Research Group on Artificial Intelligence

UNIVERSITAS SCIENTIARUM SZEGEDIENSIS

Peer-to-Peer Multi-Class Boosting

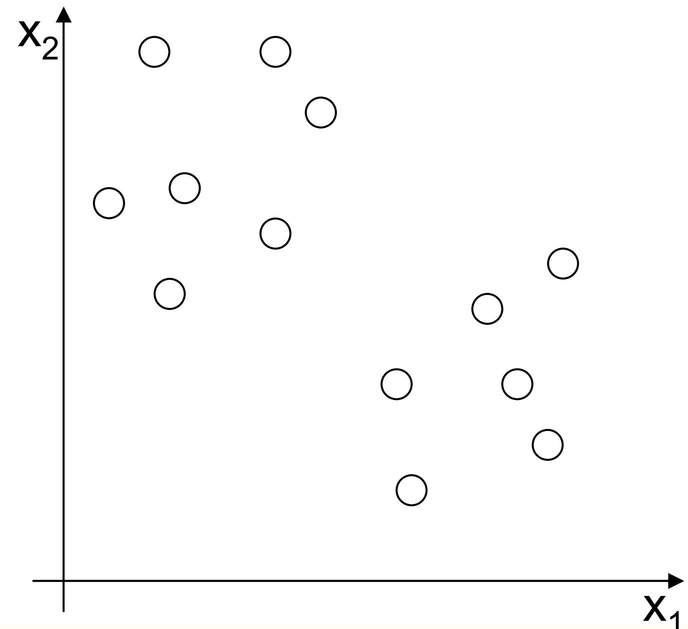
István Hegedűs, Róbert Busa-Fekete, Róbert Ormándi, Márk Jelasity, Balázs Kégl

Motivation

- Machine Learning, Data Mining
  - Identifying representative patterns in data
  - Make compact representation of the data
- Classification
  - Separating different type of patterns to each other
  - Based on (hand-) labeled data
  - E.g. Spam detection, OCR, Speech recognition, NLP, Document classification, …
- **We would like to use state of the art classification algorithms in large scale P2P environments on distributed data**

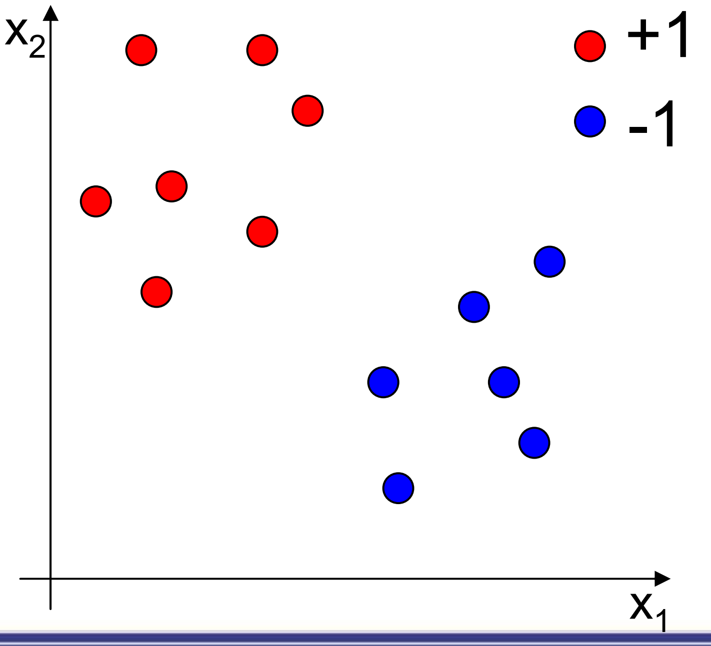
Classification

- Binary classification
  - Given a set of training samples: $(x_1, y_1), \ldots, (x_n, y_n)$ where $x_i \in \mathbb{R}^d$



Classification

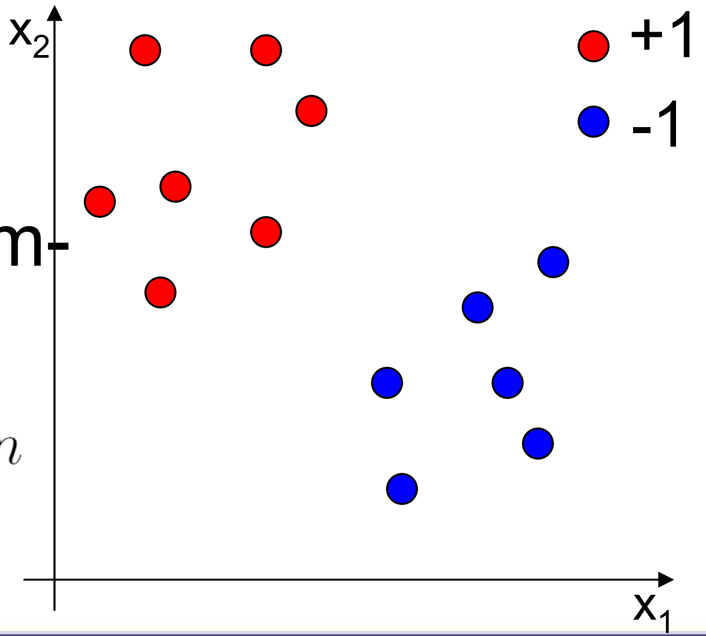
- Binary classification
  - Given a set of training samples: $(x_1, y_1), \ldots, (x_n, y_n)$ where $x_i \in \mathbb{R}^d$ and $y_i \in \{-1, 1\}$



Classification

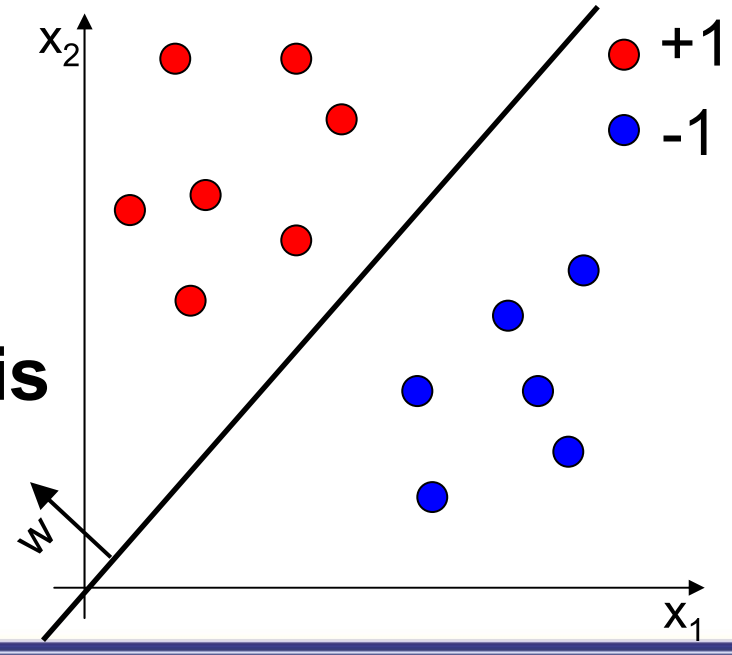
- Binary classification
  - Given $(x_1, y_1), \ldots, (x_n, y_n)$ training samples, where $x_i \in \mathbb{R}^d$ and $y_i \in \{-1, 1\}$
  - Task: looking for a **model** $f : \mathbb{R}^d \to \{-1, 1\}$ that correctly separates the samples from different classes (minimizes the number of misclassifications)

$$\min_f \sum_i (f(x_i) - y_i)^2 \quad i = 1, \ldots, n$$



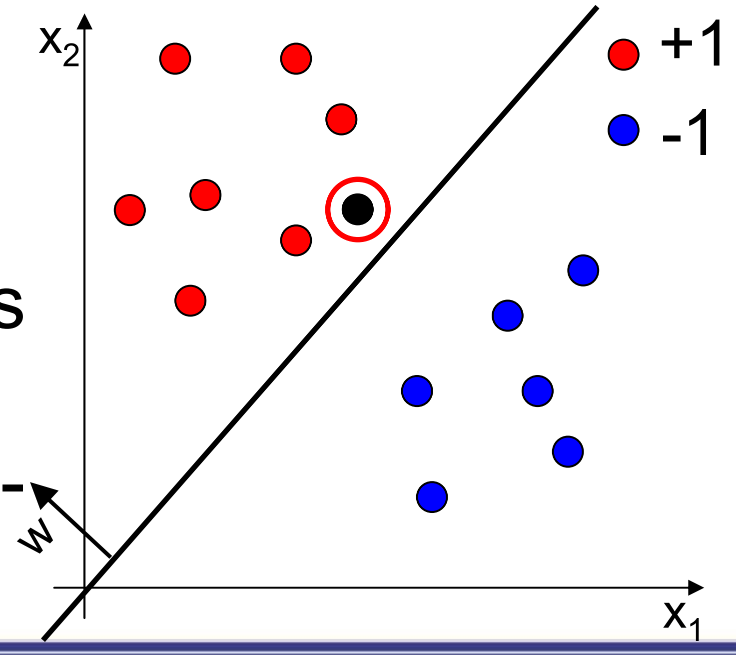
Classification

- Binary classification
  - Given $(x_1, y_1), \ldots, (x_n, y_n)$ training samples, where $x_i \in \mathbb{R}^d$ and $y_i \in \{-1, 1\}$
  - Task: looking for a model
    $$f : \mathbb{R}^d \to \{-1, 1\}$$
    minimizes the error
    $$\min_f \sum_i (f(x_i) - y_i)^2 \quad i = 1, \ldots, n$$
  - In linear case the **model is a hyper-plane** (w)

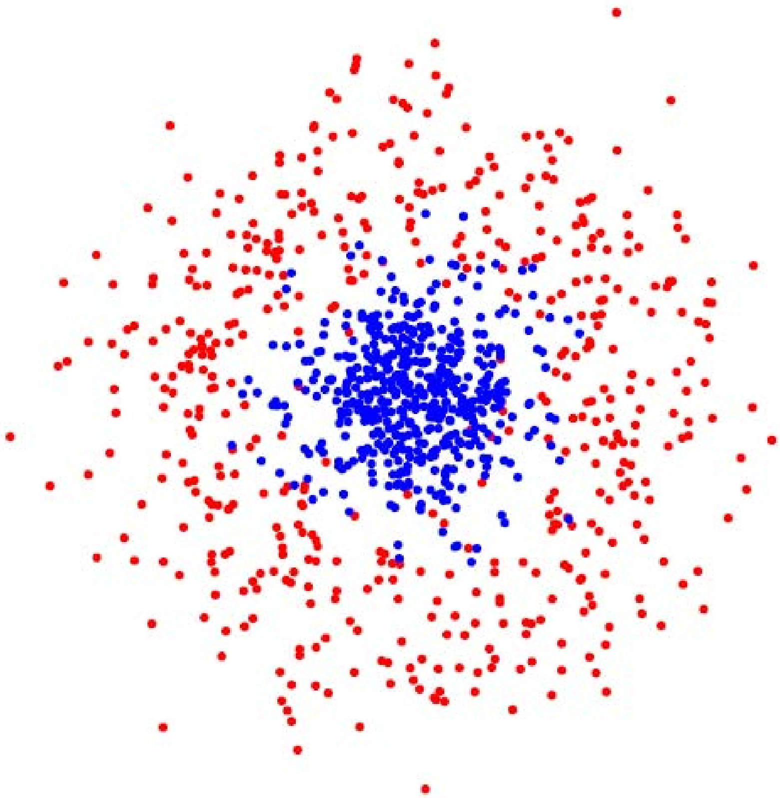


Classification

- Binary classification
  - Given $(x_1, y_1), \ldots, (x_n, y_n)$ training samples, where $x_i \in \mathbb{R}^d$ and $y_i \in \{-1, 1\}$
  - Task: looking for a model
    $$f : \mathbb{R}^d \rightarrow \{-1, 1\}$$
    minimizes the error
    $$\min_f \sum_i (f(x_i) - y_i)^2 \quad i = 1, \ldots, n$$
  - In linear case the model is a hyper-plane (w)
  - The **label** of a new instance **can be predicted**

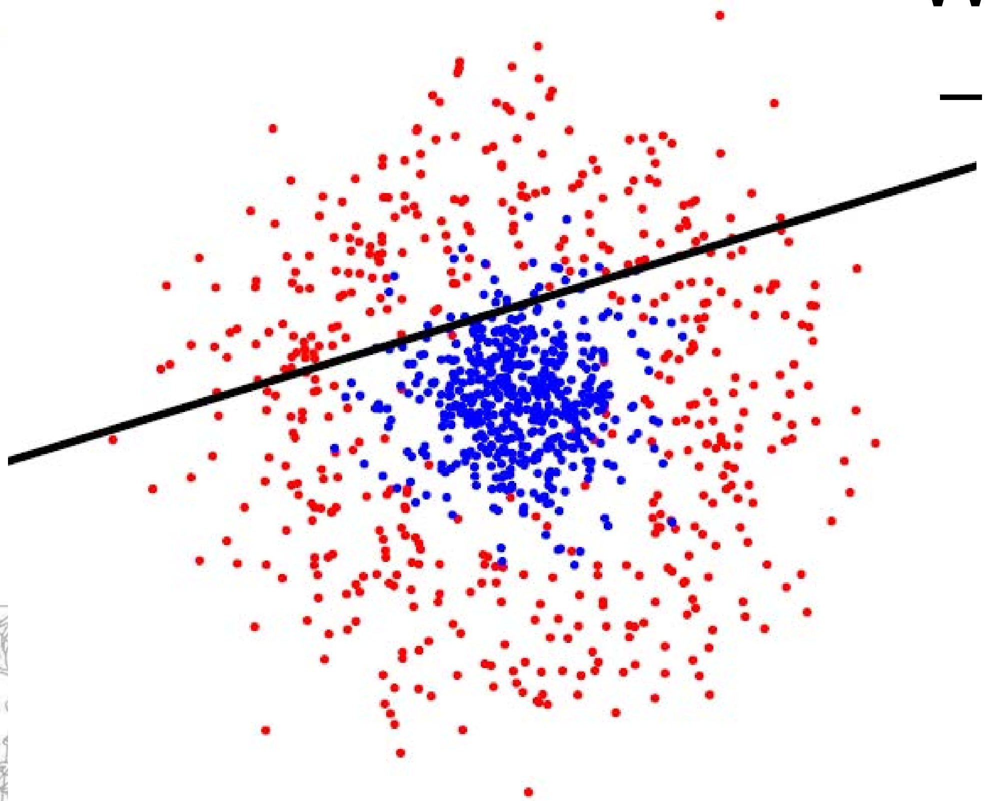


Not Linearly Separable Set



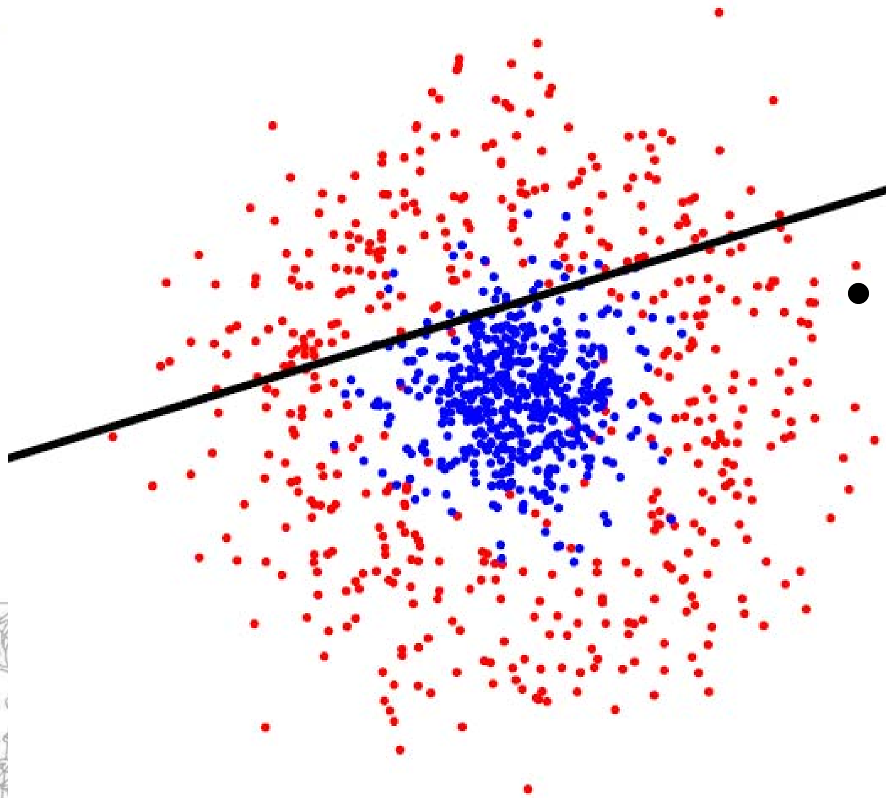
- What can we do?

Not Linearly Separable Set



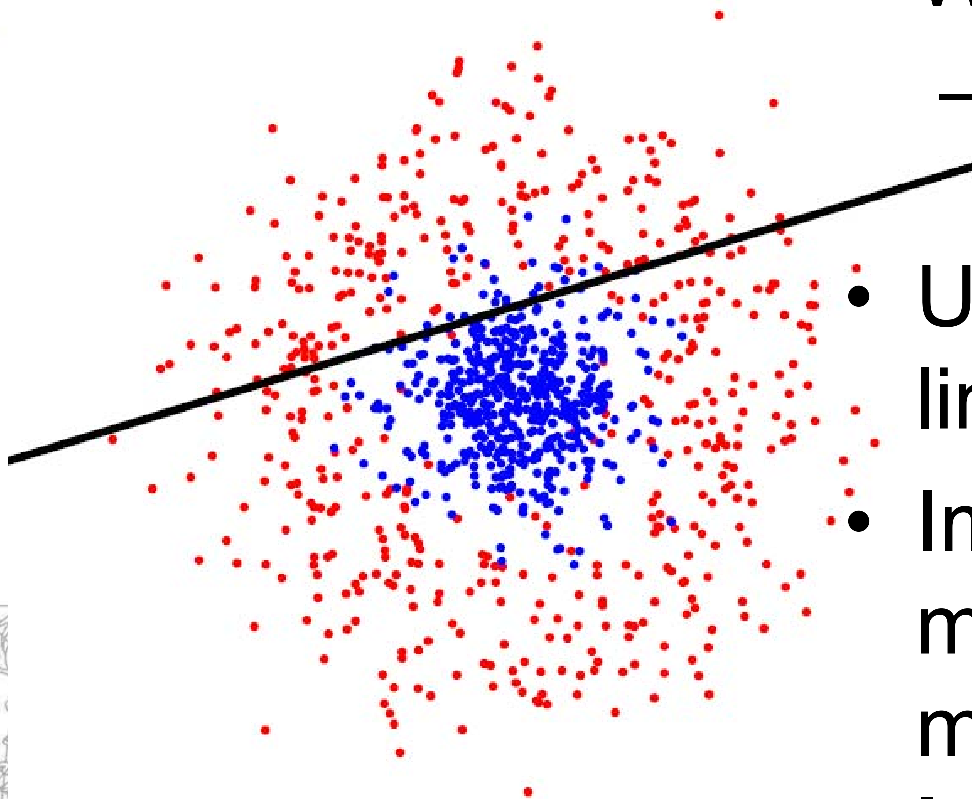
- What can we do?
  - The linear model is wrong

Not Linearly Separable Set



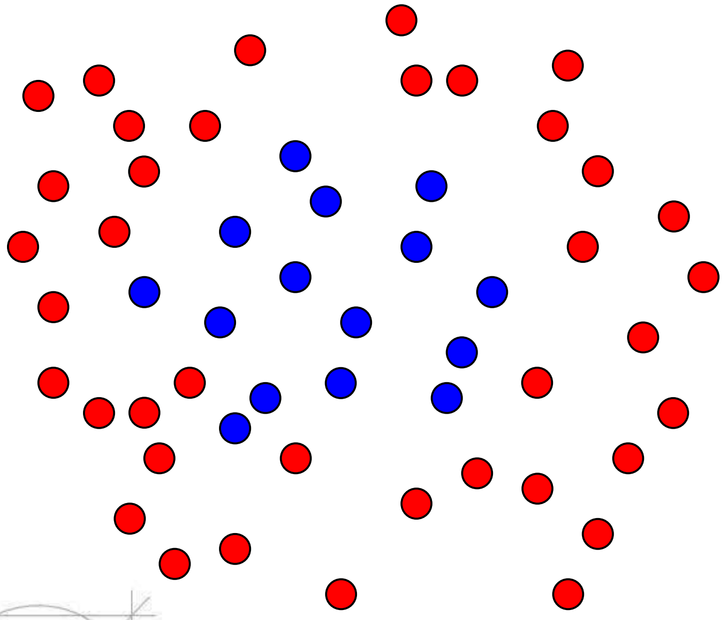
- What can we do?
  - The linear model is wrong
- Use „boosted" linear models

Not Linearly Separable Set



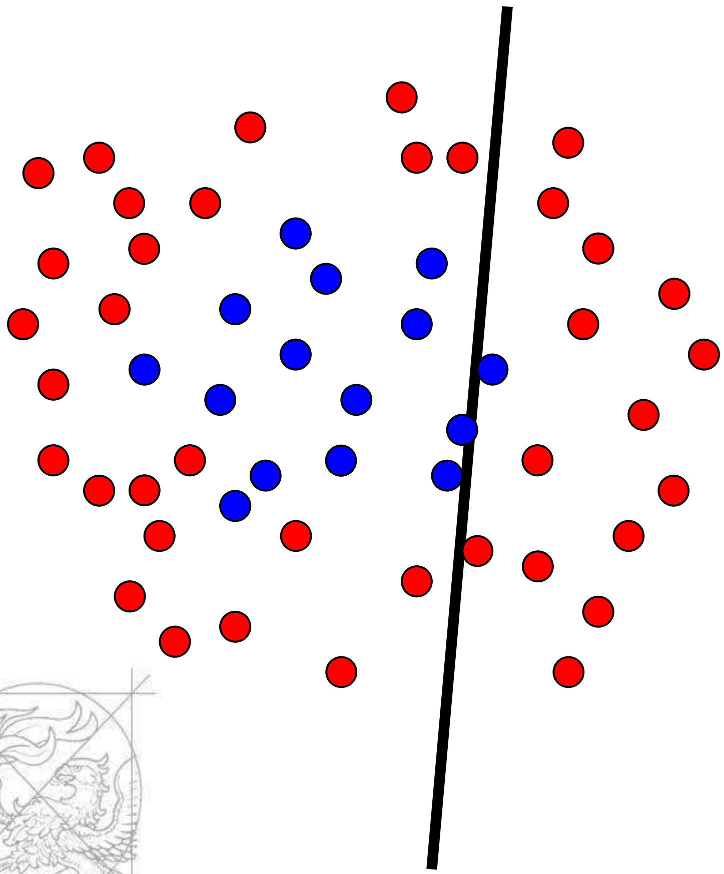
- What can we do?
  - The linear model is wrong
- Use „boosted" linear models
- Improve the performance of the linear models through the **boosting** technique

Boosting linear models



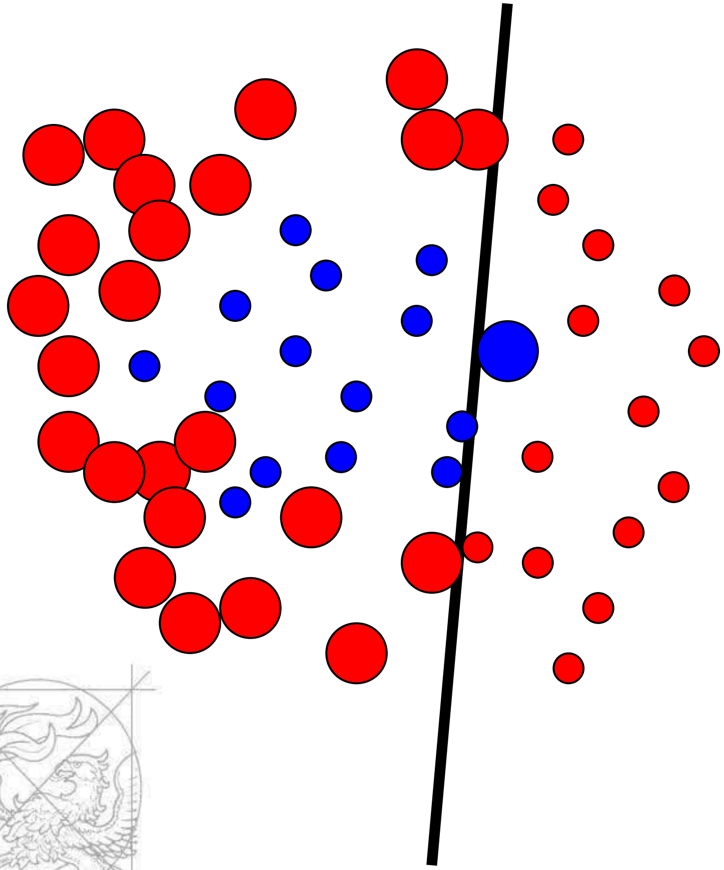
1. **Initializes equal weights for every sample**
2. Classifies instances
3. Re-Weights instances
4. Jump to 2.

Boosting linear models



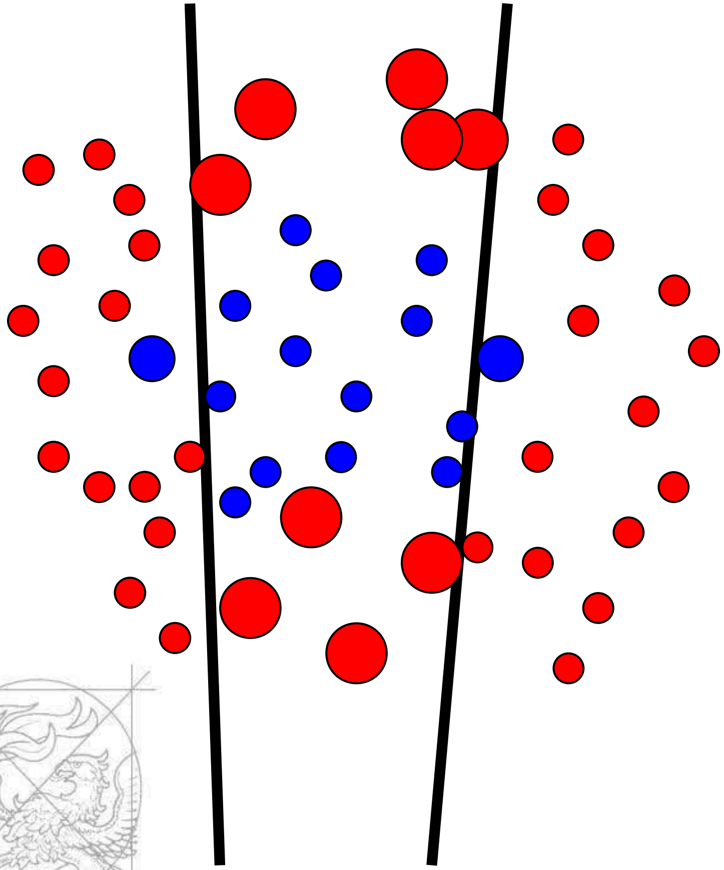
1. Initializes equal weights for every sample
2. **Classifies instances**
3. Re-Weights instances
4. Jump to 2.

Boosting linear models



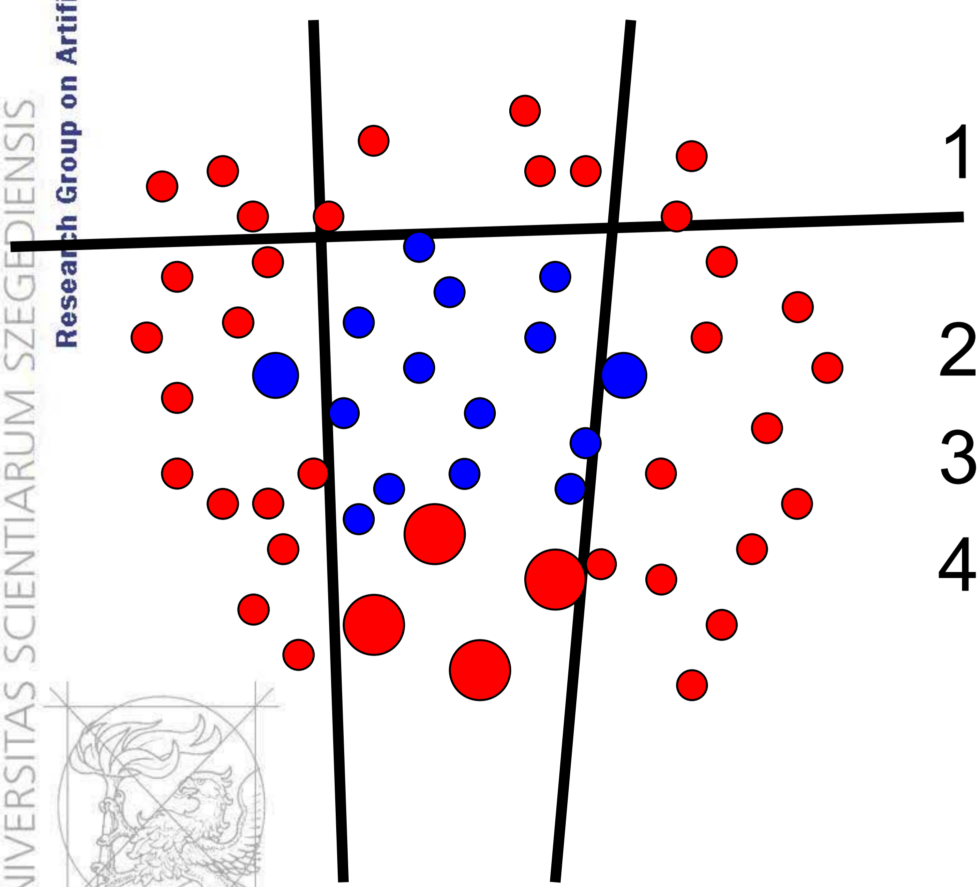
1. Initializes equal weights for every sample
2. Classifies instances
3. **Re-Weights instances**
4. Jump to 2.

Boosting linear models



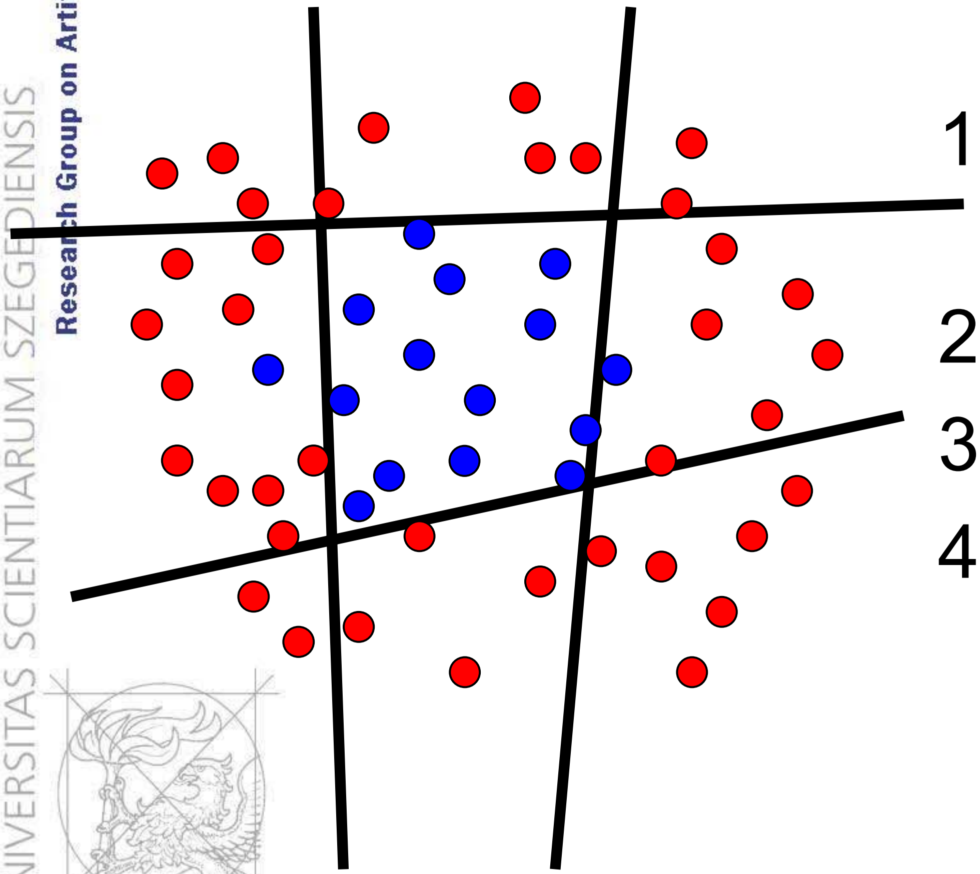
1. Initializes equal weights for every sample
2. Classifies instances
3. Re-Weights instances
4. **Jump to 2.**

Boosting linear models



1. Initialize equal weights for every sample
2. Classifies instances
3. Re-Weights instances
4. Jump to 2.

Boosting linear models



1. Initialize equal weights for every sample
2. Classifies instances
3. Re-Weights instances
4. Jump to 2.

System and Data Model

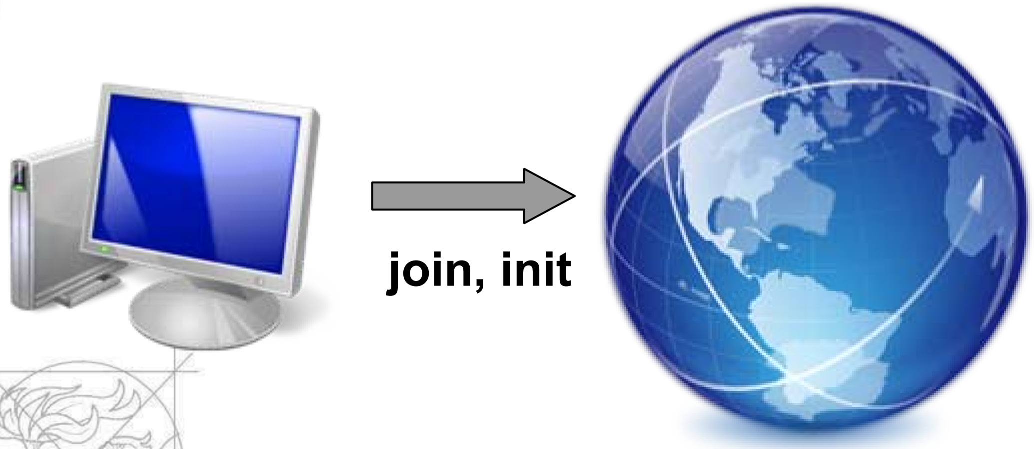
- Given a network of computers (peers or nodes)
- The database is distributed in the network
  - Every node has exactly one training sample → training set size = network size
- Every node can get the address of a randomly selected node from the network
  - using the NewsCast peer sampling service
- Every node can send messages to another node if its address is available
- Finally every node can predict labels locally

GoLF (Gossip Learning Framework)

Algorithm 1 Skeleton of original GoLF learning protocol

```
currentModel ← initModel()
loop
    wait(Δ)
    p ← selectPeer()
    sendModel(p, currentModel)

procedure onReceiveModel(m)
    m.updateModel(x, y)
    currentModel ← m
```

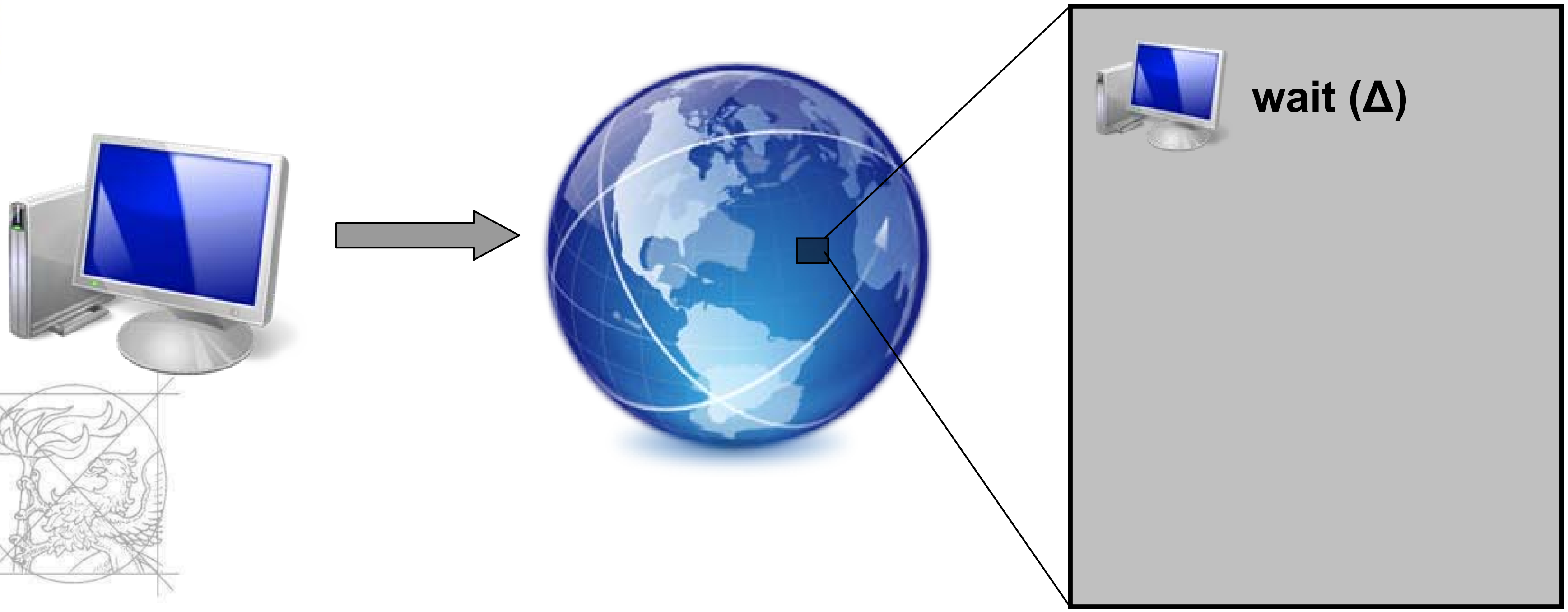


GoLF (Gossip Learning Framework)

Algorithm 1 Skeleton of original GoLF learning protocol

$currentModel \leftarrow$ initModel()
loop
    wait(Δ)
    $p \leftarrow$ selectPeer()
    sendModel(p, $currentModel$)

procedure onReceiveModel(m)
    m.updateModel(x, y)
    $currentModel \leftarrow m$

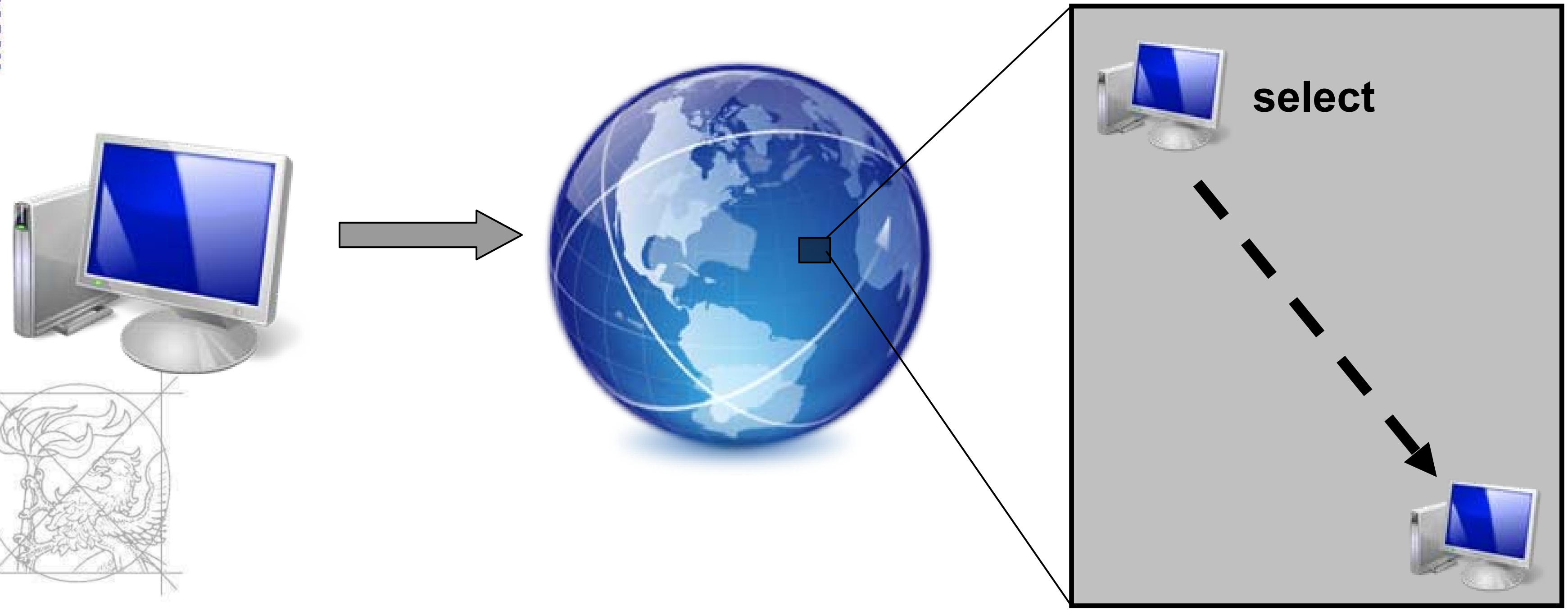


GoLF (Gossip Learning Framework)

Algorithm 1 Skeleton of original GoLF learning protocol

$currentModel \leftarrow$ initModel()
loop
    wait(Δ)
    $p \leftarrow$ selectPeer()
    sendModel(p, $currentModel$)

procedure onReceiveModel(m)
    m.updateModel(x, y)
    $currentModel \leftarrow m$

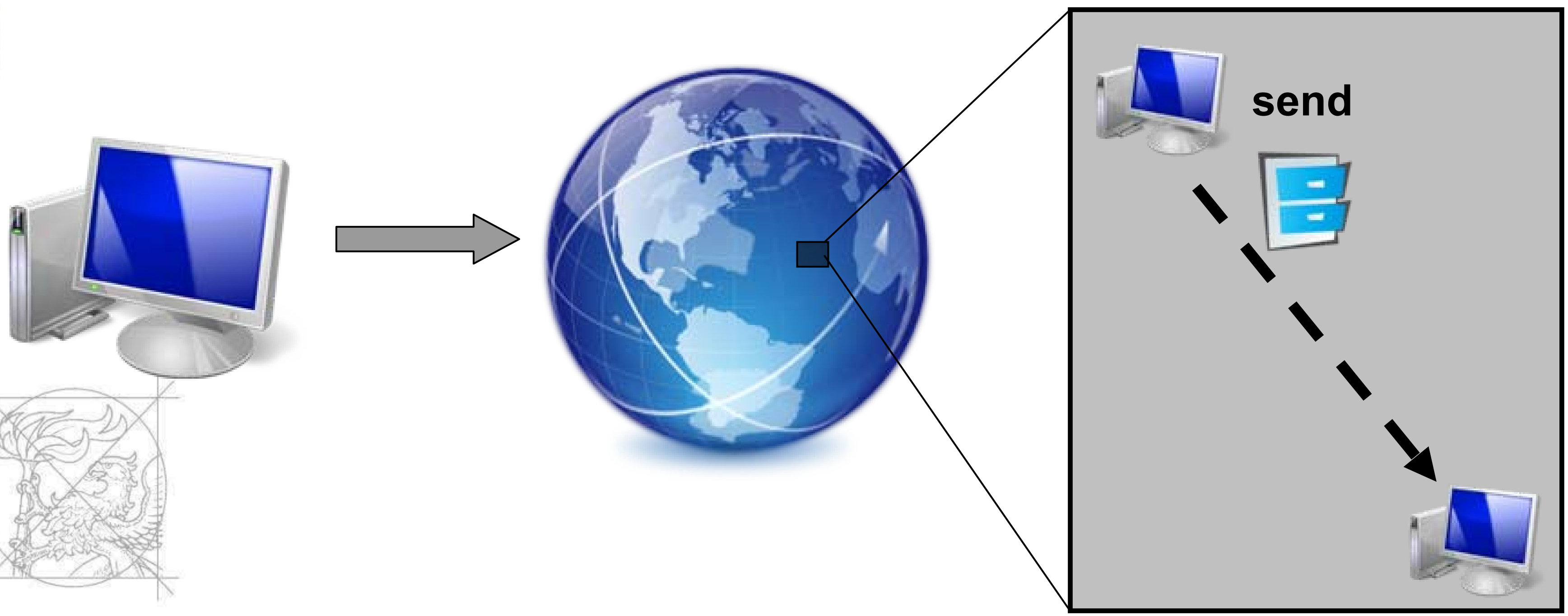


GoLF (Gossip Learning Framework)

Algorithm 1 Skeleton of original GoLF learning protocol

```
currentModel ← initModel()
loop
    wait(Δ)
    p ← selectPeer()
    sendModel(p, currentModel)

procedure onReceiveModel(m)
    m.updateModel(x, y)
    currentModel ← m
```

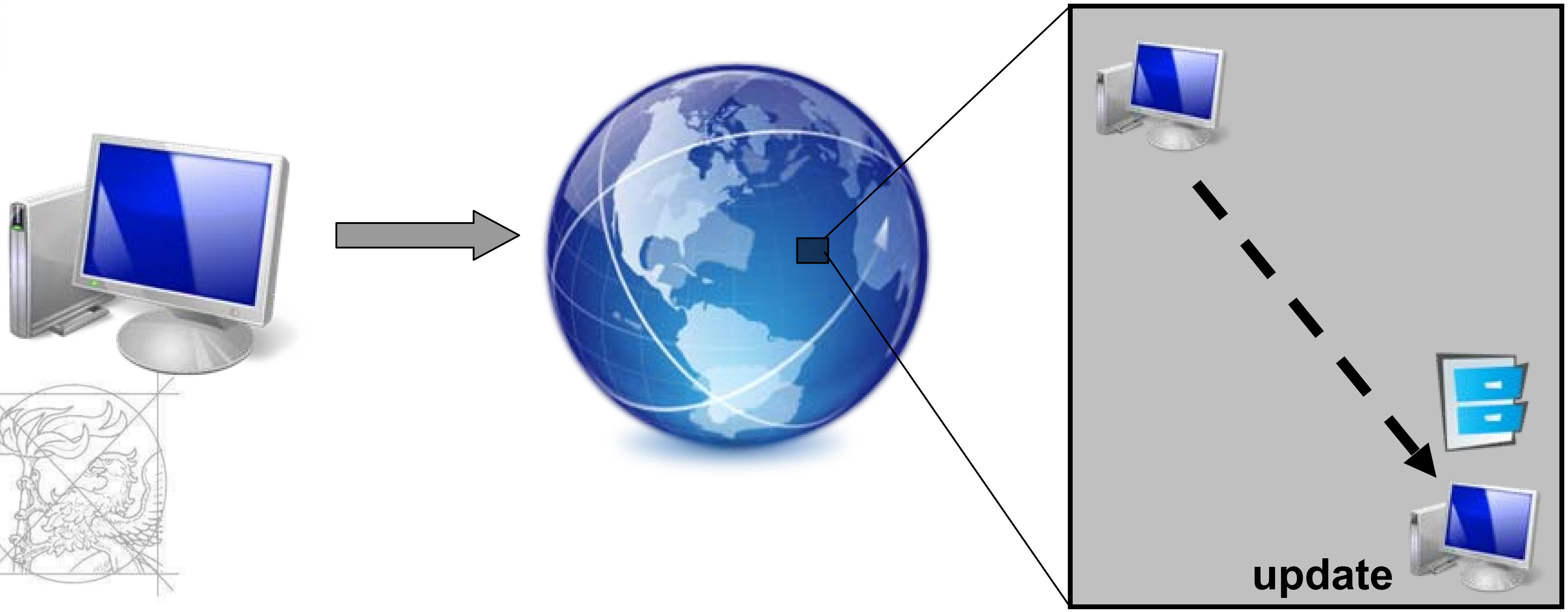


GoLF (Gossip Learning Framework)

Algorithm 1 Skeleton of original GoLF learning protocol

$currentModel \leftarrow$ initModel()
loop
    wait(Δ)
    $p \leftarrow$ selectPeer()
    sendModel(p, $currentModel$)

procedure ONRECEIVEMODEL(m)
    m.updateModel(x, y)
    $currentModel \leftarrow m$

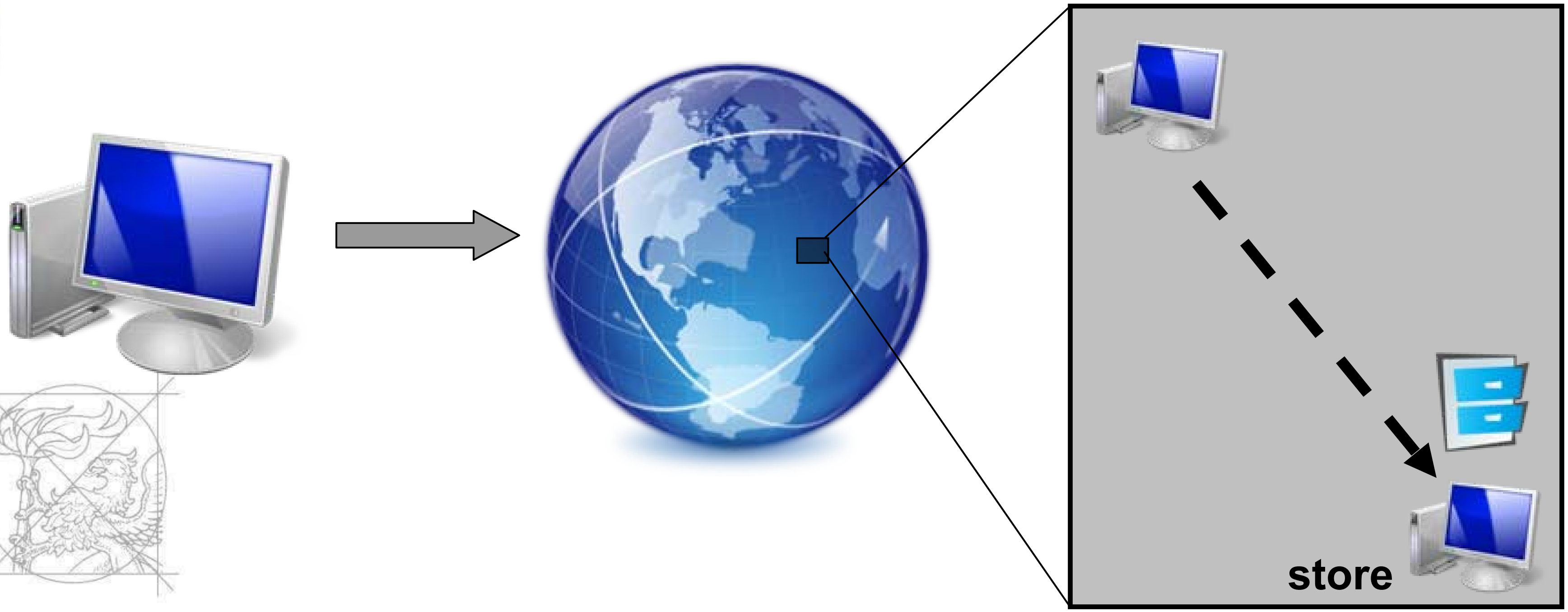


GoLF (Gossip Learning Framework)

Algorithm 1 Skeleton of original GoLF learning protocol

$currentModel \leftarrow \mathrm{initModel}()$
loop
    $\mathrm{wait}(\Delta)$
    $p \leftarrow \mathrm{selectPeer}()$
    $\mathrm{sendModel}(p, currentModel)$

procedure ONRECEIVEMODEL(m)
    $m.\mathrm{updateModel}(x, y)$
    $currentModel \leftarrow m$



GoLF (Gossip Learning Framework)

- Updating the models of the peers through gossiping that
- The models have to be updatable (**online**) → can be optimized by stochastic gradient descent method
- The models converge while make random walks in the network
- Every peer has local model for prediction
- The data never leaves the node

Online FilterBoost

- FilterBoost is a well known and efficient type of boosting algorithms
- We **adopted** a pure **online** version of this algorithm
- We **integrated** this algorithm into our learning framework
- Compared its performance to other state of the art boosting methods

The Online FilterBoost

Algorithm 2 $\text{FilterBoost}(\text{Init}(), \text{Update}(\cdot,\cdot,\cdot,\cdot), T, C)$

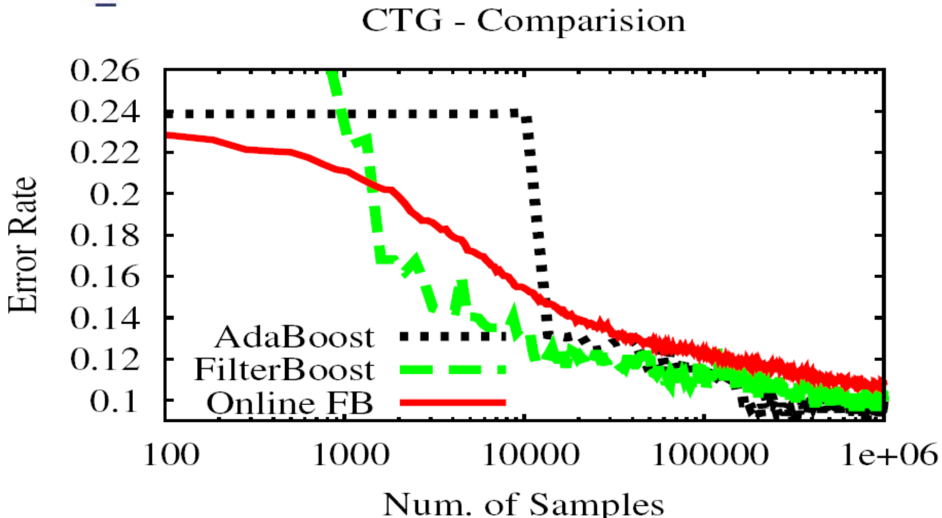
$\mathbf{f}^{(0)}(\mathbf{x}) \leftarrow 0$
for $t \leftarrow 1 \rightarrow T$ **do**
$C_t \leftarrow C \log(t+1)$
$\mathbf{h}^{(t)}(\cdot) \leftarrow \text{Init}()$
for $t' \leftarrow 1 \rightarrow C_t$ **do** ▷ Online base learning
$(\mathbf{x}, \mathbf{y}, \mathbf{w}) \leftarrow \text{Filter}(\mathbf{f}^{(t-1)}(\cdot))$ ▷ Draw a weighted random instance
$\mathbf{h}^{(t)}(\cdot) \leftarrow \text{UPDATE}(\mathbf{x}, \mathbf{y}, \mathbf{w}, \mathbf{h}^{(t)}(\cdot))$
$\gamma \leftarrow 0, W \leftarrow 0$
for $t' \leftarrow 1 \rightarrow C_t$ **do** ▷ Estimate the edge on a filtered data
$(\mathbf{x}, \mathbf{y}, \mathbf{w}) \leftarrow \text{Filter}(\mathbf{f}^{(t-1)}(\cdot))$ ▷ Draw a weighted random instance
$\gamma \leftarrow \gamma + \sum_{\ell}^{K} w_\ell h_\ell^{(t)}(\mathbf{x}) y_\ell,\ W \leftarrow W + \sum_{\ell}^{K} w_\ell$
$\gamma \leftarrow \gamma / W$ ▷ Normalize the edge
$\alpha^{(t)} \leftarrow \frac{1}{2} \log \frac{1+\gamma}{1-\gamma}$
$\mathbf{f}^{(t)}(\cdot) = \mathbf{f}^{(t-1)}(\cdot) + \alpha^{(t)} \mathbf{h}^{(t)}(\cdot)$
return $\mathbf{f}^{(T)}(\cdot) = \sum_{t=1}^{T} \alpha^{(t)} \mathbf{h}^{(t)}(\cdot)$
procedure $\text{Filter}(\mathbf{f}(\cdot))$
$(\mathbf{x}, \mathbf{y}) \leftarrow \text{RandomInstance}()$ ▷ Draw random instance
for $\ell \leftarrow 1 \rightarrow K$ **do**
$w_\ell \leftarrow \frac{\exp\left(f_\ell(\mathbf{x}) - f_{\ell(\mathbf{x})}(\mathbf{x})\right)}{\sum_{\ell'=1}^{K} \exp\left(f_{\ell'}(\mathbf{x}) - f_{\ell(\mathbf{x})}(\mathbf{x})\right)}$
return $(\mathbf{x}, \mathbf{y}, \mathbf{w})$

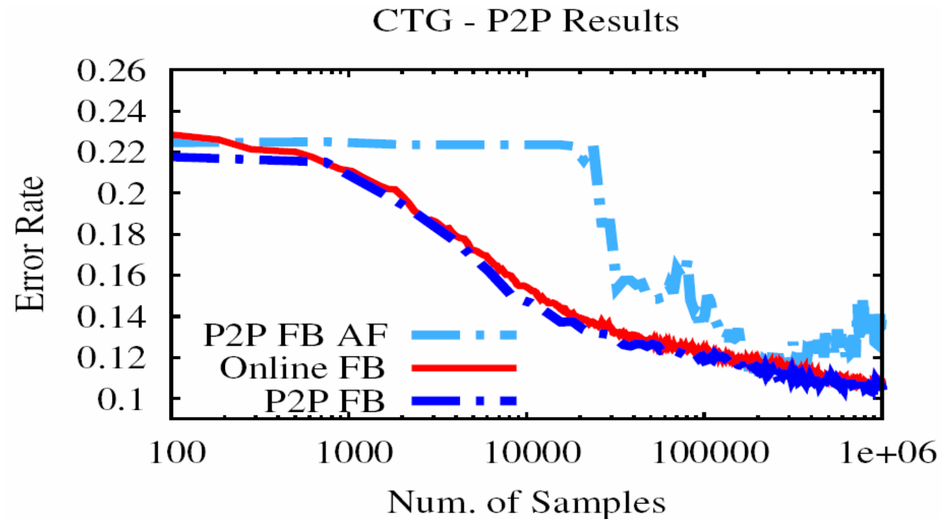
Experimental Setup

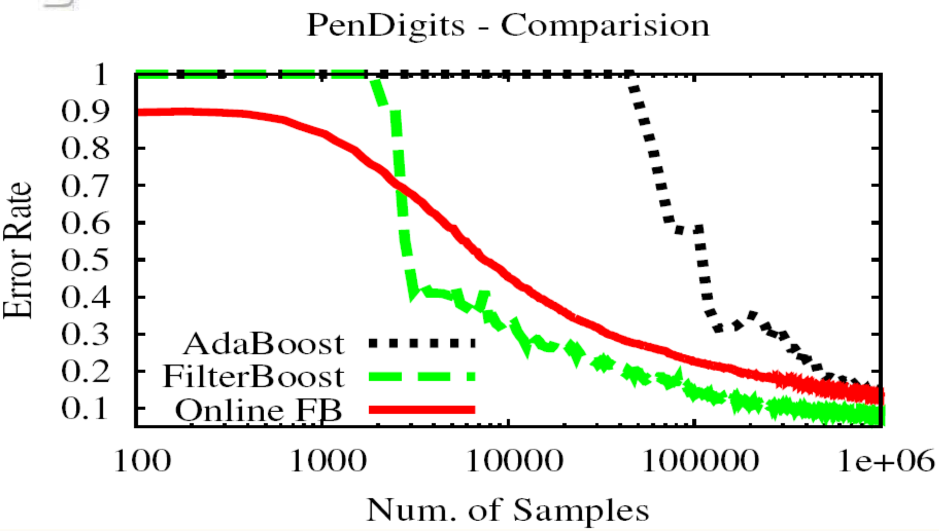
- Peersim simulation environment
- NewsCast peer sampling service
- Baselines: AdaBoost, FilterBoost
- Data sets: CTG, PenDigits, Segmentation
- Modeling environment failures
  - Msg drop, delay and node churn
- Measurement: misclassification ratio

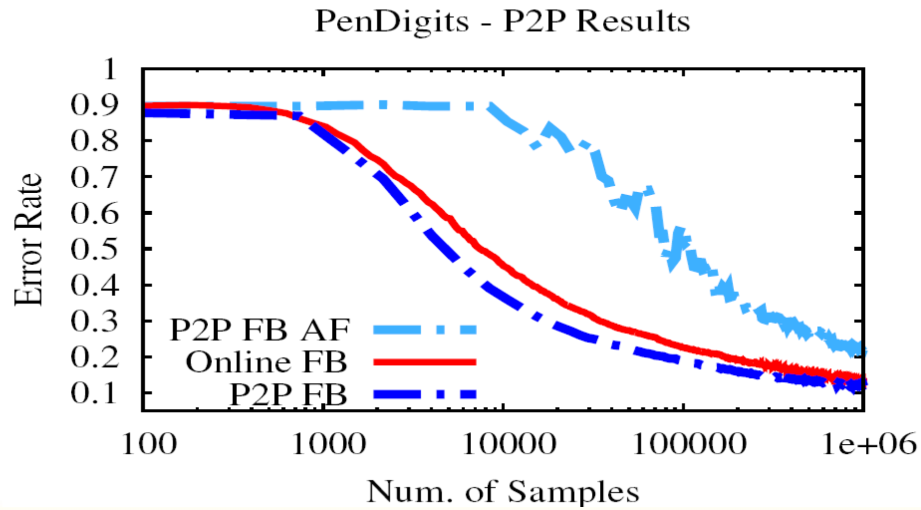
UNIVERSITAS SCIENTIARUM SZEGEDIENSIS
Research Group on Artificial Intelligence

Experimental Results





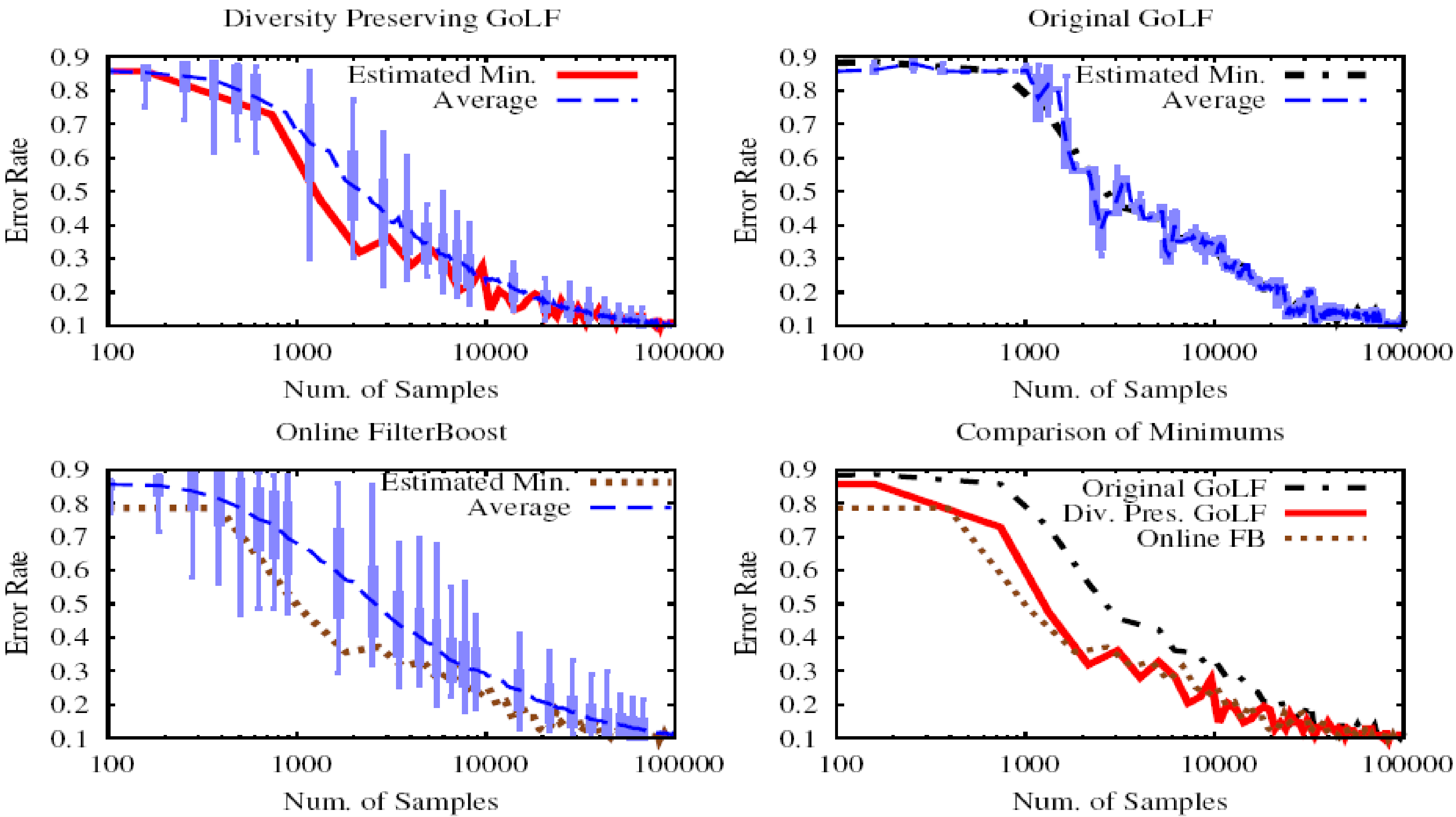




Diversity Preservation

- Since the nodes send only the last received model
  - Some model will be replicated
  - Some model will be die out
- → the diversity of the models will be decreased
- We updated our framework to preserve the diversity
- We can exploit the diversity to have better performance

Diversity Preservation



Summary

- Improved our learning framework
  - Introduced and integrated a state of the art, pure online classification technique
  - Modified the framework for preserving model diversity
- Tested our algorithm in simulated P2P environment
- We achieved good convergence rate and performance compared to the centralized AdaBoost and FilterBoost algorithms
- We showed that our method is tolerant for network failures