大数据科学与应用技术

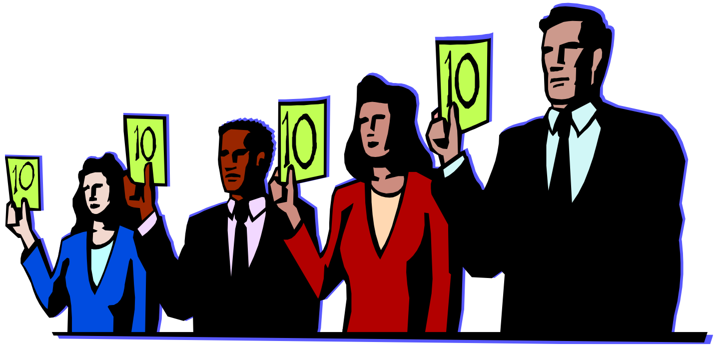
Lecture：焦在滨（ Zaibin JIAO ）

Email：jiaozaibin@mail.xjtu.edu.cn

Real World Scenarios

vs.

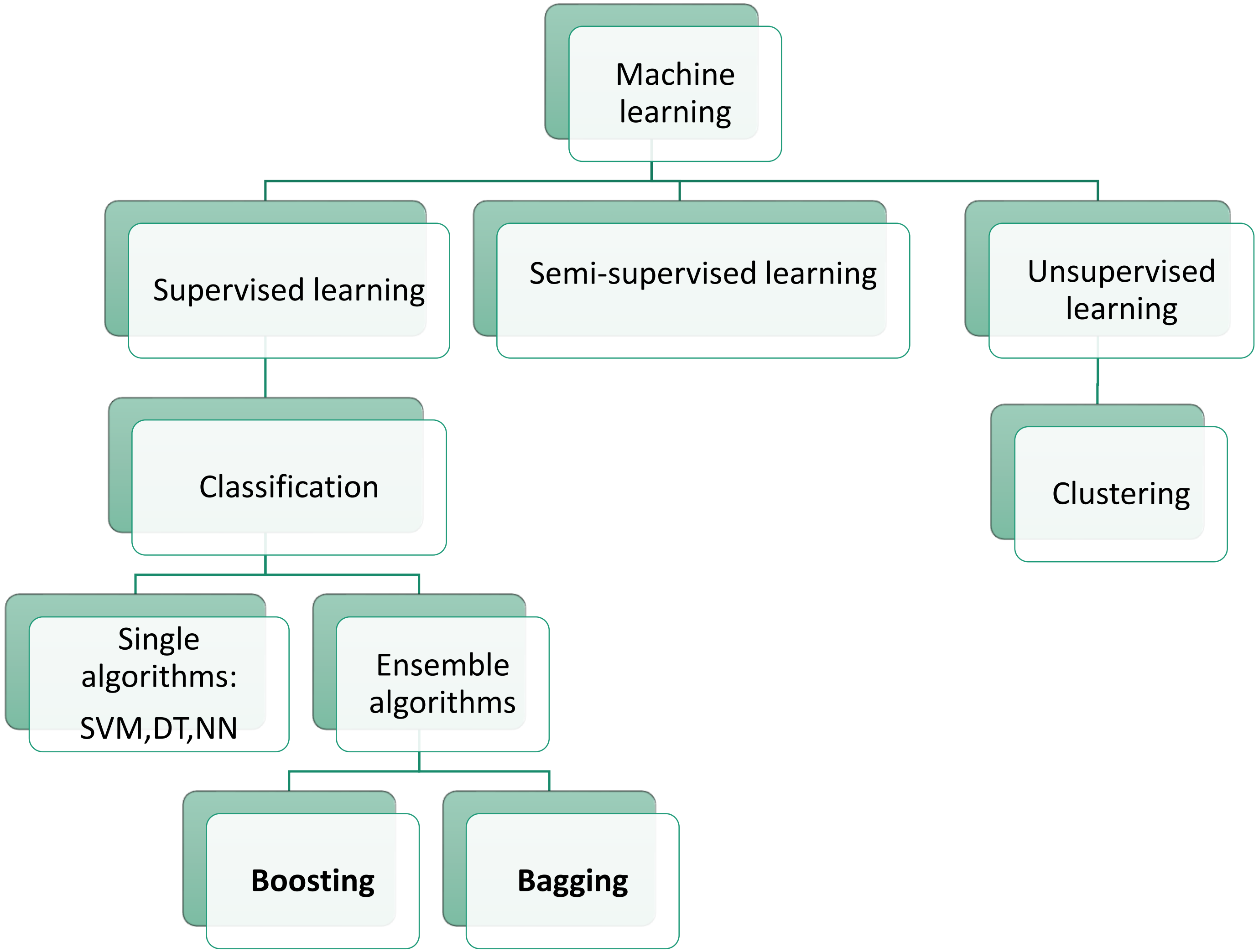
Real World Scenarios



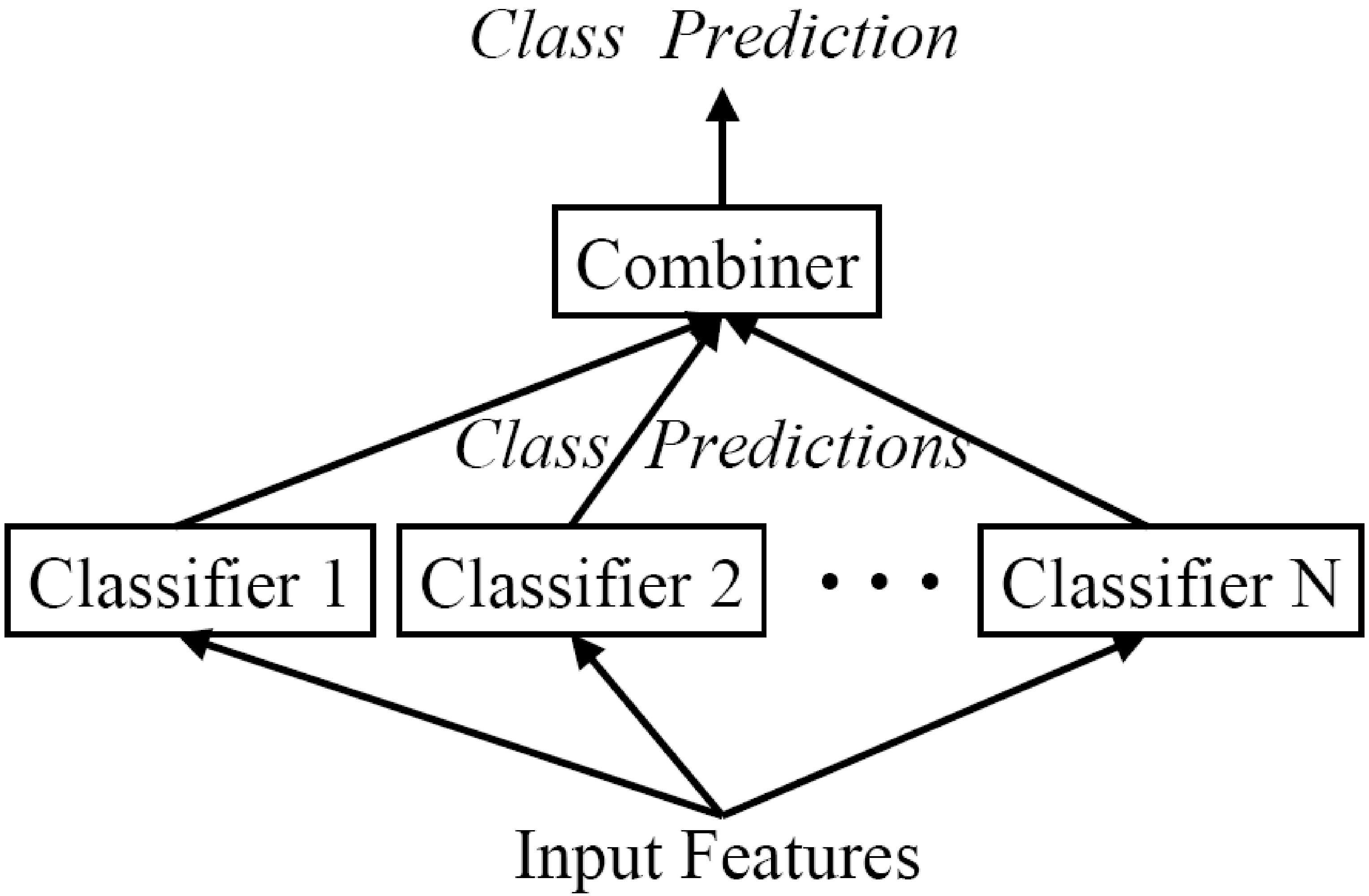
What is ensemble learning?

- Many individual learning algorithms are available:
  - Decision Trees, Neural Networks, Support Vector Machines
- The process by which multiple models are strategically generated and combined in order to better solve a particular Machine Learning problem.
- Motivations
  - To improve the performance of a single model.
  - To reduce the likelihood of an unfortunate selection of a poor model.
- Multiple Classifier Systems
- One idea, many implementations
  - Bagging
  - Boosting

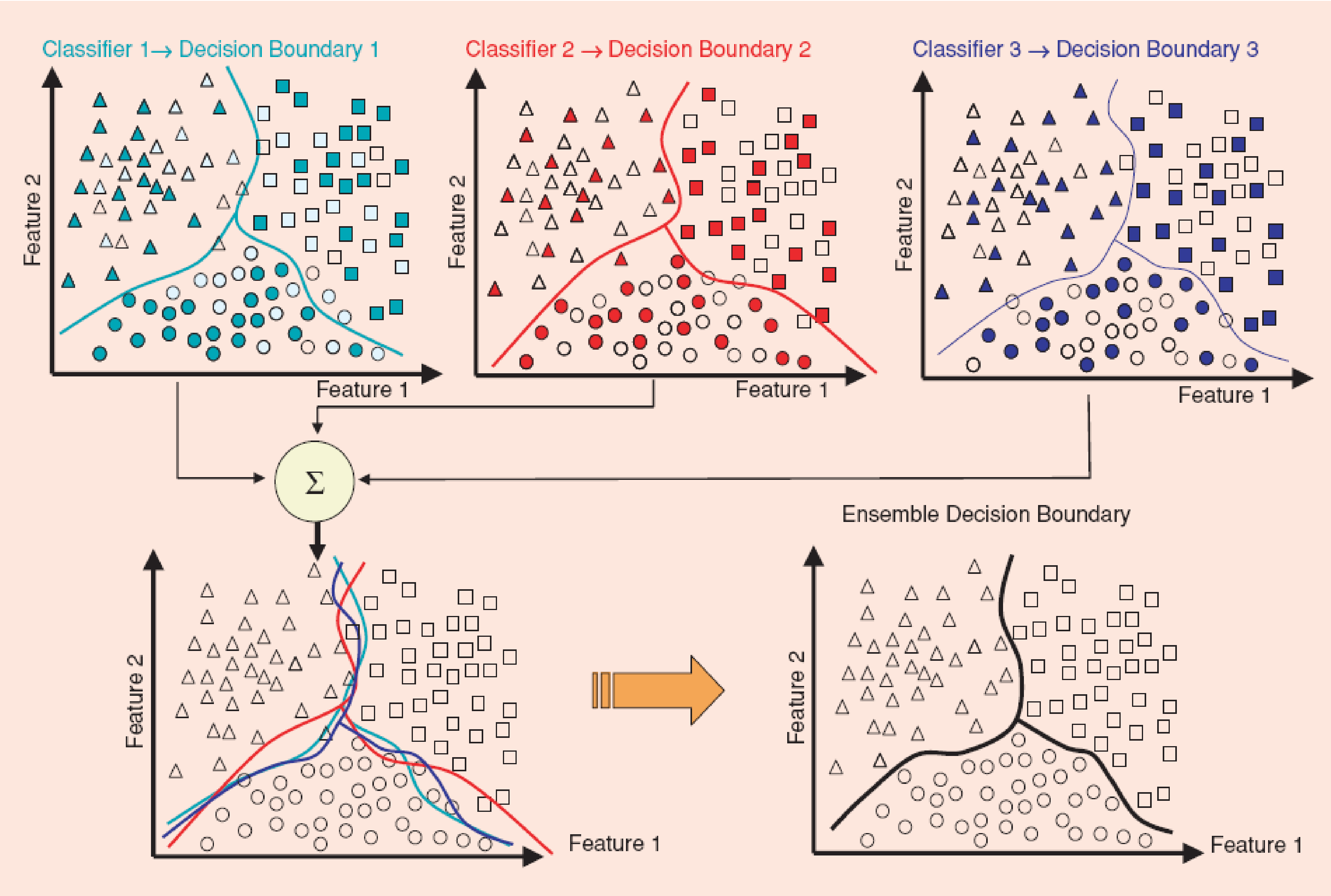
Algorithm Hierarchy



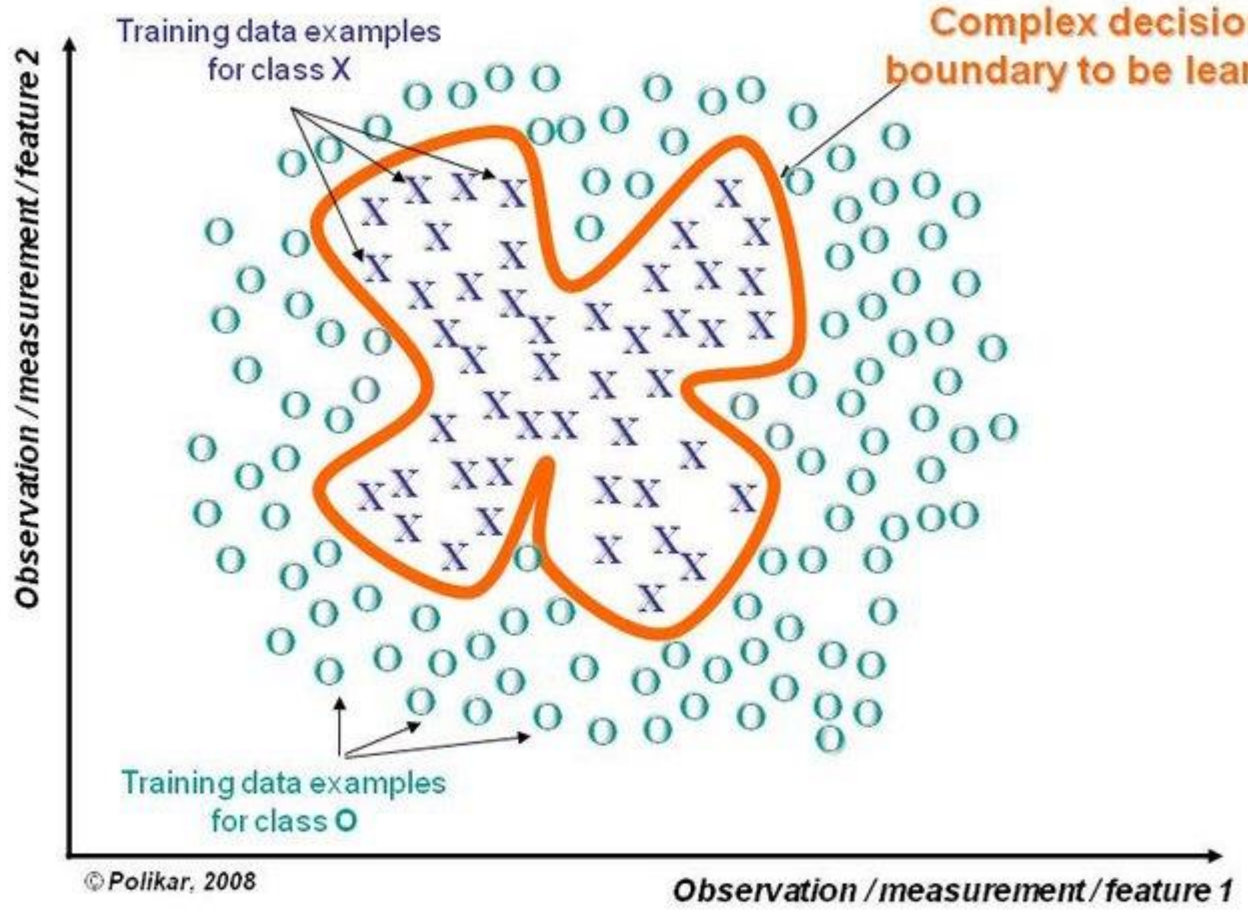
Combination of Classifiers



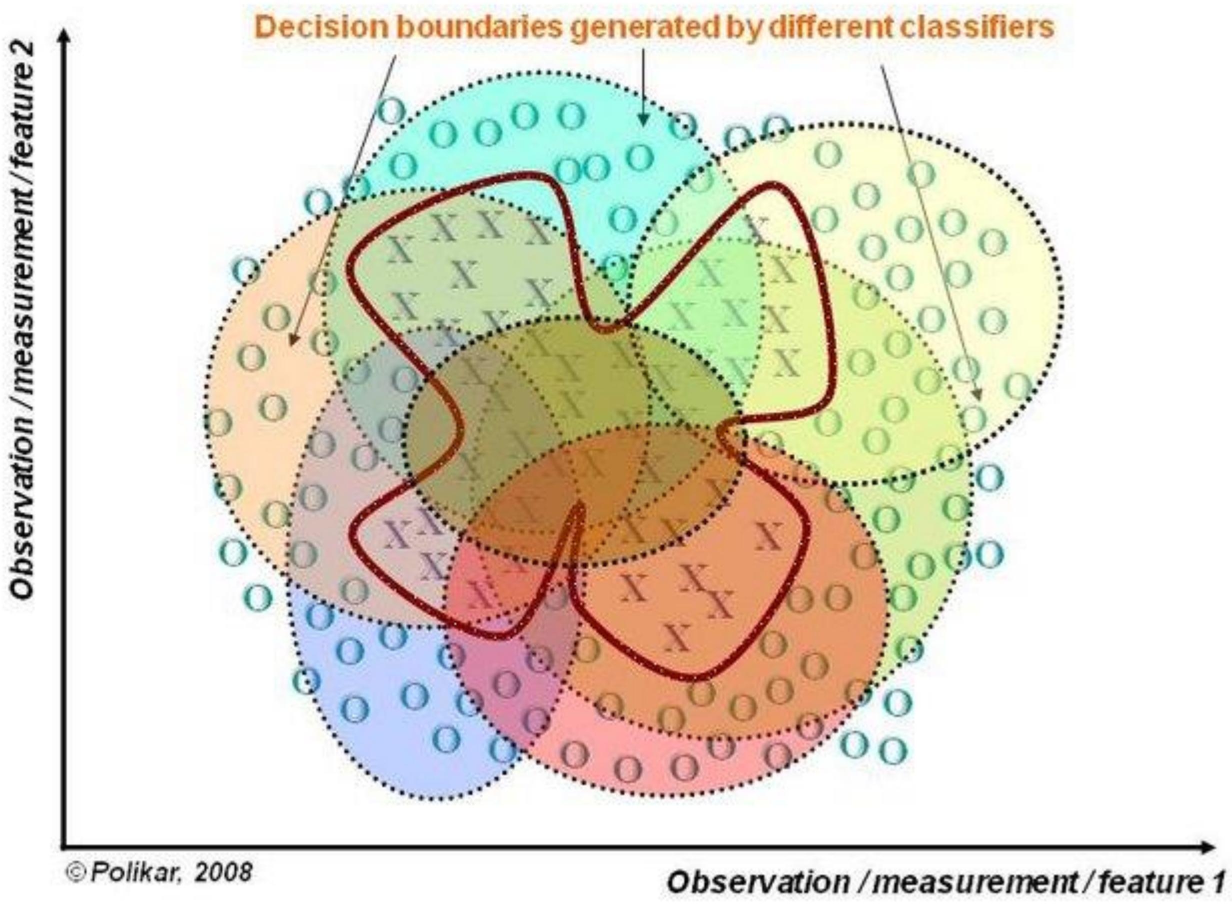
Model Selection



Divide and Conquer

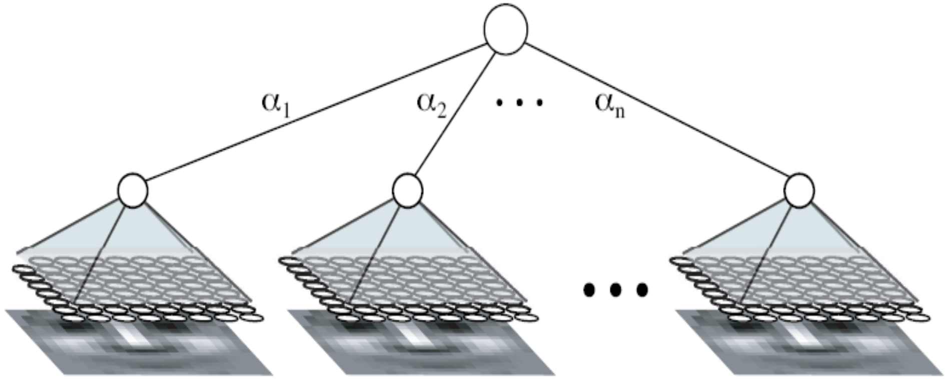


Divide and Conquer



Combiners

- How to combine the outputs of classifiers?
- Averaging
- Voting
  - Majority Voting
    - Random Forest
  - Weighted Majority Voting
    - AdaBoost
- Learning Combiner
  - General Combiner
    - Stacking
  - Piecewise Combiner
    - RegionBoost
- No Free Lunch



Diversity

- The key to the success of ensemble learning
    - Need to correct the errors made by other classifiers.
    - Does not work if all models are identical.

- Different Learning Algorithms
    - DT, SVM, NN, KNN …

- Different Training Processes
    - Different Parameters
    - Different Training Sets
    - Different Feature Sets

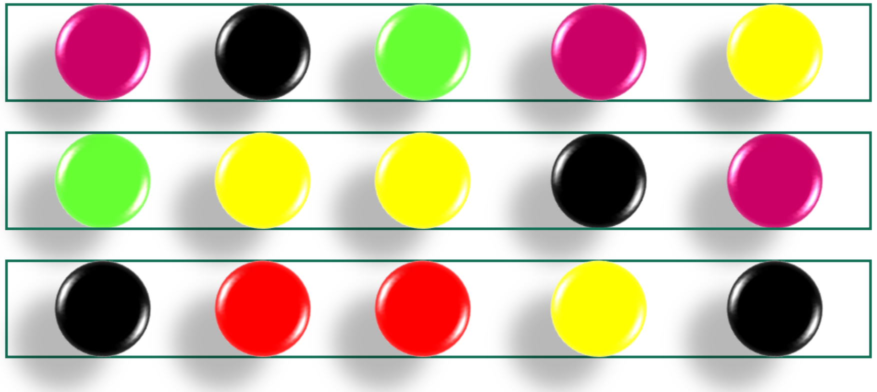
- Weak Learners
    - Easy to create different decision boundaries.
    - Stumps …

Bootstrap Samples

Sample 1

Sample 2

Sample 3



Bagging (Bootstrap Aggregating)

Algorithm: Bagging

Input:

- Training data S with correct labels $\omega_i \Omega = \{\omega_1,\ldots,\omega_C\}$ representing C classes
- Weak learning algorithm **WeakLearn,**
- Integer T specifying number of iterations.
- Percent (or fraction) F to create bootstrapped training data

Do $t=1, \ldots, T$

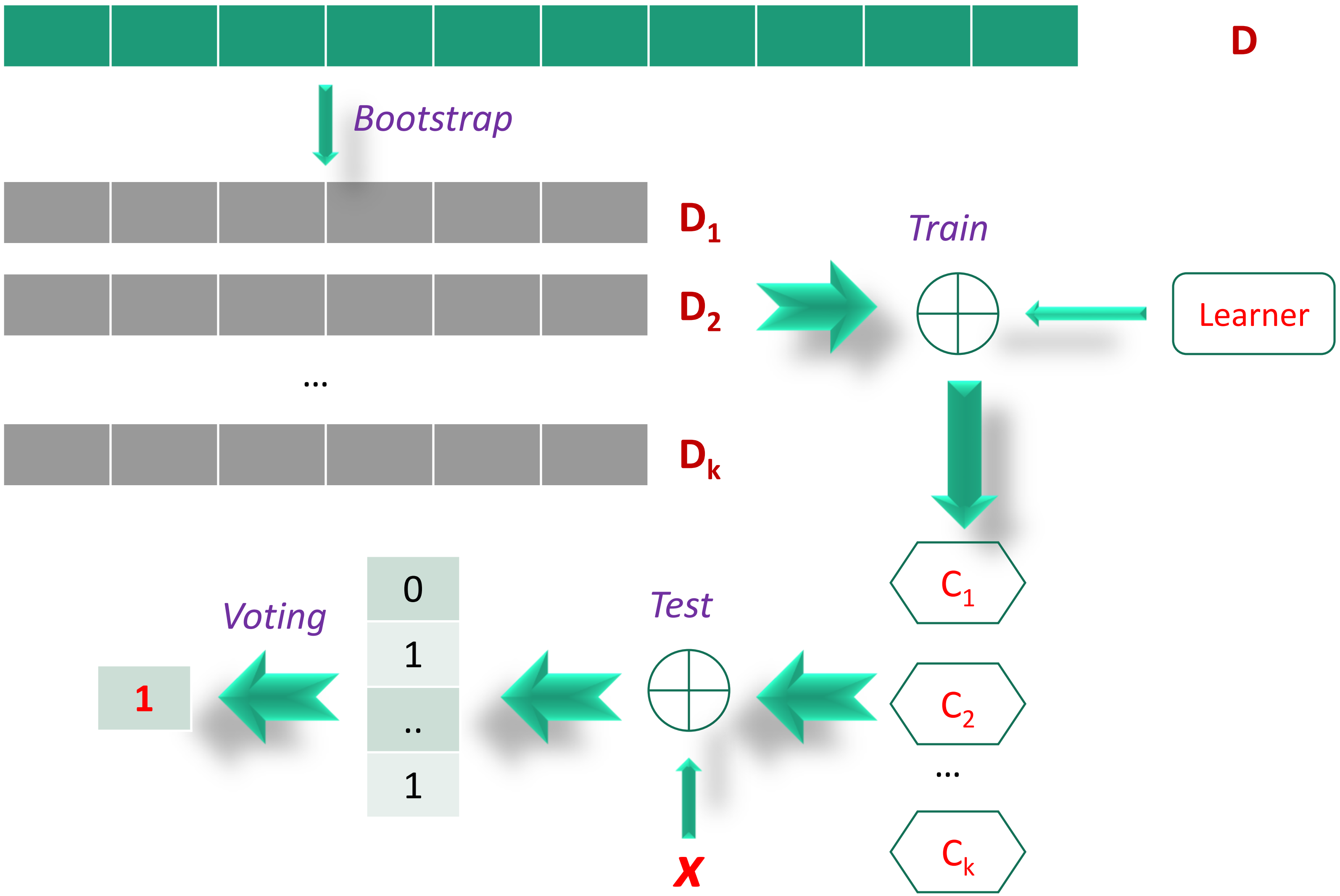
1. Take a bootstrapped replica S_t by randomly drawing F percent of S.
2. Call **WeakLearn** with S_t and receive the hypothesis (classifier) h_t.
3. Add h_t to the ensemble, $\mathcal{E}$.

End

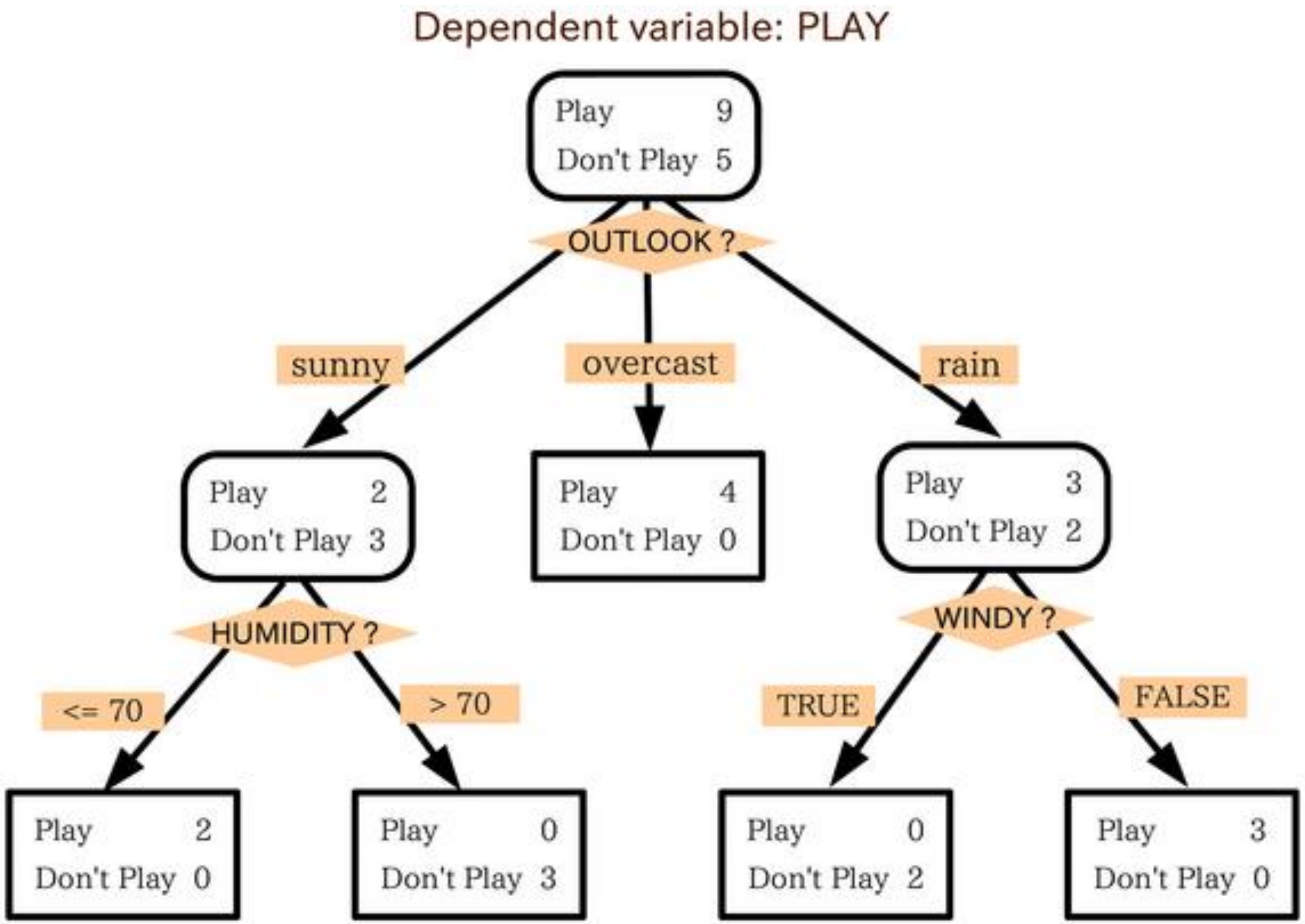
Test: Simple Majority Voting – Given unlabeled instance **x**

1. Evaluate the ensemble $\mathcal{E} = \{h_1,\ldots,h_T\}$ on **x**.
2. Let $v_{t,j} = \begin{cases} 1, & \text{if } h_t \text{ picks class } \omega_j \\ 0, & \text{otherwise} \end{cases}$ be the vote given to class ω_j by classifier h_t.
3. Obtain total vote received by each class, $V_j = \sum_{t=1}^{T} v_{t,j}$ $j = 1,\ldots,C$.
4. Choose the class that receives the highest total vote as the final classification.

Bagging



A Decision Tree



Tree vs. Forest

Random Forests

- Developed by Prof. Leo Breiman
  - Inventor of CART
  - www.stat.berkeley.edu/users/breiman/
  - Breiman, L.: Random Forests. *Machine Learning* 45(1), 5–32, 2001
- Bootstrap Aggregation (Bagging)
  - Resample with Replacement
  - Use around two third of the original data.

$$\boxed{1 - \lim_{n \to \infty} \left(1 - \frac{1}{n}\right)^n}$$

- A Collection of CART-like Trees
  - Binary Partition
  - No Pruning
  - Inherent Randomness
- Majority Voting

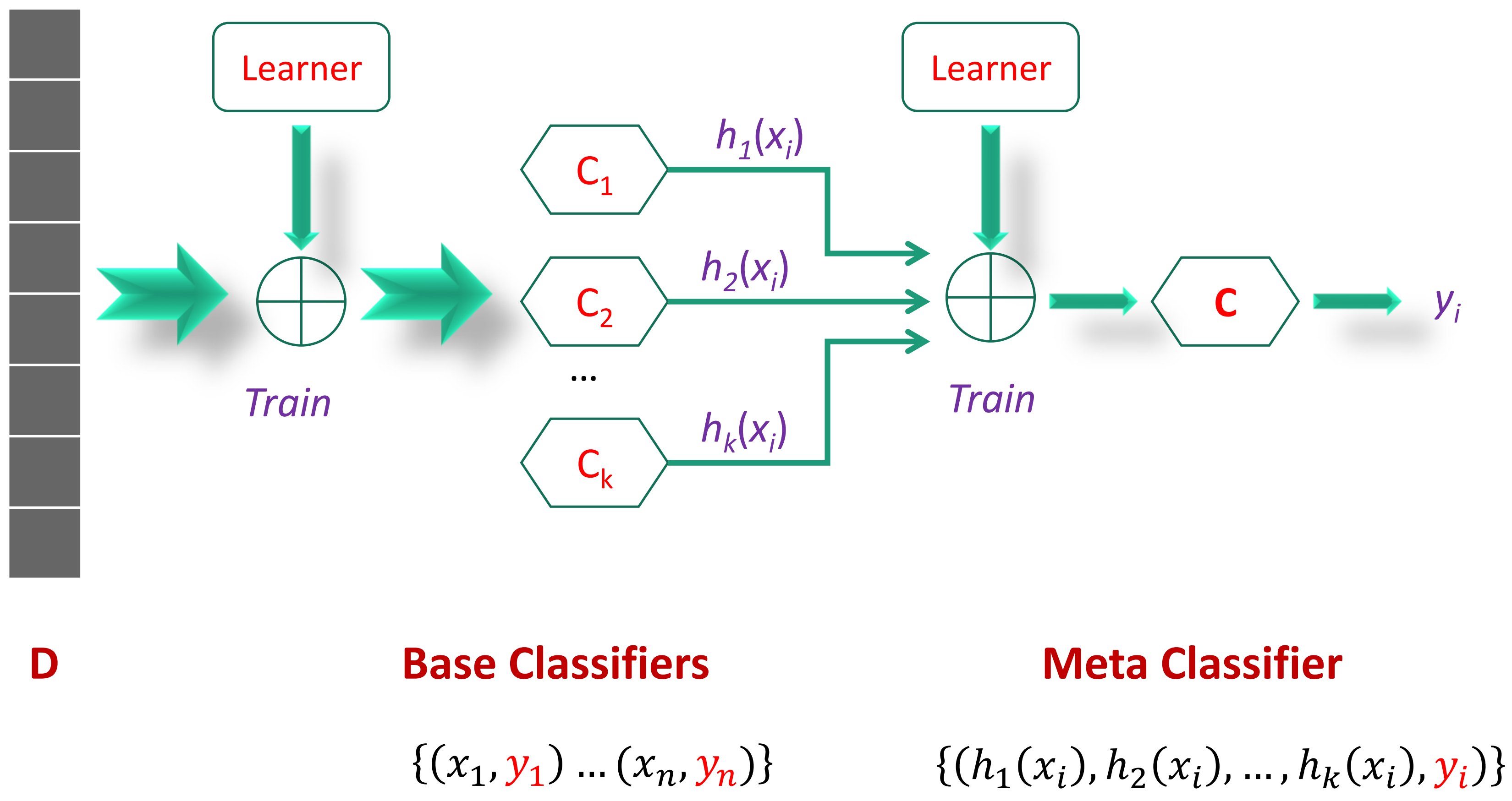
RF Main Features

- Generates substantially different trees:
  - Use random bootstrap samples of the training data.
  - Use random subsets of variables for each node.
- Number of Variables
  - Square Root (K)
  - K: total number of available variables
  - Can dramatically speed up the tree building process.
- Number of Trees
  - 500 or more
- Self-Testing
  - Around one third of the original data are left out.
  - Out of Bag (OOB)
  - Similar to Cross-Validation

RF Advantages

- All data can be used in the training process.
  - No need to leave some data for testing.
  - No need to do conventional cross-validation.
  - Data in OOB are used to evaluate the current tree.
- Performance of the entire RF
  - Each data point is tested over a subset of trees.
  - Depends on whether it is in the OOB.
- High levels of predictive accuracy
  - Only a few parameters to experiment with.
  - Suitable for both classification and regression.
- Resistant to overtraining (overfitting).
- No need for prior feature selection.

Stacking


Learner
Learner
Train
Train
C_1
C_2
...
C_k
$h_1(x_i)$
$h_2(x_i)$
$h_k(x_i)$
C
y_i
D
Base Classifiers
Meta Classifier
$\{(x_1, y_1) \dots (x_n, y_n)\}$
$\{(h_1(x_i), h_2(x_i), \dots, h_k(x_i), y_i)\}$

Stacking

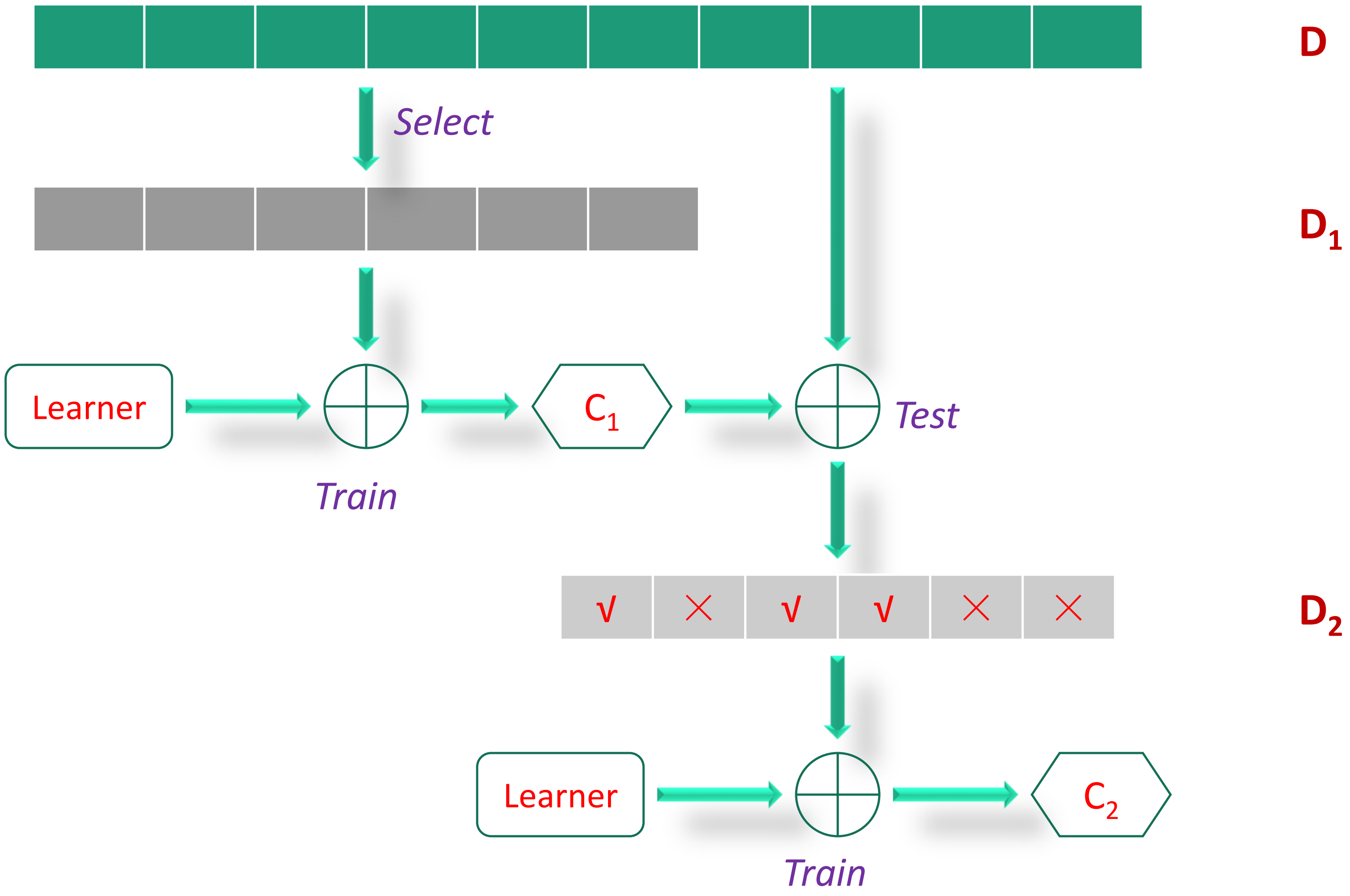
Input: Data set $\mathcal{D} = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \cdots, (\boldsymbol{x}_m, y_m)\}$;
First-level learning algorithms $\mathcal{L}_1, \cdots, \mathcal{L}_T$;
Second-level learning algorithm $\mathcal{L}$.

Process:

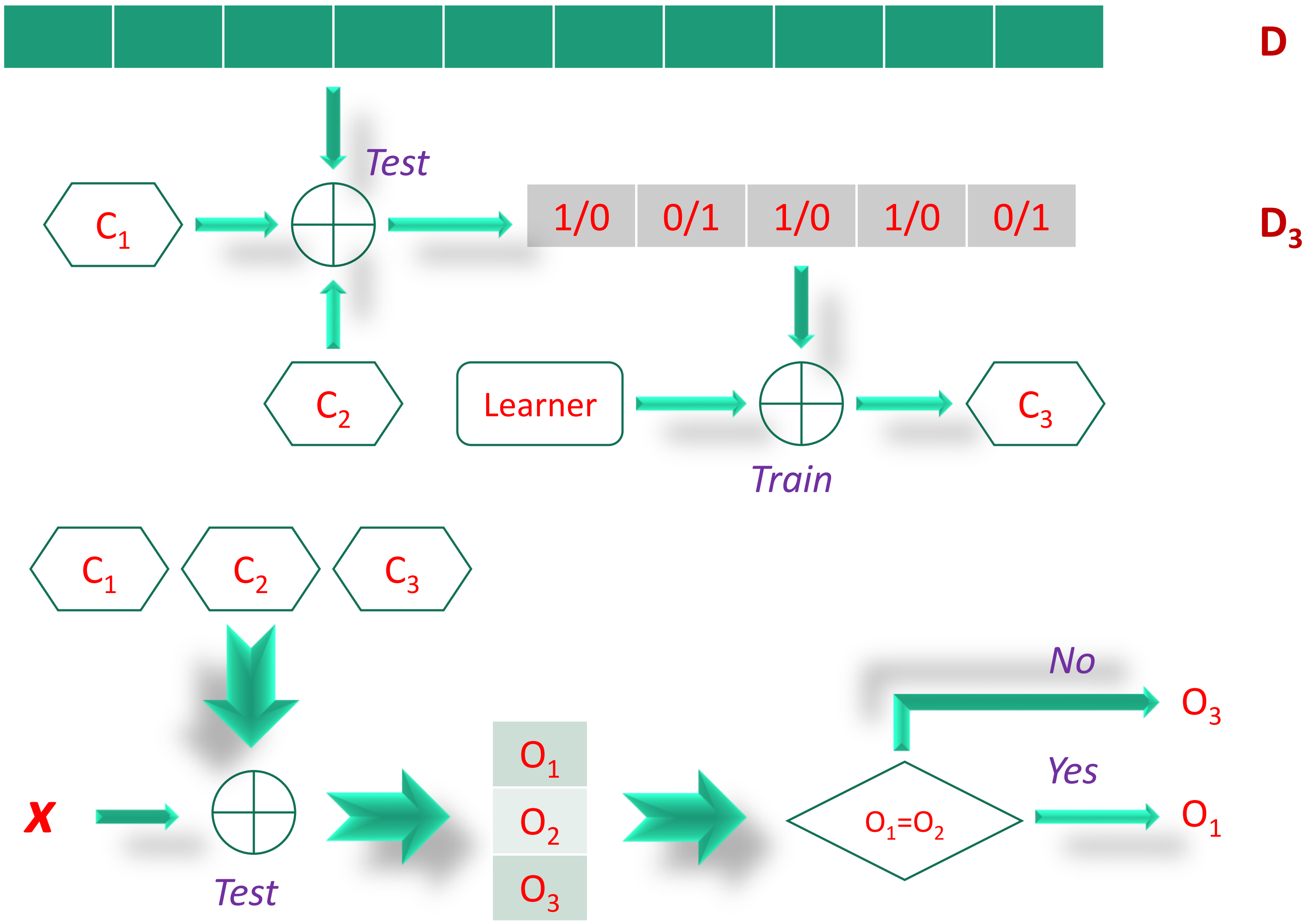
for $t = 1, \cdots, T$:
    $h_t = \mathcal{L}_t(\mathcal{D})$ % Train a first-level individual learner h_t by applying the first-level
end;   % learning algorithm $\mathcal{L}_t$ to the original data set $\mathcal{D}$
$\mathcal{D}' = \emptyset$;   % Generate a new data set
for $i = 1, \cdots, m$:
    for $t = 1, \cdots, T$:
        $z_{it} = h_t(\boldsymbol{x}_i)$ % Use h_t to classify the training example $\boldsymbol{x}_i$
    end;
    $\mathcal{D}' = \mathcal{D}' \cup \{((z_{i1}, z_{i2}, \cdots, z_{iT}), y_i)\}$
end;
$h' = \mathcal{L}(\mathcal{D}')$.   % Train the second-level learner h' by applying the second-level
    % learning algorithm $\mathcal{L}$ to the new data set $\mathcal{D}'$

Output: $H(\boldsymbol{x}) = h'(h_1(\boldsymbol{x}), \cdots, h_T(\boldsymbol{x}))$

Boosting



Boosting


D
Test
C_1
1/0
0/1
1/0
1/0
0/1
D_3
C_2
Learner
C_3
Train
C_1
C_2
C_3
No
O_3
O_1
O_2
O_3
Yes
x
Test
$O_1=O_2$
O_1

Boosting

Input: Instance distribution $\mathcal{D}$;
Base learning algorithm $\mathcal{L}$;
Number of learning rounds T.

Process:

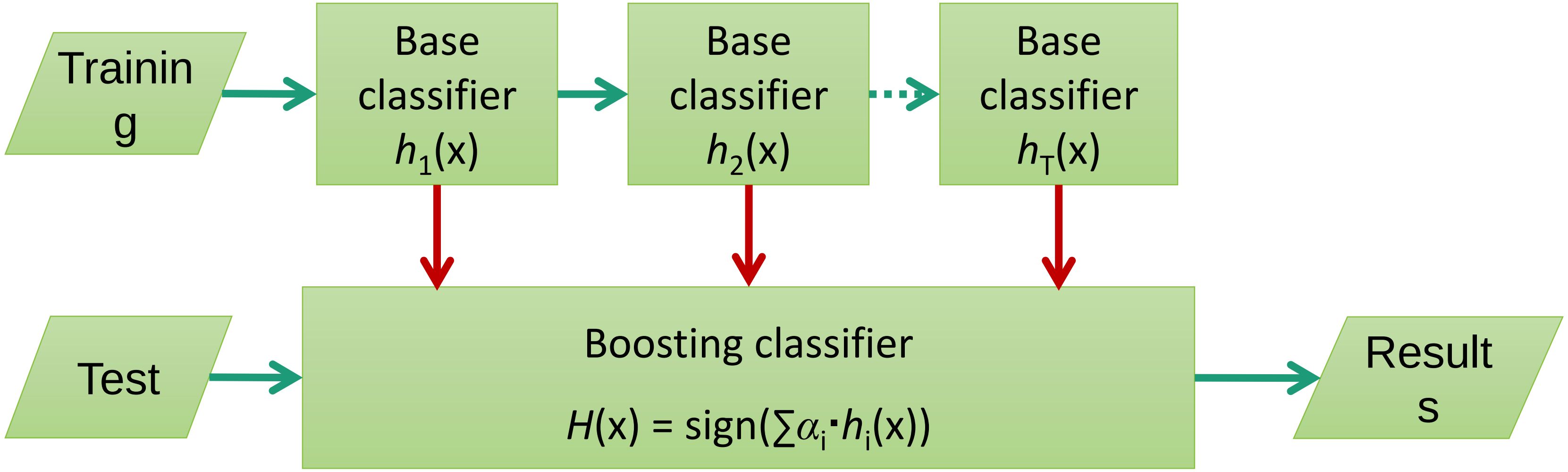
1. $\mathcal{D}_1 = \mathcal{D}$. % Initialize distribution
2. **for** $t = 1, \cdots, T$:
3. $h_t = \mathcal{L}(\mathcal{D}_t)$; % Train a weak learner from distribution $\mathcal{D}_t$
4. $\epsilon_t = \Pr_{\boldsymbol{x} \sim D_t, y} \boldsymbol{I}[h_t(\boldsymbol{x}) \neq y]$; % Measure the error of h_t
5. $\mathcal{D}_{t+1} = Adjust\_Distribution(\mathcal{D}_t, \epsilon_t)$
6. **end**

Output: $H(\boldsymbol{x}) = Combine\_Outputs(\{h_t(\boldsymbol{x})\})$

Boosting

- Bagging aims at reducing variance, not bias.
- In Boosting, classifiers are generated sequentially.
- Focuses on most informative data points.
- Training samples are weighted.
- Outputs are combined via weighted voting.
- Can create arbitrarily strong classifiers.
- The base learners can be arbitrarily weak.
- As long as they are better than random guess!

Boosting



Trainin
g
Base
classifier
h_1(x)
Base
classifier
h_2(x)
Base
classifier
h_T(x)
Test
Boosting classifier
$H(x) = \text{sign}(\sum \alpha_i \cdot h_i(x))$
Result
s

AdaBoost

Input: Data set $D = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \cdots, (\boldsymbol{x}_m, y_m)\}$;
Base learning algorithm $\mathcal{L}$;
Number of learning rounds T.

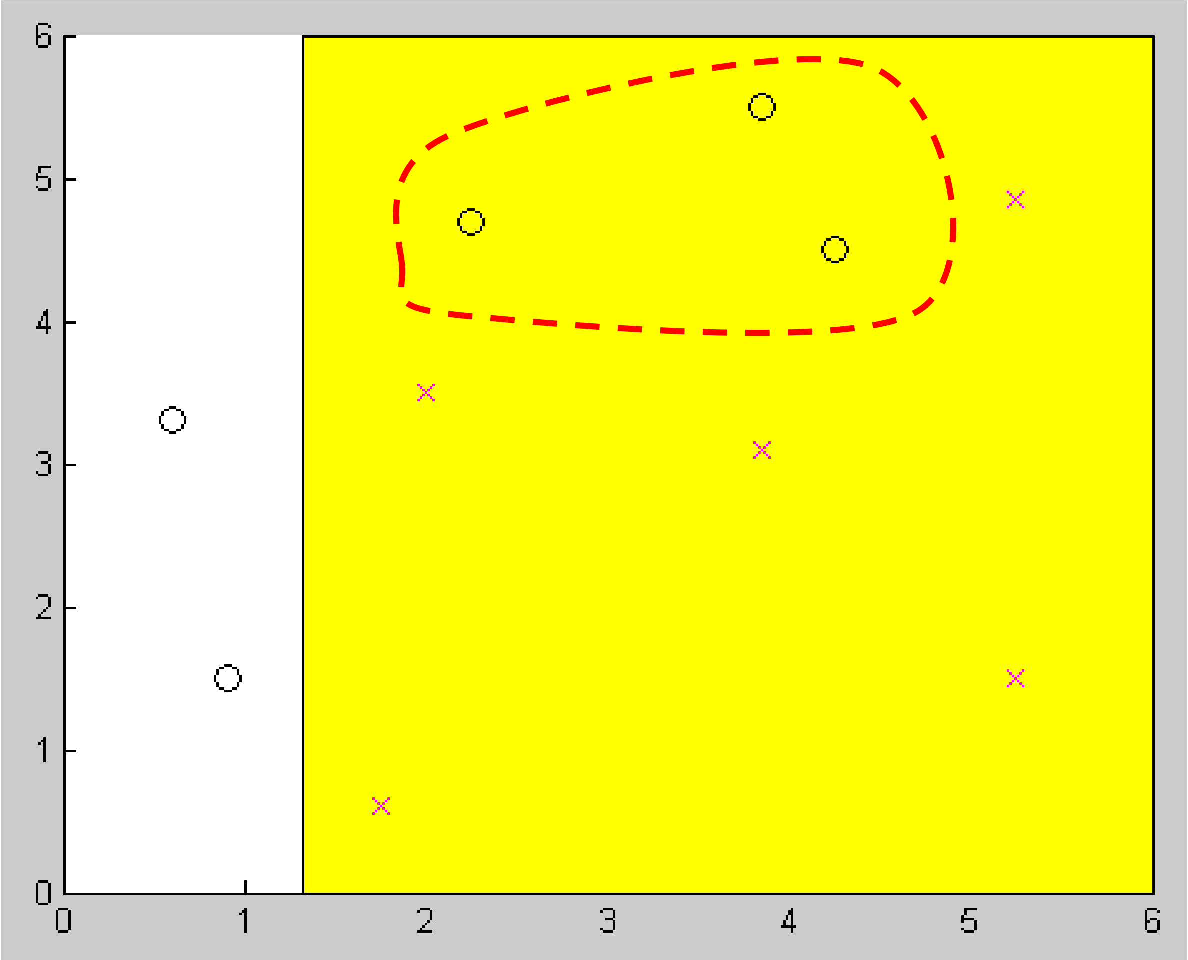
Process:

1. $\mathcal{D}_1(i) = 1/m$. % Initialize the weight distribution
2. **for** $t = 1, \cdots, T$:
3. $h_t = \mathcal{L}(D, \mathcal{D}_t)$; % Train a learner h_t from D using distribution $\mathcal{D}_t$
4. $\epsilon_t = \Pr_{\boldsymbol{x} \sim \mathcal{D}_t, y} \boldsymbol{I}[h_t(\boldsymbol{x}) \neq y]$; % Measure the error of h_t
5. **if** $\epsilon_t > 0.5$ **then break**
6. $\alpha_t = \frac{1}{2} \ln \left(\frac{1-\epsilon_t}{\epsilon_t} \right)$; % Determine the weight of h_t
7. $$\mathcal{D}_{t+1}(i) = \frac{\mathcal{D}_t(i)}{Z_t} \times \begin{cases} \exp(-\alpha_t) & \text{if } h_t(\boldsymbol{x}_i) = y_i \\ \exp(\alpha_t) & \text{if } h_t(\boldsymbol{x}_i) \neq y_i \end{cases}$$
   $$= \frac{\mathcal{D}_t(i) \exp(-\alpha_t y_i h_t(\boldsymbol{x}_i))}{Z_t}$$
   % Update the distribution, where
   % Z_t is a normalization factor which
   % enables $\mathcal{D}_{t+1}$ to be a distribution
8. **end**

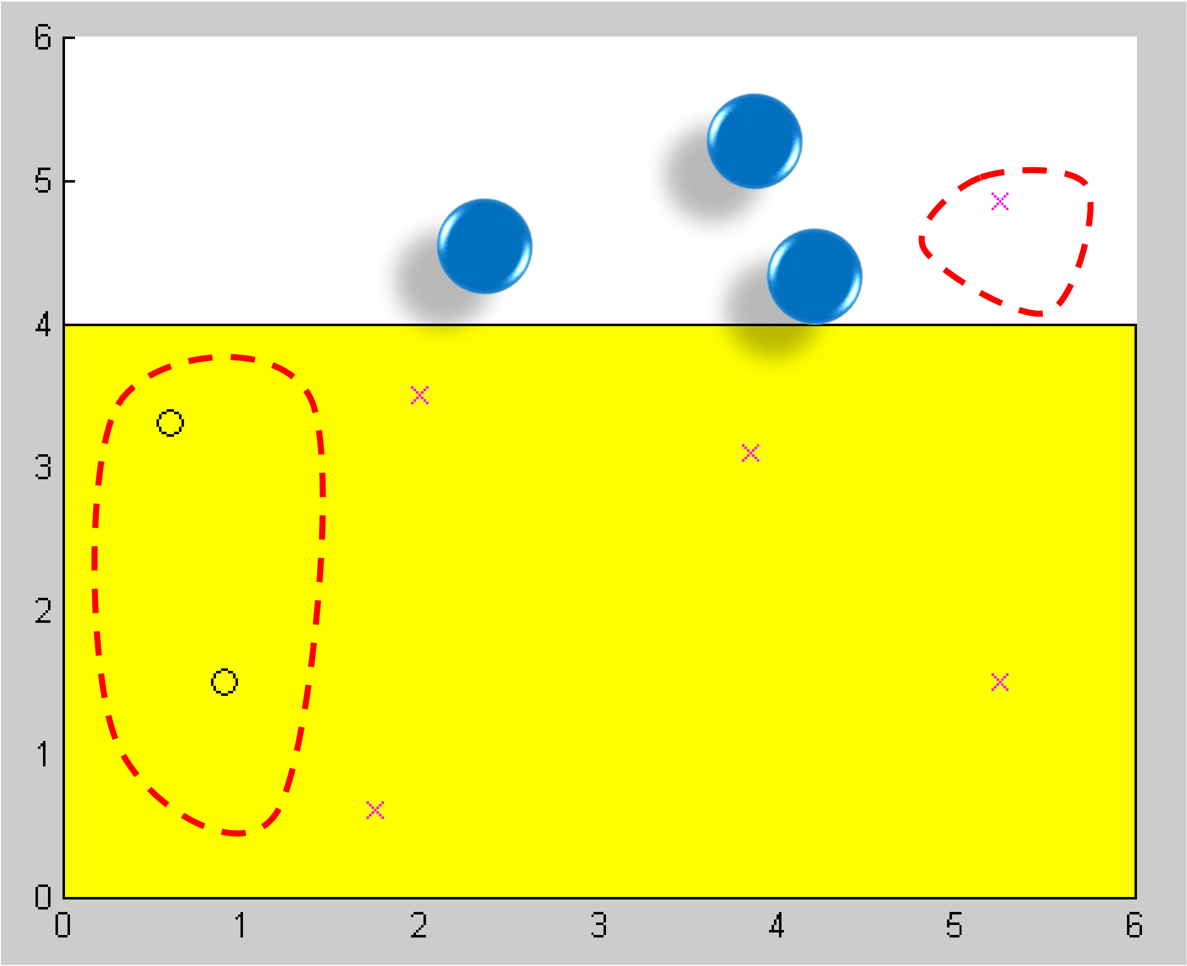
Output: $H(\boldsymbol{x}) = \text{sign} \left(\sum_{t=1}^{T} \alpha_t h_t(\boldsymbol{x}) \right)$

$$Z = \sum_i D_i e^{-\alpha y_i h(x_i)}$$

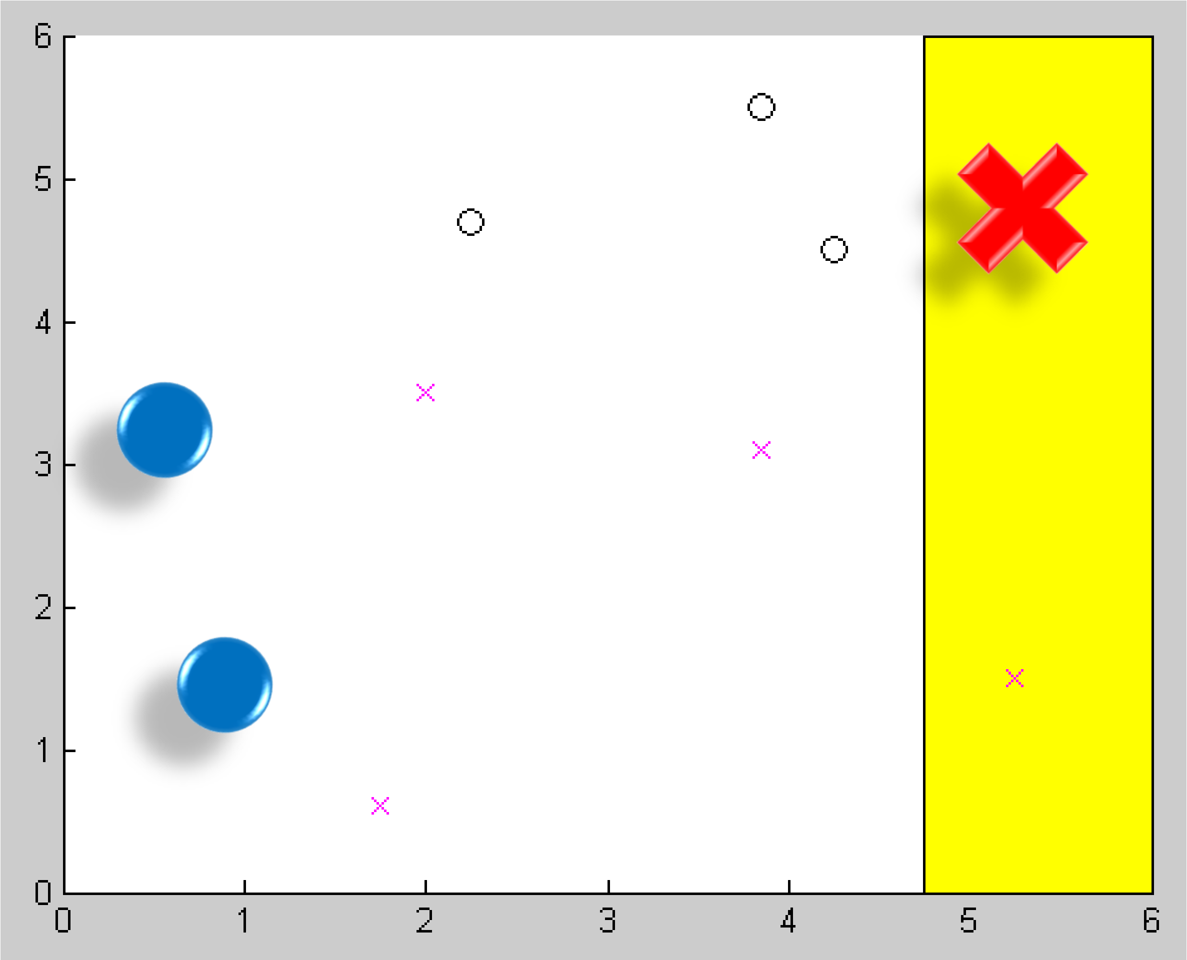
Demo: Classifier 1



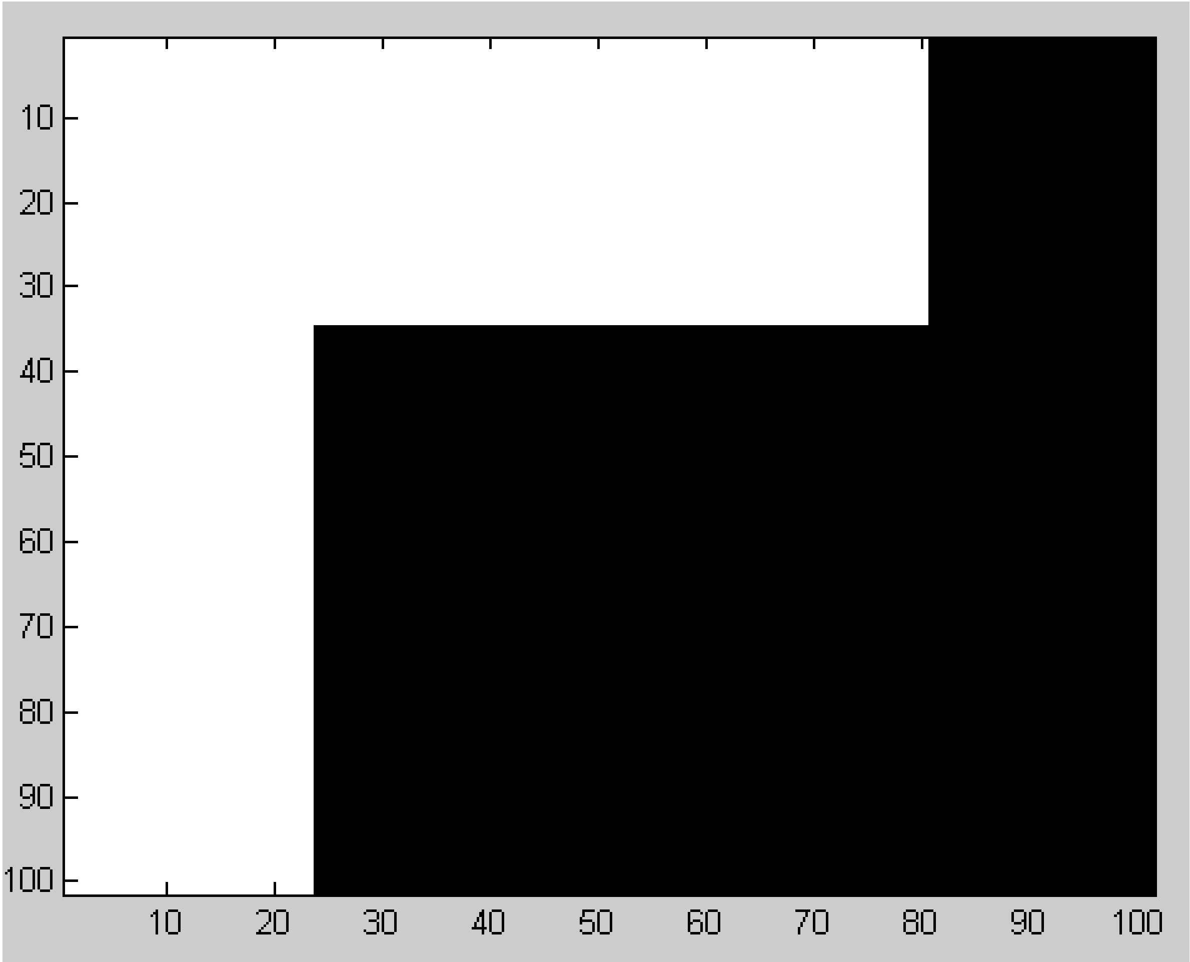
Demo: Classifier 2



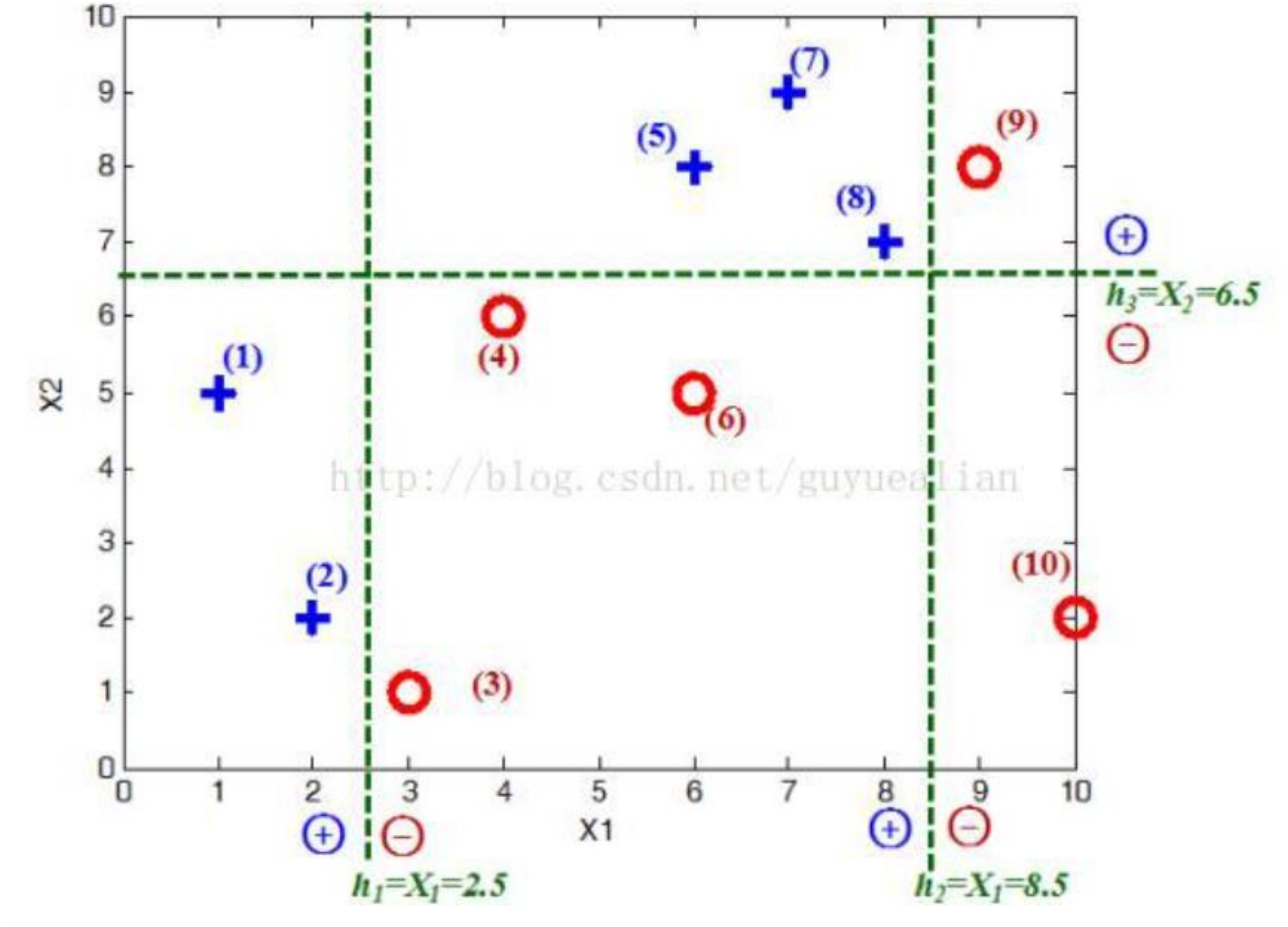
Demo: Classifier 3



Demo: Combined Classifier



Case Study



X2
X1
(1)
(2)
(3)
(4)
(5)
(6)
(7)
(8)
(9)
(10)
h₁=X₁=2.5
h₂=X₁=8.5
h₃=X₂=6.5
http://blog.csdn.net/guyuealian

Case Study

样本序号	1	2	3	4	5	6	7	8	9	10
样本点 X	(1,5)	(2,2)	(3,1)	(4,6)	(6,8)	(6,5)	(7,9)	(8,7)	(9,8)	(10,2)
类别 Y	1	1	-1	-1	1	-1	1	1	-1	-1
权值分布 D_1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1

在分类器H1(x)=h1情况下，样本点“5 7 8”被错分，因此基本分类器H1(x)的误差率为：

误差率为：

$$e_1 = (0.1 + 0.1 + 0.1) = 0.3$$

根据误差率 e_1 计算 H_1 的权重：

$$\alpha_1 = \frac{1}{2}\ln\left(\frac{1-e_1}{e_1}\right) = \frac{1}{2}\ln\left(\frac{1-0.3}{0.3}\right) = 0.4236$$

PS：这个 α_1 代表 $H_1(x)$ 在最终的分类函数中所占的权重为 0.4236

可见，被误分类样本的权值之和影响误差率e，误差率e影响基本分类器在最终分类器中所占的权重α。

Case Study

样本序号	1	2	3	4	5	6	7	8	9	10
样本点 X	(1,5)	(2,2)	(3,1)	(4,6)	(6,8)	(6,5)	(7,9)	(8,7)	(9,8)	(10,2)
类别 Y	1	1	-1	-1	1	-1	1	1	-1	-1
权值分布 D_1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1

在分类器H1(x)=h1情况下，样本点“5 7 8”被错分，因此基本分类器H1(x)的误差率为：

误差率为：

$$e_1 = (0.1 + 0.1 + 0.1) = 0.3$$

根据误差率 e_1 计算 H_1 的权重：

$$\alpha_1 = \frac{1}{2}\ln(\frac{1-e_1}{e_1}) = \frac{1}{2}\ln(\frac{1-0.3}{0.3}) = 0.4236$$

http://blog.csdn.net/guyuealian

PS：这个 α_1 代表 $H_1(x)$ 在最终的分类函数中所占的权重为 0.4236

可见，被误分类样本的权值之和影响误差率e，误差率e影响基本分类器在最终分类器中所占的权重α。

Case Study

然后，更新训练样本数据的权值分布，用于下一轮迭代，对于正确分类的训练样本“1 2 3 4 6 9 10”（共7个）的权值更新为:

$$D_2 = \frac{D_1}{2(1-\varepsilon_1)} = \frac{1}{10} \times \frac{1}{2 \times (1-0.3)} = \frac{1}{14}$$

PS：可见，**正确分类**的样本的权值由原来 1/10 减小到 1/14。

对于所有错误分类的训练样本 “5 7 8” （共 3 个）的权值更新为：

$$D_2(i) = \frac{D_1(i)}{2e_1} = \frac{1}{10} \times \frac{1}{2 \times 03} = \frac{1}{6}$$

PS：可见，**错误分类**的样本的权值由原来 1/10 增大到 1/6。

这样，第1轮迭代后，最后得到各个样本数据新的权值分布:

D2=[1/14,1/14,1/14,1/14,1/6,1/14,1/6,1/6,1/14,1/14]

Case Study

样本序号	1	2	3	4	5	6	7	8	9	10
样本点 X	(1,5)	(2,2)	(3,1)	(4,6)	(6,8)	(6,5)	(7,9)	(8,7)	(9,8)	(10,2)
类别 Y	1	1	-1	-1	1	-1	1	1	-1	-1
权值分布 $D1$	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
权值分布 $D2$	1/14	1/14	1/14	1/14	1/6	1/14	1/6	1/6	1/14	1/14
$sign(f_1(x))$	1	1	-1	-1	-1	-1	-1	-1	-1	-1

被 $H_1(x)$分错的样本

PS：用浅绿色底纹标记的表格，是被 $H_1(x)$分错的样本“5 7 8”；没有底纹（白色的）是正确分类的样本

Case Study

$$H_2 = \begin{cases} 1, & X_1 < 8.5 \\ -1, & X_1 > 8.5 \end{cases}$$

显然，H2(x)把样本“3 4 6”分错了，根据D2可知它们的权值为D2(3)=1/14，D2(4)=1/14，D2(6)=1/14，所以H2(x)在训练数据集上的误差率：

序号	1	2	3	4	5	6	7	8	9	10
样本点 X	(1,5)	(2,2)	(3,1)	(4,6)	(6,8)	(6,5)	(7,9)	(8,7)	(9,8)	(10,2)
类别 Y	1	1	-1	-1	1	-1	1	1	-1	-1
权值分布 $D1$	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
权值分布 $D2$	1/14	1/14	1/14	1/14	1/6	1/14	1/6	1/6	1/14	1/14
$sign(f_1(x))$	1	1	-1	-1	-1	-1	-1	-1	-1	-1
权值分布 $D3$	1/22	1/22	1/6	1/6	7/66	1/6	7/66	7/66	1/22	1/22
$sign(f_2(x))$	1	1	1	1	1	1	1	1	-1	-1

被 H_2(x)分错的样本

D3=[1/22,1/22,1/6,1/6,7/66,1/6,7/66,7/66,1/22,1/22]

Case Study

序号	1	2	3	4	5	6	7	8	9	10
样本点 X	(1,5)	(2,2)	(3,1)	(4,6)	(6,8)	(6,5)	(7,9)	(8,7)	(9,8)	(10,2)
类别 Y	1	1	-1	-1	1	-1	1	1	-1	-1
权值分布 $D1$	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
权值分布 $D2$	1/14	1/14	1/14	1/14	1/6	1/14	1/6	1/6	1/14	1/14
$sign(f_1(x))$	1	1	-1	-1	-1	-1	-1	-1	-1	-1
权值分布 $D3$	1/22	1/22	1/6	1/6	7/66	1/6	7/66	7/66	1/22	1/22
$sign(f_2(x))$	1	1	1	1	1	1	1	1	-1	-1
权值分布 $D4$	1/6	1/6	11/114	11/114	7/114	11/114	7/114	7/114	1/6	1/38
$sign(f_3(x))$	1	1	-1	-1	1	-1	1	1	-1	-1

$$H_{final} = sign\left(\sum_{t=1}^{T} \alpha_t H_t(x)\right) = sign\left(0.4236H_1(x) + 0.6496H_2(x) + 0.9229H_3(x)\right)$$

The Choice of α

Theorem 1: *Error is minimized by minimizing* Z_t

Proof:

$$D_{T+1}(i) = \frac{1}{m} \cdot \frac{e^{-y_i \alpha_1 h_1(x_i)}}{Z_1} \cdot \ldots \cdot \frac{e^{-y_i \alpha_T h_T(x_i)}}{Z_T}$$

$$= \frac{e^{\sum_t -y_i \alpha_t h_t(x_i)}}{m \prod_t Z_t} = \frac{e^{-y_i \sum_t \alpha_t h_t(x_i)}}{m \prod_t Z_t}$$

$$= \frac{e^{-y_i f(x_i)}}{m \prod_t Z_t}$$

$$Z = \sum_i D_i e^{-\alpha y_i h(x_i)}$$

$$f(x_i) = \sum_t \alpha_t h_t(x_i)$$

$$H(x_i) \neq y_i \Rightarrow y_i f(x_i) \leq 0 \Rightarrow e^{-y_i f(x_i)} \geq 1$$

$$[\![H(x_i) \neq y_i]\!] \leq e^{-y_i f(x_i)}$$

$$\frac{1}{m} \sum_i [\![H(x_i) \neq y_i]\!] \leq \frac{1}{m} \sum_i e^{-y_i f(x_i)}$$

Model Error

The Choice of α

$$D_{T+1}(i) = \frac{e^{-y_i\ f(x_i)}}{m\prod_t Z_t}$$

Combining these results,

$$\begin{aligned} \frac{1}{m}\sum_i [\![H(x_i) \neq y_i]\!] &\leq \frac{1}{m}\sum_i e^{-y_i f(x_i)} \\ &= \sum_i \left(\prod_t Z_t\right) D_{T+1}(i) \\ &= \prod_t Z_t \quad \text{(since } D_{T+1} \text{ sums to 1).} \end{aligned}$$

Thus, we can see that minimizing Z_t will minimize this error bound.

$$\min_\alpha Z_t \Rightarrow \min \prod_t Z_t$$

The Choice of α

$$y, h(x) \in \{-1, +1\} \qquad Z = \sum_i D_i e^{-\alpha y_i h(x_i)}$$

$$e^{-\alpha y_i h(x_i)} = \boxed{e^{-\alpha} P(y_i = h(x_i)) + e^{\alpha} P(y_i \neq h(x_i))}$$

$$\frac{\partial Z}{\partial \alpha} = -e^{-\alpha} \sum_i D_i P\big(y_i = h(x_i)\big) + e^{\alpha} \sum_i D_i P\big(y_i \neq h(x_i)\big) = 0$$

$$\alpha = \frac{1}{2} \ln \frac{\sum_i D_i \big(1 - P\big(y_i \neq h(x_i)\big)\big)}{\sum_i D_i P\big(y_i \neq h(x_i)\big)} = \boxed{\frac{1}{2} \ln \frac{1-\varepsilon}{\varepsilon}}$$

Error Bounds

$$r = \sum_i D_i y_i h(x_i) \longrightarrow \varepsilon = \frac{1-r}{2} \longrightarrow \alpha = \frac{1}{2}\ln\frac{1+r}{1-r}$$

$$Z = \sum_i D_i e^{-\alpha y_i h(x_i)} = \sum_i D_i e^{-\left(\frac{1}{2}\ln\frac{1+r}{1-r}\right) y_i h(x_i)} = \sum_i D_i \left(\sqrt{\frac{1+r}{1-r}}\right)^{-y_i h(x_i)}$$

$$= \sum_i D_i \left(\sqrt{\frac{1+r}{1-r}} P\big(y_i \neq h(x_i)\big) + \sqrt{\frac{1-r}{1+r}} P\big(y_i = h(x_i)\big)\right)$$

$$= \sqrt{\frac{1+r}{1-r}}\varepsilon + \sqrt{\frac{1-r}{1+r}}(1-\varepsilon) = \frac{1}{1-r}\sqrt{1-r^2}\frac{1-r}{2} + \frac{1}{1+r}\sqrt{1-r^2}\frac{1+r}{2}$$

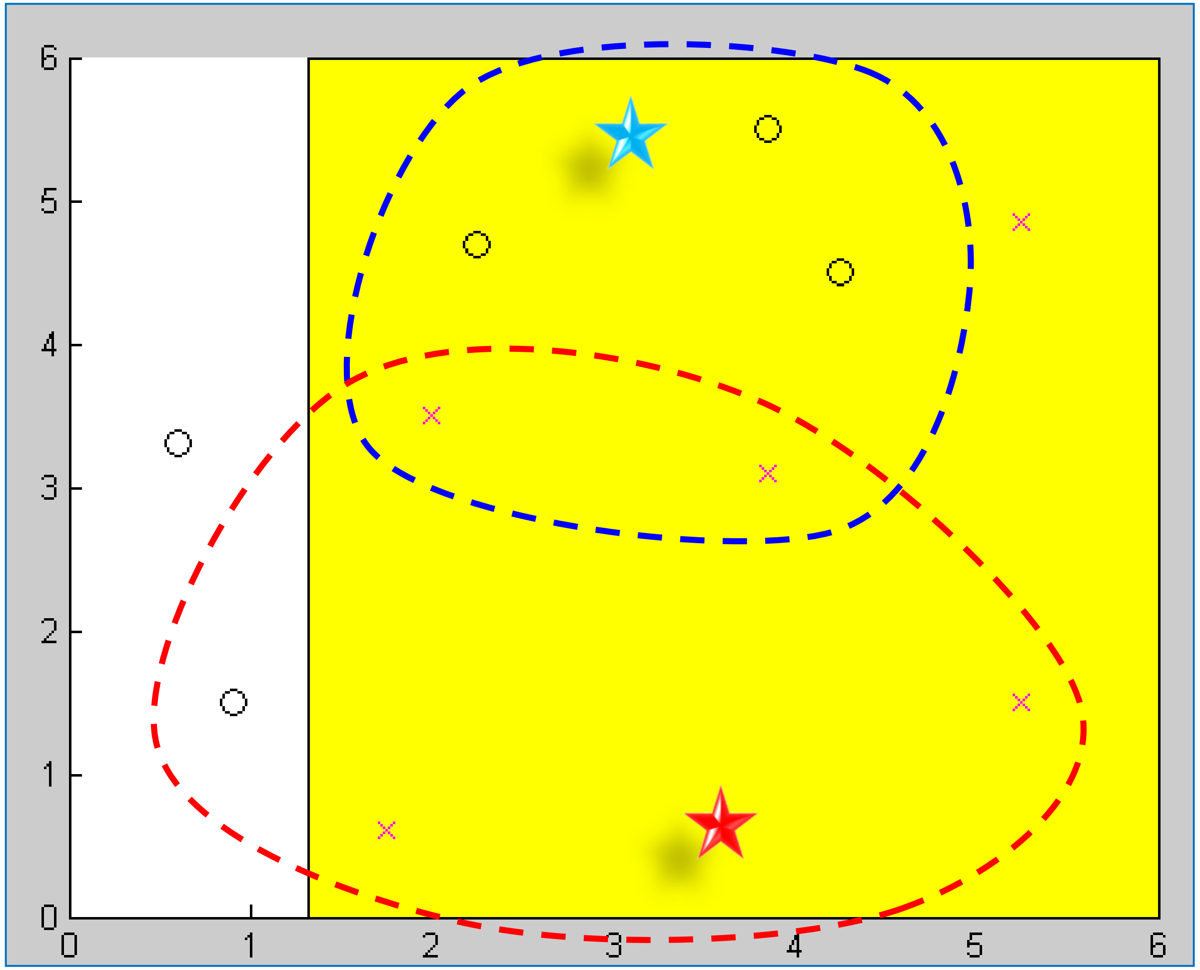
$$= \sqrt{1-r^2}$$

$$\frac{1}{m}[\![H(x_i) \neq y_i]\!] \leq \prod_t Z_t = \prod_t \sqrt{1-{r_t}^2}$$

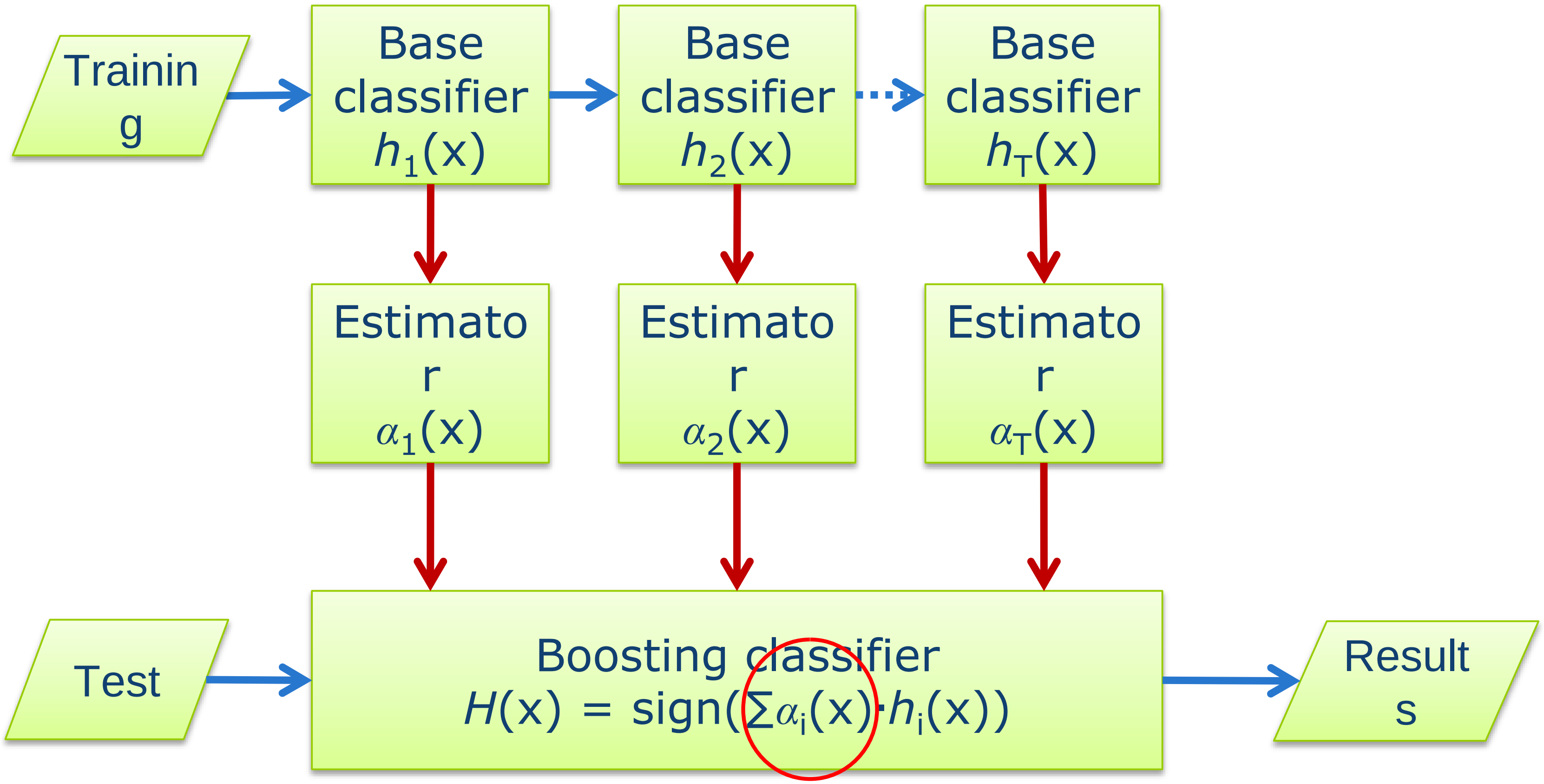
Summary of AdaBoost

- Advantages
  - Simple and easy to implement
  - Almost no parameters to tune
  - Proven upper bounds on training set
  - Immune to overfitting
- Disadvantages
  - Suboptimal α values
  - Steepest descent
  - Sensitive to noise
- Future Work
  - Theory
  - Comprehensibility
  - New Framework

Fixed Weighting Scheme



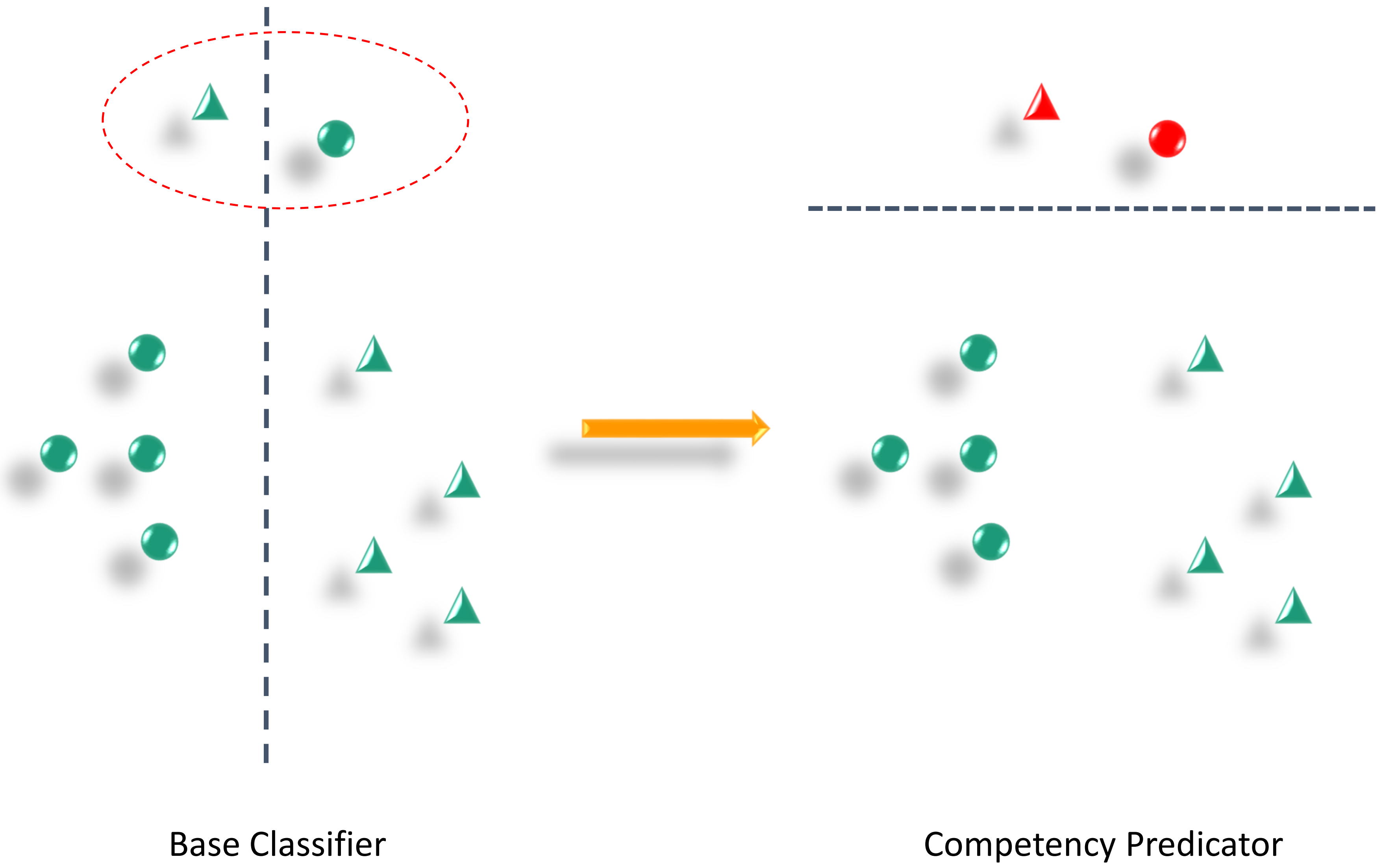
Dynamic Weighting Scheme



RegionBoost

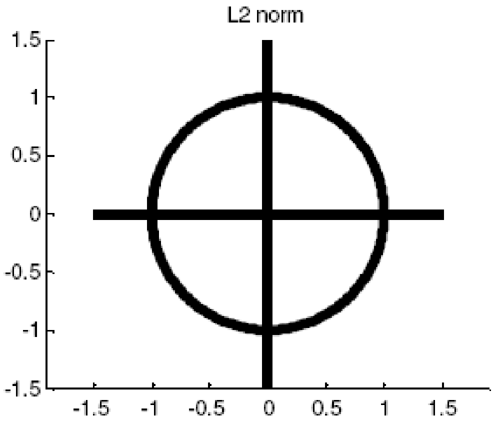
- AdaBoost assigns fixed weights to models.
- However, different models emphasize different regions.
- The weights of models should be input-dependent.
- Given an input, only invoke appropriate models.
- Train a competency predictor for each model.
- Estimate whether the model is likely to make a right decision.
- Use this information as the weight.
- Maclin, R.: Boosting classifiers regionally. AAAI, 700-705, 1998.

RegionBoost

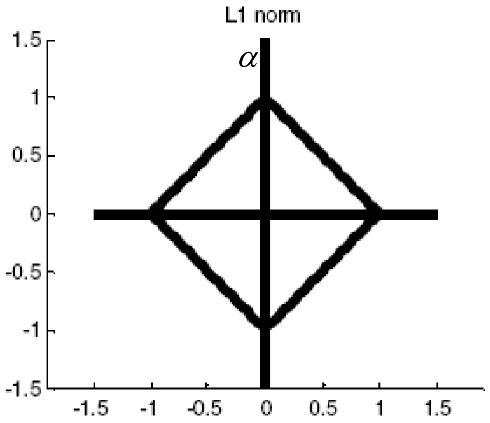


RegionBoost with KNN

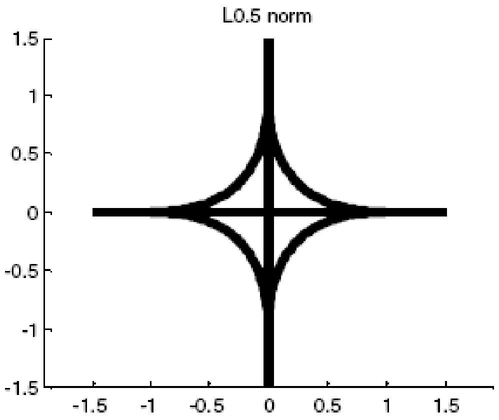
- To calculate $\alpha_j(x_i)$:
  - Find the K nearest neighbors of x_i in the training set.
  - Calculate the percentage of points correctly classified by h_j.



L_2 norm

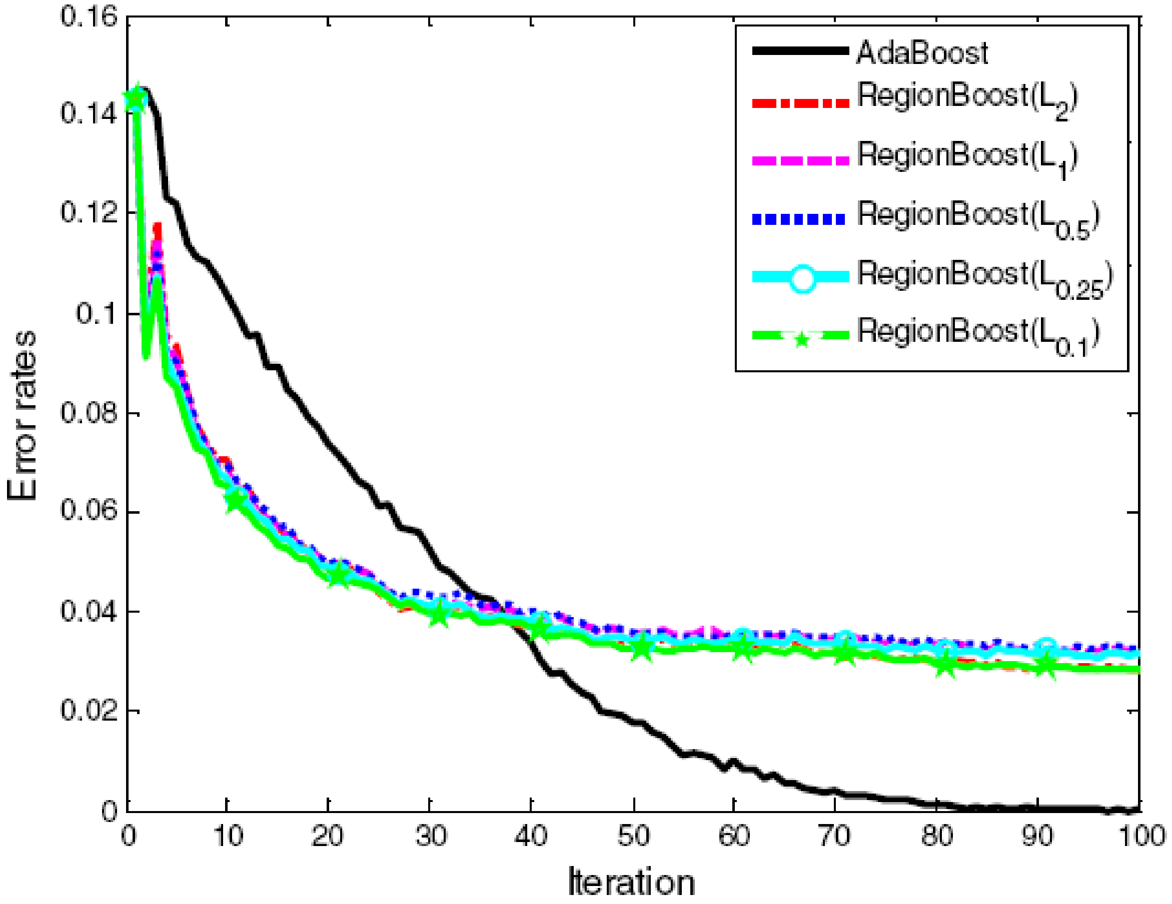


L_1 norm



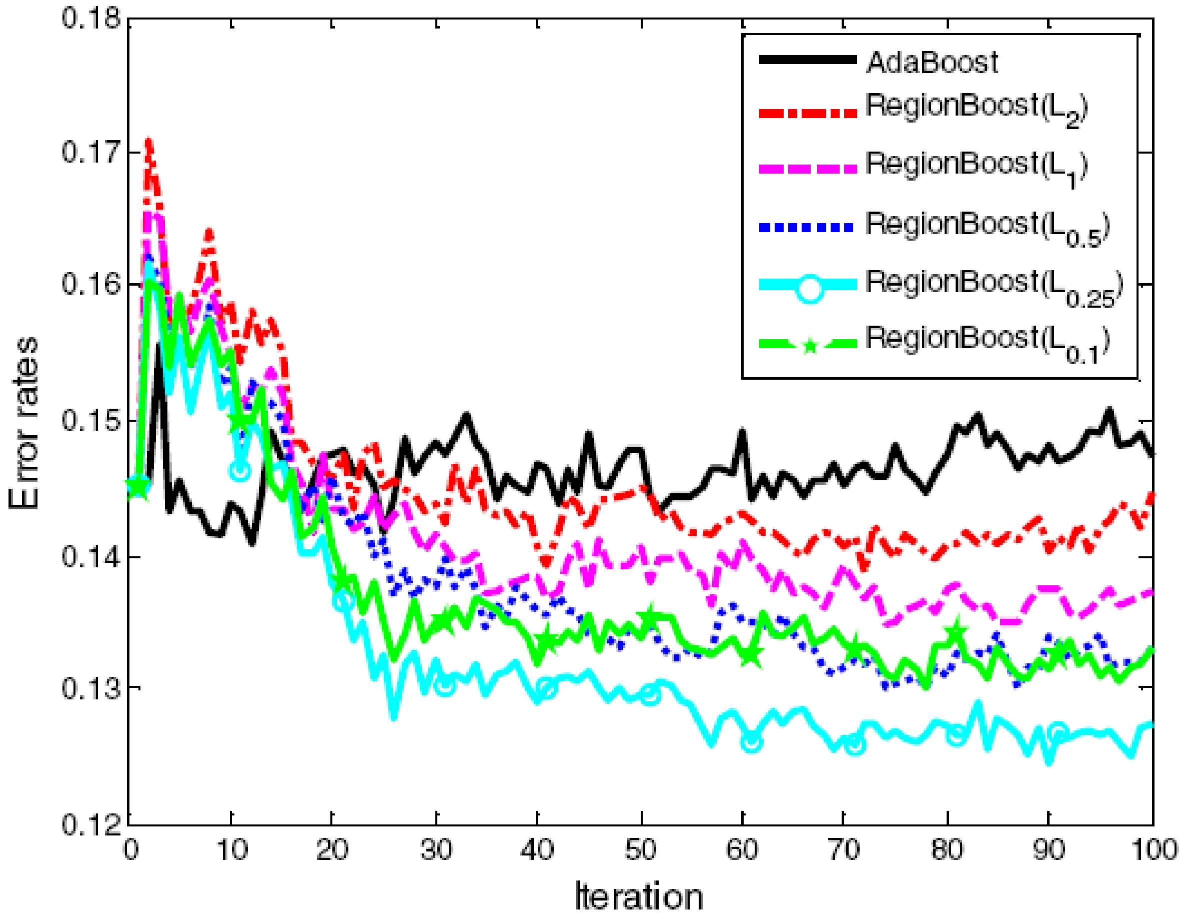
$L_{0.5}$ norm

RegionBoost Results



(a) Training error rates on Credit_Aus

RegionBoost Results



(b) Test error rates on Credit_Aus

Review

- What is ensemble learning?
- What can ensemble learning help us?
- Two major types of ensemble learning:
  - Parallel (Bagging)
  - Sequential (Boosting)
- Different ways to combine models:
  - Average
  - Majority Voting
  - Weighted Majority Voting
- Some representative algorithms:
  - Random Forests
  - AdaBoost
  - RegionBoost

BIG
DATA
ALSO
TOOLS
DISK
TARGET
APPLIED
SENSOR
SHARED
EXAMPLES
EVERY
DEFINITION
CURRENT
MAY
MOVING
MANAGE
SET
USE
RELATIONAL
LARGER
CONTINUES
SOCIAL
COMPLEX
TENS
CITATION
ANALYTICS
PRACTITIONERS
CAPTURE
WITHIN
ARCHIVES
TECHNOLOGIES
USING
INTERNET
DISTRIBUTED
BUSINESS
SETS
GENOMICS
NEEDED
TIME
ONE
LARGE
MANAGEMENT
TERABYTES
CASE
ABILITY
SOFTWARE
INFORMATION
PROCESS
BIOLOGICAL
PROCESSING
CAPACITY
HUNDREDS
NETWORKS
DATABASES
RECORDS
SEARCH
GARTNER
WORKING
ELAPSED
PETABYTES
INCLUDE
PERFORMANCE
STORE
SYSTEMS
TOLERABLE
SIZE
RESEARCH
STORAGE
MPP
LOGS
COST
PARALLEL
SAN
MASSIVELY
GROW
DIFFICULTY