irvine ML databases
40 ms network latency
2,66 GHz Intel Core i7

Decision Tree

Tree Specs.		Time / Protocol			Total	Comm.	Inter.
Nodes	Depth	Lin. Class.	ES Switch	Decision Tree (FHE)			
4	4	0.45 s	1.64 s	0.27 s	2.3 s	2639 kB	30
6	4	1.41 s	7.41 s	0.93 s	9.8 s	3555 kB	44

Run sequentially, can be parallelized

Building blocks library

- Designed to be modular

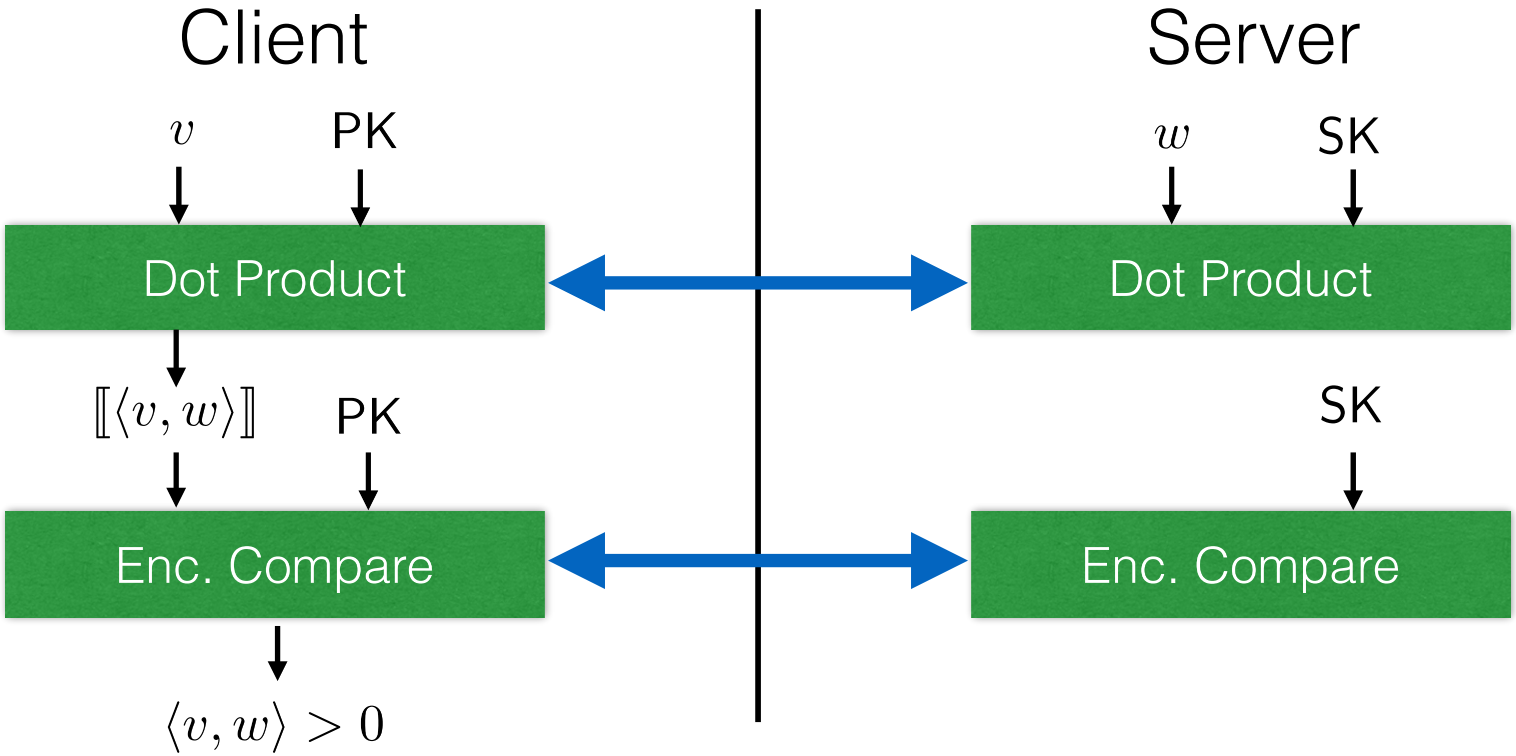
  Easy composition

- Easy to construct new secure classifiers

  Face detection algorithm (Viola & Jones)

Building blocks library

E.g.: Linear Classifier



Building blocks library

E.g.: Linear Classifier

Client

```
bool Linear_Classifier_Client::run()
{
  exchange_keys();

  // values_ is a vector of integers
  // compute the dot product
  mpz_class v = compute_dot_product(values_);
  mpz_class w = 1; // encryption of 0

  // compare the dot product with 0
return enc_comparison(v, w, bit_size_, false);
}
```

Server

```
void Linear_Classifier_Server_session:: run_session()
{
  exchange_keys();

// enc_model_ is the encrypted model vector
// compute the dot product
help_compute_dot_product(enc_model_, true);

// help the client to get
// the sign of the dot product
  help_enc_comparison(bit_size_, false);
}
```

In conclusion

- Composable building blocks for secure classifiers
- Library with practical performances

Future work :

- Less roundtrips (work on the protocols)
- More parallelism (work on the implementation)

Questions?