Chapter X: Classification*

1. Basic idea
2. Decision trees
3. Naïve Bayes classifier
4. Support vector machines
5. Ensemble methods

* Zaki & Meira: Ch. 24, 26, 28 & 29; Tan, Steinbach & Kumar: Ch. 4, 5.3–5.6

X.5 Ensemble methods*

1. Basic idea
2. Bagging
3. Boosting
   - 3.1. AdaBoost

* Zaki & Meira: Ch. 29; Tan, Steinbach & Kumar: Ch. 5.6; Bishop: Ch. 14.2–3

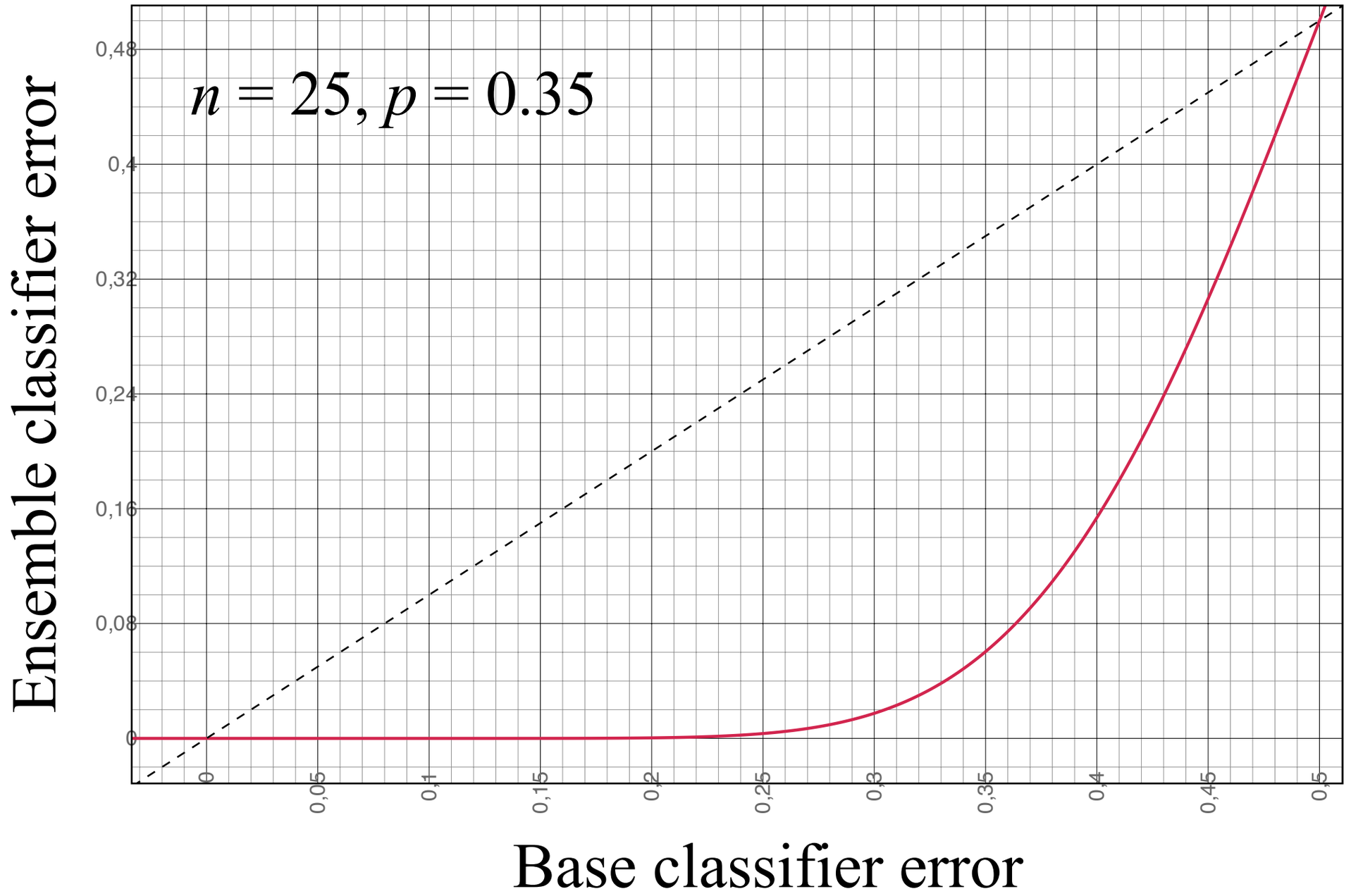
Basic idea

- Suppose we have multiple classifiers for the data
  - Each is good in some parts and bad in other parts
- Can we get better results if we combine these methods?
- How can combine them?
  - Simple committee solution: take the majority label
  - If we have confidence to the classifiers, we can weight their solutions and take the weighted majority label

Rationale of ensembles

- 25 binary classifiers (*base classifiers*)
  - Each base classifier has error rate 0.35
  - Majority vote to select the class label
- If the base classifiers are identical, the ensemble will have error rate 0.35
  - If the base classifiers are independent, the ensemble will have error rate $\Pr[X \geq 13]$ with $X \sim \text{Binom}(25, 0.35)$
  - $\Pr[X \geqslant 13] = \sum_{i=13}^{25} \binom{25}{i} 0.35^i (1 - 0.35)^{25-i} = 0.06$
- Two conditions:
  - Base classifiers must be (reasonably) independent
  - Base classifiers must do better than purely random

Ensemble error example



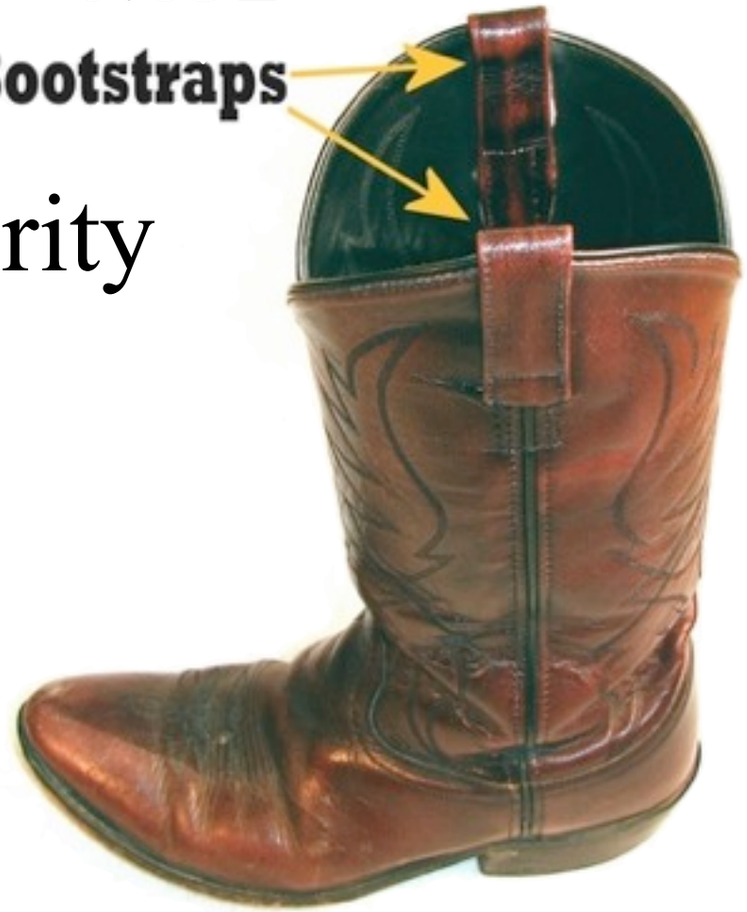
How to make independent classifers

- Manipulate the training set
  - Bagging
  - Boosting
- Manipulate the input features
  - Random forest
- Manipulate the class labels
  - Different splits from multi-class to two-class
- Manipulate the learning algorithm
  - Add randomness

Bagging (a.k.a. bootstrapping)

- Sample the data *uniformly with replacements*
  - Each sample D_i has the same size as the original data D
  - Each data point $\boldsymbol{x} \in D$ has
    $\Pr[\boldsymbol{x} \in D_i] = 1 - (1 - 1/|D|)^{|D|} \rightarrow 1 - 1/e \approx 0.632$
    when $|D| \rightarrow \infty$
- Final classifier usually uses the majority voting

Image: http://www.lemen.com



Bagging example

x	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
y	+1	+1	+1	–1	–1	–1	–1	+1	+1	+1

Bagging example

x	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
y	+1	+1	+1	−1	−1	−1	−1	+1	+1	+1

$x \leq 0.35$

Single one-split decision tree's accuracy $\leq 70\%$

Bagging example

x	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
y	+1	+1	+1	−1	−1	−1	−1	+1	+1	+1

$x \leq 0.35$

Single one-split decision tree's accuracy $\leq 70\%$

10 bagging samples:

Bagging example

x	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
y	+1	+1	+1	−1	−1	−1	−1	+1	+1	+1

$x \leq 0.35$

Single one-split decision tree's accuracy ≤ 70%

10 bagging samples:

x	0.1	0.2	0.2	0.3	0.4	0.4	0.5	0.6	0.9	1.0
y	+1	+1	+1	+1	−1	−1	−1	−1	+1	+1

Bagging example

x	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
y	+1	+1	+1	−1	−1	−1	−1	+1	+1	+1

$x \leq 0.35$

Single one-split decision tree's accuracy ≤ 70%

10 bagging samples:

x	0.1	0.2	0.2	0.3	0.4	0.4	0.5	0.6	0.9	1.0
y	+1	+1	+1	+1	−1	−1	−1	−1	+1	+1

x	0.1	0.2	0.3	0.4	0.5	0.8	0.9	1.0	1.0	1.0
y	+1	+1	+1	−1	−1	+1	+1	+1	+1	+1

Bagging example

x	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
y	+1	+1	+1	−1	−1	−1	−1	+1	+1	+1

$x \leq 0.35$

Single one-split decision tree's accuracy ≤ 70%

10 bagging samples:

x	0.1	0.2	0.2	0.3	0.4	0.4	0.5	0.6	0.9	1.0
y	+1	+1	+1	+1	−1	−1	−1	−1	+1	+1
x	0.1	0.2	0.3	0.4	0.5	0.8	0.9	1.0	1.0	1.0
y	+1	+1	+1	−1	−1	+1	+1	+1	+1	+1
x	0.1	0.1	0.1	0.1	0.3	0.3	0.8	0.8	0.9	0.9
y	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1

Bagging example

x	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
y	+1	+1	+1	–1	–1	–1	–1	+1	+1	+1

$x \leq 0.35$

Single one-split decision tree's accuracy ≤ 70%

10 bagging samples:

x	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
Σ	2	2	2	–6	–6	–6	–6	2	2	2

Bagging example

x	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
y	+1	+1	+1	–1	–1	–1	–1	+1	+1	+1

$x \leq 0.35$

Single one-split decision tree's accuracy ≤ 70%

10 bagging samples:

x	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
Σ	2	2	2	–6	–6	–6	–6	2	2	2

Estim: +1 +1 +1 -1 -1 -1 -1 +1 +1 +1

Bagging example

x	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
y	+1	+1	+1	–1	–1	–1	–1	+1	+1	+1

Estim: +1 +1 +1 -1 -1 -1 -1 +1 +1 +1

10 bagging samples:

x	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
Σ	2	2	2	–6	–6	–6	–6	2	2	2

Bagging example

x	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
y	+1	+1	+1	–1	–1	–1	–1	+1	+1	+1

Estim: +1 +1 +1 -1 -1 -1 -1 +1 +1 +1

10 bagging samples:

2-level decision tree!

x	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
Σ	2	2	2	–6	–6	–6	–6	2	2	2

Boosting

- In bagging, each item gets sampled u.a.r. independent of how hard they're to classify
  - Easy items don't need ensembles
  - Shouldn't we concentrate on hard cases?
- Boosting adds weights to the training items
  - More often misclassified items get bigger weights
  - Weights can be used to learn biased classifiers
    - Pay more for misclassifying heavier items
  - Weights can be used to weight bootstrap sampling
    - Heavier items get selected more often

Boosting

- In bagging, each item gets sampled u.a.r. independent of how hard they're to classify
  - Easy items don't need ensembles
  - Shouldn't we concentrate on hard cases?
- Boosting adds weights to the training items
  - More often misclassified items get bigger weights
  - Weights can be used to learn biased classifiers
    - Pay more for misclassifying heavier items
  - Weights can be used to weight bootstrap sampling
    - Heavier items get selected more often

This is what we cover

Basic idea

1. Initialize all weights to $1/N$
   - Uniform distribution
2. Perform classification
3. Increase the weights of misclassified items and reduce the weight of correctly classified items

4. Aggregate the predictions

Basic idea

1. Initialize all weights to $1/N$
   - Uniform distribution
2. Perform classification
3. Increase the weights of misclassified items and reduce the weight of correctly classified items

4. Aggregate the predictions

- Methods differ on how the weights are changed and how the aggregation works

AdaBoost

- Let C_i be a base classifier
  - *Error rate* of C_i is $\epsilon_i = N^{-1} \sum_{j=1}^{N} w_j \mathbf{1}(C_i(\mathbf{x}_j) \neq y_j)$
    - $\mathbf{1}(p) = 1$ if p is true and 0 o/w
    - w_j is the weight of training item $(\boldsymbol{x}_j, y_j)$
  - *Importance* of C_i is $\alpha_i = \ln(1/\epsilon_i - 1)/2$
  - *Weight* of $(\boldsymbol{x}_j, y_j)$ for iteration i+1 is

$$w_j^{i+1} = \frac{w_j^{(i)}}{Z_i} \times \begin{cases} e^{-\alpha_i} & \text{if } C_i(\mathbf{x}_j) = y_j \\ e^{\alpha_i} & \text{if } C_i(\mathbf{x}_j) \neq y_j \end{cases}$$

    - Z_i is a normalization constant s.t. w_j's sum to 1
  - If error rate goes above 0.5, all weights are set to $1/N$
- For aggregation, each classifier is weighted by α_i

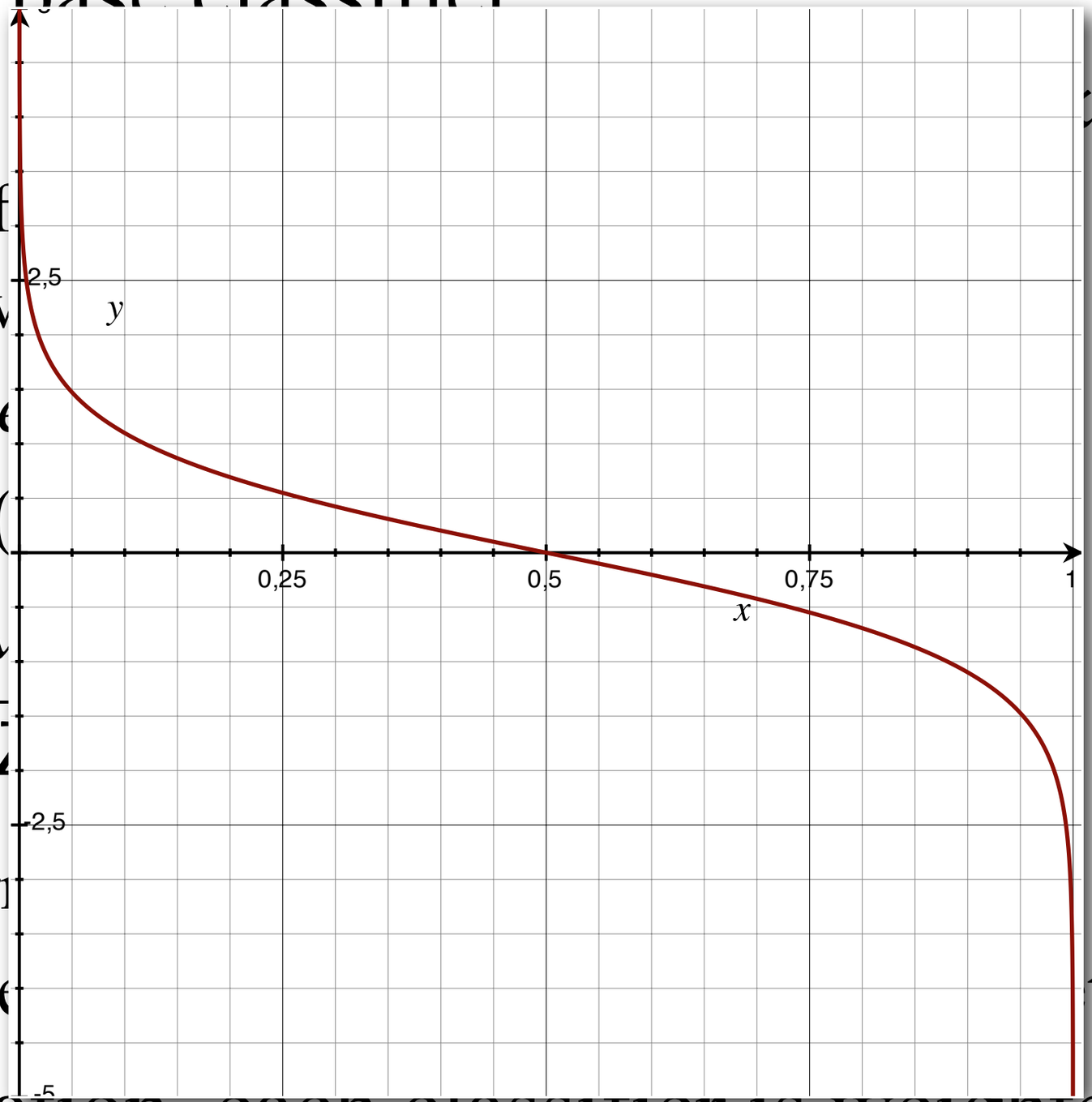
AdaBoost

Importance w.r.t. error rate

- Let C_i be a base classifier
  - *Error rate* $\ldots \neq y_j)$
    - $\mathbf{1}(p) = 1$ if
    - w_j is the w
  - *Importance*
  - *Weight* of (

$$w_j^{i+1} = \frac{}{}$$

    - Z_i is a nor
  - If error rate t to $1/N$
- For aggregation, each classifier is weighted by α_i



error rate

AdaBoost

- Let C_i be a base classifier
  - *Error rate* of C_i is $\epsilon_i = N^{-1} \sum_{j=1}^{N} w_j \mathbf{1}(C_i(\mathbf{x}_j) \neq y_j)$
    - $\mathbf{1}(p) = 1$ if p is true and 0 o/w
    - w_j is the weight of training item $(\boldsymbol{x}_j, y_j)$
  - *Importance* of C_i is $\alpha_i = \ln(1/\epsilon_i - 1)/2$
  - *Weight* of $(\boldsymbol{x}_j, y_j)$ for iteration i+1 is

$$w_j^{i+1} = \frac{w_j^{(i)}}{Z_i} \times \begin{cases} e^{-\alpha_i} & \text{if } C_i(\mathbf{x}_j) = y_j \\ e^{\alpha_i} & \text{if } C_i(\mathbf{x}_j) \neq y_j \end{cases}$$

    - Z_i is a normalization constant s.t. w_j's sum to 1
  - If error rate goes above 0.5, all weights are set to $1/N$
- For aggregation, each classifier is weighted by α_i

Error rate

- Let ε_i be the error rate of classifier i in boosting
  - Let $\varepsilon_i < 0.5$ for all i and write $\varepsilon_i = 0.5 - \gamma_i$
- The total error rate of the ensemble can now be bounded by

$$\epsilon_E \leqslant \prod_i \sqrt{\epsilon_i(1-\epsilon_i)} = \prod_i \sqrt{1/4 - \gamma_i^2}$$

$$= \exp\left(-O\left(\sum_i \gamma_i^2\right)\right)$$

  - The error decreases exponentially if $\gamma_i > \gamma^* > 0$ for all i's
    $\Rightarrow$ Fast convergence

Summary of ensemble methods

- Combining many classifiers usually helps
- Base classifiers must be reasonably independent
- Bagging is simple, but doesn't improve bad classifiers much
- Boosting helps in particular with almost-random classifiers
  - Boosting is prone to overfitting
- How many classifiers can be combined depends on how much time we can spend