Ensemble Learning

Slides from Cong Li

Outline

- **Ensemble Methods in Machine Learning**
- Bagging
- Boosting

Ensemble Methods in ML

- **Ensemble Methods in Machine Learning**
  - Ensembles of Classifiers
  - Application
- Bagging
- Boosting

Different Classifiers (1)

- **Different Classifiers**
  - Conduct classification on same set of class labels
  - May use different input or have different parameters
  - May produce different output for a certain example
- **Learning Different Classifiers**
  - Use different training examples
  - Use different features

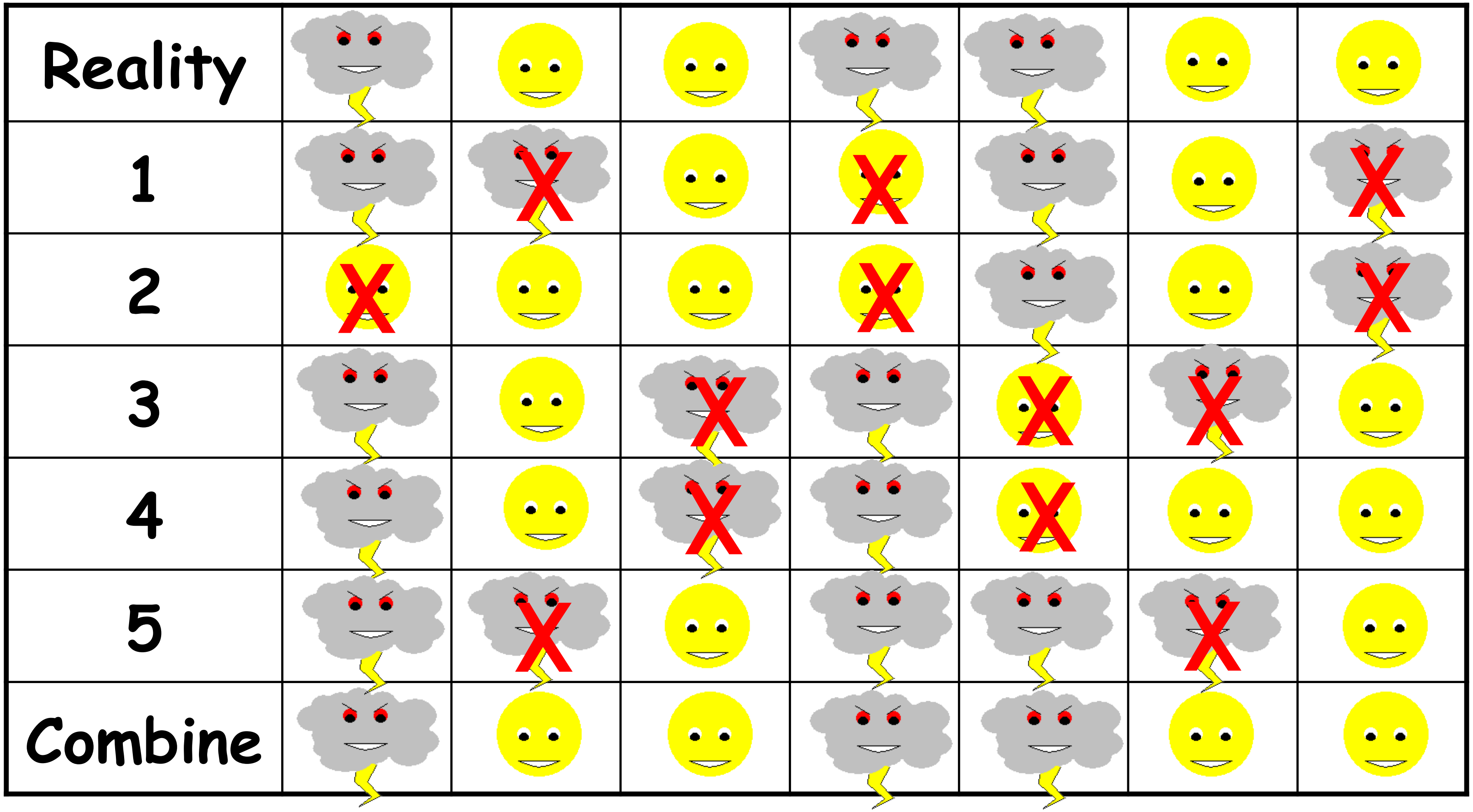
Different Classifiers (2)

- **Performance**
  - None of the classifiers is perfect
  - Complementary
    - Examples which are not correctly classified by one classifier may be correctly classified by the other classifiers
- **Potential Improvements?**
  - Utilize the complementary property

Ensembles of Classifiers

- **Idea**
  - Combine the classifiers to improve the performance
- **Ensembles of Classifiers**
  - Combine the classification results from different classifiers to produce the final output
    - Unweighted voting
    - Weighted voting

Example: Weather Forecast



Ensemble Methods in ML

- **Ensemble Methods in Machine Learning**
  - Ensembles of Classifiers
  - Application
- Bagging
- Boosting

Application: WSD (Pedersen 2000)

- **Ensembles of classifiers using different features**
  - Use different features in training and classification in each classifier
- **Ensembles of naive Bayesian classifiers for WSD**
  - Use different context windows to create different naive Bayesian classifiers

WSD = Word Sense Disambiguation

Implementation

- **81 Base Classifiers**
  - Context window, num of words *left*, *right*
  - Possible values for *l* and *r* : 0, 1, 2,(*narrow*) 3, 4, 5, (*medium*) 10, 25, 50 (*wide*)
- **9 Selected Range Classifiers**
  - For each range (e.g., (*narrow*, *medium*)), select the best classifiers from 9 candidates (using a development set)
- **Combination**
  - Unweighted voting of the 9 classifiers

WSD Results

- **Benchmark: <u>Interest</u>**
  - Six senses
  - 2368 examples for training and testing
- **Results**
  - Ensembles of naive Bayesian classifiers: 89% (Pedersen 2000)
  - Achieve the best performance reported

Outline

- Ensemble Methods in Machine Learning
- **Bagging**
- Boosting

Bagging

- **An Important Strategy for Ensemble Learning**
  - Create different training sets
- **Bootstrap AGGregatING**
  - Take created bootstrap samples to create a sequence of training sets
  - Train classifiers using the training sets
  - Classification by majority voting
    (or averaging for, e.g., estimation problems)

Replicating Data Sets

- **Original Training Set**
  - $\{(\mathbf{x}^{(1)}, y^{(1)}), (\mathbf{x}^{(2)}, y^{(2)}), \cdots, (\mathbf{x}^{(m)}, y^{(m)})\}$
- **Sample with Replacement**
  - At each time, randomly draw m examples according to the uniform distribution on the original training set
  - Allow duplicating and missing
  - Used for training classifiers

Bagging decision trees

Table 1 Mis~~s~~classification Rates (Percent)

Data Set	$\bar{e}_S$	$\bar{e}_B$	**Decrease**
waveform	29.0	19.4	33%
heart	10.0	5.3	47%
breast cancer	6.0	4.2	30%
ionosphere	11.2	8.6	23%
diabetes	23.4	18.8	20%
glass	32.0	24.9	22%
soybean	14.5	10.6	27%

1. Splitting the data set into training set T1 and test set T2.
2. Bagging using 50 bootstrap samples.
3. Repeat Steps 1-2 100 times, and calculate average
   test set misclassification rate.

How many bootstrap samples are needed?

Bagged Mis~~s~~classification Rates (%)

No. Bootstrap Replicates	Mi~~s~~sclassification Rate
10	21.8
25	19.5
50	19.4
100	19.4

Bagging decision trees for the waveform task:

- Unbagged rate is 29.0%.
- We are getting most of the improvement using only 10 bootstrap samples.

Bagging k-nearest neighbor classifiers

Mis~~s~~classification Rates for Nearest Neighbor

Data Set	$\bar{e}_S$	$\bar{e}_B$
waveform	26.1	26.1
heart	6.3	6.3
breast cancer	4.9	4.9
ionosphere	35.7	35.7
diabetes	16.4	16.4
glass	21.6	21.6

100 bootstrap samples. 100 iterations.
Bagging does not help.

Experiment results

- **Bagging works well for “unstable” learning algorithms.**

- **Bagging can slightly degrade the performance of “stable” learning algorithms.**

Learning algorithms

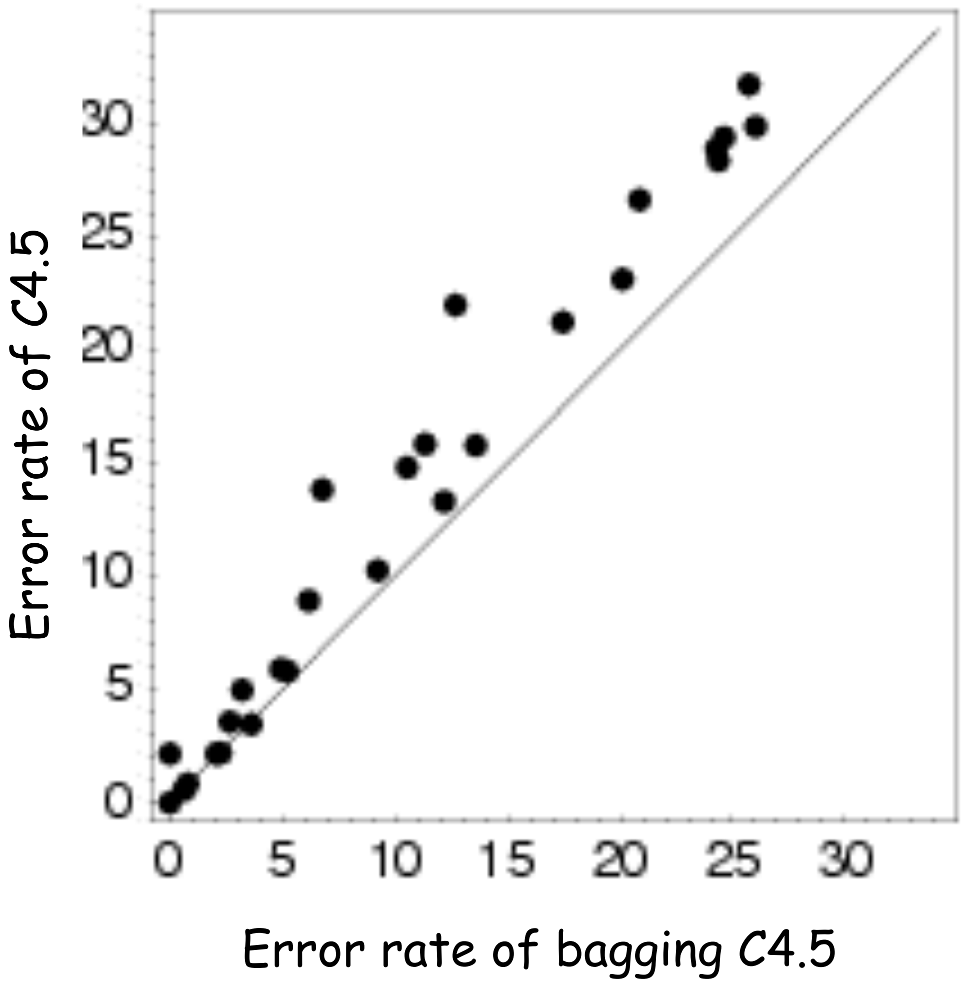
- **Unstable learning algorithms: small changes in the training set result in large changes in predictions.**
  - Neural network
  - Decision tree (in particular: regression trees)

- **Stable learning algorithms:**
  - K-nearest neighbors

First Performance tests

- **Data Set**
  - 27 data sets from UCI ML Repository
- **Methods for Comparison**
  - Decision tree classifier: C4.5
  - Bagging: ensembles of 100 C4.5 classifiers

Results (Freund and Schapire 1996)



Seems to improve performance

Majority vote

Suppose we have 5 completely independent classifiers…

- If accuracy is 70% for each
  - 10 (.7^3)(.3^2)+5(.7^4)(.3)+(.7^5)
  - **83.7% majority vote accuracy**
- 101 such classifiers
  - **99.9% majority vote accuracy**

Outline

- Ensemble Methods in Machine Learning
- Bagging
- Boosting

Boosting

- Ensemble Methods in Machine Learning
- Bagging
- **Boosting**
  - Basic Idea
  - AdaBoost Algorithm
  - Performance

Strong and Weak Learners

- **Strong Learner**
  - Take labeled data for training
  - Produce a classifier which can be arbitrarily accurate
  - Objective of machine learning
- **Weak Learner**
  - Take labeled data for training
  - Produce a classifier which is more accurate than random guessing

Boosting

- **Learners**
  - Strong learners are very difficult to construct
  - Constructing weaker Learners is relatively easy
- **Strategy**
  - Derive strong learner from weak learner
  - Boost weak classifiers to a strong learner

Construct Weak Classifiers

- **Using Different Data Distribution**
  - Start with uniform weighting
  - During each step of learning
    - Increase weights of the examples which are not correctly learned by the weak learner
    - Decrease weights of the examples which are correctly learned by the weak learner
- **Idea**
  - Focus on difficult examples which are not correctly classified in the previous steps

Combine Weak Classifiers

- **Weighted Voting**
  - Construct strong classifier by weighted voting of the weak classifiers
- **Idea**
  - Better weak classifier gets a larger weight
  - Iteratively add weak classifiers
    - Increase accuracy of the combined classifier through minimization of a cost function

Boosting

- Ensemble Methods in Machine Learning
- Bagging
- **Boosting**
  - Basic Idea
  - AdaBoost Algorithm
  - Performance

Principle of AdaBoost

$$F(x) = \alpha_1 f_1(x) + \alpha_2 f_2(x) + \alpha_3 f_3(x) + \ldots$$

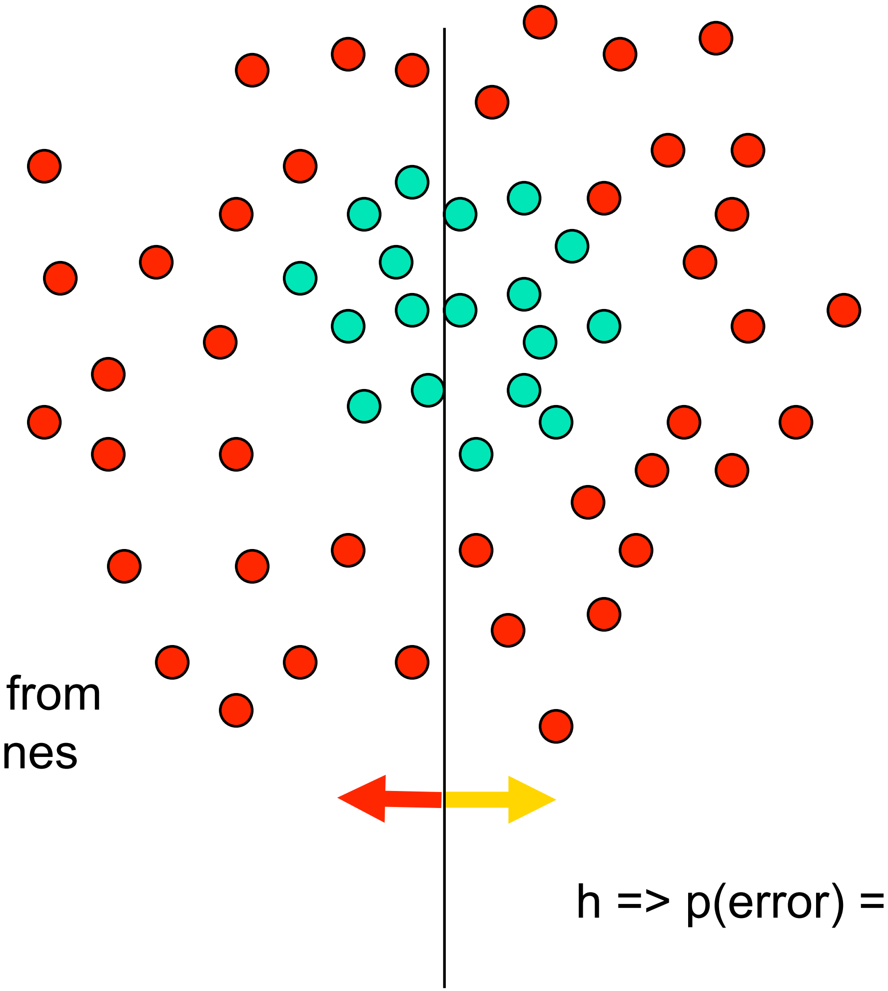
Strong classifier

Features vector

Weight

Weak classifier

Toy Example – taken from Torralba @MIT



Each data point has

a class label:

$$y_t = \begin{cases} +1 & (\text{red}) \\ -1 & (\text{green}) \end{cases}$$

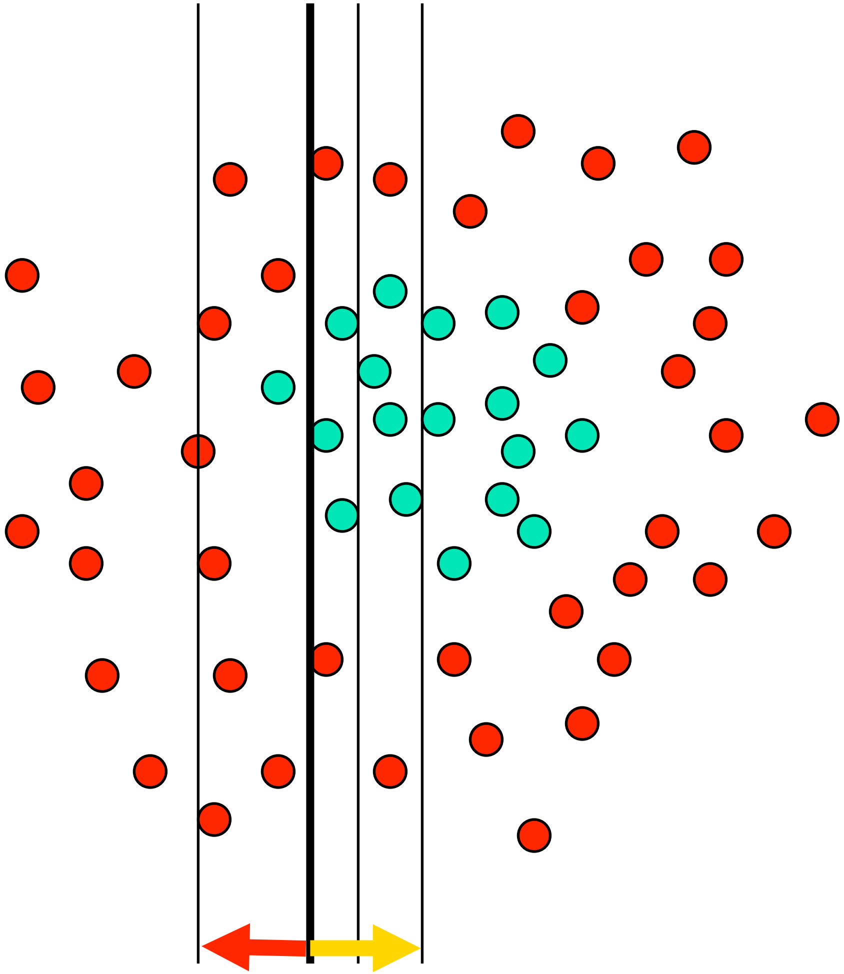
and a distribution:

$D_t = 1/m$

Weak learners from the family of lines

h => p(error) = 0.5 it is at chance

Toy example



Each data point has

a class label:

$$y_t = \begin{cases} +1 \ (\text{red}) \\ -1 \ (\text{cyan}) \end{cases}$$

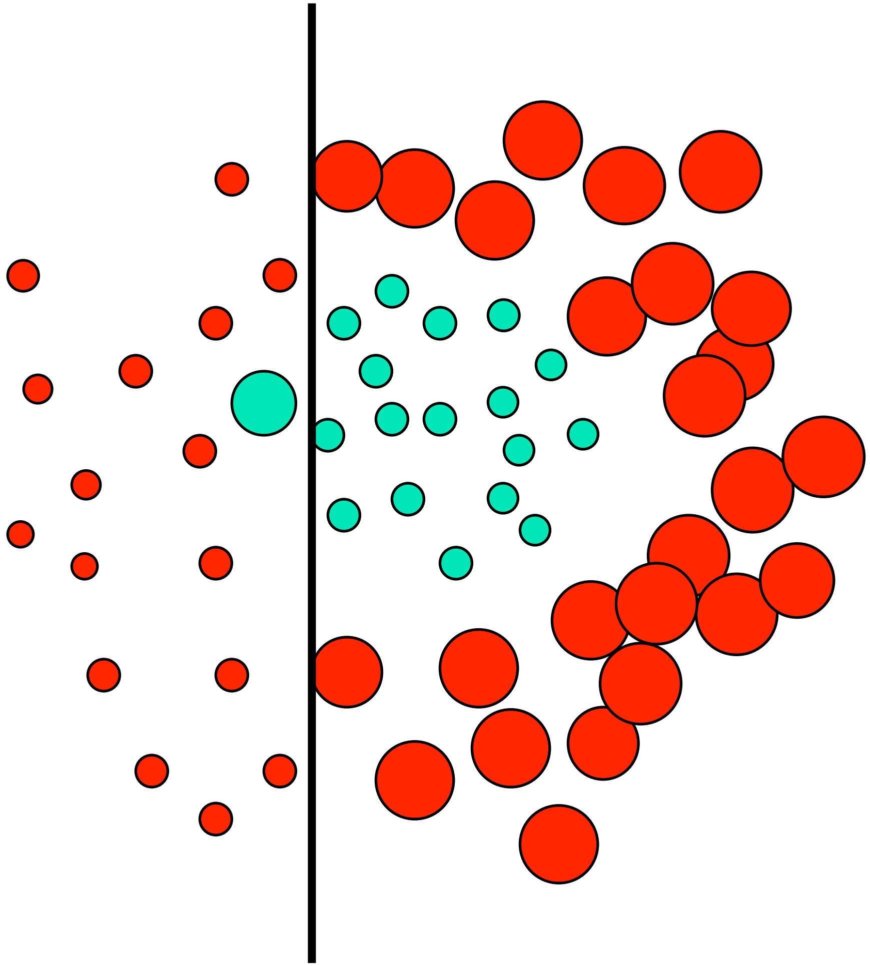
and a distribution:

$D_t = 1/m$

This one seems to be the best

This is a '**weak classifier**': It performs slightly better than chance.

Toy example



Each data point has

a class label:

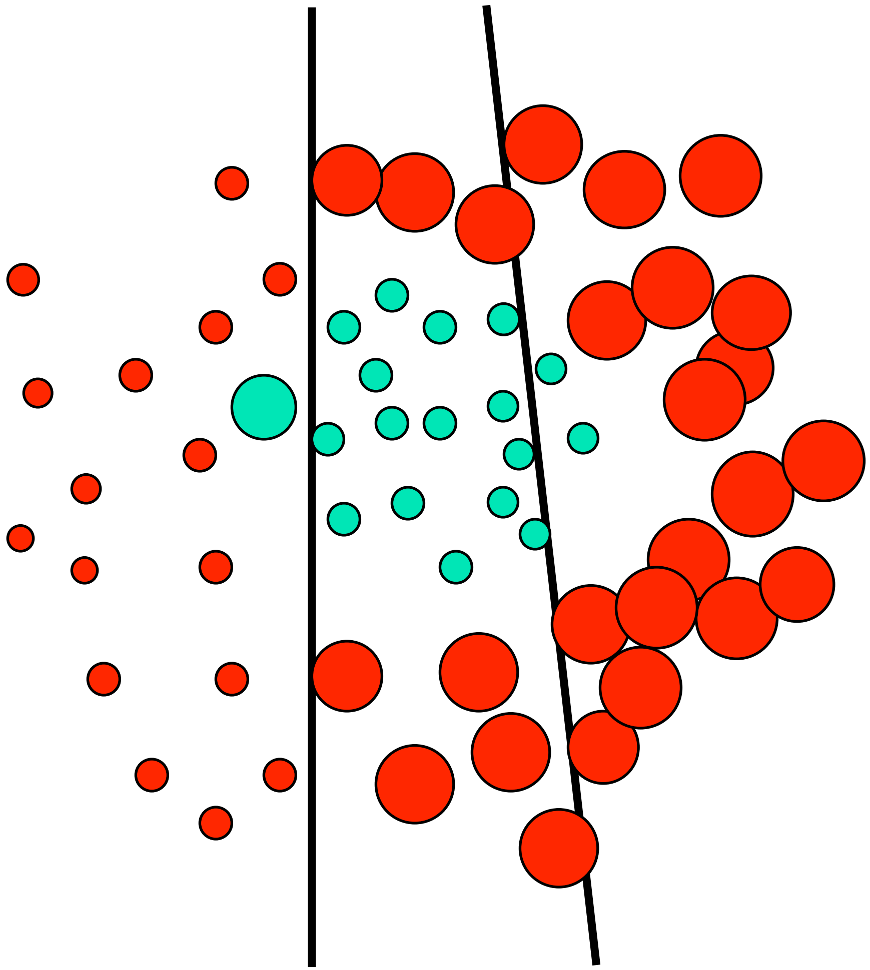
$$y_t = \begin{cases} +1 \ (\bullet) \\ -1 \ (\bullet) \end{cases}$$

We update D:

$$D_{t+1} \leftarrow D_t \, f(-y_t, h_t)$$

We set a new problem for which the previous weak classifier performs at chance again

Toy example



Each data point has

a class label:

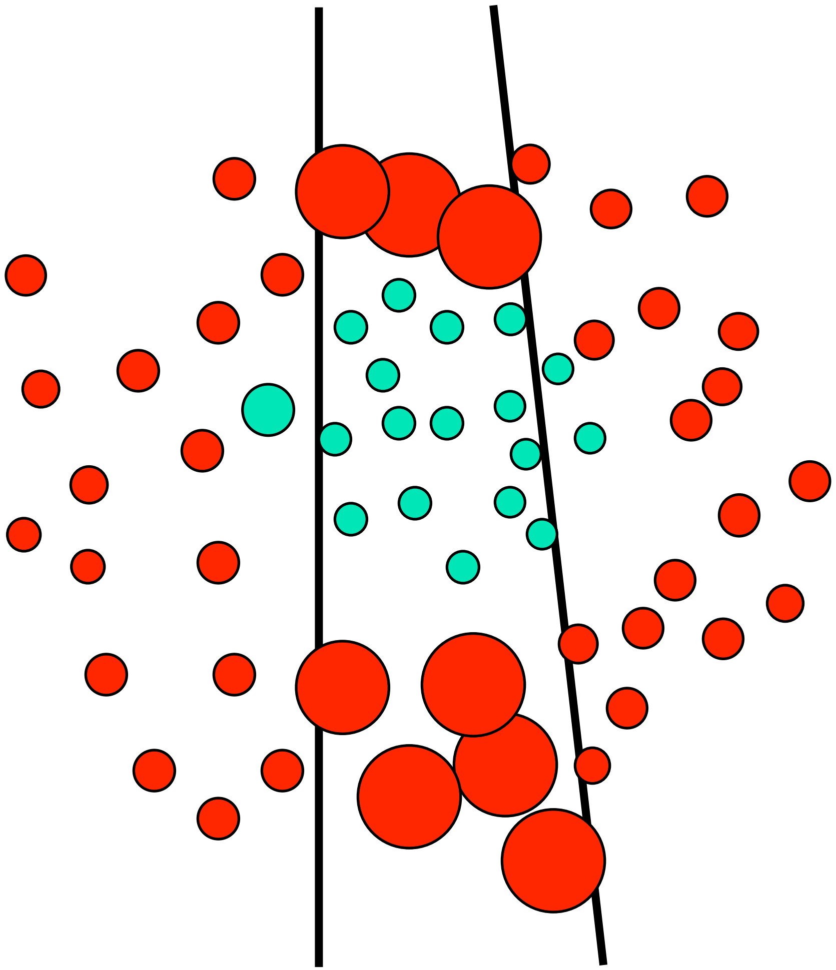
$$y_t = \begin{cases} +1 \ (\text{red}) \\ -1 \ (\text{cyan}) \end{cases}$$

We update D:

$$D_{t+1} \leftarrow D_t \, f(-y_t, h_t)$$

We set a new problem for which the previous weak classifier performs at chance again

Toy example



Each data point has

a class label:

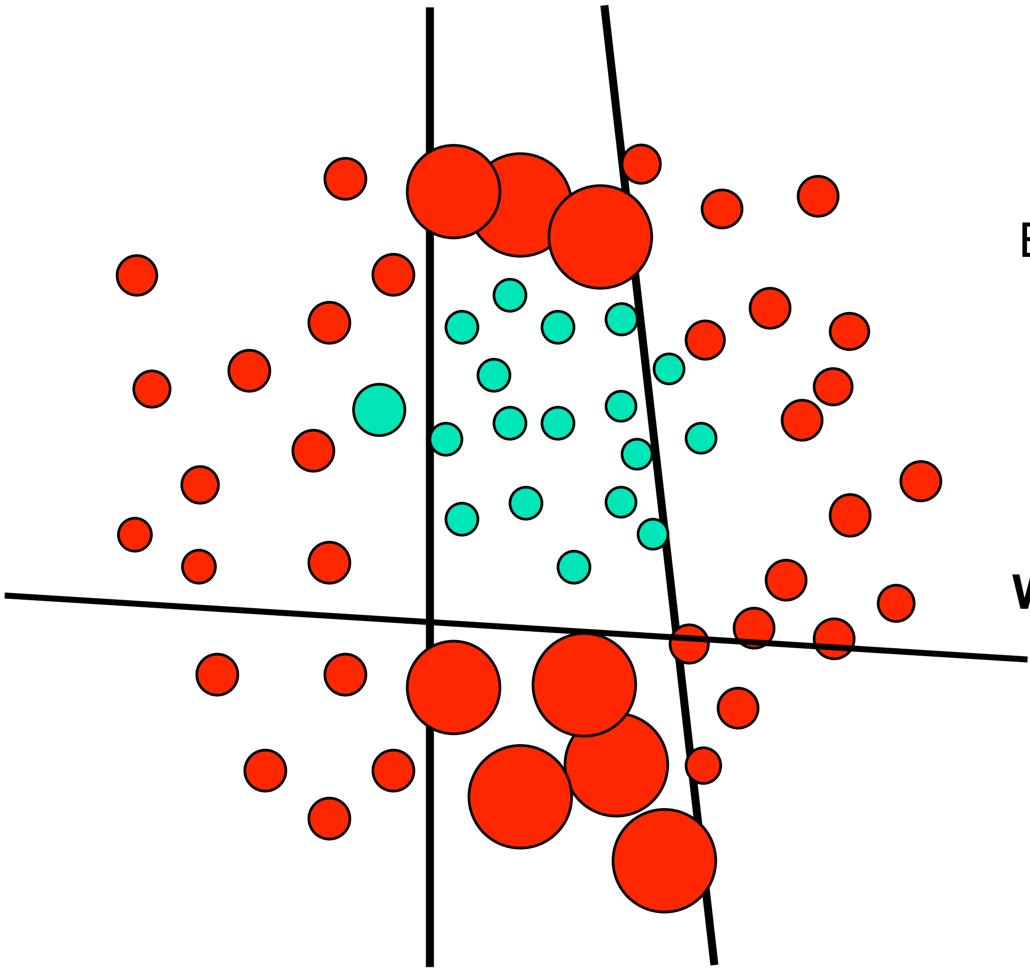
$$y_t = \begin{cases} +1 & (\text{red}) \\ -1 & (\text{green}) \end{cases}$$

We update D:

$$D_{t+1} \leftarrow D_t \, f(-y_t, h_t)$$

We set a new problem for which the previous weak classifier performs at chance again

Toy example



Each data point has

a class label:

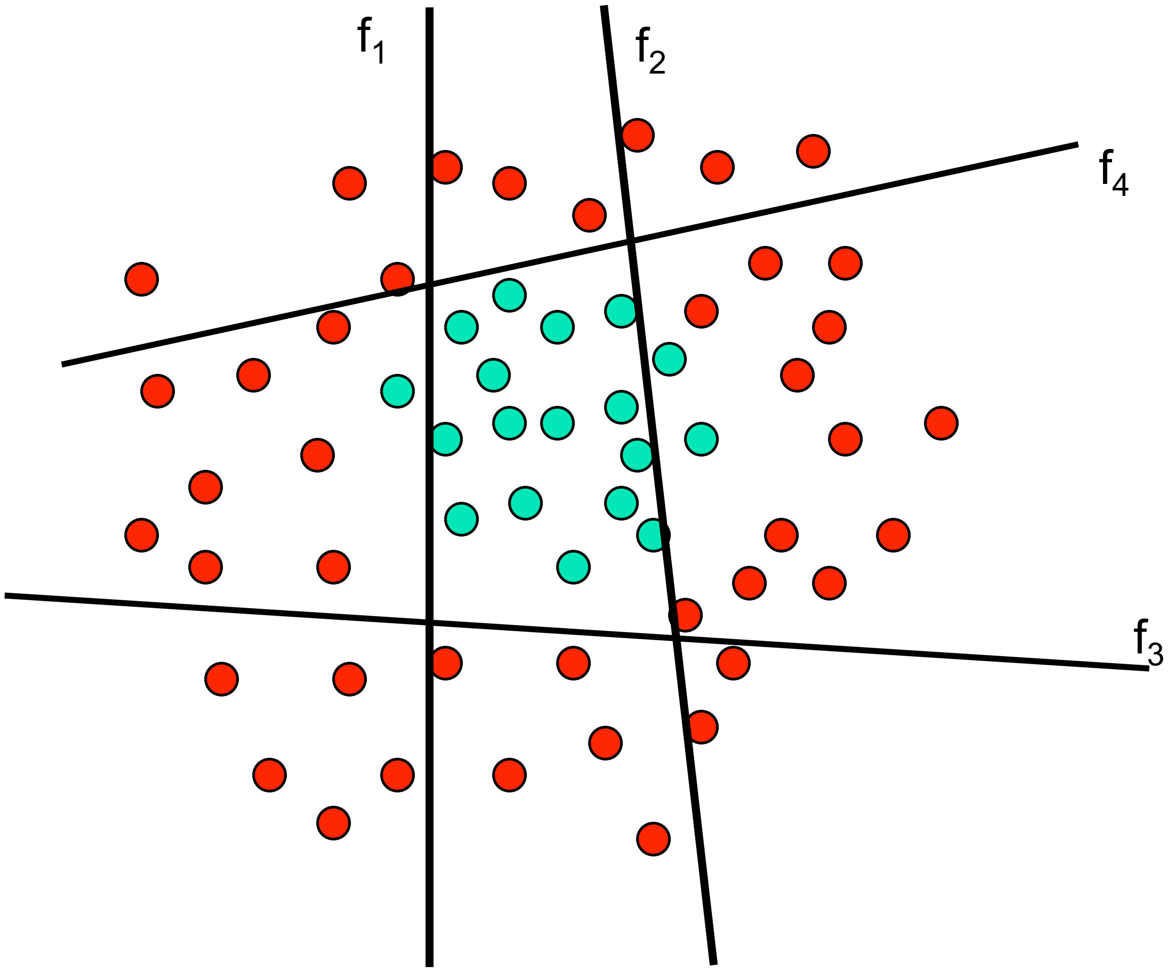
$$y_t = \begin{cases} +1 & (\bullet) \\ -1 & (\bullet) \end{cases}$$

We update D:

$D_{t+1} \leftarrow D_t \, f(-y_{t,} \; h_t)$

We set a new problem for which the previous weak classifier performs at chance again

Toy example



The strong (non- linear) classifier is built as the combination of all the weak (linear) classifiers.

AdaBoost: Algorithm

- given training set $(x_1, y_1), \ldots, (x_m, y_m)$
- $y_i \in \{-1, +1\}$ correct label of instance $x_i \in X$
- for $t = 1, \ldots, T$:
  - construct distribution D_t on $\{1, \ldots, m\}$
  - find weak classifier ("rule of thumb")

    $$h_t : X \to \{-1, +1\}$$

    with small error ϵ_t on D_t:

    $$\epsilon_t = \Pr_{D_t}[h_t(x_i) \neq y_i]$$
- output final classifier H_{final}

AdaBoost: Algorithm

- constructing D_t:
  - $D_1(i) = 1/m$
  - given D_t and h_t:

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

$$= \frac{D_t(i)}{Z_t} \exp(-\alpha_t \, y_i \, h_t(x_i))$$

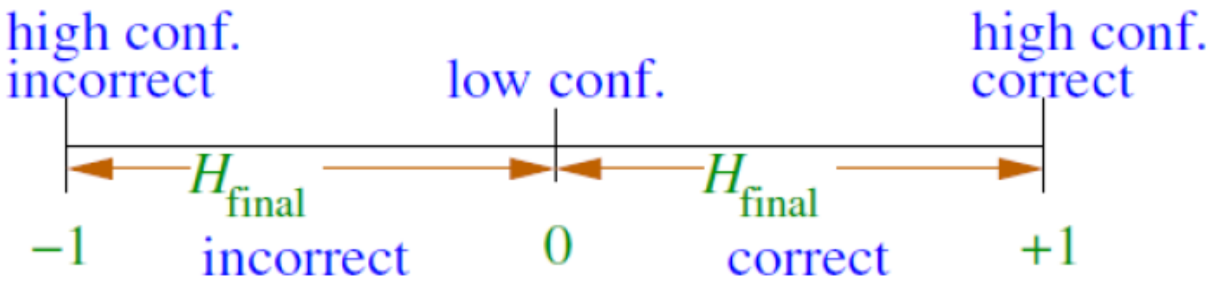
where $Z_t =$ normalization constant

$$\alpha_t = \frac{1}{2} \ln \left(\frac{1 - \epsilon_t}{\epsilon_t} \right) > 0$$

- final classifier:
  - $H_{\text{final}}(x) = \text{sign}\left(\sum_t \alpha_t h_t(x) \right)$

The margin

- key idea:
  - training error only measures whether classifications are right or wrong
  - should also consider confidence of classifications
- recall: H_{final} is weighted majority vote of weak classifiers
- measure confidence by margin = strength of the vote
  = (fraction voting correctly) − (fraction voting incorrectly)



AdaBoost: Final

- **Output**

$$H(\mathbf{x}) = sign[\sum_t \alpha_t h_t(\mathbf{x})]$$

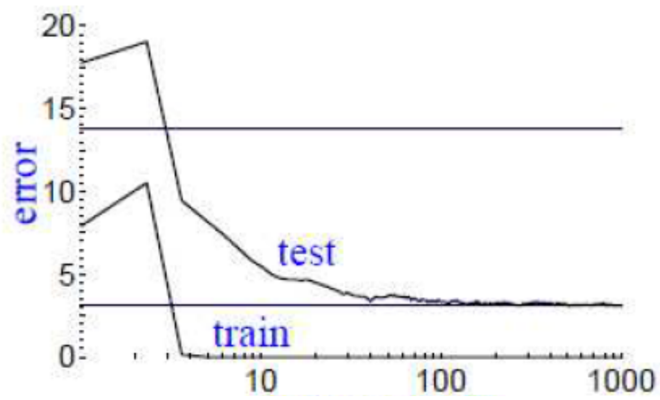
- **Margin Classifier**
  - Margin in majority vote classifiers

$$Margin(\mathbf{x}^{(i)}, y^{(i)}) = y^{(i)} \frac{\sum_t \alpha_t h_t(\mathbf{x}^{(i)})}{\sum_t \alpha_t}$$

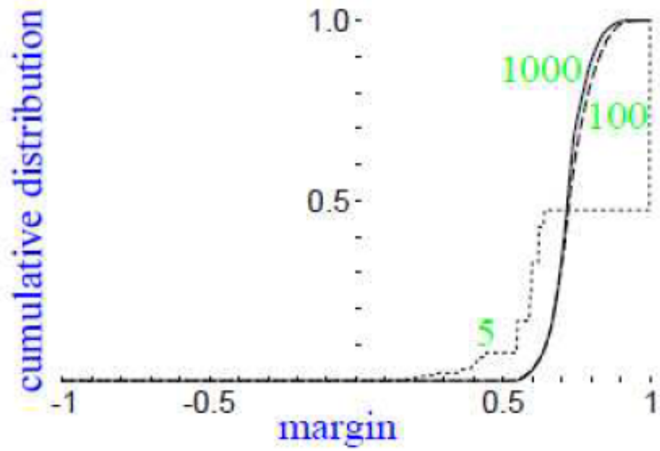
  - AdaBoost often optimizes the margins

The margin explanation

- margin distribution
  = cumulative distribution of margins of training examples



# of classifiers

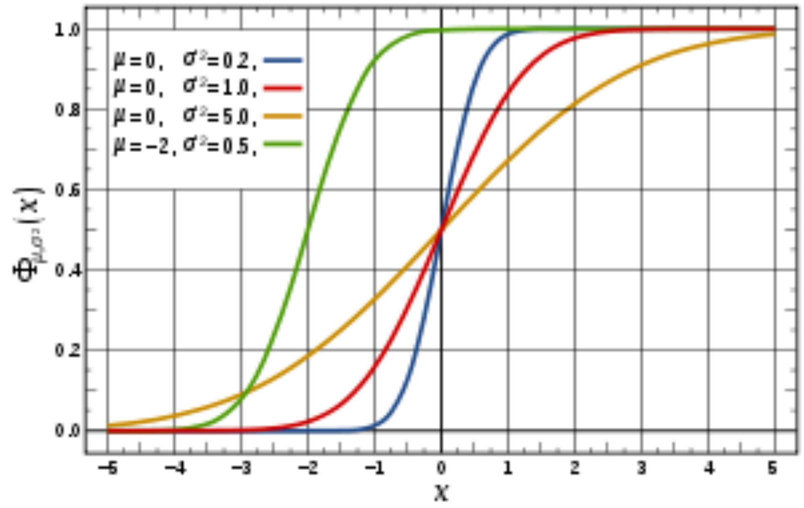


# of classifiers

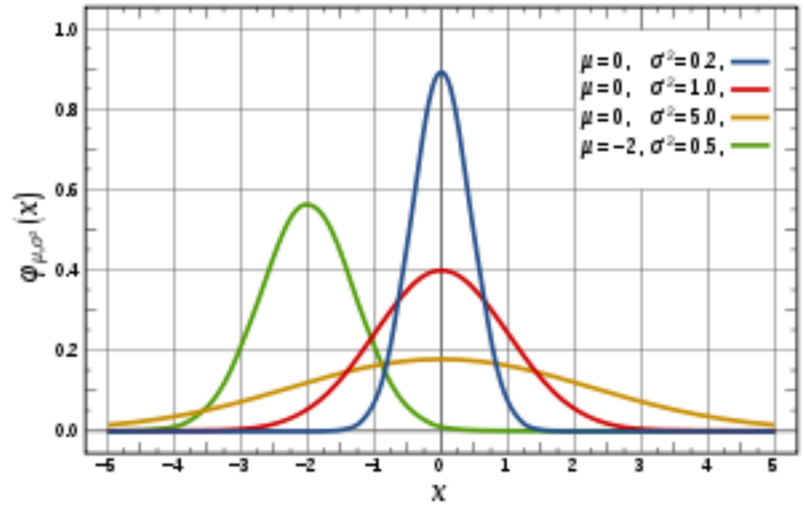
	5	100	1000
train error	0.0	0.0	0.0
test error	8.4	3.3	3.1
% margins ≤ 0.5	7.7	0.0	0.0
minimum margin	0.14	0.52	0.55

Cumulative distribution function

- **Describes the probability that a real-valued random variable X with a given probability distribution will be found at a value less than or equal to x**
- **„Area so far“**



$\mu=0,\ \sigma^2=0.2,$
$\mu=0,\ \sigma^2=1.0,$
$\mu=0,\ \sigma^2=5.0,$
$\mu=-2,\ \sigma^2=0.5,$
$\Phi_{\mu,\sigma^2}(x)$
x



$\mu=0,\ \sigma^2=0.2,$
$\mu=0,\ \sigma^2=1.0,$
$\mu=0,\ \sigma^2=5.0,$
$\mu=-2,\ \sigma^2=0.5,$
$\varphi_{\mu,\sigma^2}(x)$
x

Message

- From 5 to 100 there is a payoff
- From 100 to 1000 there is hardly a payoff
- 100 are enough, or
- You cannot become arbitrarily confident

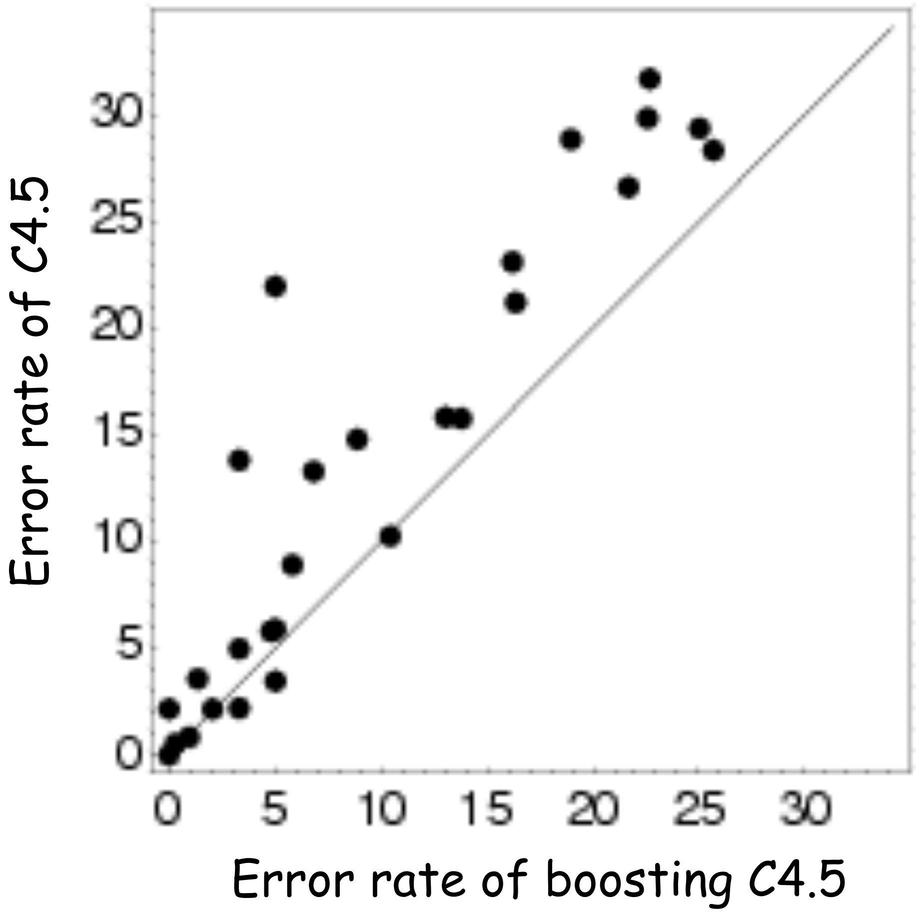
Boosting

- Ensemble Methods in Machine Learning
- Bagging
- **Boosting**
  - Basic Idea
  - AdaBoost Algorithm
  - Performance

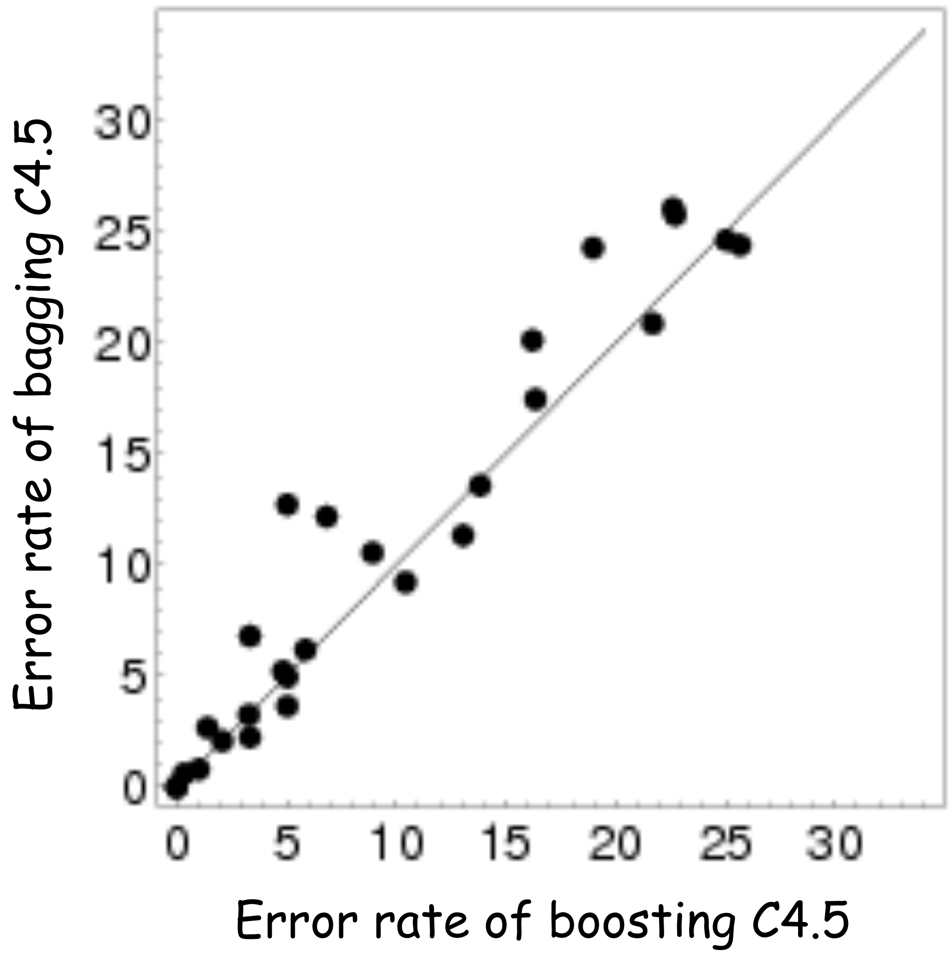
Performance

- **Data Set**
  - 27 data sets from UCI ML Repository
- **Methods for Comparison**
  - Decision tree classifier: C4.5
  - Bagging: ensembles of 100 C4.5 classifiers
  - Boosting: AdaBoost using C4.5 as the weak learner

Results (Freund and Schapire 1996)



Results (Freund and Schapire 1996)



References

- L. Breiman (1995). Bagging Predictors. Machine Learning, 24(2), 123-140.
- Y. Freund and R. Schapire (1996). Experiments with a New Boosting Algorithm. In Proc. ICML-1996, 148-156.
- T. Pedersen (2000). A Simple Approach to Build Ensembles of Naive Bayesian Classifiers for Word Sense Disambiguation. In Proc. NAACL-2000, 63-69.
- R. Schapire, Y. Freund, P. Bartlett, and W. Lee (1997). Boosting the Margin: A New Explanation for the Effectiveness of Voting Methods. In Proc. ICML-1997, 322-330.