Machine Learning Algorithms for Small Datasets

Mohamadsadegh Khosravani
Farnam Mansouri
Marjan Movahedan
Mohammad Hossein Nikravan
Mahsasadat Razavi
Sandra Zilles

University of Regina and Amii
zandra.zilles@gmail.com

June 2026

Introduction

problem: data-hungry machine learning algorithms

- infeasibility of data acquisition
- environmental impact

problem: imbalanced training data

- data points from one class far less common than data points from another class

in practice: often ad-hoc approaches / workarounds for individual problem cases

Introduction

computational learning theory:

- design and analyze domain-independent machine learning algorithms to address such problems
- formal guarantees under idealized conditions
- often aligned with practical performance even when idealized conditions are not met

Introduction

examples:

- Topic 1: Precision-based Boosting of Simple Classifiers
- Topic 2: Active Learning with Uncertainty Sampling
- Topic 3: (Active) Learning from Positive and Unlabeled Data

Topic 1 – Precision-based Boosting of Simple Classifiers

predictions from two people:

- person 1: canola seed
  overall error rate in classifying North American plant seeds: 15%

- person 2: mustard seed
  overall error rate in classifying North American plant seeds: 38%

whom to believe?

Topic 1 – Precision-based Boosting of Simple Classifiers

predictions from two people:

- person 1: canola seed
  overall error rate in classifying North American plant seeds: 15%

- person 2: mustard seed
  overall error rate in classifying North American plant seeds: 38%

whom to believe?

Topic 1 – Precision-based Boosting of Simple Classifiers

predictions from two people:

- person 1: canola seed
  overall error rate in classifying North American plant seeds: 15%
  plant biologist specializing in the anatomy of North American plants
  error rate when assessing food seeds: 29%
- person 2: mustard seed
  overall error rate in classifying North American plant seeds: 38%
  experienced seed tester
  error rate when assessing food seeds: 9%

whom to believe?

Topic 1 – Precision-based Boosting of Simple Classifiers

predictions from 100 experts:

experts predicting canola seed	experts predicting mustard seed
C1: overall error 15%	M1: overall error 38%
C2: overall error 40%	M2: overall error 12%
C3: overall error 27%	M3: overall error 24%
…	…

what to predict? $\rightsquigarrow$ weighted majority vote!

Topic 1 – Precision-based Boosting of Simple Classifiers

predictions from 100 experts:

experts predicting canola seed

C1: overall error 15%
error when predicting canola 29%
C2: overall error 40%
error when predicting canola 26%
C3: overall error 27%
error when predicting canola 39%
…

experts predicting mustard seed

M1: overall error 38%
error when predicting mustard 9%
M2: overall error 12%
error when predicting mustard 16%
M3: overall error 24%
error when predicting mustard 35%
…

what to predict? $\rightsquigarrow$ weighted majority vote!

Topic 1 – Precision-based Boosting of Simple Classifiers

Classification with ensembles

1. train multiple base classifiers
2. prediction on unseen data point x:
   (a) let each base classifier make a prediction on x
   (b) form a weighted average of these predictions, giving more weight to base classifiers with higher training accuracy
   (c) output that weighted average as the prediction for x

idea: refine this weighting method assess **class-specific precision** instead of total accuracy of a base classifier

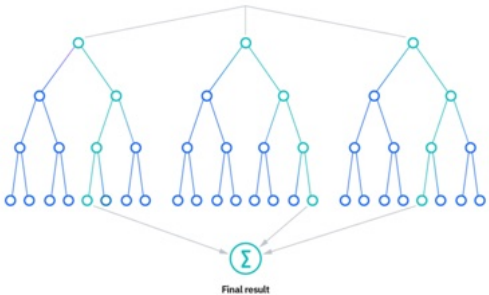
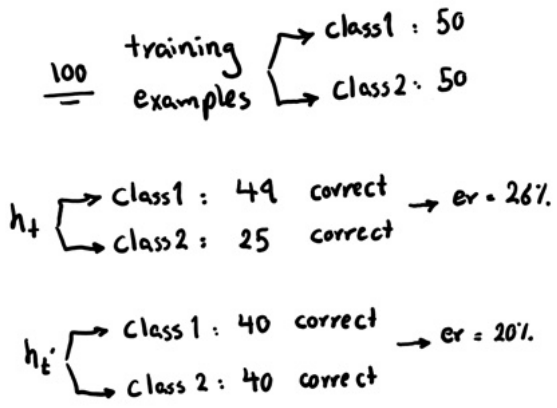


image source:
www.ibm.com/topics/random-forest

An intuitive example:

Why Class-specific Precision?

An intuitive example:



Example: AdaBoost (Freund and Schapire 1997)

1. Train a simple base classifier h_t and observe its performance on the training data
2. Reassign the training data weights
3. Iterate this procedure T times
4. Final classifier: weighted majority vote

AdaBoost trains and aggregates base classifiers into arbitrarily strong classifiers*

Topic 1 – Precision-based Boosting of Simple Classifiers
Precision-based AdaBoost

- Refining the principle behind AdaBoost's weighting scheme
- Assessing **class-specific precision** instead of total accuracy of a base classifier
- **PrAdaBoost**: use this idea both to reweight training examples and to assign weights to base classifiers in the final classifier
- Obtain similar theoretical guarantees as for AdaBoost, plus overall better performance in empirical tests

Topic 1 – Related Work

AdaBoost's theoretical foundation

- Based on the greedy minimization of an exponential loss function

Using AdaBoost's theoretical foundation

- Multi-class version of AdaBoost, using Hamming loss (Schapire and Singer 1999)
- Asymmetric learning methods (Nikolaou and Brown 2015): address imbalances in data/misclassification costs, e.g., AdaCost (Fan et al. 1999)

$\rightsquigarrow$ target modified application settings

One existing method using an idea similar to ours: InfoBoost (Aslam 2000)

Topic 1 – AdaBoost Framework

Algorithm 1: AdaBoost Scheme (Freund and Schapire 1997)

Input: A training set $S = \{(x_1, y_1), \ldots, (x_n, y_n)\}$,
BaseInducer, i.e., a weak learning algorithm,
Number of iterations $T \in \mathbb{N}$.
Initialize $D_1(x_i) = \frac{1}{n}$ for all $i \in \{1, \ldots, n\}$.
For $t = 1$ to T **do**

1. Call BaseInducer with the distribution D_t to obtain a hypothesis $h_t : X \to Y$.
2. Calculate the total error ε_t of h_t w.r.t. D_t, i.e.,

$$\varepsilon_t = \sum_{i=1}^{n} D_t(x_i) \mathbb{I}(h_t(x_i) \neq y_i).$$

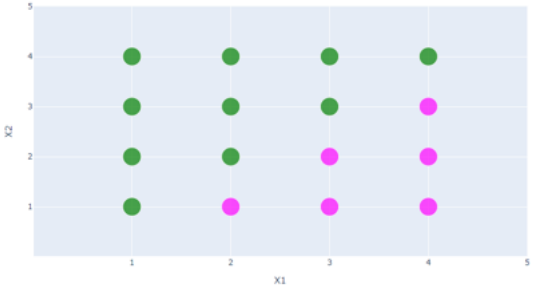
   If $\varepsilon_t > 0.5$, then set T to $t - 1$ and abort the loop.
3. Set β_t as a decreasing function of ε_t.
4. Redistribute weights as follows, for all $i \in \{1, \ldots, n\}$:

$$D_{t+1}(x_i) = \frac{D_t(x_i)}{Z_t} \times \begin{cases} e^{-\beta_t} & \text{if } h_t(x_i) = y_i \\ e^{\beta_t} & \text{otherwise} \end{cases}$$

   where Z_t is defined such that $\sum_{i=1}^{n} D_{t+1}(x_i) = 1$.

end
Output: the final hypothesis

$$H(x) = \begin{cases} 1, & \text{if } \sum_{t=1}^{T} \beta_t h_t(x) \geq 0, \\ -1, & \text{otherwise}. \end{cases}$$



Topic 1 – AdaBoost Framework

Algorithm 1: AdaBoost Scheme (Freund and Schapire 1997)

Input: A training set $S = \{(x_1, y_1), \ldots, (x_n, y_n)\}$,
BaseInducer, i.e., a weak learning algorithm,
Number of iterations $T \in \mathbb{N}$.
Initialize $D_1(x_i) = \frac{1}{n}$ for all $i \in \{1, \ldots, n\}$.
For $t = 1$ to T **do**

1. Call BaseInducer with the distribution D_t to obtain a hypothesis $h_t : X \to Y$.
2. Calculate the total error ε_t of h_t w.r.t. D_t, i.e.,

$$\varepsilon_t = \sum_{i=1}^{n} D_t(x_i) \mathbb{I}(h_t(x_i) \neq y_i).$$

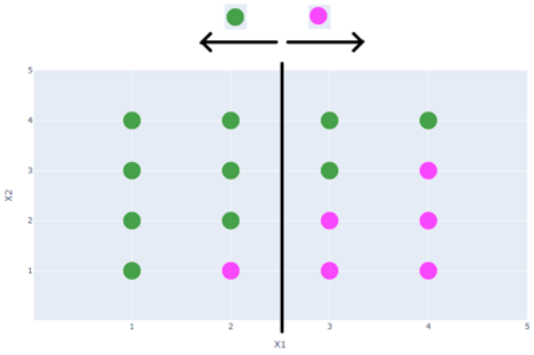
   If $\varepsilon_t > 0.5$, then set T to $t - 1$ and abort the loop.
3. Set β_t as a decreasing function of ε_t.
4. Redistribute weights as follows, for all $i \in \{1, \ldots, n\}$:

$$D_{t+1}(x_i) = \frac{D_t(x_i)}{Z_t} \times \begin{cases} e^{-\beta_t} & \text{if } h_t(x_i) = y_i \\ e^{\beta_t} & \text{otherwise} \end{cases}$$

   where Z_t is defined such that $\sum_{i=1}^{n} D_{t+1}(x_i) = 1$.

end
Output: the final hypothesis

$$H(x) = \begin{cases} 1, & \text{if } \sum_{t=1}^{T} \beta_t h_t(x) \geq 0, \\ -1, & \text{otherwise}. \end{cases}$$



Topic 1 – AdaBoost Framework

Algorithm 1: AdaBoost Scheme (Freund and Schapire 1997)

Input: A training set $S = \{(x_1, y_1), \ldots, (x_n, y_n)\}$,
BaseInducer, i.e., a weak learning algorithm,
Number of iterations $T \in \mathbb{N}$.
Initialize $D_1(x_i) = \frac{1}{n}$ for all $i \in \{1, \ldots, n\}$.
For $t = 1$ to T **do**

1. Call BaseInducer with the distribution D_t to obtain a hypothesis $h_t : X \to Y$.
2. Calculate the total error ε_t of h_t w.r.t. D_t, i.e.,

$$\varepsilon_t = \sum_{i=1}^{n} D_t(x_i) \mathbb{I}(h_t(x_i) \neq y_i).$$

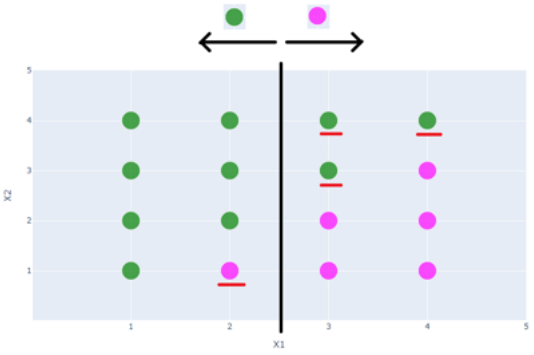
   If $\varepsilon_t > 0.5$, then set T to $t - 1$ and abort the loop.
3. Set β_t as a decreasing function of ε_t.
4. Redistribute weights as follows, for all $i \in \{1, \ldots, n\}$:

$$D_{t+1}(x_i) = \frac{D_t(x_i)}{Z_t} \times \begin{cases} e^{-\beta_t} & \text{if } h_t(x_i) = y_i \\ e^{\beta_t} & \text{otherwise} \end{cases}$$

   where Z_t is defined such that $\sum_{i=1}^{n} D_{t+1}(x_i) = 1$.

end
Output: the final hypothesis

$$H(x) = \begin{cases} 1, & \text{if } \sum_{t=1}^{T} \beta_t h_t(x) \geq 0, \\ -1, & \text{otherwise}. \end{cases}$$



$\varepsilon_1 = 0.25$

Topic 1 – AdaBoost Framework

Algorithm 1: AdaBoost Scheme (Freund and Schapire 1997)

Input: A training set $S = \{(x_1, y_1), \ldots, (x_n, y_n)\}$,
BaseInducer, i.e., a weak learning algorithm,
Number of iterations $T \in \mathbb{N}$.
Initialize $D_1(x_i) = \frac{1}{n}$ for all $i \in \{1, \ldots, n\}$.
For $t = 1$ to T **do**

1. Call BaseInducer with the distribution D_t to obtain a hypothesis $h_t : X \to Y$.
2. Calculate the total error ε_t of h_t w.r.t. D_t, i.e.,

$$\varepsilon_t = \sum_{i=1}^{n} D_t(x_i) \mathbb{I}(h_t(x_i) \neq y_i).$$

If $\varepsilon_t > 0.5$, then set T to $t - 1$ and abort the loop.

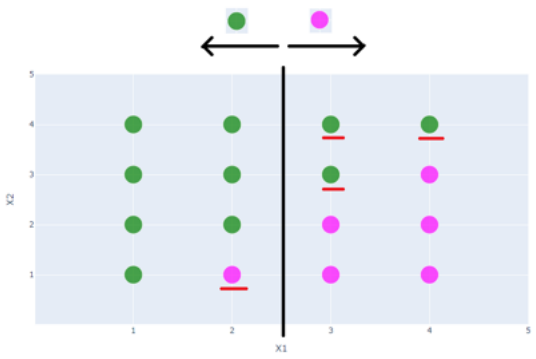
3. Set β_t as a decreasing function of ε_t.
4. Redistribute weights as follows, for all $i \in \{1, \ldots, n\}$:

$$D_{t+1}(x_i) = \frac{D_t(x_i)}{Z_t} \times \begin{cases} e^{-\beta_t} & \text{if } h_t(x_i) = y_i \\ e^{\beta_t} & \text{otherwise} \end{cases}$$

where Z_t is defined such that $\sum_{i=1}^{n} D_{t+1}(x_i) = 1$.

end
Output: the final hypothesis

$$H(x) = \begin{cases} 1, & \text{if } \sum_{t=1}^{T} \beta_t h_t(x) \geq 0, \\ -1, & \text{otherwise}. \end{cases}$$



$$\beta_t = \frac{1}{2} \ln \frac{1 - \varepsilon_t}{\varepsilon_t}$$

$\varepsilon_1 = 0.25$

$\beta_1 = 0.55$

Topic 1 – AdaBoost Framework

Algorithm 1: AdaBoost Scheme (Freund and Schapire 1997)

Input: A training set $S = \{(x_1, y_1), \ldots, (x_n, y_n)\}$,
BaseInducer, i.e., a weak learning algorithm,
Number of iterations $T \in \mathbb{N}$.
Initialize $D_1(x_i) = \frac{1}{n}$ for all $i \in \{1, \ldots, n\}$.
For $t = 1$ to T **do**

1. Call BaseInducer with the distribution D_t to obtain a hypothesis $h_t : X \to Y$.
2. Calculate the total error ε_t of h_t w.r.t. D_t, i.e.,

$$\varepsilon_t = \sum_{i=1}^{n} D_t(x_i) \mathbb{I}(h_t(x_i) \neq y_i).$$

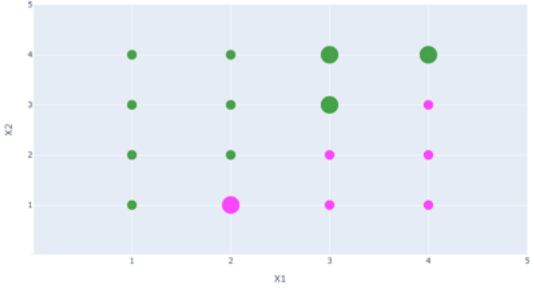
   If $\varepsilon_t > 0.5$, then set T to $t - 1$ and abort the loop.
3. Set β_t as a decreasing function of ε_t.
4. Redistribute weights as follows, for all $i \in \{1, \ldots, n\}$:

$$D_{t+1}(x_i) = \frac{D_t(x_i)}{Z_t} \times \begin{cases} e^{-\beta_t} & \text{if } h_t(x_i) = y_i \\ e^{\beta_t} & \text{otherwise} \end{cases}$$

   where Z_t is defined such that $\sum_{i=1}^{n} D_{t+1}(x_i) = 1$.

end
Output: the final hypothesis

$$H(x) = \begin{cases} 1, & \text{if } \sum_{t=1}^{T} \beta_t h_t(x) \geq 0, \\ -1, & \text{otherwise}. \end{cases}$$



Topic 1 – Precision-based Boosting

- AdaBoost weights a classifier based on β_t
- Our method: one more degree of freedom
- Two values based on h_t's predictions: $\beta_{t,\text{canola}}$ and $\beta_{t,\text{mustard}}$
- AdaBoost: mathematical approach (Friedman, Hastie, and Tibshirani 2000)
  $\rightsquigarrow$ derivation of optimal β_t
- PrAdaBoost: same approach to derive $\beta_{t,\text{canola}}$ and $\beta_{t,\text{mustard}}$

Topic 1 – PrAdaBoost

Algorithm 2: PrAdaBoost

Input: A training set $S = \{(x_1, y_1), \ldots, (x_n, y_n)\}$,
BaseInducer, i.e., a weak learning algorithm,
Number of iterations $T \in \mathbb{N}$.
Initialize $D_1(x_i) = \frac{1}{n}$ for all $i \in \{1, \ldots, n\}$.
For $t = 1$ to T **do**

1. Call BaseInducer with the distribution D_t to obtain a hypothesis $h_t : X \to Y$.
2. If $\varepsilon_t > \min_y r_{t,y}$, set T to $t - 1$ and abort the loop.
3. Set $\beta_{t,1} = \frac{1}{2} \ln \frac{r_{t,1} - \varepsilon_{t,-1}}{\varepsilon_{t,1}}$ and $\beta_{t,-1} = \frac{1}{2} \ln \frac{r_{t,-1} - \varepsilon_{t,1}}{\varepsilon_{t,-1}}$.
4. Redistribute weights as follows, for all $i \in \{1, \ldots, n\}$:

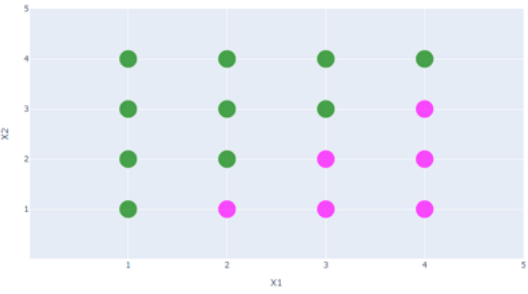
$$D_{t+1}(x_i) = \frac{D_t(x_i)}{Z_t} \times \begin{cases} e^{-\beta_{t,h_t(x_i)}} & \text{if } h_t(x_i) = y_i \\ e^{\beta_{t,h_t(x_i)}} & \text{if } h_t(x_i) \neq y_i \end{cases}$$

where Z_t is defined such that $\sum_{i=1}^{n} D_{t+1}(x_i) = 1$.

end
Output: the final hypothesis

$$H(x) = \begin{cases} h_{t_\infty}(x), & \text{if } T_\infty(x) \neq \emptyset,\ t_\infty = \min T_\infty(x), \\ 1, & \text{else if } \sum_{t=1}^{T} \beta_{t,h_t(x)} h_t(x) \geq 0, \\ -1, & \text{otherwise}. \end{cases}$$

$$T_\infty(x) = \{t \mid \beta_{t,h_t(x)} = \infty\}$$



Topic 1 – PrAdaBoost

Algorithm 2: PrAdaBoost

Input: A training set $S = \{(x_1, y_1), \ldots, (x_n, y_n)\}$,
BaseInducer, i.e., a weak learning algorithm,
Number of iterations $T \in \mathbb{N}$.
Initialize $D_1(x_i) = \frac{1}{n}$ for all $i \in \{1, \ldots, n\}$.
For $t = 1$ to T **do**

1. Call BaseInducer with the distribution D_t to obtain a hypothesis $h_t : X \to Y$.
2. If $\varepsilon_t > \min_y r_{t,y}$, set T to $t - 1$ and abort the loop.
3. Set $\beta_{t,1} = \frac{1}{2} \ln \frac{r_{t,1} - \varepsilon_{t,-1}}{\varepsilon_{t,1}}$ and $\beta_{t,-1} = \frac{1}{2} \ln \frac{r_{t,-1} - \varepsilon_{t,1}}{\varepsilon_{t,-1}}$.
4. Redistribute weights as follows, for all $i \in \{1, \ldots, n\}$:

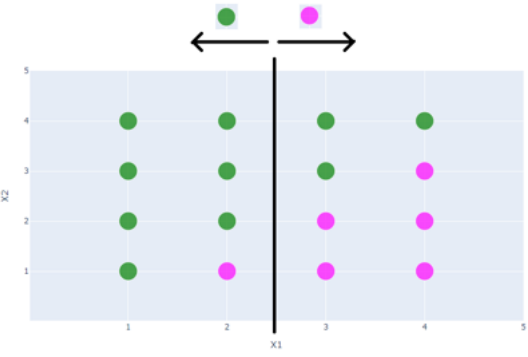
$$D_{t+1}(x_i) = \frac{D_t(x_i)}{Z_t} \times \begin{cases} e^{-\beta_{t,h_t(x_i)}} & \text{if } h_t(x_i) = y_i \\ e^{\beta_{t,h_t(x_i)}} & \text{if } h_t(x_i) \neq y_i \end{cases}$$

where Z_t is defined such that $\sum_{i=1}^{n} D_{t+1}(x_i) = 1$.

end
Output: the final hypothesis

$$H(x) = \begin{cases} h_{t_\infty}(x), & \text{if } T_\infty(x) \neq \emptyset,\ t_\infty = \min T_\infty(x), \\ 1, & \text{else if } \sum_{t=1}^{T} \beta_{t,h_t(x)} h_t(x) \geq 0, \\ -1, & \text{otherwise}. \end{cases}$$

$$T_\infty(x) = \{t \mid \beta_{t,h_t(x)} = \infty\}$$



Topic 1 – PrAdaBoost

Algorithm 2: PrAdaBoost

Input: A training set $S = \{(x_1, y_1), \ldots, (x_n, y_n)\}$,
BaseInducer, i.e., a weak learning algorithm,
Number of iterations $T \in \mathbb{N}$.
Initialize $D_1(x_i) = \frac{1}{n}$ for all $i \in \{1, \ldots, n\}$.
For $t = 1$ to T **do**

1. Call BaseInducer with the distribution D_t to obtain a hypothesis $h_t : X \to Y$.
2. If $\varepsilon_t > \min_y r_{t,y}$, set T to $t-1$ and abort the loop.
3. Set $\beta_{t,1} = \frac{1}{2} \ln \frac{r_{t,1} - \varepsilon_{t,-1}}{\varepsilon_{t,1}}$ and $\beta_{t,-1} = \frac{1}{2} \ln \frac{r_{t,-1} - \varepsilon_{t,1}}{\varepsilon_{t,-1}}$.
4. Redistribute weights as follows, for all $i \in \{1, \ldots, n\}$:

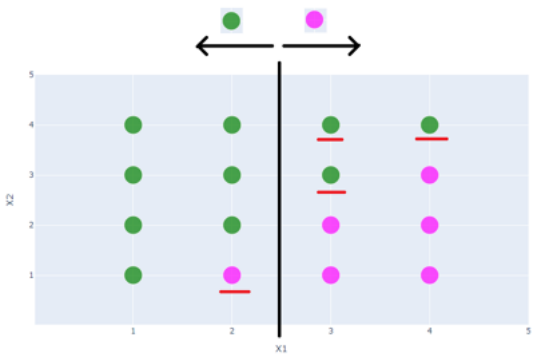
$$D_{t+1}(x_i) = \frac{D_t(x_i)}{Z_t} \times \begin{cases} e^{-\beta_{t,h_t(x_i)}} \text{ if } h_t(x_i) = y_i \\ e^{\beta_{t,h_t(x_i)}} \text{ if } h_t(x_i) \neq y_i \end{cases}$$

where Z_t is defined such that $\sum_{i=1}^{n} D_{t+1}(x_i) = 1$.

end
Output: the final hypothesis

$$H(x) = \begin{cases} h_{t_\infty}(x), & \text{if } T_\infty(x) \neq \emptyset,\ t_\infty = \min T_\infty(x), \\ 1, & \text{else if } \sum_{t=1}^{T} \beta_{t,h_t(x)} h_t(x) \geq 0, \\ -1, & \text{otherwise.} \end{cases}$$

$$T_\infty(x) = \{t \mid \beta_{t,h_t(x)} = \infty\}$$



ε_1 = 4/16

$\varepsilon_{1\bullet}$ = 3/16
$\varepsilon_{1\bullet}$ = 1/16

$r_{1\bullet}$ = 6/16
$r_{1\bullet}$ = 10/16

Topic 1 – PrAdaBoost

Algorithm 2: PrAdaBoost

Input: A training set $S = \{(x_1, y_1), \ldots, (x_n, y_n)\}$,
BaseInducer, i.e., a weak learning algorithm,
Number of iterations $T \in \mathbb{N}$.
Initialize $D_1(x_i) = \frac{1}{n}$ for all $i \in \{1, \ldots, n\}$.
For $t = 1$ to T **do**

1. Call BaseInducer with the distribution D_t to obtain a hypothesis $h_t : X \to Y$.
2. If $\varepsilon_t > \min_y r_{t,y}$, set T to $t-1$ and abort the loop.
3. Set $\beta_{t,1} = \frac{1}{2} \ln \frac{r_{t,1} - \varepsilon_{t,-1}}{\varepsilon_{t,1}}$ and $\beta_{t,-1} = \frac{1}{2} \ln \frac{r_{t,-1} - \varepsilon_{t,1}}{\varepsilon_{t,-1}}$.
4. Redistribute weights as follows, for all $i \in \{1, \ldots, n\}$:

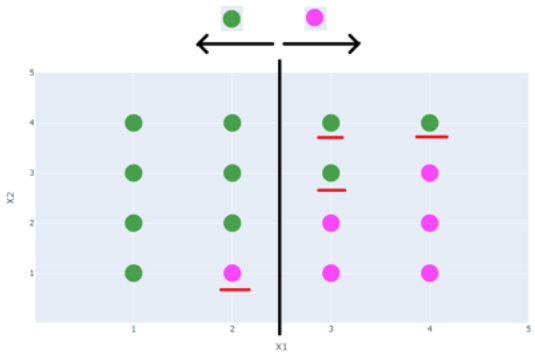
$$D_{t+1}(x_i) = \frac{D_t(x_i)}{Z_t} \times \begin{cases} e^{-\beta_{t,h_t(x_i)}} & \text{if } h_t(x_i) = y_i \\ e^{\beta_{t,h_t(x_i)}} & \text{if } h_t(x_i) \neq y_i \end{cases}$$

where Z_t is defined such that $\sum_{i=1}^{n} D_{t+1}(x_i) = 1$.

end
Output: the final hypothesis

$$H(x) = \begin{cases} h_{t_\infty}(x), & \text{if } T_\infty(x) \neq \emptyset,\ t_\infty = \min T_\infty(x), \\ 1, & \text{else if } \sum_{t=1}^{T} \beta_{t,h_t(x)} h_t(x) \geq 0, \\ -1, & \text{otherwise.} \end{cases}$$

$$T_\infty(x) = \{t \mid \beta_{t,h_t(x)} = \infty\}$$



ε_1 = 4/16

$\varepsilon_{1\,\text{●}}$ = 3/16
$\varepsilon_{1\,\text{●}}$ = 1/16

$r_{1\,\text{●}}$ = 6/16
$r_{1\,\text{●}}$ = 10/16

$\beta_{1\,\text{●}}$ = 0.26
$\beta_{1\,\text{●}}$ = 0.97

Topic 1 – PrAdaBoost

Algorithm 2: PrAdaBoost

Input: A training set $S = \{(x_1, y_1), \ldots, (x_n, y_n)\}$,
BaseInducer, i.e., a weak learning algorithm,
Number of iterations $T \in \mathbb{N}$.
Initialize $D_1(x_i) = \frac{1}{n}$ for all $i \in \{1, \ldots, n\}$.
For $t = 1$ to T **do**

1. Call BaseInducer with the distribution D_t to obtain a hypothesis $h_t : X \to Y$.
2. If $\varepsilon_t > \min_y r_{t,y}$, set T to $t - 1$ and abort the loop.
3. Set $\beta_{t,1} = \frac{1}{2} \ln \frac{r_{t,1} - \varepsilon_{t,-1}}{\varepsilon_{t,1}}$ and $\beta_{t,-1} = \frac{1}{2} \ln \frac{r_{t,-1} - \varepsilon_{t,1}}{\varepsilon_{t,-1}}$.
4. Redistribute weights as follows, for all $i \in \{1, \ldots, n\}$:

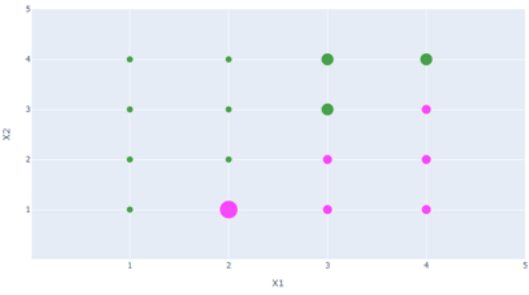
$$D_{t+1}(x_i) = \frac{D_t(x_i)}{Z_t} \times \begin{cases} e^{-\beta_{t,h_t(x_i)}} & \text{if } h_t(x_i) = y_i \\ e^{\beta_{t,h_t(x_i)}} & \text{if } h_t(x_i) \neq y_i \end{cases}$$

where Z_t is defined such that $\sum_{i=1}^{n} D_{t+1}(x_i) = 1$.

end
Output: the final hypothesis

$$H(x) = \begin{cases} h_{t_\infty}(x), & \text{if } T_\infty(x) \neq \emptyset,\ t_\infty = \min T_\infty(x), \\ 1, & \text{else if } \sum_{t=1}^{T} \beta_{t,h_t(x)} h_t(x) \geq 0, \\ -1, & \text{otherwise.} \end{cases}$$

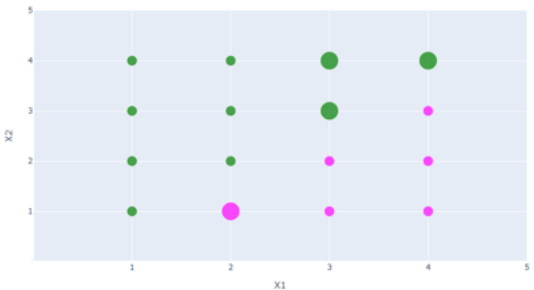
$$T_\infty(x) = \{t \mid \beta_{t,h_t(x)} = \infty\}$$



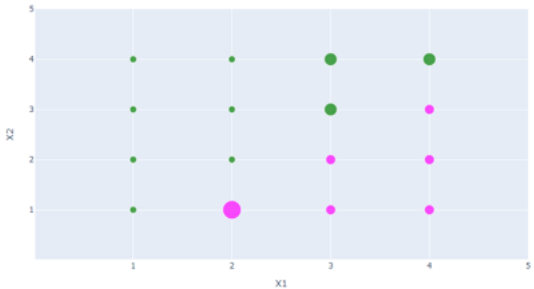
Topic 1 – AdaBoost vs PrAdaBoost
Distribution D_2

Training sample distribution after one iteration

AdaBoost



PrAdaBoost



Topic 1 – Formal Analysis

- AdaBoost
  - boosts weak learners into a strong classifier
  - upper bound on training error decreases exponentially quickly with time (Freund and Schapire 1996; Mohri, Rostamizadeh, and Talwalkar 2018)

$$\epsilon_H \leq \prod_{t=1}^{T} 2\sqrt{\epsilon_t(1-\epsilon_t)},$$

- PrAdaBoost
  - training error upper bound is less than or equal to that for AdaBoost
  - …and in some cases, *much* smaller

A hypothetical situation:

- Goal: train classifier H with $\epsilon_H \leq 0.02$
- Assumptions:

  $\epsilon_t \leq 0.48$
  $r_{t,1} \leq 0.4$ and $r_{t,-1} \geq 0.6$
  $\epsilon_{t,1} \leq 0.285$ and $\epsilon_{t,-1} \leq 0.195$

- With a pessimistic estimate for both algorithms:

  PrAdaBoost meets goal at $T \geq 186$
  AdaBoost meets goal at $T \geq 4828$

Topic 1 – Variants for Other Application Settings

So far: classification into two classes

- extension to multi-class classification (e.g., multiple seed types)
- extension to regression (prediction of a numerical value)

Experimental setup

- 23 binary UCI datasets (Lichman 2013)
- 18 multi-class UCI datasets
- 16 UCI regression datasets

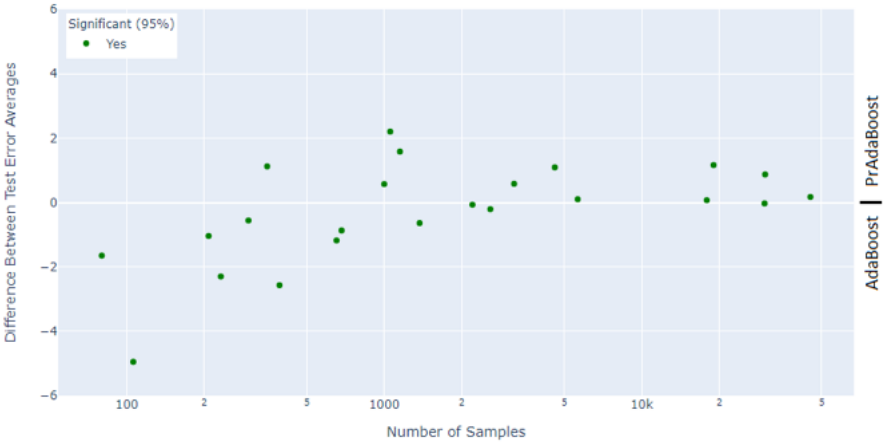
Topic 1 – Empirical Analysis
Results – Comparison to InfoBoost

- Aslam (2000) proposed InfoBoost as a first AdaBoost-type method using a similar notion of class-specific precision

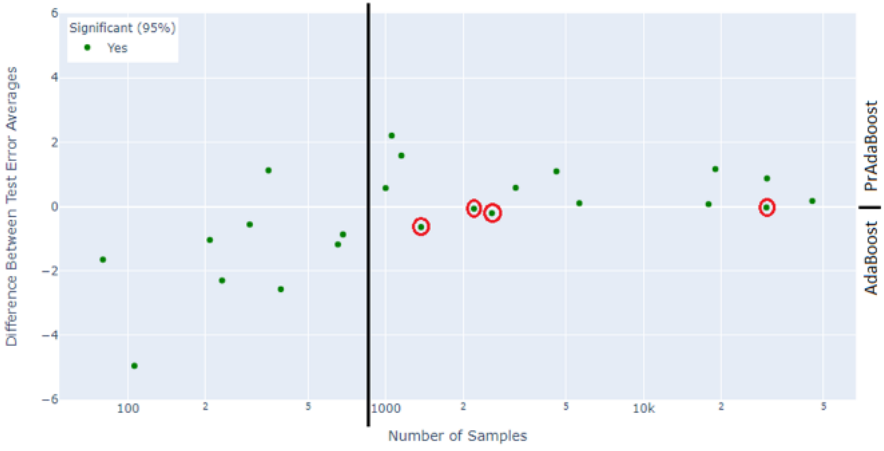
- parameters in InfoBoost differ from those used in PrAdaBoost

- In terms of test error:
  PrAdaBoost significantly outperforms InfoBoost on 21 of 23 tested datasets
  ties on one dataset
  loses to InfoBoost on only one dataset (promoters)

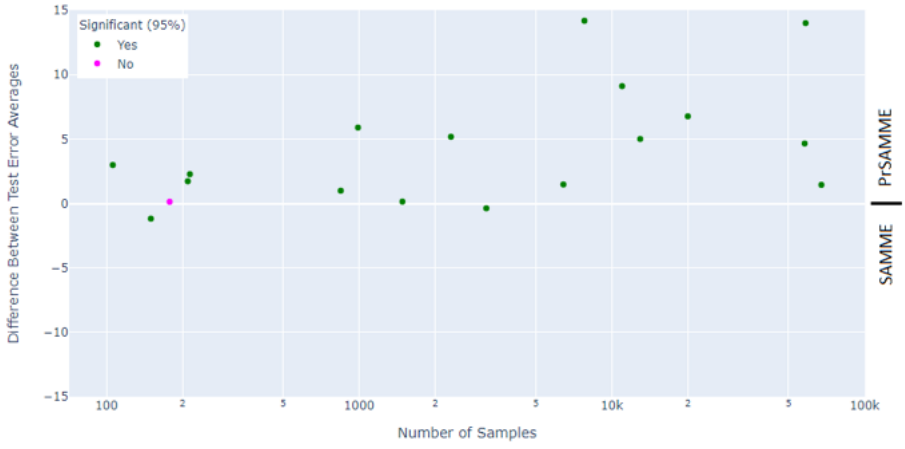
Topic 1 – Empirical Analysis
Results – Comparison to AdaBoost (Binary)



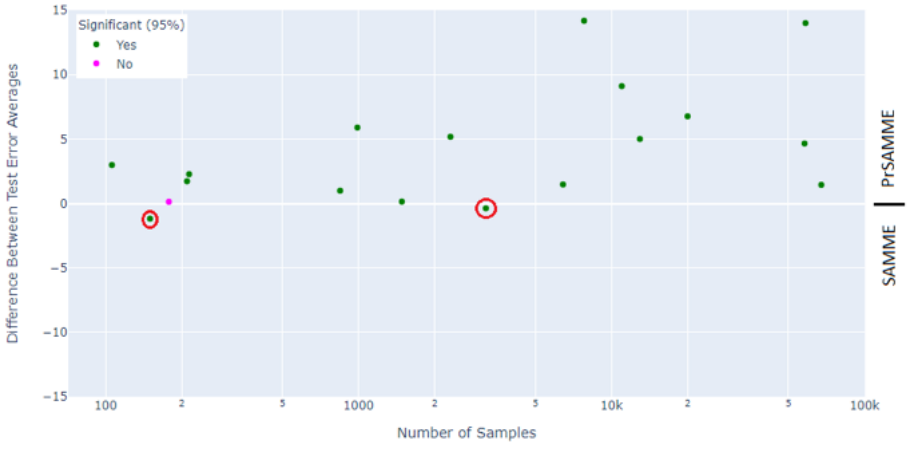
Topic 1 – Empirical Analysis
Results – Comparison to AdaBoost (Binary)



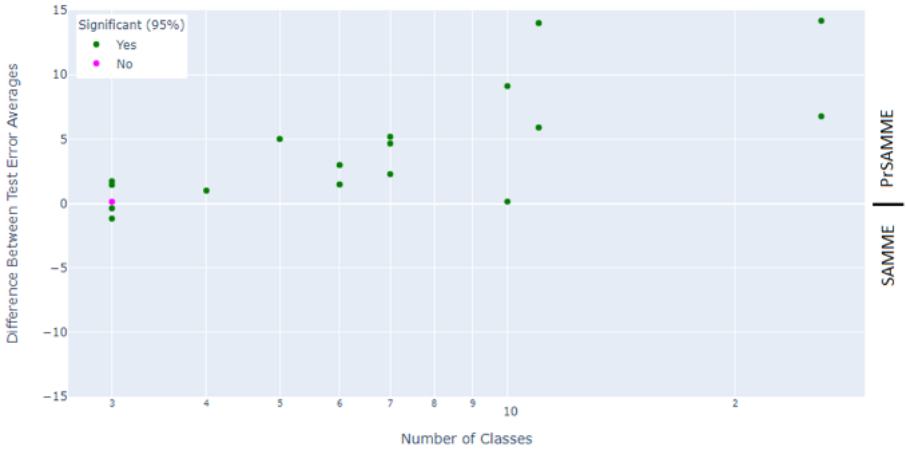
Topic 1 – Empirical Analysis
Results – Multi-Class



Topic 1 – Empirical Analysis
Results – Multi-Class



Topic 1 – Empirical Analysis
Results – Multi-Class



Topic 1 – Empirical Analysis

Dataset	PrSAMME-R	RF	R2	RT	GB
Concrete Slump Test	0.023	**0.021**	0.038	0.035	0.032
Parkinsons Telemonitoring	0.045	**0.039**	0.057	0.047	0.041
Automobile	**0.041**	0.047	0.062	0.058	0.051
Auto MPG	**0.049**	0.056	0.081	0.062	0.090
Air Quality	0.033	**0.029**	0.045	0.049	0.053
Energy prediction	**0.029**	0.034	0.047	0.039	0.035
Superconductivity	0.042	**0.038**	0.064	0.052	0.046
Demand Forecasting	**0.048**	0.062	0.078	0.051	0.049
Parking Birmingham	**0.049**	0.057	0.076	0.055	0.061
Protein Tertiary Structure	**0.022**	0.034	0.045	0.039	0.043
Metro Traffic Volume	**0.031**	0.038	0.065	0.042	0.039
Survival modelling	**0.032**	0.043	0.054	0.059	0.046
Accelerometer	**0.045**	0.059	0.086	0.051	0.063
Video Transcoding	**0.042**	0.059	0.084	0.066	0.061
Beijing Air-Quality Data	**0.027**	0.048	0.069	0.041	0.056
Power Consumption	**0.031**	0.043	0.065	0.048	0.054

MAE comparison

- PrSAMME-Regression with K=50, EM, vs.

- Random Forest (Breiman 2001)
- AdaBoost.R2 (Drucker 1997)
- AdaBoost.RT (Solomatine and Shrestha 2004)
- Gradient Boosting (Zemel and Pitassi 2000)

Topic 1 – Conclusions

Introduction of **class-specific precision** into AdaBoost framework

- Formally derived **optimal weighting scheme** in a setting that generalizes the one on which AdaBoost is built
- **PrAdaBoost** is **guaranteed** to convert a weak learner into a strong learner
- Extensions to multi-class classification and regression
- Precision-based approach seems preferable when running an AdaBoost-type method on datasets with **more than 1,000 instances or with more than 3 classes**
- Precision-based approach tends to be more effective for imbalanced classes

Ongoing work: incorporate precision-based approach directly into other methods (not just AdaBoost)!

Topic 2 – Active Learning with Uncertainty Sampling

problem situation: many unlabeled data points, high cost of labeling

e.g., performing tests on seedlings

(image generated by ChatGPT)

solution: **active learning**

- learning algorithm selects a small set of data points to be labeled
- training performed on the resulting labeled data points
- repeat this in rounds

⇝ better classifier with fewer data points

one criterion for selecting data points to be labeled:
uncertainty (Lewis and Gale 1994)

Topic 2 – Active Learning with Uncertainty Sampling

problem: how to assess uncertainty?

- two new methods for assessing a learning algorithm's uncertainty
- both apply also in deep learning
- one uses formally proven error bounds in its design

$\rightsquigarrow$ empirically superior to existing methods

Topic 3 – Learning from Positive and Unlabeled Data

e.g., cheap and simple test to confirm bad seedling, but low recall

problem: a pool of unlabeled data points is given;
some of these will be labeled class 1, others remain unlabeled

- formal study with theoretical guarantees
- under certain assumptions, highly accurate classifiers can be learned

Possible New Studies of Relevance?

problem: data points with missing features

e.g., deciding when to assess seedling

(image generated by ChatGPT)

- features are revealed over time
- cost associated with late classification
- trade-off between accuracy and cost of obtaining classifier

$\rightsquigarrow$ AI for decision making: when to deploy classifier? performance guarantees?

Thank You to …

Mohamadsadegh Khosravani, Farnam Mansouri, Marjan Movahedan,

Mohammad Hossein Nikravan, Mahsasadat Razavi

NSERC CIFAR Amii

ISTA Organizing Team

…and you all :)