分类：集成学习-丁兆云

分类：集成学习-丁兆云

例子

VS.

例子

动机

- 在机器学习中，直接建立一个高性能的分类器是很困难的。
- 但是，如果能找到一系列性能较差的分类器，并把它们集成起来的话，也许就能得到更好的分类器。

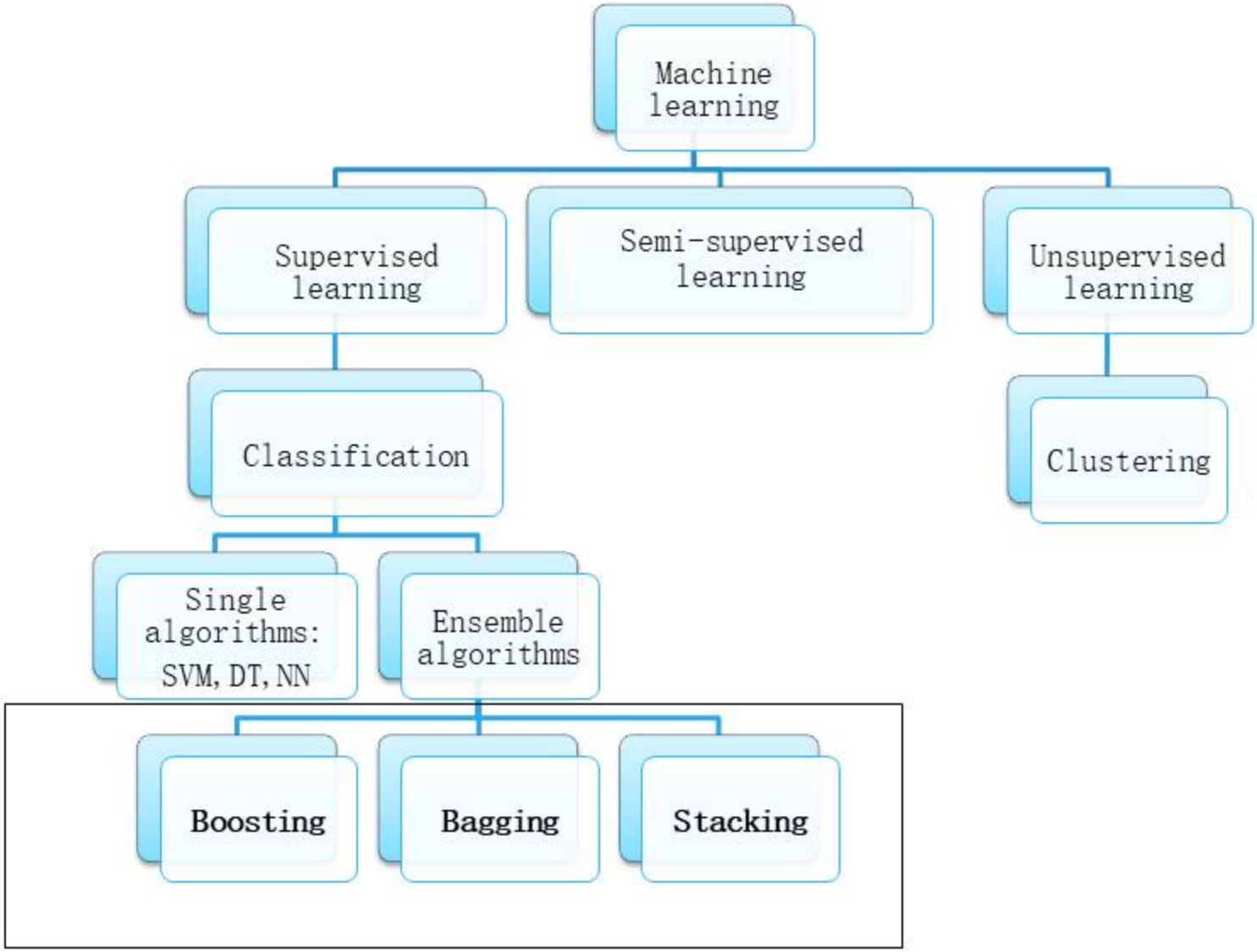
主要内容

- 1集成学习简介
- 2集成学习算法
  - Bagging
  - Boosting
  - Stacking
- 3集成学习应用

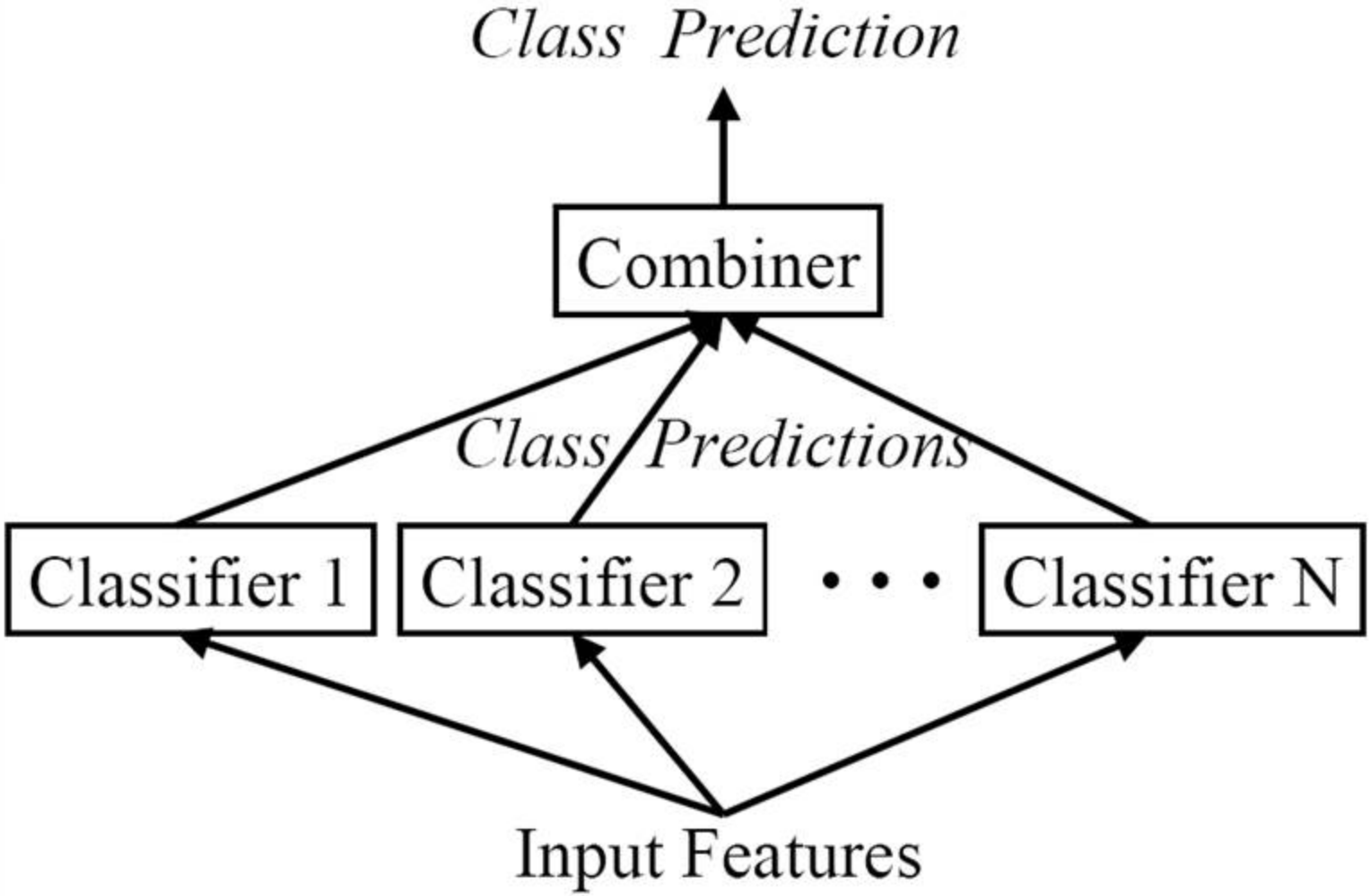
1. 集成学习简介

- 定义
  - The process by which multiple models are strategically generated and combined in order to better solve a particular Machine Learning problem.
- 动机
  - 提高单分类器的性能

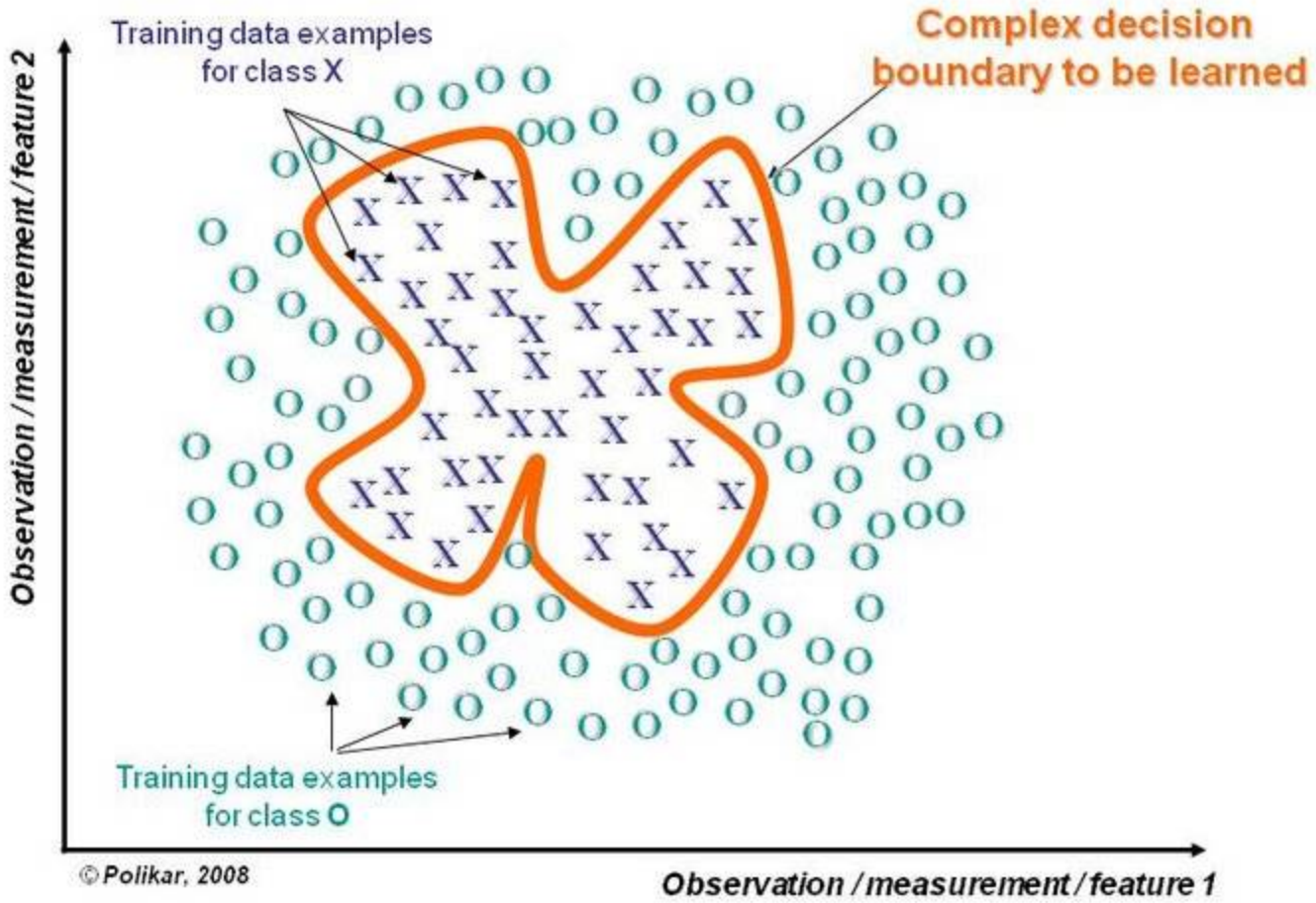
1. 集成学习简介


Machine learning
Supervised learning
Semi-supervised learning
Unsupervised learning
Classification
Clustering
Single algorithms: SVM, DT, NN
Ensemble algorithms
Boosting
Bagging
Stacking

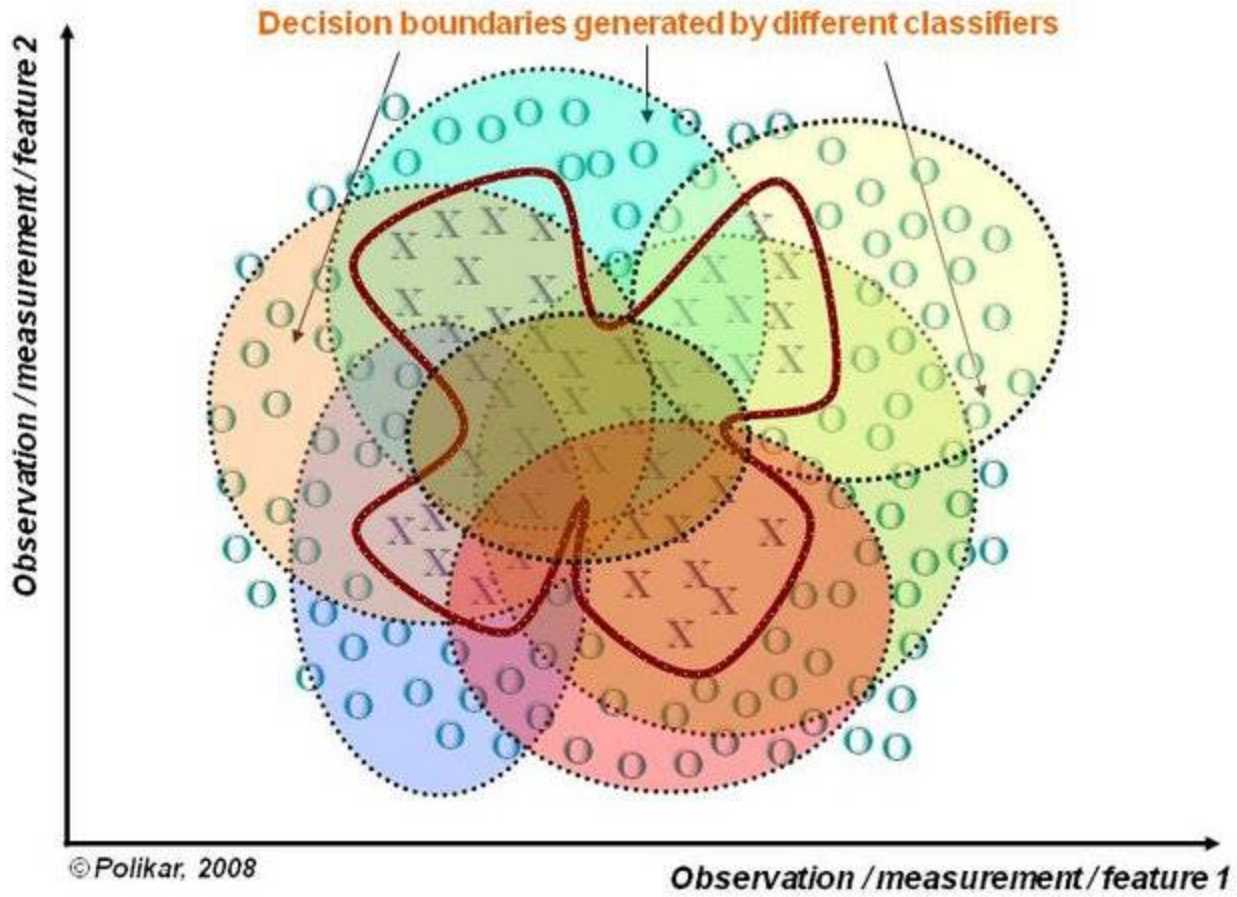
1Combination of Classifiers



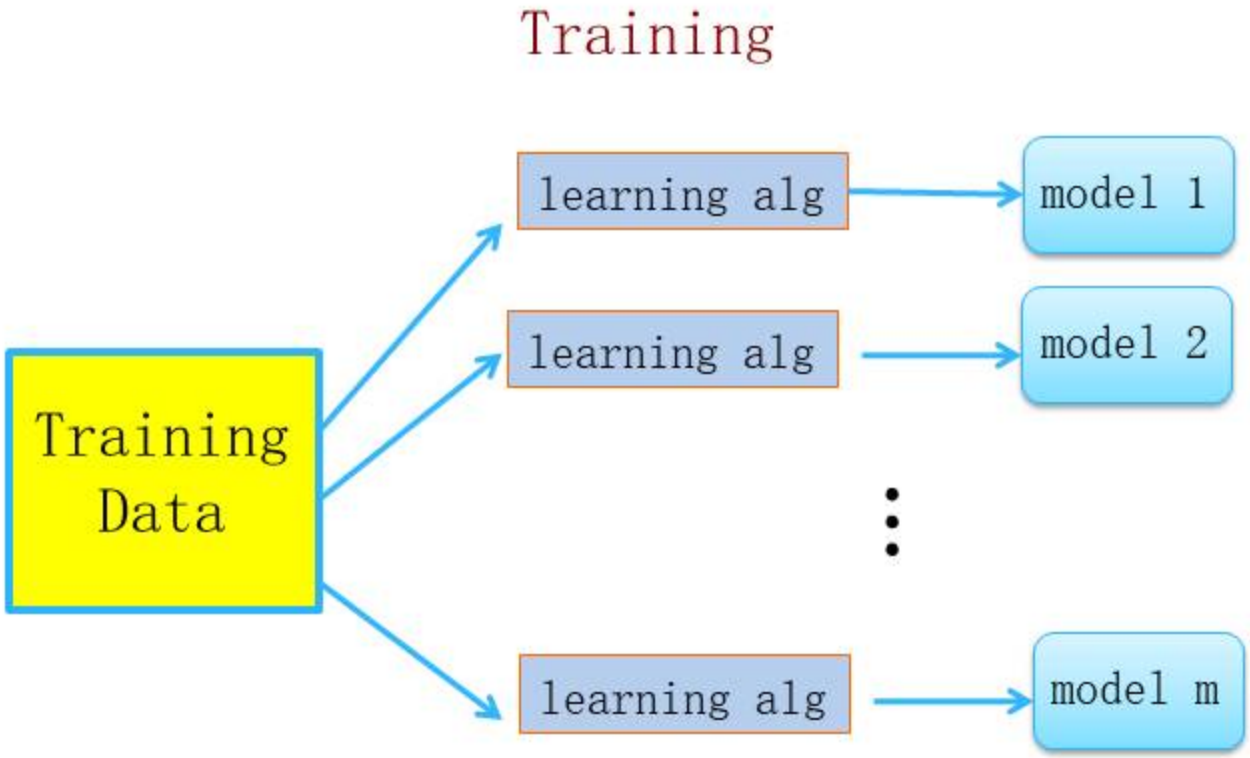
1分而治之


Training data examples
for class X
Complex decision
boundary to be learned
Training data examples
for class O
Observation / measurement / feature 2
Observation / measurement / feature 1
© Polikar, 2008

1分而治之

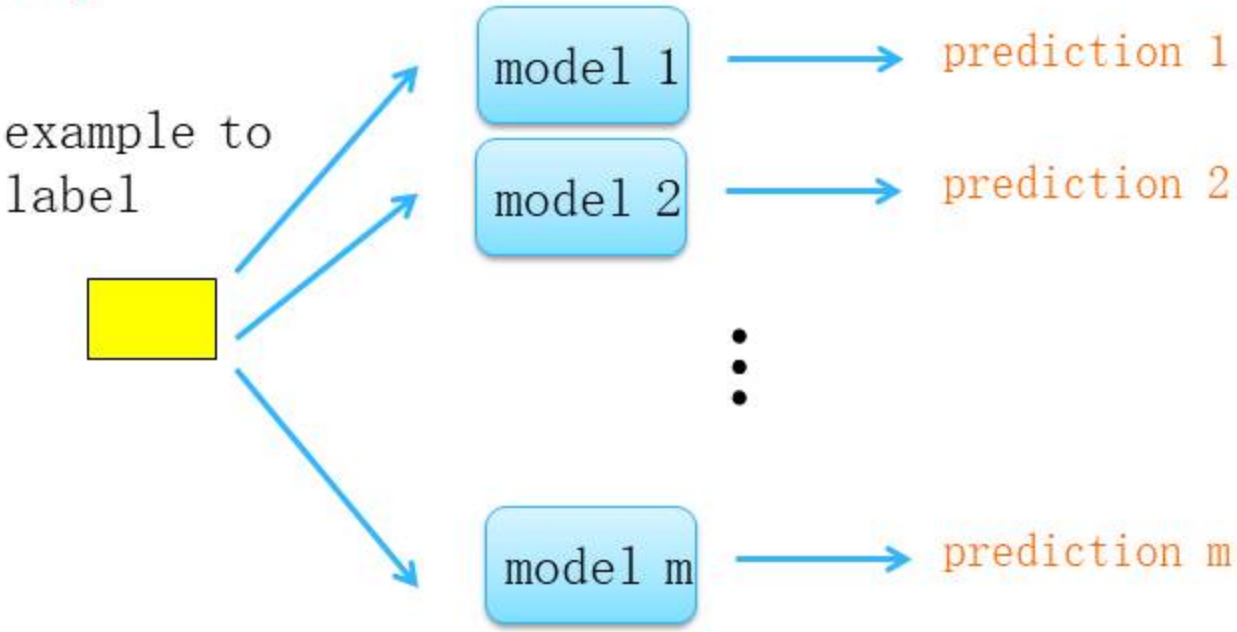

Decision boundaries generated by different classifiers
Observation / measurement / feature 2
Observation / measurement / feature 1
© Polikar, 2008

1.1训练阶段


Training
Training Data
learning alg
learning alg
learning alg
model 1
model 2
model m

1.2测试阶段

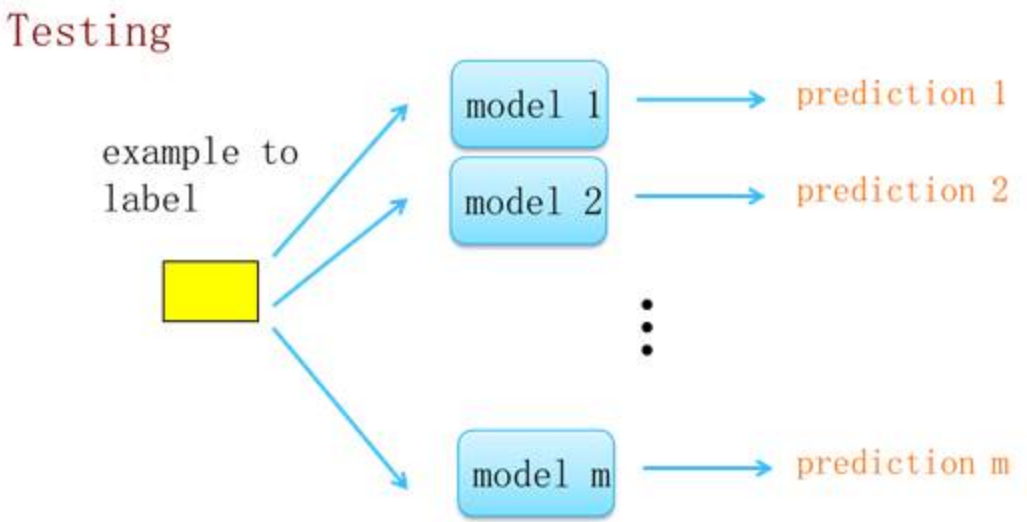
Testing



讨论：如何组合多个分类器呢？

1.3如何组合多个分类器?

- 平均
- 投票
  - 多数投票：Bagging
  - 带权重的多数投票：Adaboost
- 学习组合器
  - Stacking
  - RegionBoost



填空题 1分

设置

假如每个分类器的错误率为40%

model 1	model 2	model 3	prob
C	C	C	.6*.6*.6=0.216
C	C	I	.6*.6*.4=0.144
C	I	C	.6*.4*.6=0.144
C	I	I	.6*.4*.4=0.096
I	C	C	.4*.6*.6=0.144
I	C	I	.4*.6*.4=0.096
I	I	C	.4*.4*.6=0.096
I	I	I	.4*.4*.4=0.064

那么3个基分类器的错误率= [填空1]

正常使用填空题需3.0以上版本雨课堂

作答

1.4集成学习优势示例

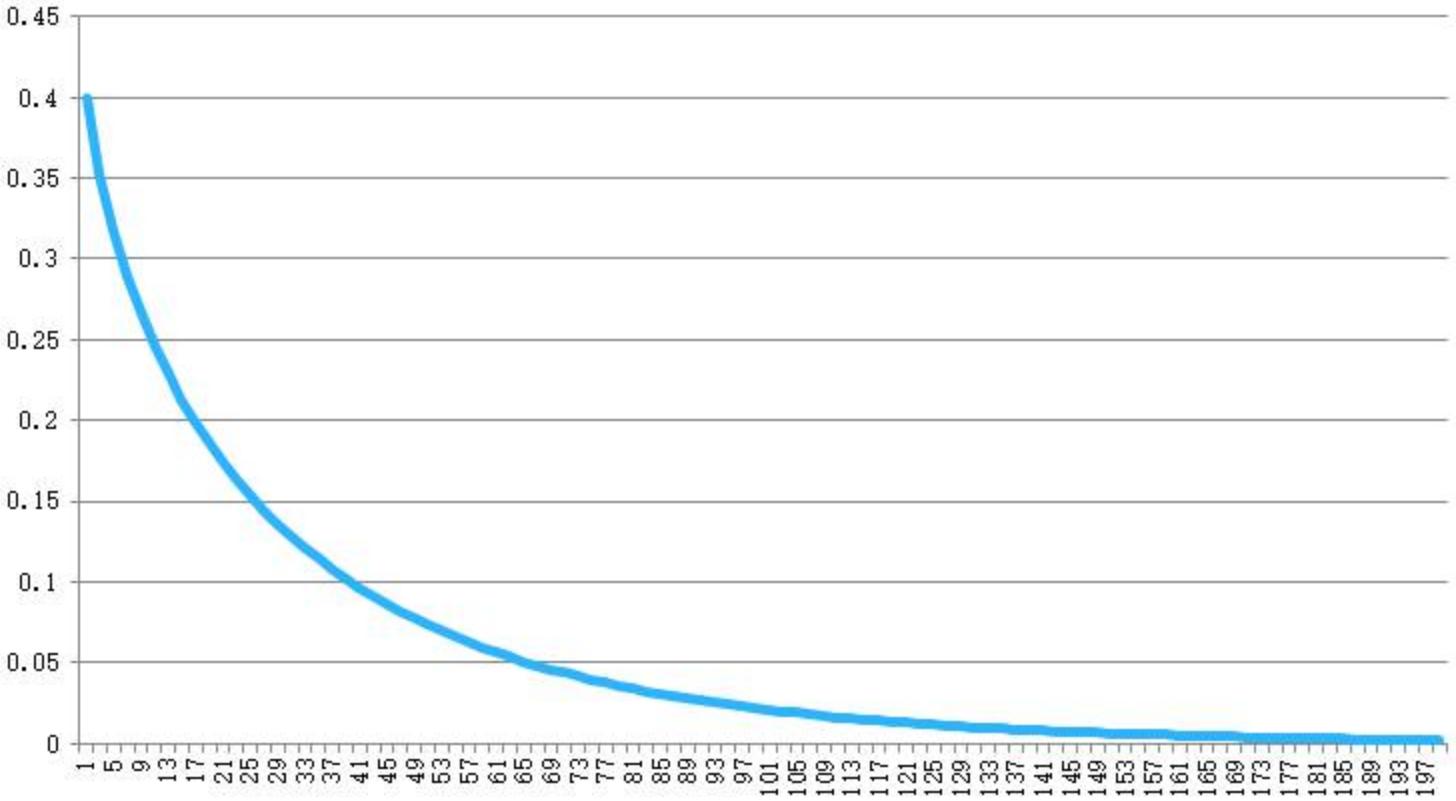
假如每个分类器的错误率为40%

model 1	model 2	model 3	prob
C	C	C	.6*.6*.6=0.216
C	C	I	.6*.6*.4=0.144
C	I	C	.6*.4*.6=0.144
C	I	I	.6*.4*.4=0.096
I	C	C	.4*.6*.6=0.144
I	C	I	.4*.6*.4=0.096
I	I	C	.4*.4*.6=0.096
I	I	I	.4*.4*.4=0.064

0.096+
0.096+
0.096+
0.064 =
35% error!

1.4集成学习优势

$$p(error) = \sum_{i=(m+1)/2}^{m} \binom{m}{i} r^i (1-r)^{m-i}$$

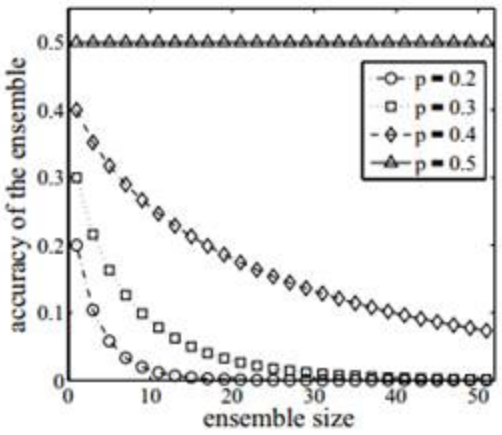


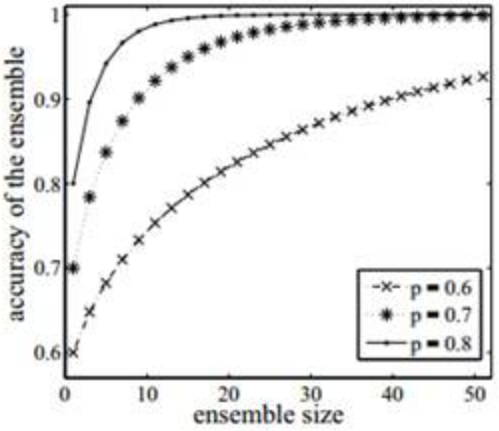
r = 0.4

1.5集成学习中单分类器的条件

- 通过集成学习提高分类器的整体泛化能力是有条件的:
  - 分类器之间应该具有差异性
  - 分类器的精度，每个个体分类器的分类精度必须大于0.5

$$p(error) = \sum_{i=(m+1)/2}^{m} \binom{m}{i} r^i (1-r)^{m-i}$$



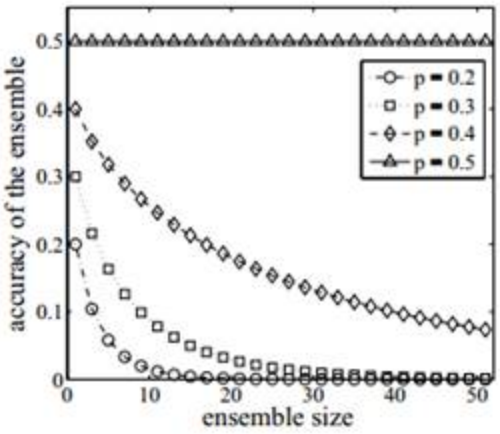


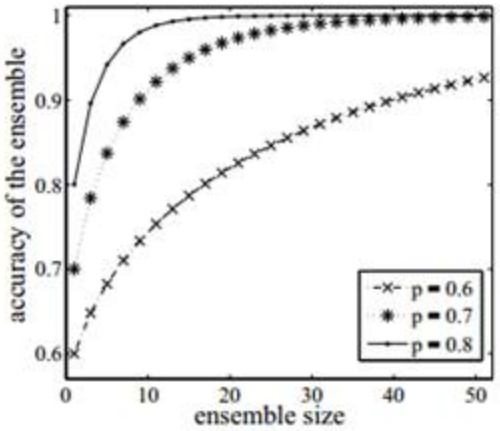
设置

集成学习对基分类器的要求

A 基分类器具有差异性

B 单个基分类器精度均高于50%





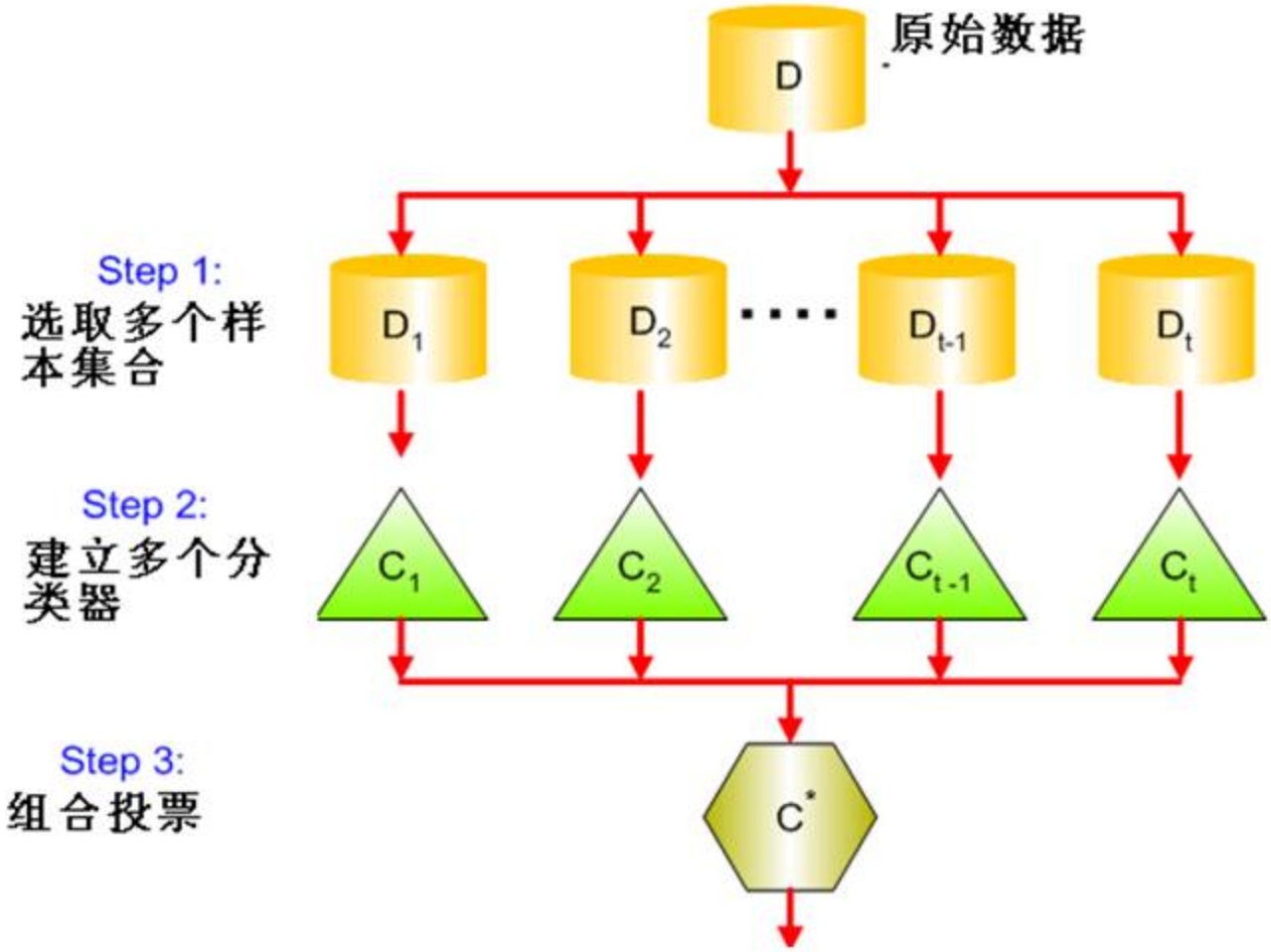
提交

主要内容

- 1集成学习简介
- 2集成学习算法
  - Bagging
  - Boosting
  - Stacking
- 3集成学习应用

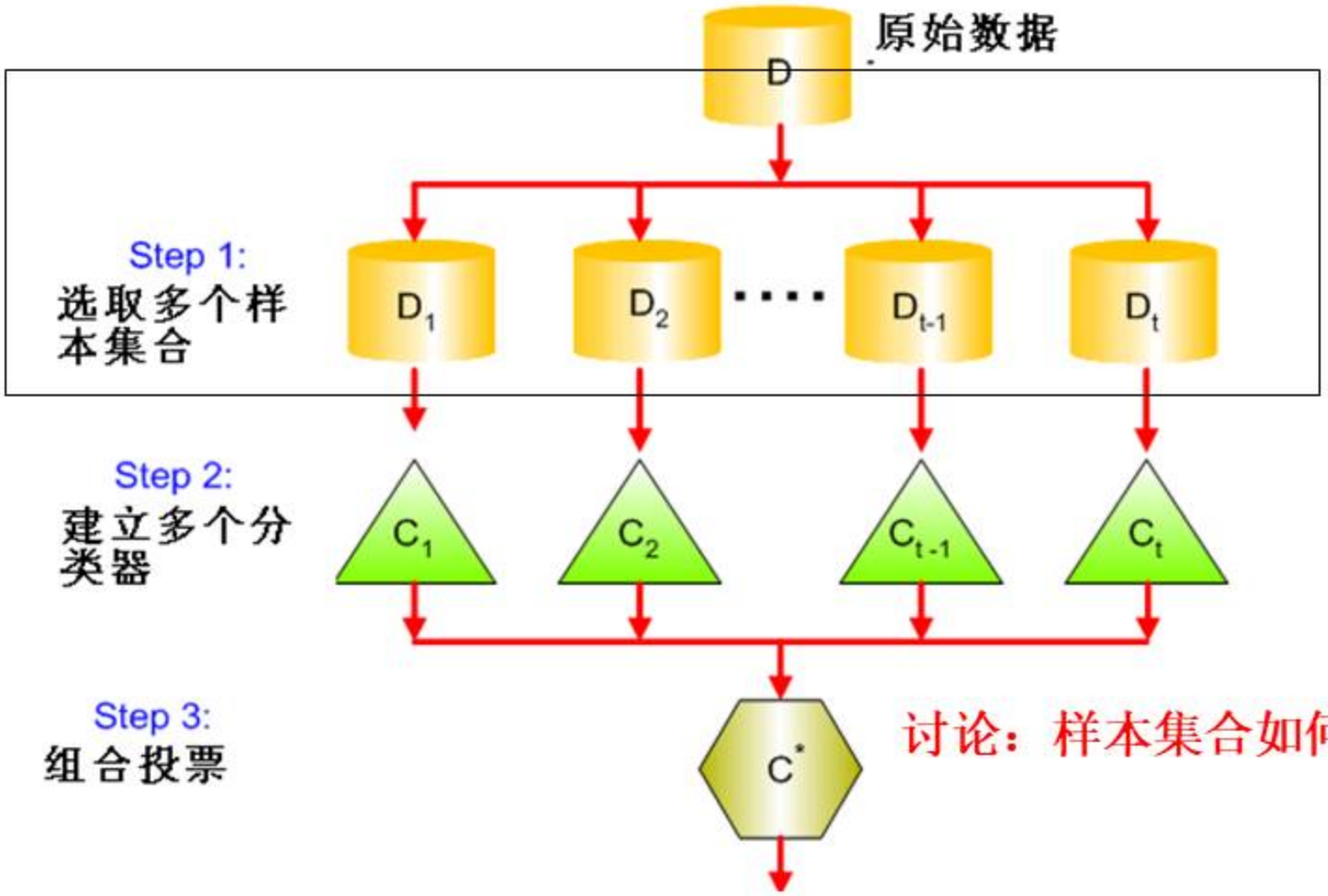
2.1.1Bagging：有放回的重采样

- 三个臭皮匠，一个诸葛亮



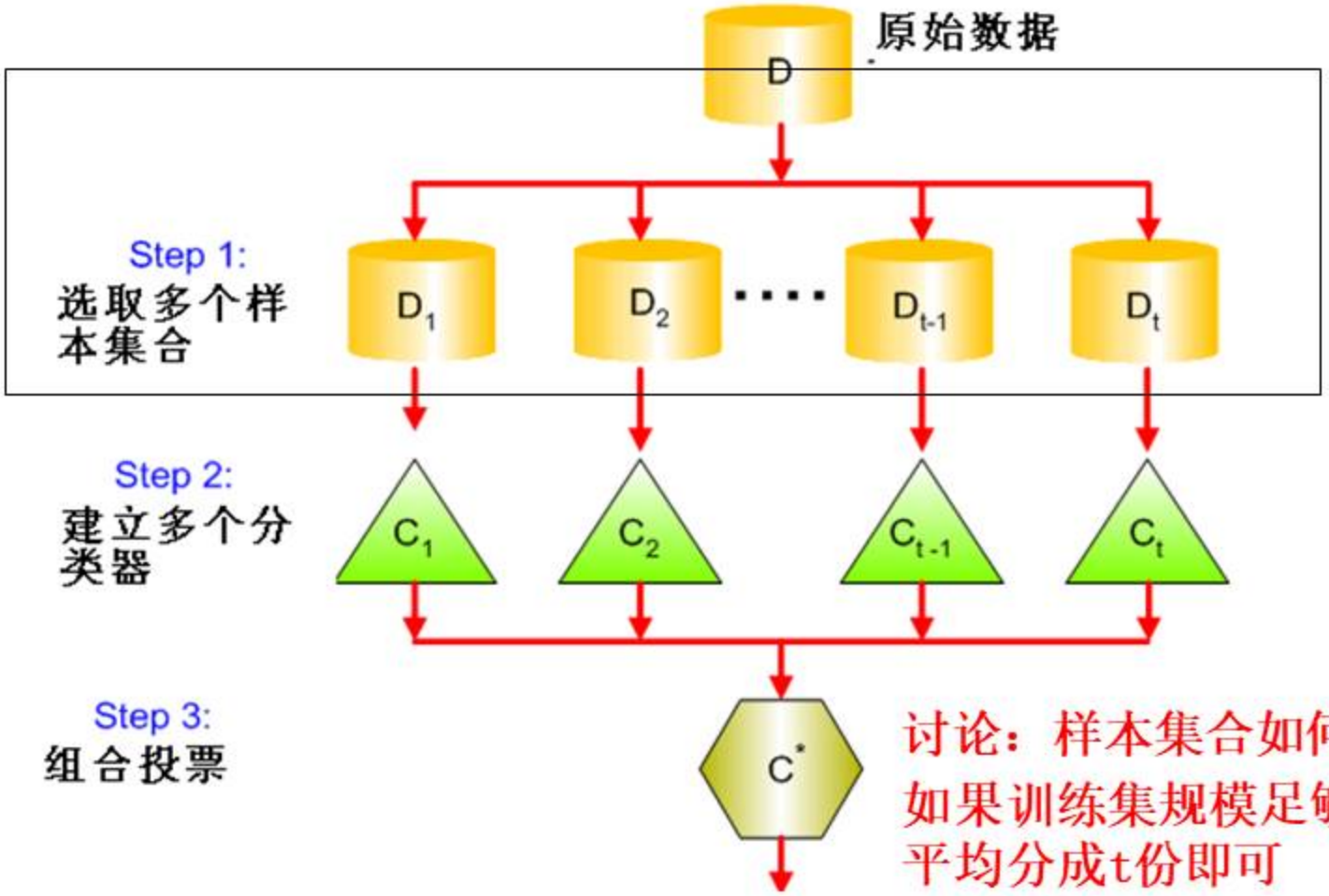
2.1.1Bagging：有放回的重采样

- 三个臭皮匠，一个诸葛亮



2.1.1Bagging：有放回的重采样

- 三个臭皮匠，一个诸葛亮

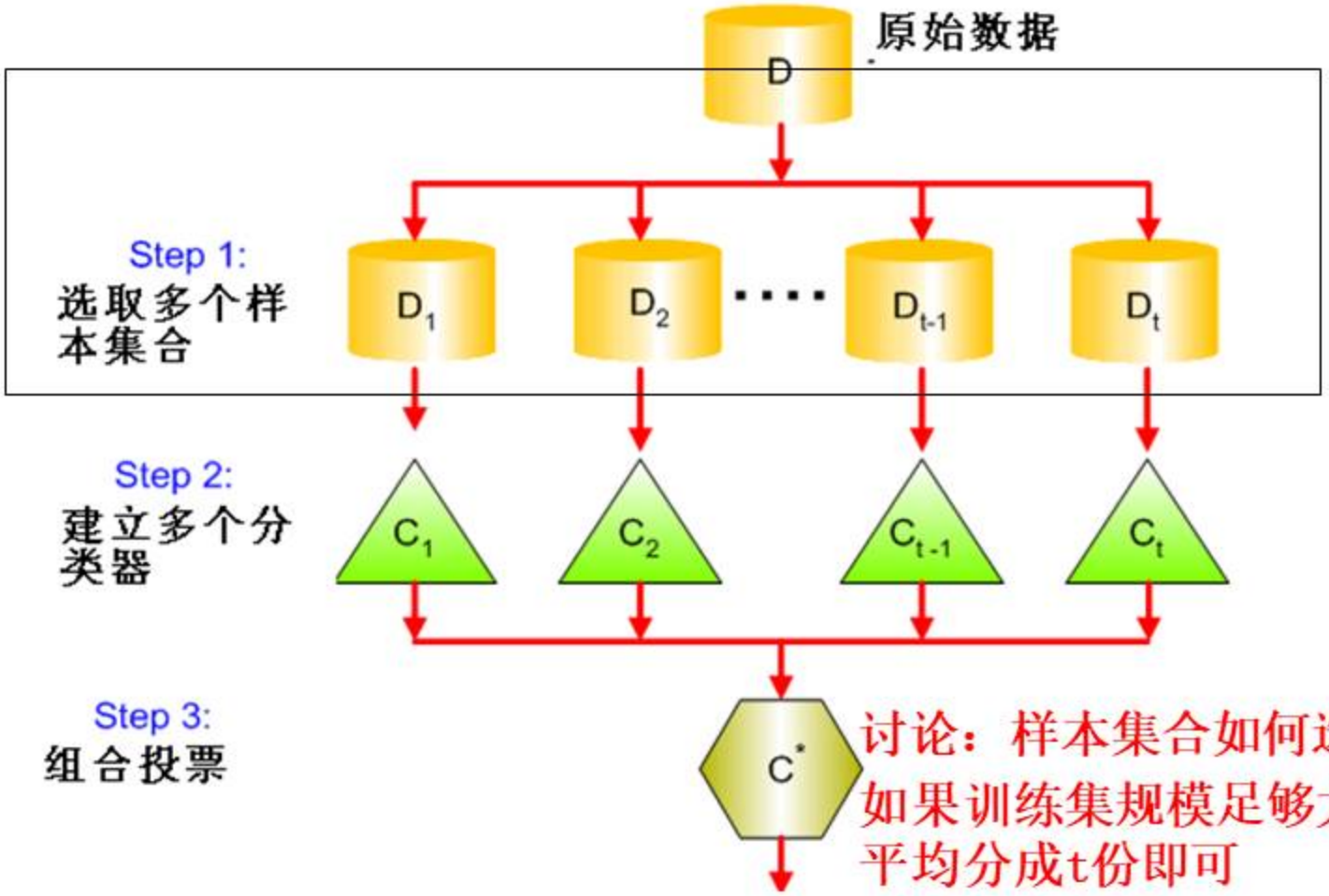


讨论：样本集合如何选取？

如果训练集规模足够大，平均分成t份即可

2.1.1Bagging：有放回的重采样

- 三个臭皮匠，一个诸葛亮

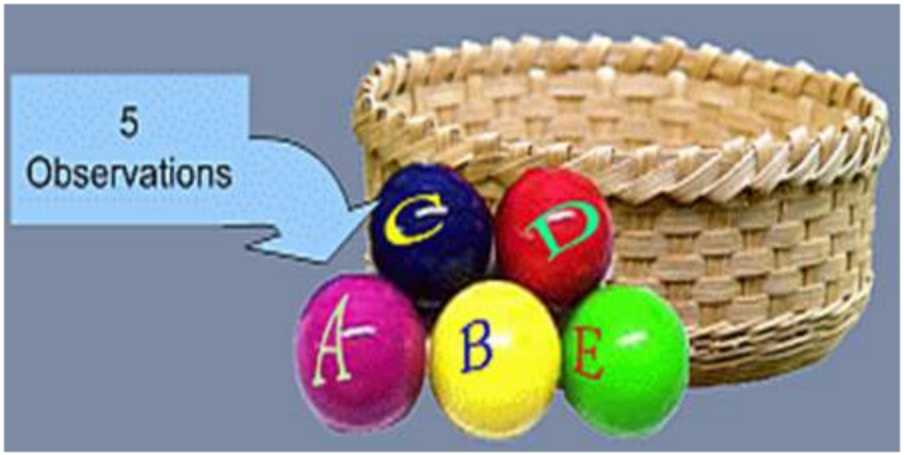


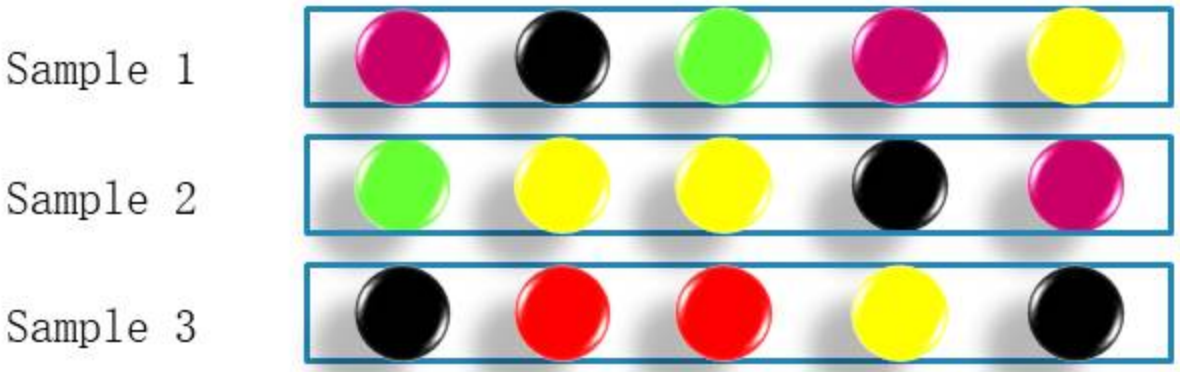
讨论：样本集合如何选取？

如果训练集规模足够大，平均分成t份即可

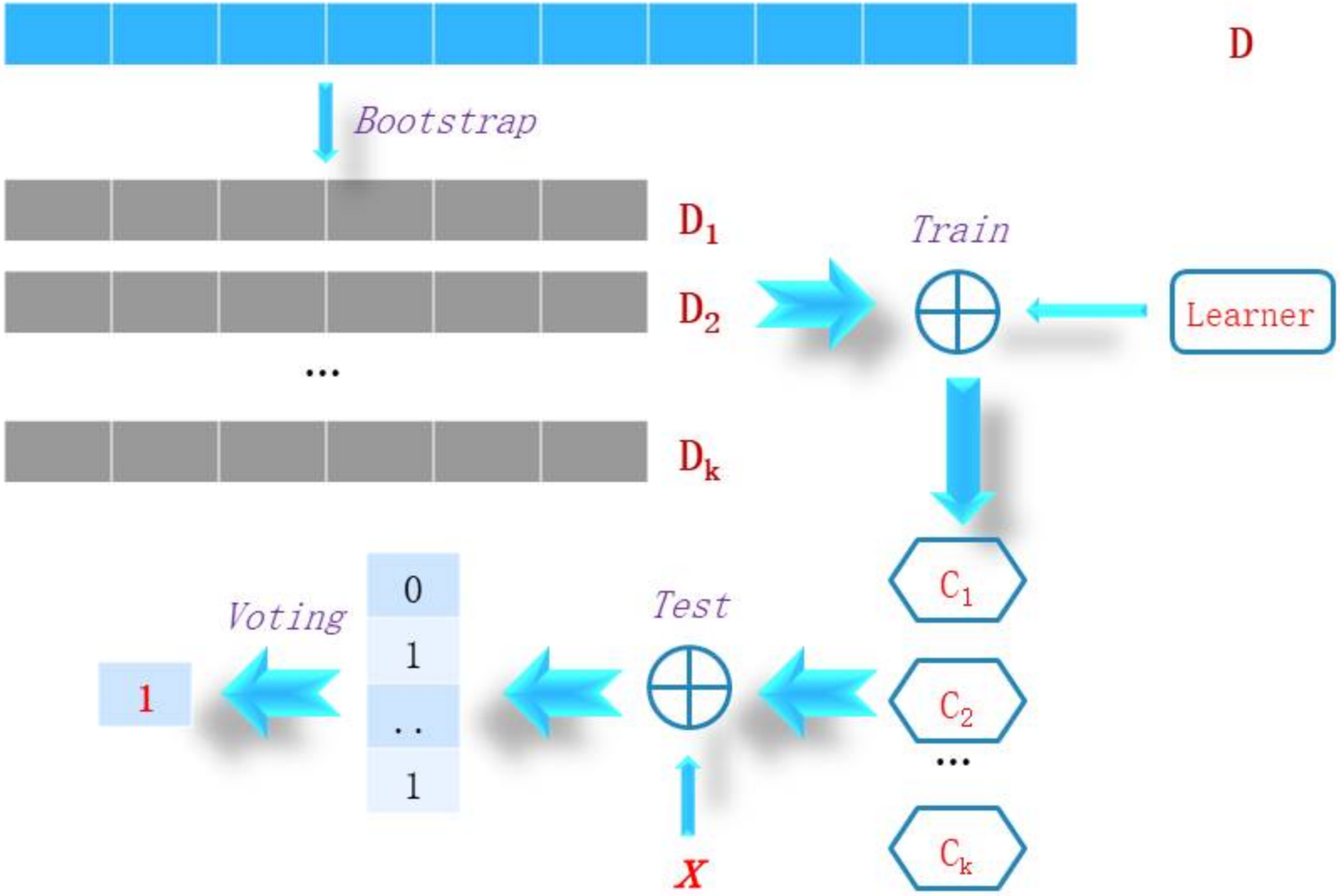
如果训练集规模不大，怎么办？

2.1.1Bagging：有放回的重采样



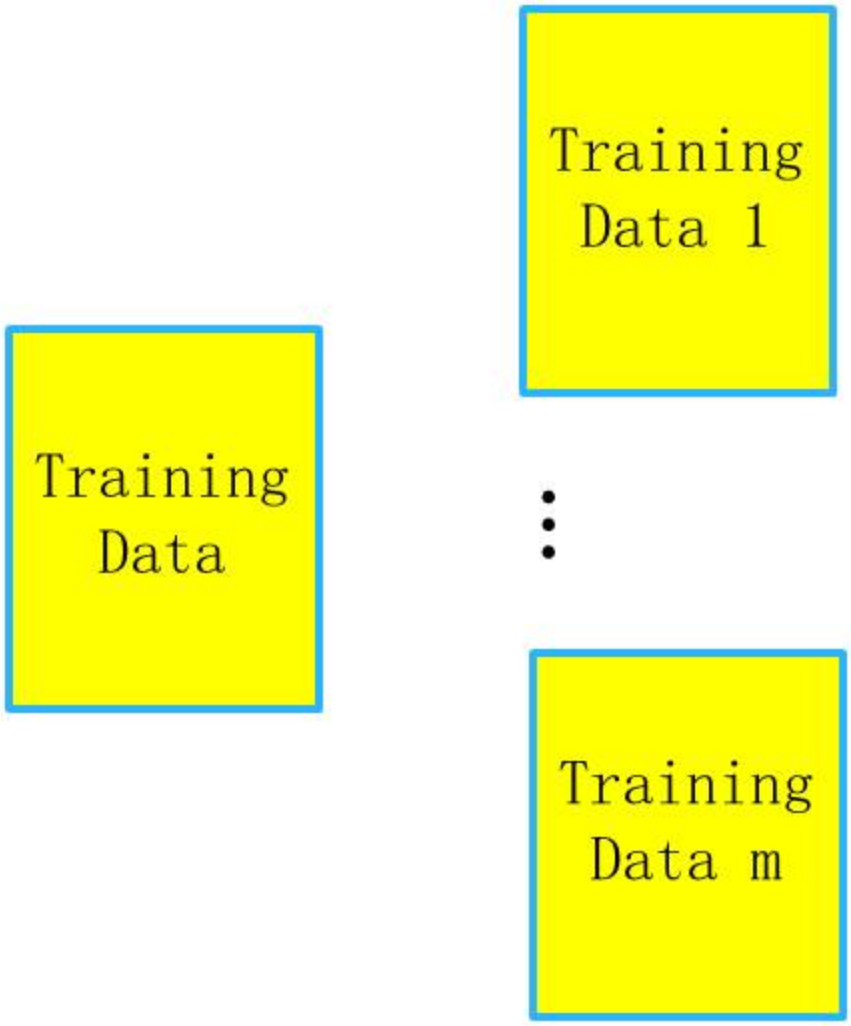


2.1.2Bagging-Bootstrap采样法



2.1.2Bagging-Bootstrap采样法优点

原始数据中，仅仅约63%的样本被采样到



2.1.2Bagging-Bootstrap采样法优点

原始数据中，仅仅约63%的样本被采样到

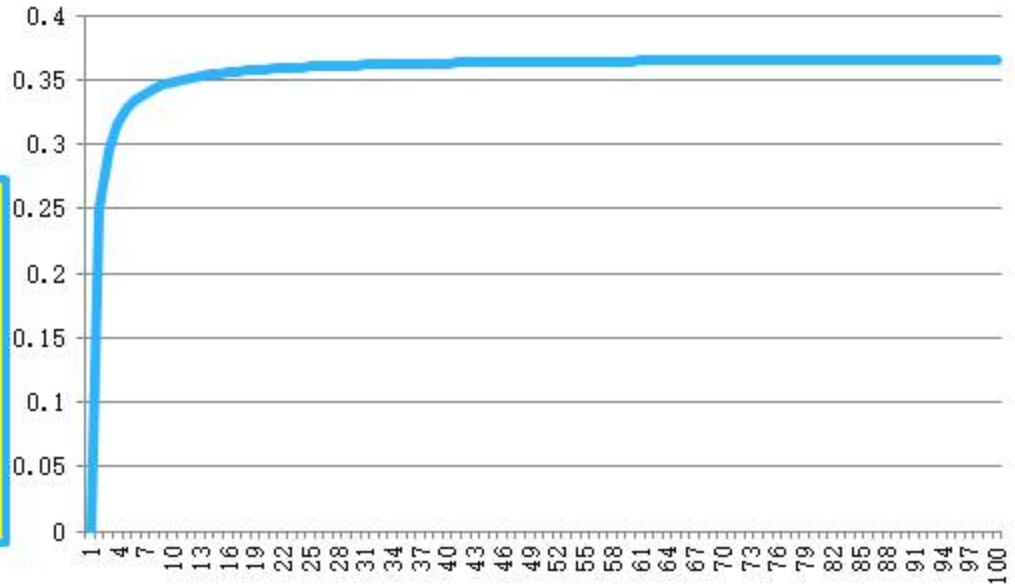
Training Data

Training Data 1

⋮

Training Data m

$$P_{not\_choose} = \left(1 - \frac{1}{n}\right)^{n} = \frac{1}{e}$$



2.1.2Bagging-Bootstrap采样法优点

原始数据中，仅仅约63%的样本被采样到

也就是说样本训练集中，约37%的错误训练集将被忽略掉，提高训练集的质量

2.1.2Bagging-Bootstrap采样法优点

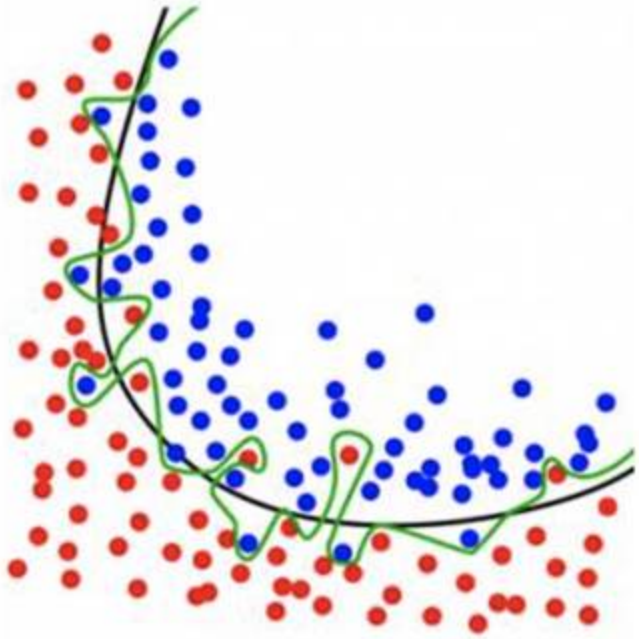
Training Data

Training Data 1

Training Data m

原始数据中，仅仅约63%的样本被采样到

也就是说样本训练集中，约37%的错误训练集将被忽略掉，提高训练集的质量



2.1.2Bagging-Bootstrap采样法优点

Training Data

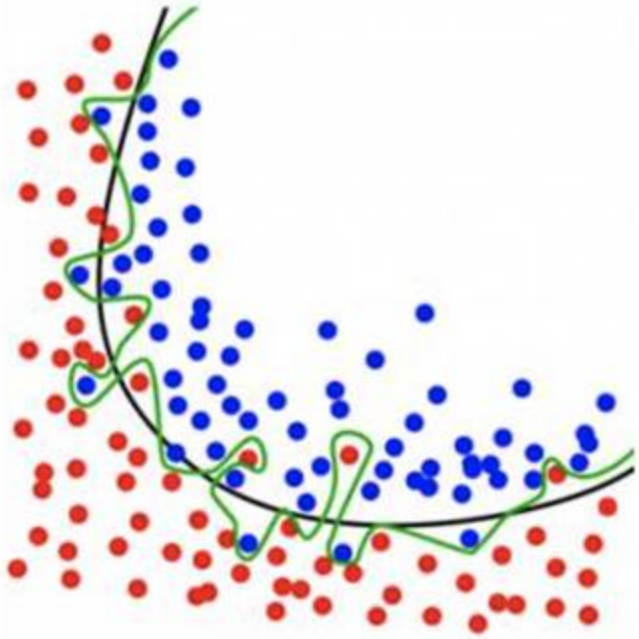
Training Data 1

Training Data m

讨论：正常数据也有37%的被忽略掉，那对训练模型的质量是不是有影响呢？

原始数据中，仅仅约63%的样本被采样到

也就是说样本训练集中，约37%的错误训练集将被忽略掉，提高训练集的质量



单选题 1分

设置

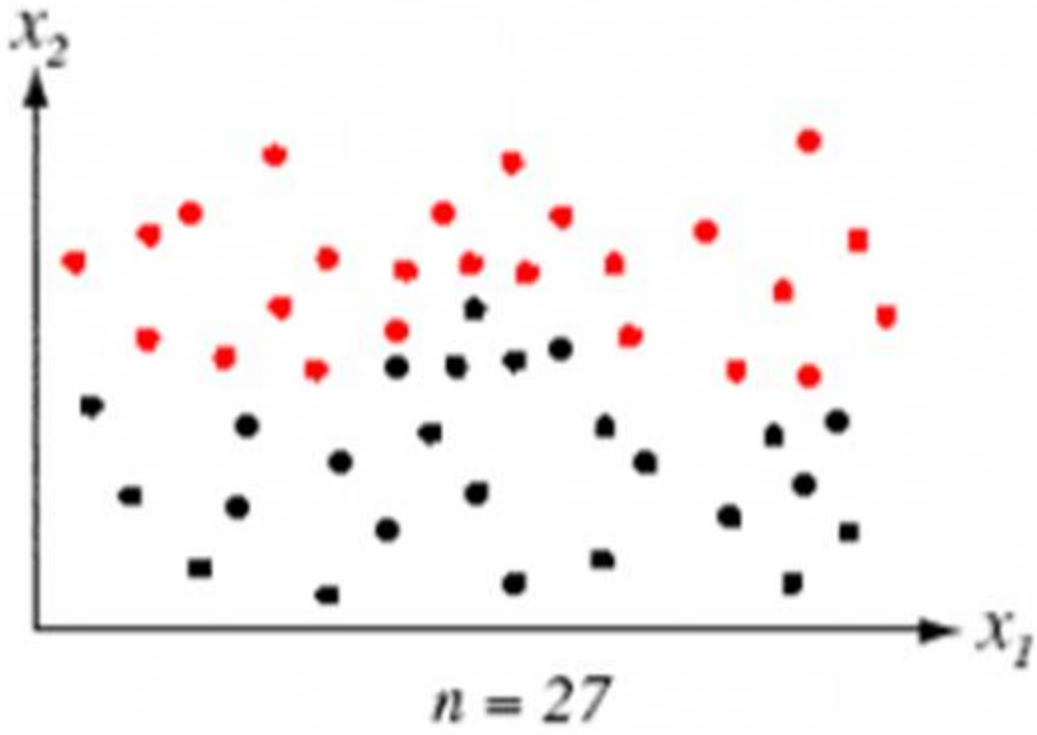
Bagging算法优点

A 降低噪音数据的影响

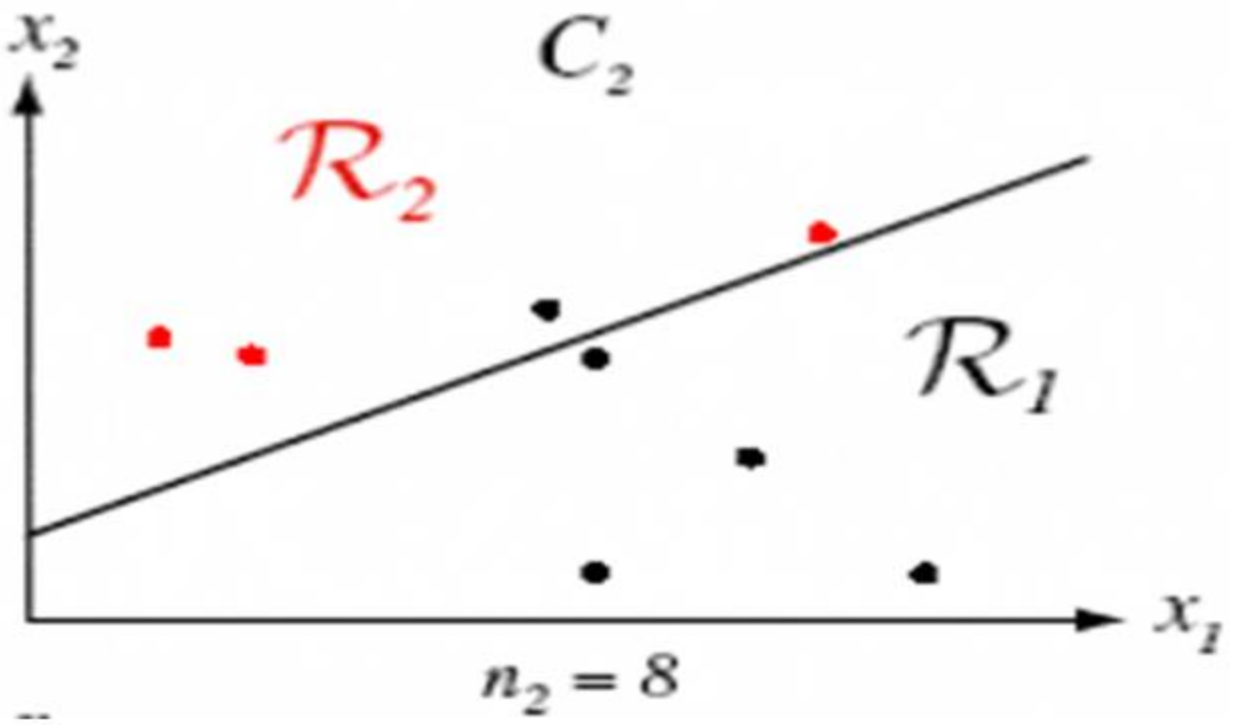
B 易过拟合

提交

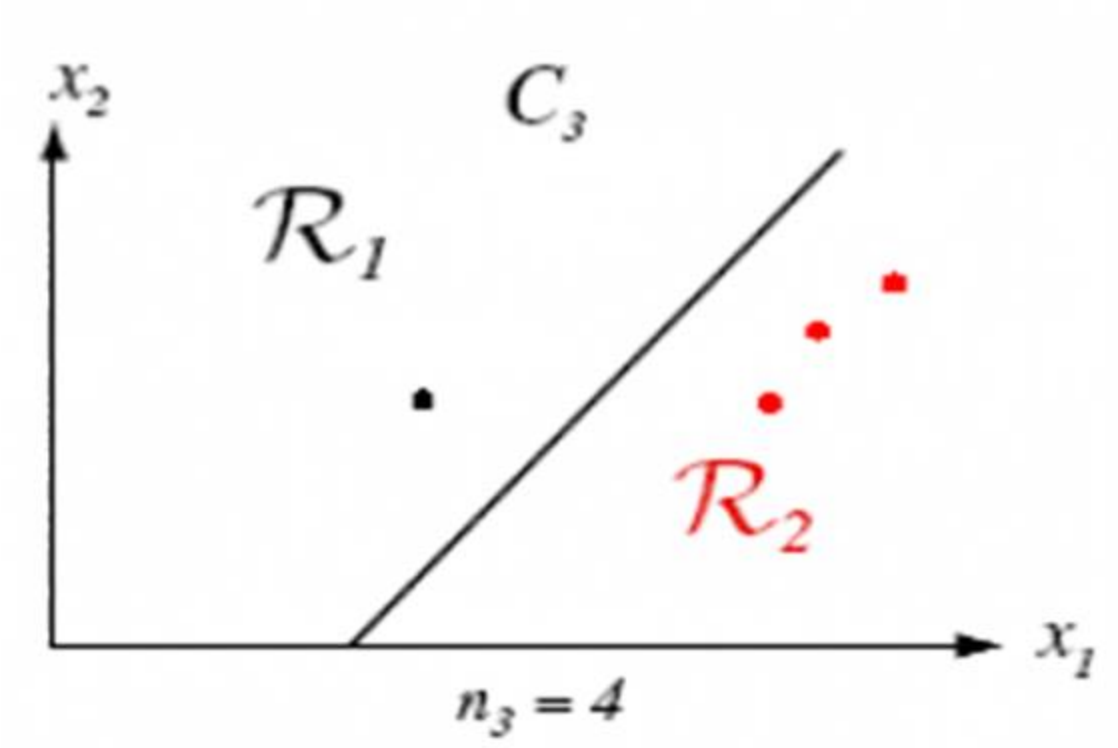
2.1.3Bagging示例



2.1.3Bagging示例

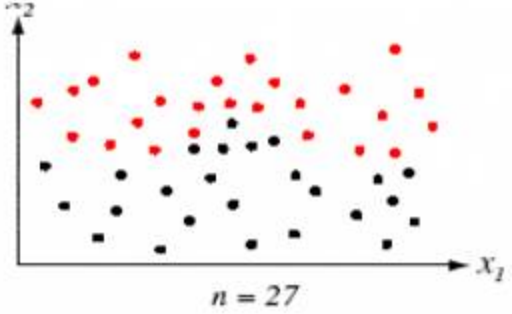


2.1.3Bagging示例

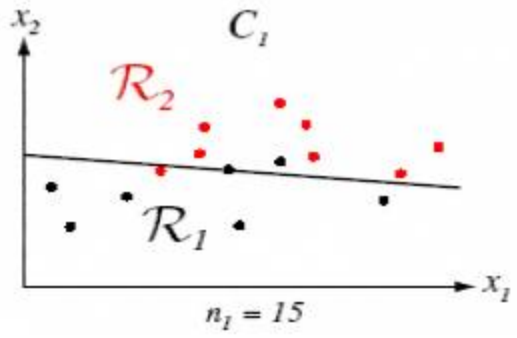


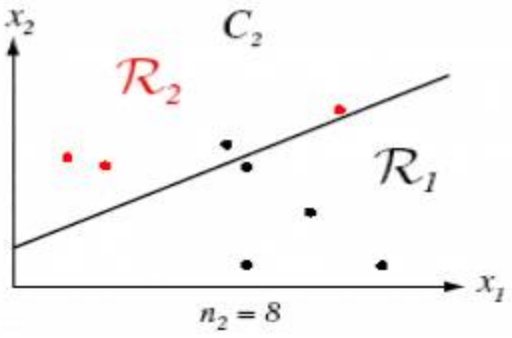
2.1.3Bagging示例

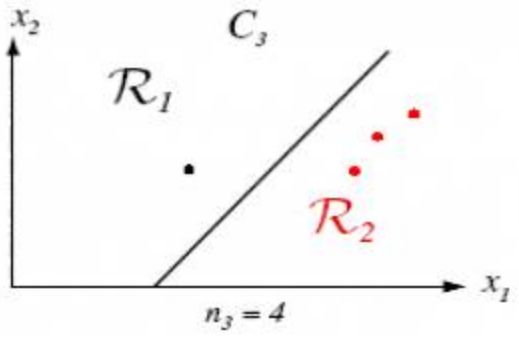
原始样本集



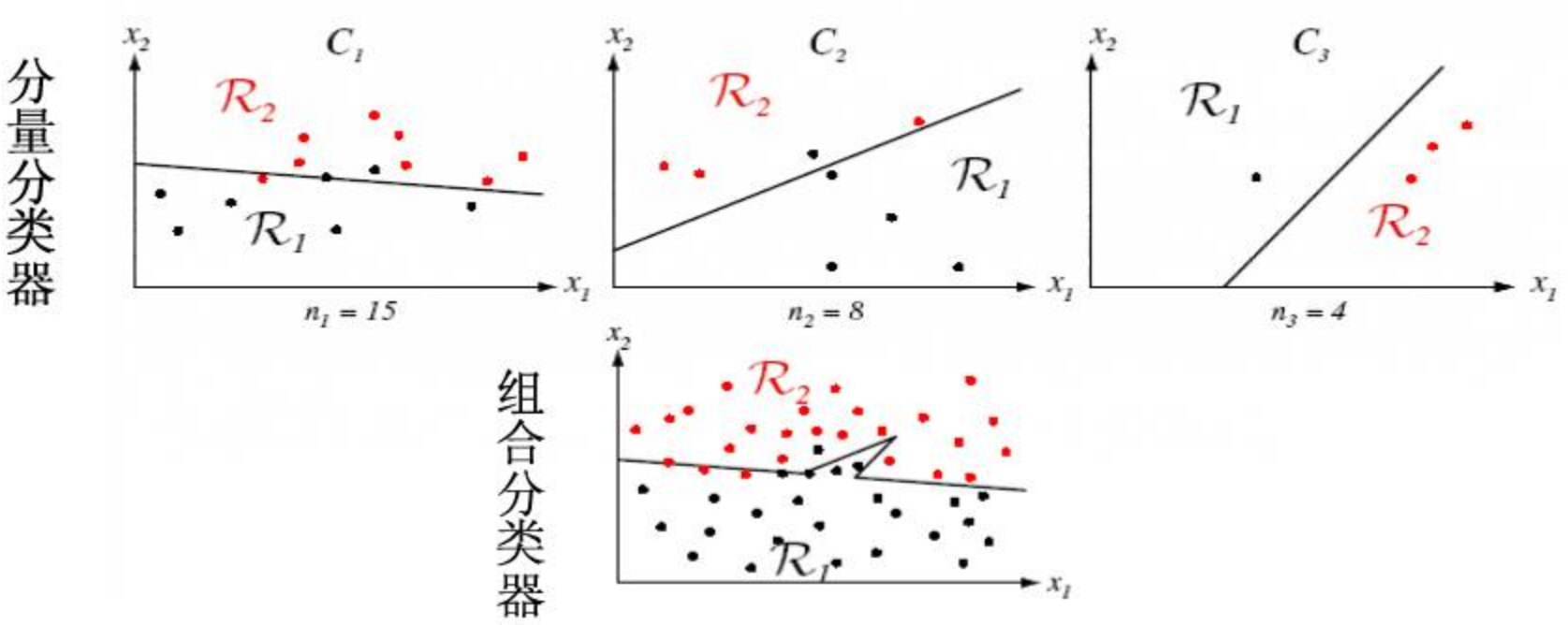
分量分类器







2.1.3Bagging示例

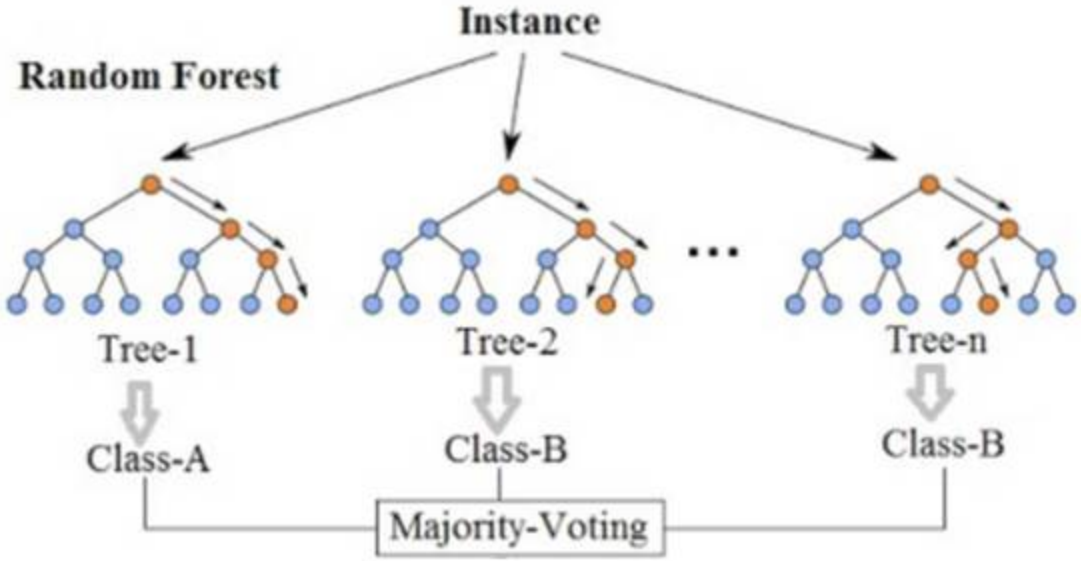


2.1.4Bagging：随机森林

- 基本思想
  - 随机森林就是通过集成学习的思想将多棵树集成的一种算法，它的基本单元是决策树
  - 随机森林的名称中有两个关键词，一个是“随机”，一个就是“森林”。“森林”很好理解，一棵叫做树，那么成百上千棵就可以叫做森林了，其实这也是随机森林的主要思想--集成思想的体现。“随机”的包括随机选取训练样本集和随机选取分裂属性集。

2.1.4Bagging：随机森林

- 基本思想
  - 随机森林就是通过集成学习的思想将多棵树集成的一种算法，它的基本单元是决策树
  - 随机森林的名称中有两个关键词，一个是“随机”，一个就是“森林”。“森林”很好理解，一棵叫做树，那么成百上千棵就可以叫做森林了，其实这也是随机森林的主要思想--集成思想的体现。“随机”的包括随机选取训练样本集和随机选取分裂属性集。
  - 行采样
  - 列采样

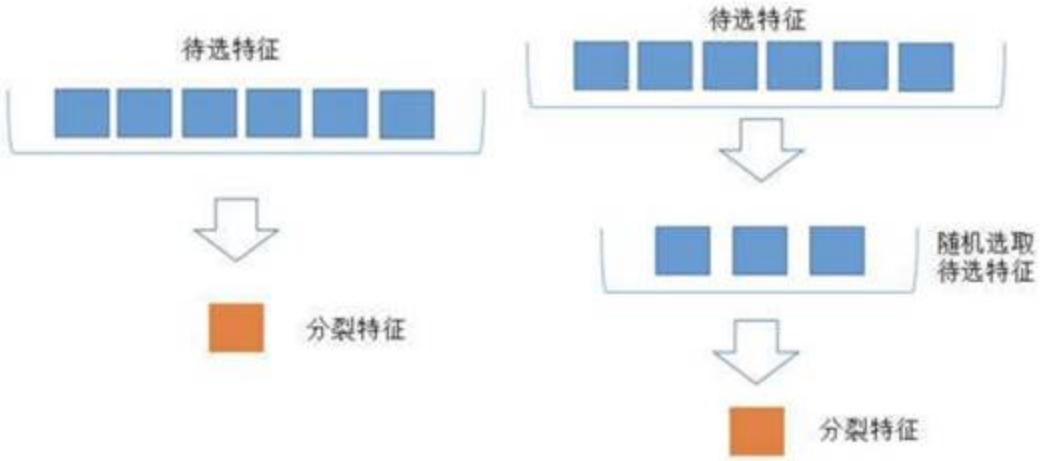


2.1.5随机森林问题

- 为什么要随机抽样训练集？【思考：基分类差异】
  - 如果不进行随机抽样，每棵树的训练集都一样，那么最终训练出的树分类结果也是完全一样的，这样的话完全没有集成的必要；
- 为什么要有放回地抽样？【思考：基分类准确率】
  - 如果不是有放回的抽样，那么每棵树的训练样本都是不同的，都是没有交集的，这样每棵树都是"有偏的"，都是"片面的"，也就是说每棵树训练出来都是有很大的差异的；
  - 而随机森林最后分类取决于多棵树（弱分类器）的投票表决，这种表决应该是"求同"，因此使用完全不同的训练集来训练每棵树这样对最终分类结果是没有帮助的。

2.1.6随机森林的生成

- 关键因素：每棵树选择特征的数量m
- 随机森林分类效果（错误率）与两个因素有关：
  - 1.森林中任意两棵树的相关性：相关性越大，错误率越大；【说明基分类器差异性越小】
  - 2.森林中每棵树的分类能力：每棵树的分类能力越强，整个森林的错误率越低。【基分类器准确率高】



多选题 1分

设置

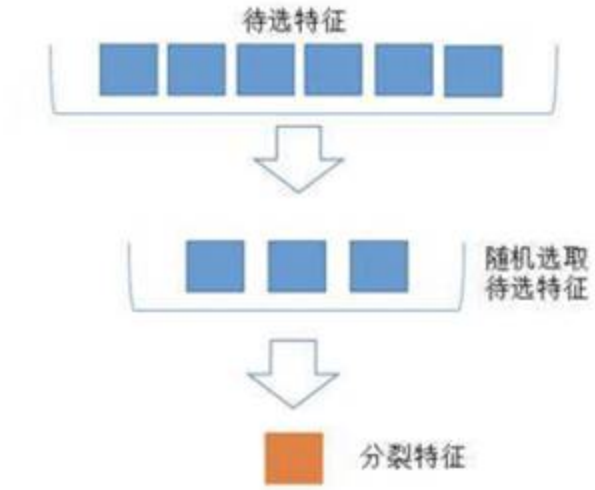
每个子树中减少特征选择个数m

A 树的相关性降低

B 分类能力降低

C 树的相关性升高

D 树的相关性升高



提交

多选题 1分

设置

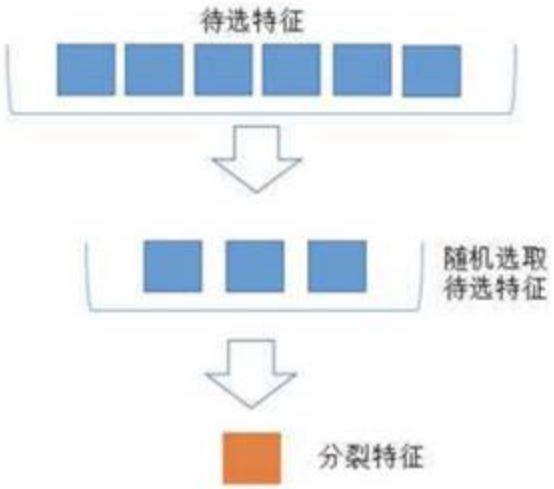
每个子树中增加特征选择个数m

A 树的相关性降低

B 分类能力降低

C 树的相关性升高

D 树的相关性升高



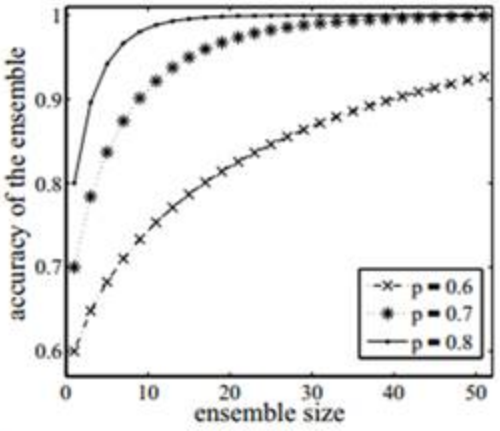
提交

2.1.6随机森林的生成

- 关键因素：每棵树选择特征的数量m
- 随机森林分类效果（错误率）与两个因素有关：
  - 1.森林中任意两棵树的相关性：相关性越大，错误率越大；【说明基分类器差异性越小】
  - 2.森林中每棵树的分类能力：每棵树的分类能力越强，整个森林的错误率越低。【基分类器准确率高】
- 减小特征选择个数m，树的相关性和分类能力也会相应的降低；增大m，两者也会随之增大。所以关键问题是如何选择最优的m

2.1.6随机森林参数与评价

- 特征数量m的选择
  - Square Root (K)
  - K: total number of available variables
  - 网格搜索法调参GridSearch
  - 参考: https://blog.csdn.net/qq_35040963/article/details/88832030
- 决策树的数量
  - 500 or more
- 自测方法
  - Around one third of the original data are left out.
  - Similar to Cross-Validation



2.1.7Bagging：随机森林

优点：

1.两个随机性的引入，使得随机森林不容易陷入过拟合；

2.两个随机性的引入，使得随机森林具有很好的抗噪声能力；

3.对数据集的适应能力强：既能处理离散型数据，也能处理连续型数据，数据集无需规范化且能够有效地运行在大数据集上；

4.能够处理具有高维特征的输入样本，而且不需要降维；

5.对于缺省值问题也能够获得很好得结果。

多选题 1分

设置

随机森林优点

A 不容易陷入过拟合

B 容易陷入过拟合

C 具有很好的抗噪声能力

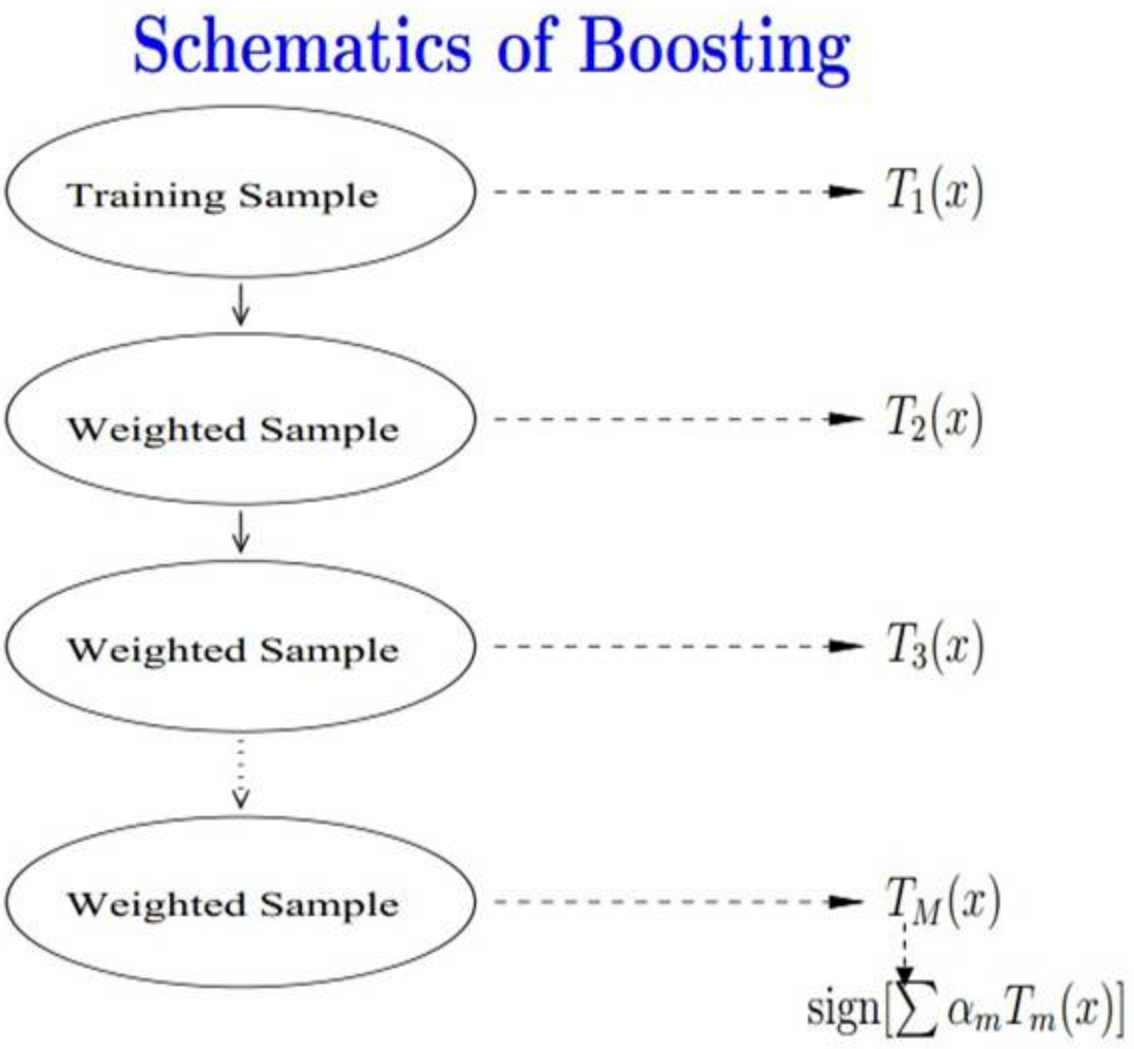
D 不需要降维

提交

主要内容

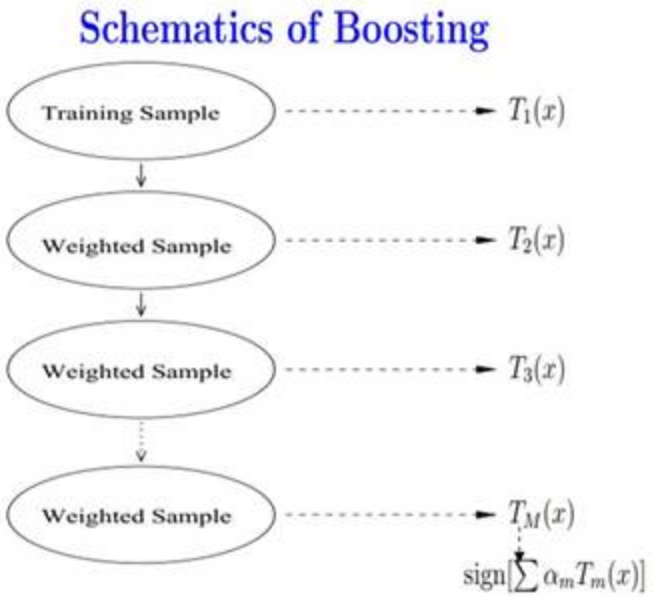
- 集成学习简介
- 集成学习算法
  - Bagging
  - Boosting
  - Stacking
- 集成学习应用

2.2.1Boosting提出



2.2.1Boosting提出

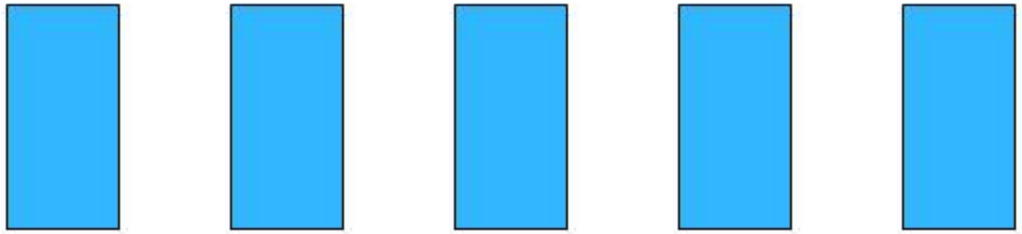
- 提升方法是一个迭代的过程
  - 通过改变样本分布，使得分类器聚集在那些很难分的样本上，对那些容易错分的数据加强学习，增加错分数据的权重
  - 这样错分的数据再下一轮的迭代就有更大的作用（对错分数据进行惩罚）



2.2.2Boosting提出-示例1

Start with equal weighted examples

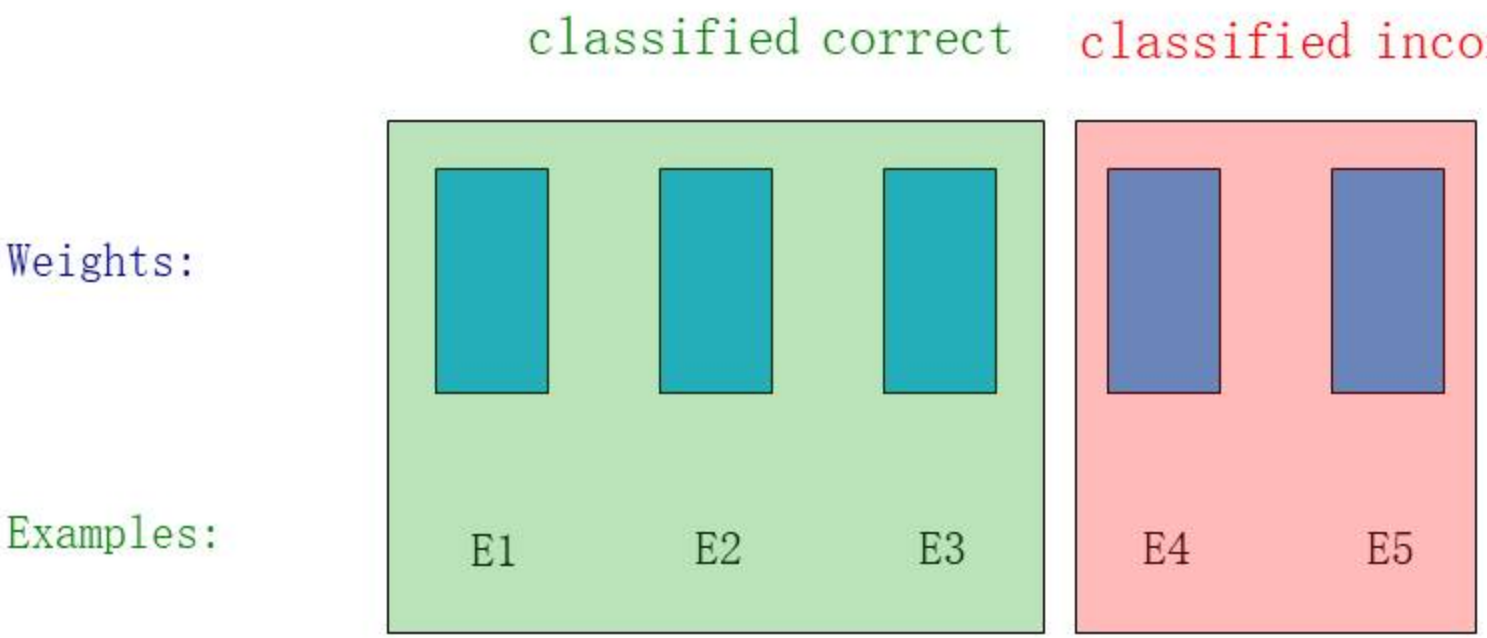
Weights:



Examples: E1 E2 E3 E4 E5

Learn a weak classifier:

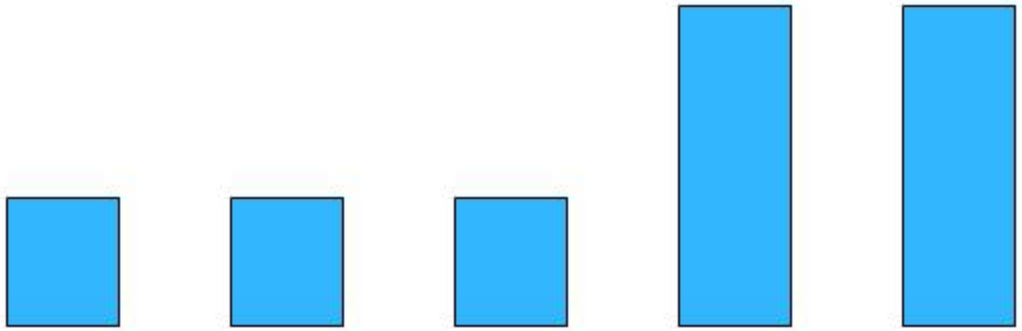
2.2.2Boosting提出-示例1



We want to 提升被错误分类样本权重，学习新的弱分类器

2.2.2Boosting提出-示例1

Weights:



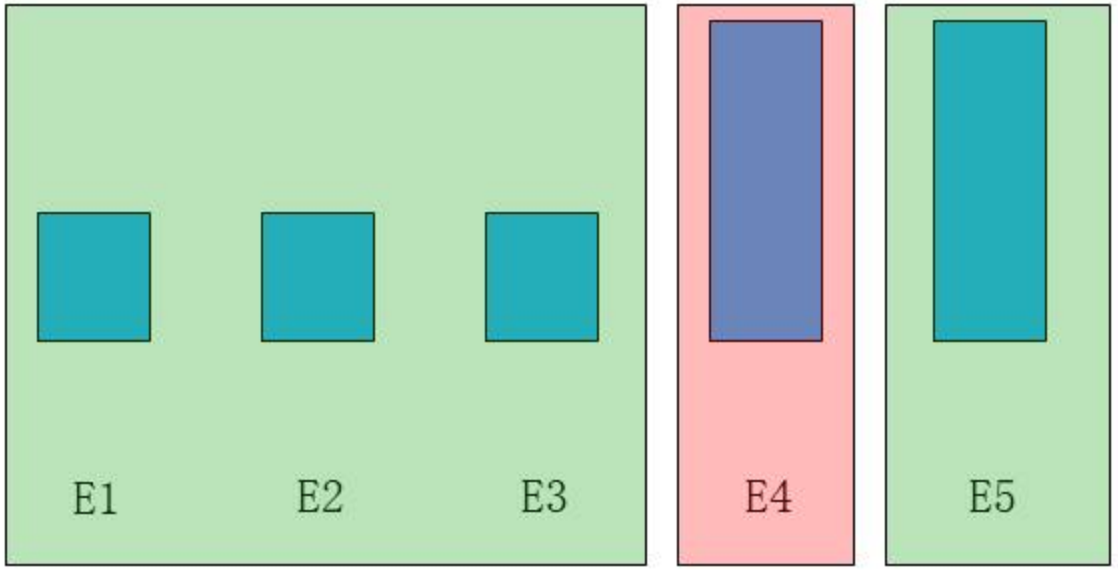
Examples: E1 E2 E3 E4 E5

Learn another weak classifier:

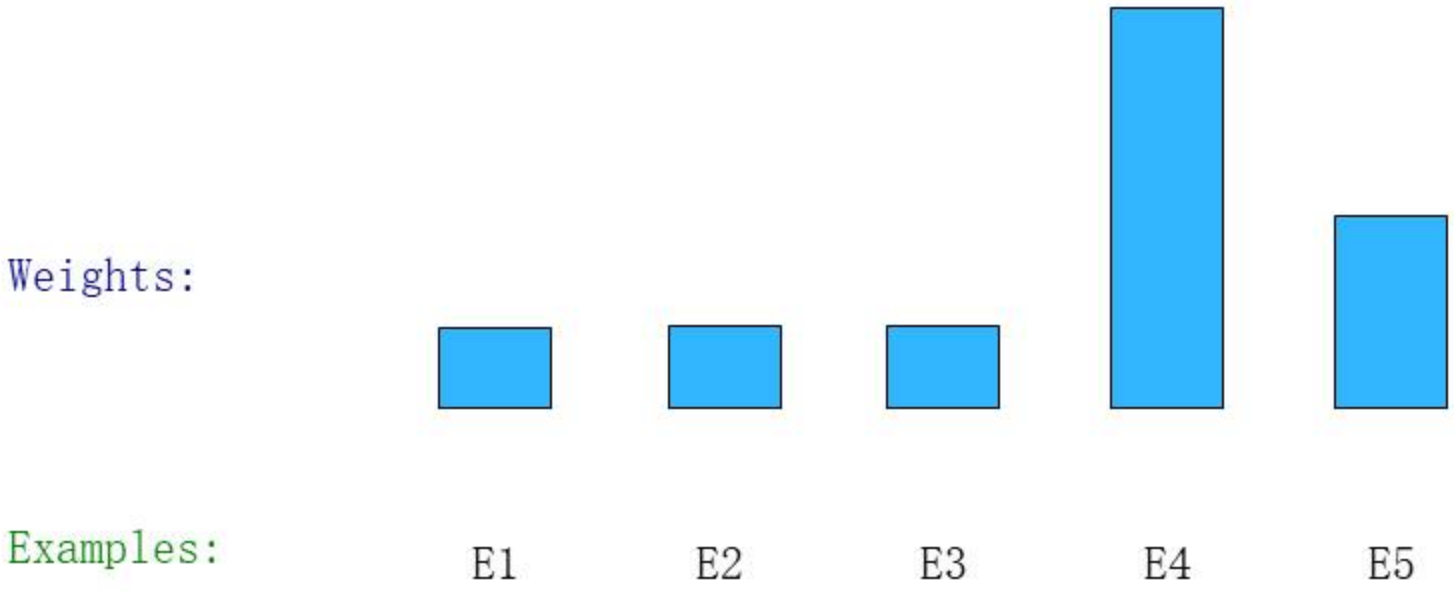
2.2.2Boosting提出-示例1

Weights:

Examples:

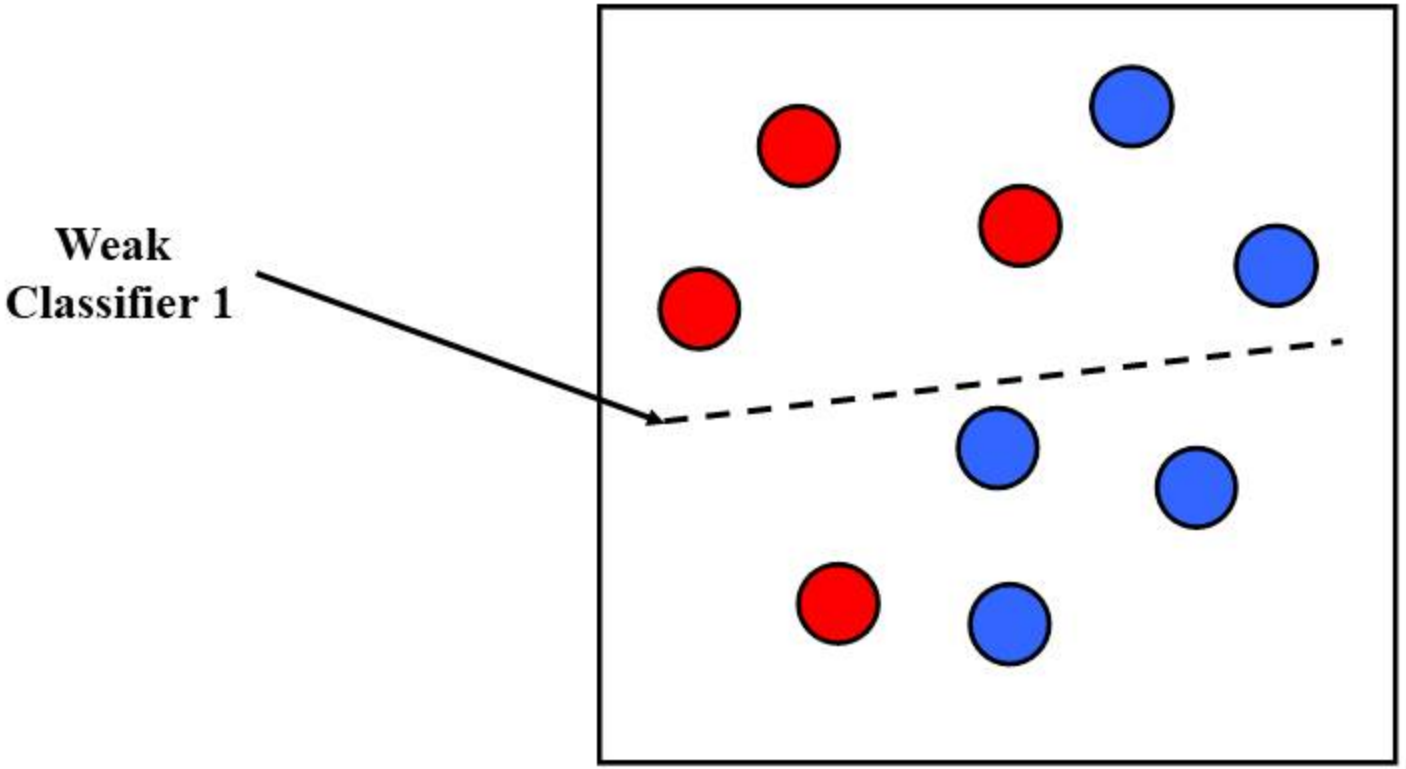


2.2.2Boosting提出-示例1

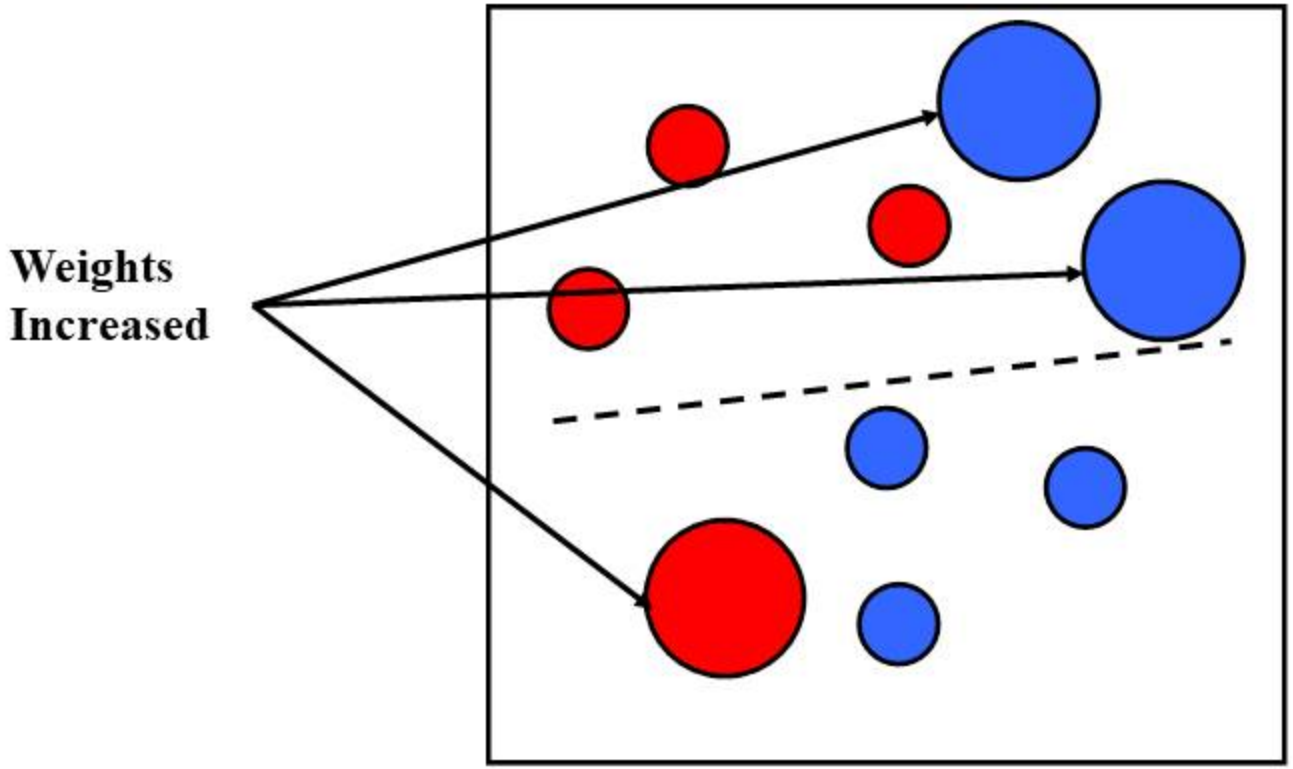


We want to 提升被错误分类样本权重，学习新的弱分类器

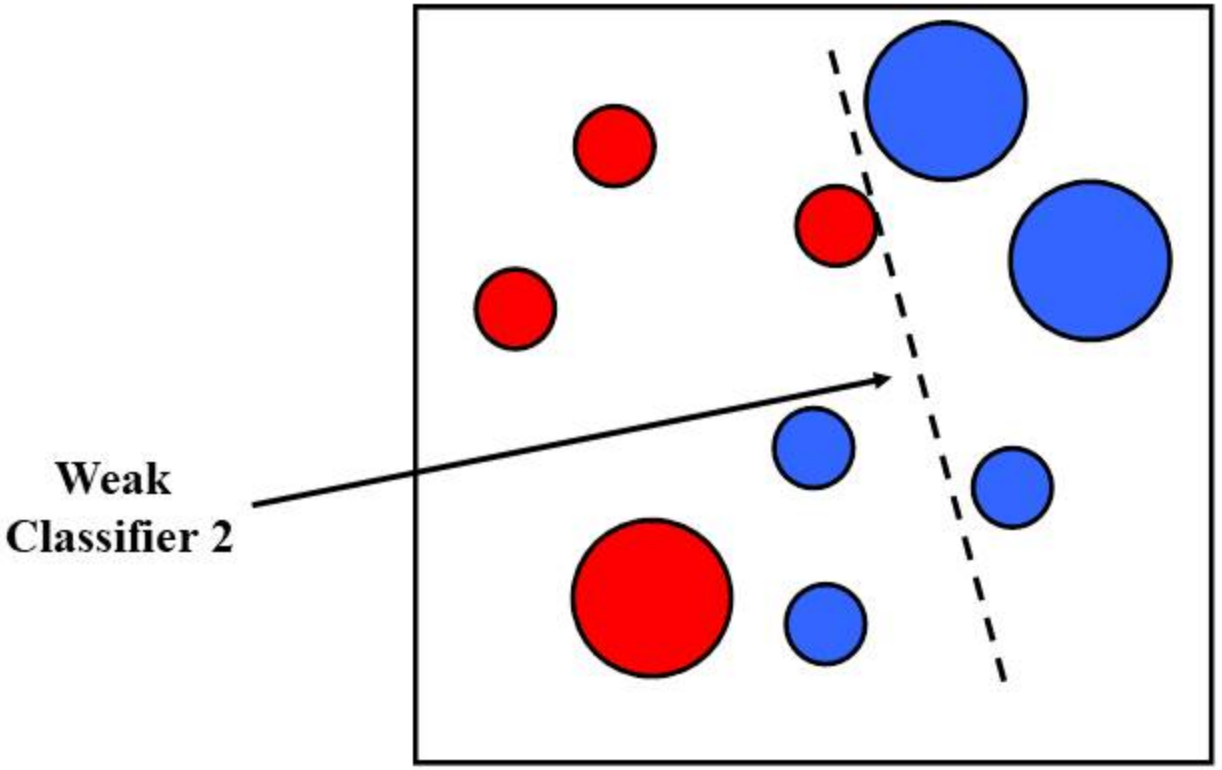
2.2.2Boosting提出-示例2



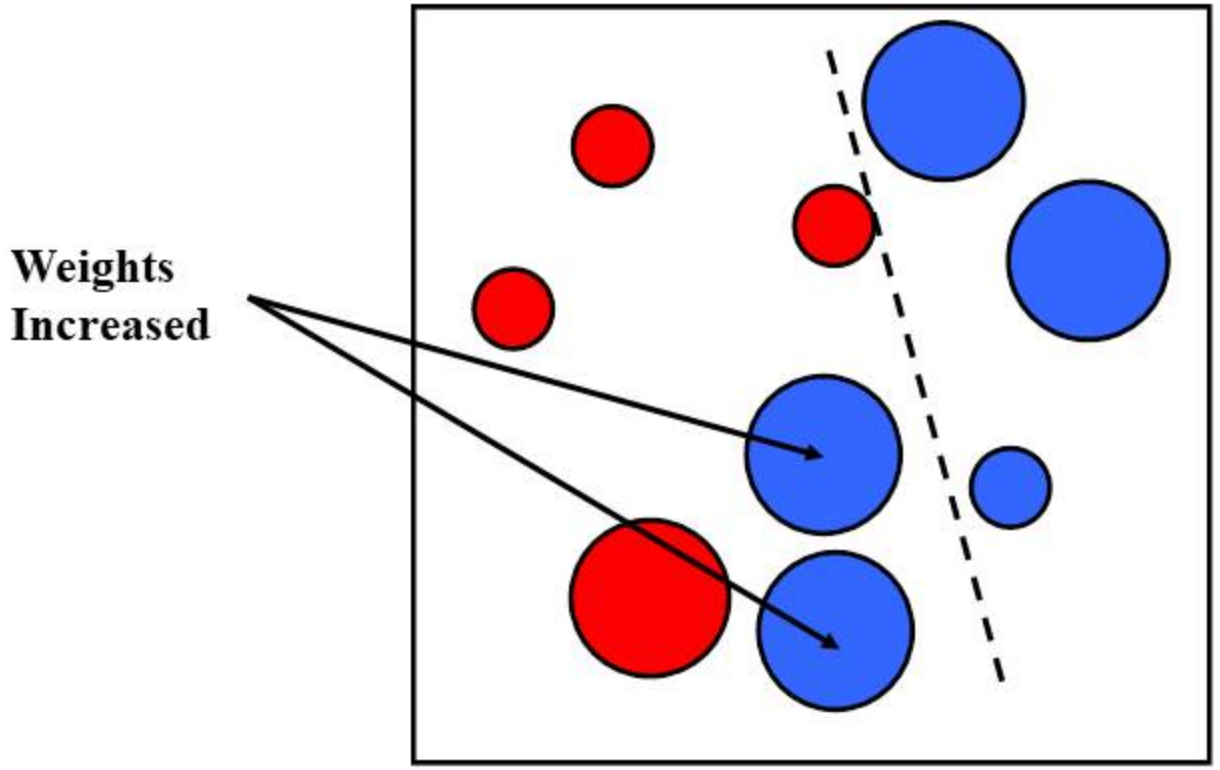
2.2.2Boosting提出-示例2



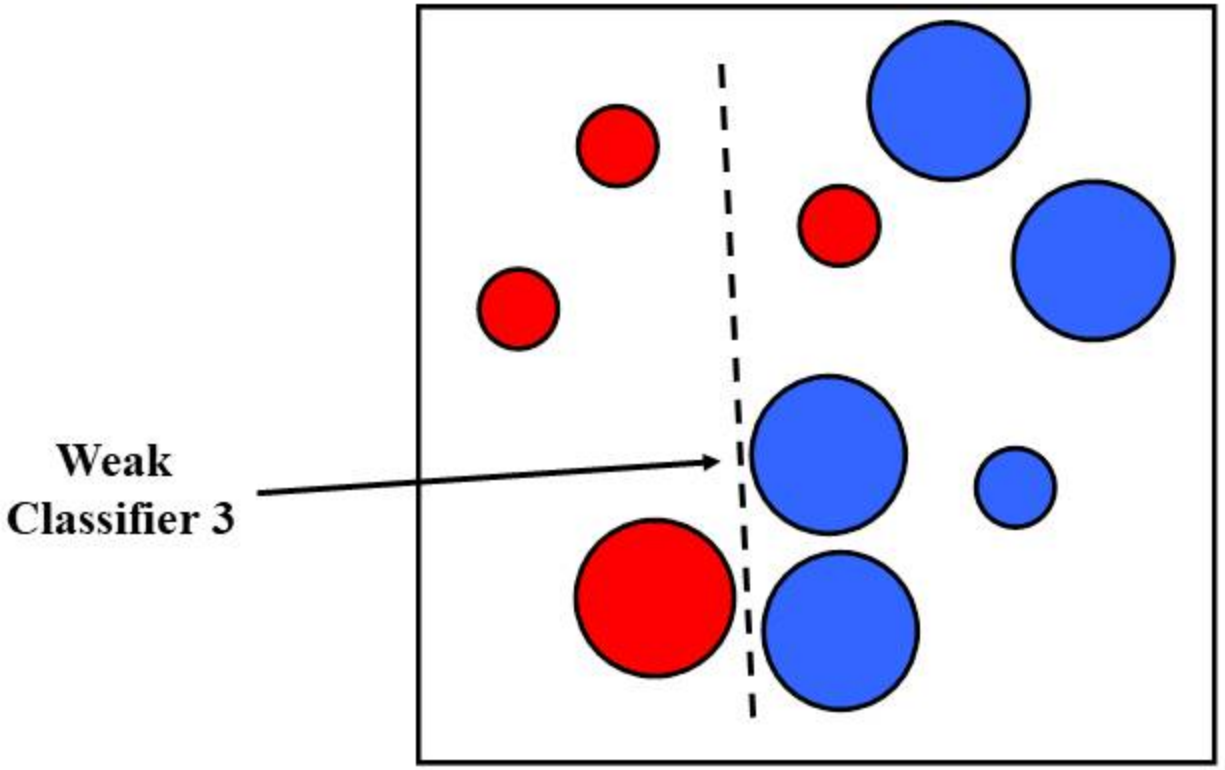
2.2.2Boosting提出-示例2



2.2.2Boosting提出-示例2

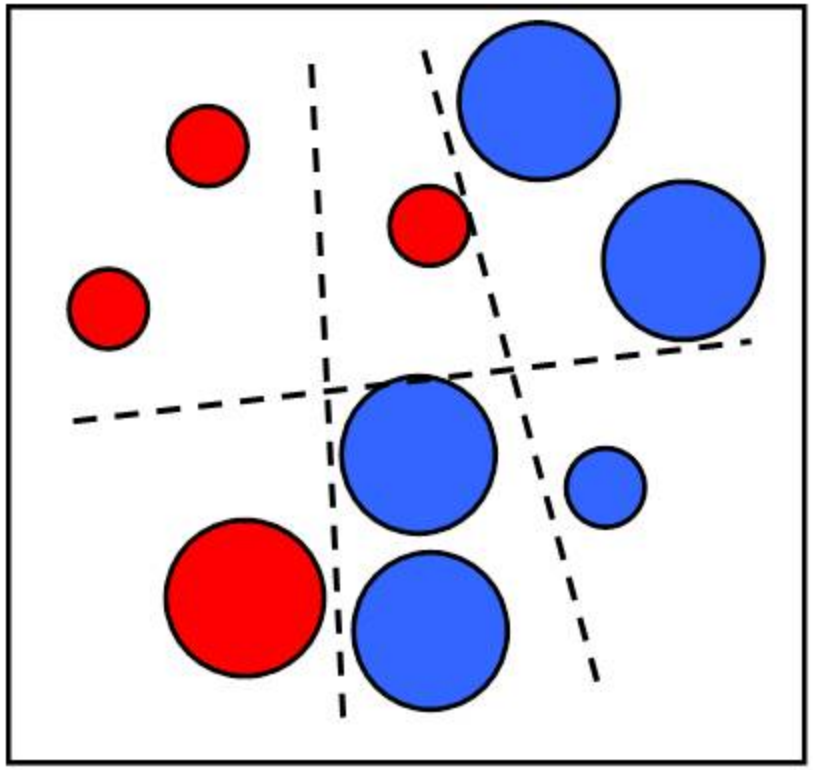


2.2.2Boosting提出-示例2



2.2.2Boosting提出-示例2

Final classifier is a combination of weak classifiers



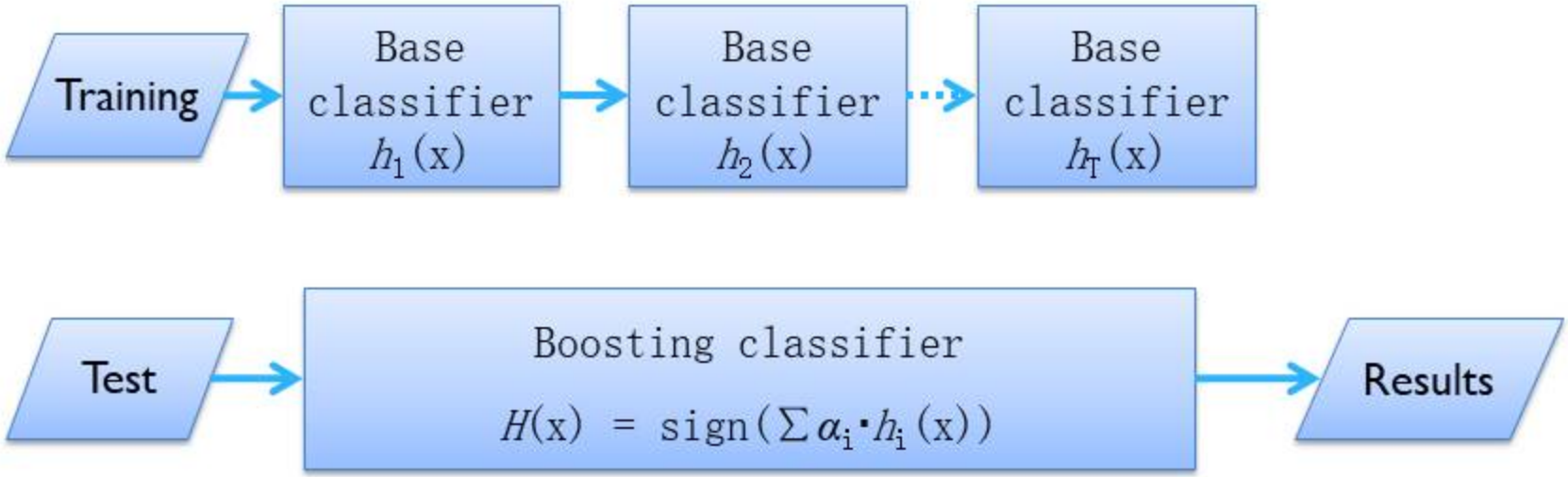
2.2.3Boosting Weight

- 数据的权重有两个作用
  - 使用这些权值作为抽样分布，进行对数据的抽样
  - 分类器可以使用权值学习有利于高权重样本的分类器。把一个弱分类器提升为一个强分类器
- Boosting算法之一：AdaBoost算法

2.2.3Boosting Weight

- 核心思想
- 样本的权重
  - 没有先验知识的情况下，初始的分布应为等概分布，也就是训练集如果有N个样本，每个样本的分布概率为1/N
  - 每次循环一后提高错误样本的分布概率，分错样本在训练集中所占权重增大， 使得下一次循环的弱学习机能够集中力量对这些错误样本进行判断。
- 弱学习机的权重
  - 准确率越高的弱学习机权重越高
- 循环控制：损失函数达到最小
  - 在强学习机的组合中增加一个加权的弱学习机，使准确率提高，损失函数值减小。

2.2.4Boosting分类过程



2.2.5AdaBoost: train

for k = 1 to *iterations*:

- $classifier_k$ = learn a weak classifier based on weights
- calculate weighted error for this classifier(加权分类误差)

$$\varepsilon_k = \sum_{i=1}^{n} w_i * 1[label_i \neq classifier_k(x_i)]$$

- calculate "score" for this classifier(分类器的系数)

$$\alpha_k = \frac{1}{2}\log\left(\frac{1-\varepsilon_i}{\varepsilon_i}\right)$$

- change the example weights(权值的更新)

$$w_i = \frac{1}{Z} w_i \exp\left(-\alpha_k * label_i * classifier_k(x_i)\right)$$

2.2.6Adaboost例子

- 训练样本，初始时样本权重相同

序号	i	1	2	3	4	5	6	7	8	9	10
数据	x	0	1	2	3	4	5	6	7	8	9
类别标签	y	1	1	1	-1	-1	-1	1	1	1	-1
初始权值	*w1i*	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1

- 1,2,3,7,8,9为同一类
- 4,5,6,10为同一类
- 根据x>v和x<v来分类

2.2.6Adaboost例子

- 第一次迭代 $G_1(x) = \begin{cases} 1, & x < 2.5 \\ -1, & x > 2.5 \end{cases}$

序号	i	1	2	3	4	5	6	7	8	9	10
数据	x	0	1	2	3	4	5	6	7	8	9
类别标签	y	1	1	1	-1	-1	-1	1	1	1	-1
分类器结果	$G1(x)$	1	1	1	-1	-1	-1	-1	-1	-1	-1
分类结果		对	对	对	对	对	对	错	错	错	对

- $G_1(x)$的误差率最低：$e_1 = P(G_1(x_i) \neq y_i) = \sum_{G_1(x_i) \neq y_i} w_{1i} = 0.1 + 0.1 + 0.1 = 0.3$
- $G_1(x)$的权重为：

$$\alpha_1 = \frac{1}{2} ln \frac{1-e_1}{e_1} = \frac{1}{2} ln \frac{1-0.3}{0.3} \approx 0.42365$$

- 分类函数：

$$f_1(x) = \alpha_1 G_1(x) = 0.42365 G_1(x)$$

for k = 1 to *iterations*:

- $classifier_k$ = learn a weak classifier based on weights
- calculate weighted error for this classifier(加权分类误差)

$$\varepsilon_k = \sum_{i=1}^{n} w_i * \mathrm{I}[label_i \neq classifier_k(x_i)]$$

- calculate "score" for this classifier(分类器的系数)

$$\alpha_k = \frac{1}{2} \log\left(\frac{1-\varepsilon_i}{\varepsilon_i}\right)$$

- change the example weights(权值的更新)

$$w_i = \frac{1}{Z} w_i \exp(-\alpha_k * label_i * classifier_k(x_i))$$

2.2.6Adaboost例子

- 第一次迭代

序号	i	1	2	3	4	5	6	7	8	9	10
数据	x	0	1	2	3	4	5	6	7	8	9
类别标签	y	1	1	1	-1	-1	-1	1	1	1	-1
初始权值	*w1i*	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1

$$G_1(x) = \begin{cases} 1, & x < 2.5 \\ -1, & x > 2.5 \end{cases} \qquad \alpha_1 = \frac{1}{2} ln \frac{1-e_1}{e_1} = \frac{1}{2} ln \frac{1-0.3}{0.3} \approx 0.42365$$

for k = 1 to *iterations*:

- classifier$_k$ = learn a weak classifier based on weights
- calculate weighted error for this classifier(加权分类误差)

$$\varepsilon_k = \sum_{i=1}^{n} w_i * \mathrm{I}[label_i \neq classifier_k(x_i)]$$

- calculate "score" for this classifier(分类器的系数)

$$\alpha_k = \frac{1}{2} \log\left(\frac{1-\varepsilon_i}{\varepsilon_i}\right)$$

- change the example weights(权值的更新)

$$w_i = \frac{1}{Z} w_i \exp(-\alpha_k * label_i * classifier_k(x_i))$$

$$Z_1 = \sum_{i=1}^{n} w_{1i} exp(-y_i \alpha_1 G_1(x_i))$$

$$= \sum_{i=1}^{3} 0.1 \times exp(-[1 \times 0.4263 \times 1]) + \sum_{i=4}^{4-6,10} 0.1 \times exp(-[(-1) \times 0.4263 \times (-1)]) + \sum_{i=7}^{9} 0.1 \times exp(-[1 \times 0.4263 \times (-1)])$$

$$\approx 0.3928 + 0.4582 + 0.0655$$

$$= 0.9165$$

$$w_{2i} = \frac{w_{1i}}{Z_1} exp(-y_i \alpha_1 G_1(x_i))$$

$$= \begin{cases} \frac{0.1}{0.9165} exp(-[1 \times 0.4236 \times 1]), & i = 1,2,3 \\ \frac{0.1}{0.9165} exp(-[(-1) \times 0.4236 \times (-1)]), & i = 4,5,6,10 \\ \frac{0.1}{0.9165} exp(-[1 \times 0.4236 \times (-1)]), & i = 7,8,9 \end{cases}$$

$$\approx \begin{cases} 0.07143 & i = 1,2,3 \\ 0.07143 & i = 4,5,6,10 \\ 0.16666 & i = 7,8,9 \end{cases}$$

序号	i	1	2	3	4	5	6	7	8	9	10
数据	x	0	1	2	3	4	5	6	7	8	9
类别标签	y	1	1	1	-1	-1	-1	1	1	1	-1
初始权值1	*w1iw1i*	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
更新权值2	*w2iw2i*	0.0714	0.0714	0.0714	0.0714	0.0714	0.0714	0.1667	0.1667	0.1667	0.0714

2.2.6Adaboost例子

- 第二次迭代 $G_2(x) = \begin{cases} 1, & x < 8.5 \\ -1, & x > 8.5 \end{cases}$

序号	i	1	2	3	4	5	6	7	8	9	10
数据	x	0	1	2	3	4	5	6	7	8	9
类别标签	y	1	1	1	-1	-1	-1	1	1	1	-1
权值分布	$w2i$	0.0714	0.0714	0.0714	0.0714	0.0714	0.0714	0.1667	0.1667	0.1667	0.0714
分类器结果	$G2(x)$	1	1	1	1	1	1	1	1	1	-1
分类结果		对	对	对	错	错	错	对	对	对	对

- $G_2(x)$的误差率最低：

$e_2 = P(G_2(x_i) \neq y_i) = \sum_{G2(x_i) \neq y_i} w_{2i} = 0.07143 + 0.07143 + 0.07143 = 0.21429$

- $G_2(x)$的权重为：

$\alpha_2 = \frac{1}{2} ln \frac{1-e_2}{e_2} = \frac{1}{2} ln \frac{1-0.21429}{0.21429} \approx 0.64963$

- 分类函数：$f_2(x) = \alpha_2 G_2(x) = 0.64963 G_2(x)$

for k = 1 to *iterations*:

- $classifier_k$ = learn a weak classifier based on weights
- calculate weighted error for this classifier(加权分类误差)

$$\varepsilon_k = \sum_{i=1}^{n} w_i * 1[label_i \neq classifier_k(x_i)]$$

- calculate "score" for this classifier(分类器的系数)

$$\alpha_k = \frac{1}{2} \log\left(\frac{1-\varepsilon_i}{\varepsilon_i}\right)$$

- change the example weights(权值的更新)

$$w_i = \frac{1}{Z} w_i \exp(-\alpha_k * label_i * classifier_k(x_i))$$

2.2.6Adaboost例子

$$G_2(x) = \begin{cases} 1, & x < 8.5 \\ -1, & x > 8.5 \end{cases}$$

$$\alpha_2 = \frac{1}{2} ln \frac{1-e_2}{e_2} = \frac{1}{2} ln \frac{1-0.21429}{0.21429} \approx 0.64963$$

$$Z_2 = \sum_{i=1}^{n} w_{2i} exp(-y_i \alpha_2 G_2(x_i))$$

$$= \sum_{i=1}^{3} 0.07143 \times exp(-[1 \times 0.64963 \times 1])$$

$$+ \sum_{i=4}^{6} 0.07143 \times exp(-[(-1) \times 0.64963 \times 1])$$

$$+ \sum_{i=7}^{9} 0.16666 \times exp(-[1 \times 0.64963 \times 1])$$

$$+ \sum_{i=10}^{10} 0.07143 \times exp(-[(-1) \times 0.64963 \times (-1)]$$

$$\approx 0.11191 + 0.41033 + 0.26111 + 0.03730$$

$$= 0.82065$$

for k = 1 to *iterations*:

- $classifier_k$ = learn a weak classifier based on weights
- calculate weighted error for this classifier(加权分类误差)

$$\varepsilon_k = \sum_{i=1}^{n} w_i * \mathrm{I}[label_i \neq classifier_k(x_i)]$$

- calculate "score" for this classifier(分类器的系数)

$$\alpha_k = \frac{1}{2} \log\left(\frac{1-\varepsilon_i}{\varepsilon_i}\right)$$

- change the example weights(权值的更新)

$$w_i = \frac{1}{Z} w_i \exp(-\alpha_k * label_i * classifier_k(x_i))$$

$$w_{3i} = \frac{w_{2i}}{Z_2} exp(-y_i \alpha_2 G_2(x))$$

$$= \begin{cases} \frac{0.07143}{0.82065} exp(-[1 \times 0.64963 \times 1]) \approx 0.04546, & i = 1,2,3 \\ \frac{0.07143}{0.82065} exp(-[(-1) \times 0.64963 \times 1]) \approx 0.16667, & i = 4,5,6 \\ \frac{0.16666}{0.82065} exp(-[1 \times 0.64963 \times 1]) \approx 0.10606, & i = 7,8,9 \\ \frac{0.07143}{0.82065} exp(-[(-1) \times 0.64963 \times (-1)]) \approx 0.04546, & i = 10 \end{cases}$$

序号	i	1	2	3	4	5	6	7	8	9	10
数据	x	0	1	2	3	4	5	6	7	8	9
类别标签	y	1	1	1	-1	-1	-1	1	1	1	-1
初始权值1	*w1i*	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
权值2	*w2i*	0.0714	0.0714	0.0714	0.0714	0.0714	0.0714	0.1667	0.1667	0.1667	0.0714
更新权值3	$w_{3i}	0.0455	0.0455	0.0455	0.1667	0.1667	0.1667	0.1061	0.1061	0.1061	0.0455

2.2.6Adaboost例子

- 第三次迭代

$$G_3(x) = \begin{cases} -1, & x < 5.5 \\ 1, & x > 5.5 \end{cases}$$

序号	i	1	2	3	4	5	6	7	8	9	10
数据	x	0	1	2	3	4	5	6	7	8	9
类别标签	y	1	1	1	-1	-1	-1	1	1	1	-1
权值分布	$w3i$	0.0455	0.0455	0.0455	0.1667	0.1667	0.1667	0.1061	0.1061	0.1061	0.0455
分类器结果	$G3(x)$	-1	-1	-1	-1	-1	-1	1	1	1	1
分类结果		错	错	错	对	对	对	对	对	对	错

- $G_3(x)$的误差率最低：

$$e_3 = P(G_3(x_i) \neq y_i) = \sum_{G_3(x_i) \neq y_i} w_{3i} = 0.04546 + 0.04546 + 0.04546 + 04546 = 0.18184$$

- $G_3(x)$的权重为：

$$\alpha_3 = \frac{1}{2} ln \frac{1-e_3}{e_3} = \frac{1}{2} ln \frac{1-0.18188}{0.18184} \approx 0.75197$$

- 分类函数：$f_3(x) = \alpha_3 G_3(x) = 0.75197 G_3(x)$

for k = 1 to *iterations*:

- $classifier_k$ = learn a weak classifier based on weights
- calculate weighted error for this classifier(加权分类误差)

$$\varepsilon_k = \sum_{i=1}^{n} w_i * 1[label_i \neq classifier_k(x_i)]$$

- calculate "score" for this classifier(分类器的系数)

$$\alpha_k = \frac{1}{2} \log\left(\frac{1-\varepsilon_i}{\varepsilon_i}\right)$$

- change the example weights(权值的更新)

$$w_i = \frac{1}{Z} w_i \exp(-\alpha_k * label_i * classifier_k(x_i))$$

2.2.6Adaboost例子

- 第三次迭代

序号	i	1	2	3	4	5	6	7	8	9	10
数据	x	0	1	2	3	4	5	6	7	8	9
类别标签	y	1	1	1	-1	-1	-1	1	1	1	-1
初始权值1	*w1i*	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
权值2	*w2i*	0.0714	0.0714	0.0714	0.0714	0.0714	0.0714	0.1667	0.1667	0.1667	0.0714
权值3	*w3i*	0.0455	0.0455	0.0455	0.1667	0.1667	0.1667	0.1061	0.1061	0.1061	0.0455
更新权值4	*w4i*	0.125	0.125	0.125	0.1019	0.1019	0.1019	0.0648	0.0648	0.0648	0.125

- 最终分类器：
  - $G_m(x)=sign(0.42365G_1(x)+0.64963+G_2(x)+0.75197G_3(x))$

2.2.7AdaBoost: train再看算法

for k = 1 to *iterations*:

- $classifier_k$ = learn a weak classifier based on weights
- calculate weighted error for this classifier(加权分类误差)

$$\varepsilon_k = \sum_{i=1}^{n} w_i * 1[label_i \neq classifier_k(x_i)]$$

- calculate "score" for this classifier(分类器的系数)

$$\alpha_k = \frac{1}{2}\log\left(\frac{1-\varepsilon_i}{\varepsilon_i}\right)$$

为什么？

- change the example weights(权值的更新)

$$w_i = \frac{1}{Z} w_i \exp\left(-\alpha_k * label_i * classifier_k(x_i)\right)$$

2.2.8如何确定α

- 集成学习模型 $f(x)=\sum_{m=1}^{M}\alpha_m G_m(x)$
- 指数损失函数 $L(y,f(x))=\exp[-yf(x)]$
- 公式推导
  - 1、已知 $f_m(x)=f_{m-1}(x)+\alpha_m G_m(x)$
  - 2、公式代入

将 $f_m(x)=f_{m-1}(x)+\alpha_m G_m(x)$ 代入损失函数得：

$$L(y,f(x))=\sum_{i=1}^{N}\exp[-y_i(f_{m-1}(x)+\alpha_m G_m(x))]$$

  - 3、求导 $\alpha_k=\frac{1}{2}\log\left(\frac{1-\varepsilon_i}{\varepsilon_i}\right)$

https://www.cnblogs.com/liuwu265/p/4692347.html

2.2.9AdaBoost: train第三次加深算法

for k = 1 to *iterations*:

- $classifier_k$ = learn a weak classifier based on weights
- calculate weighted error for this classifier(加权分类误差)

$$\varepsilon_k = \sum_{i=1}^{n} w_i * 1[label_i \neq classifier_k(x_i)]$$

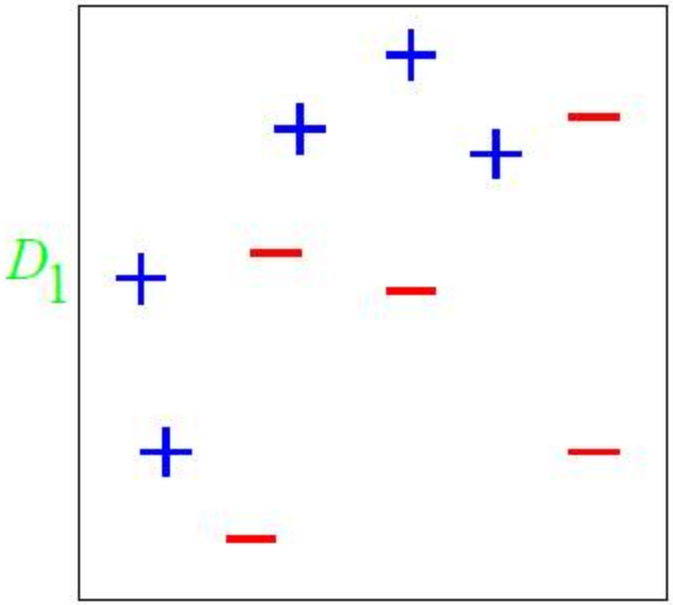
- calculate "score" for this classifier(分类器的系数)

$$\alpha_k = \frac{1}{2}\log\left(\frac{1-\varepsilon_i}{\varepsilon_i}\right)$$

- change the example weights(权值的更新)

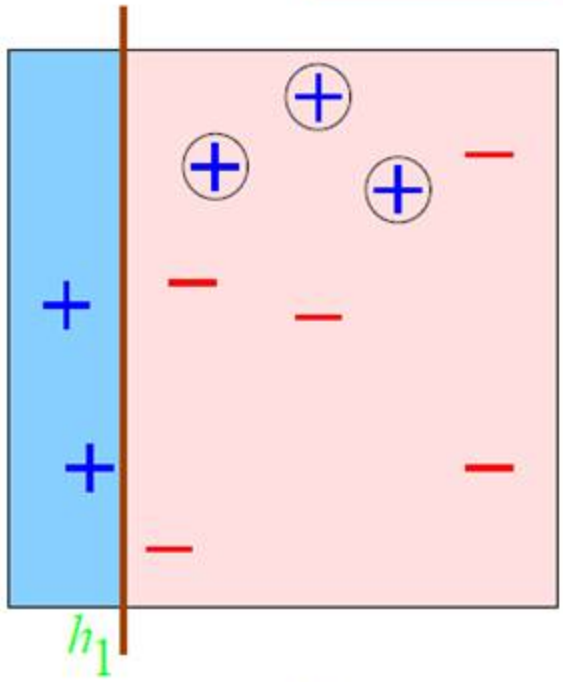
$$w_i = \frac{1}{Z} w_i \exp\left(-\alpha_k * label_i * classifier_k(x_i)\right)$$

2.2.10A Toy Example



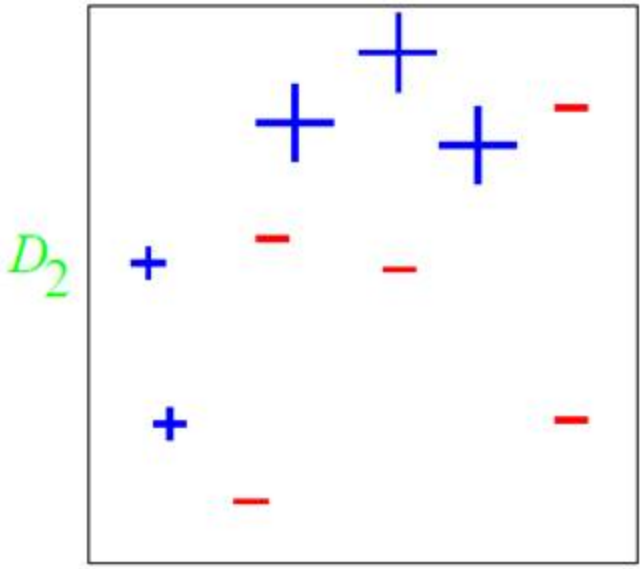
A Toy E

Round 1

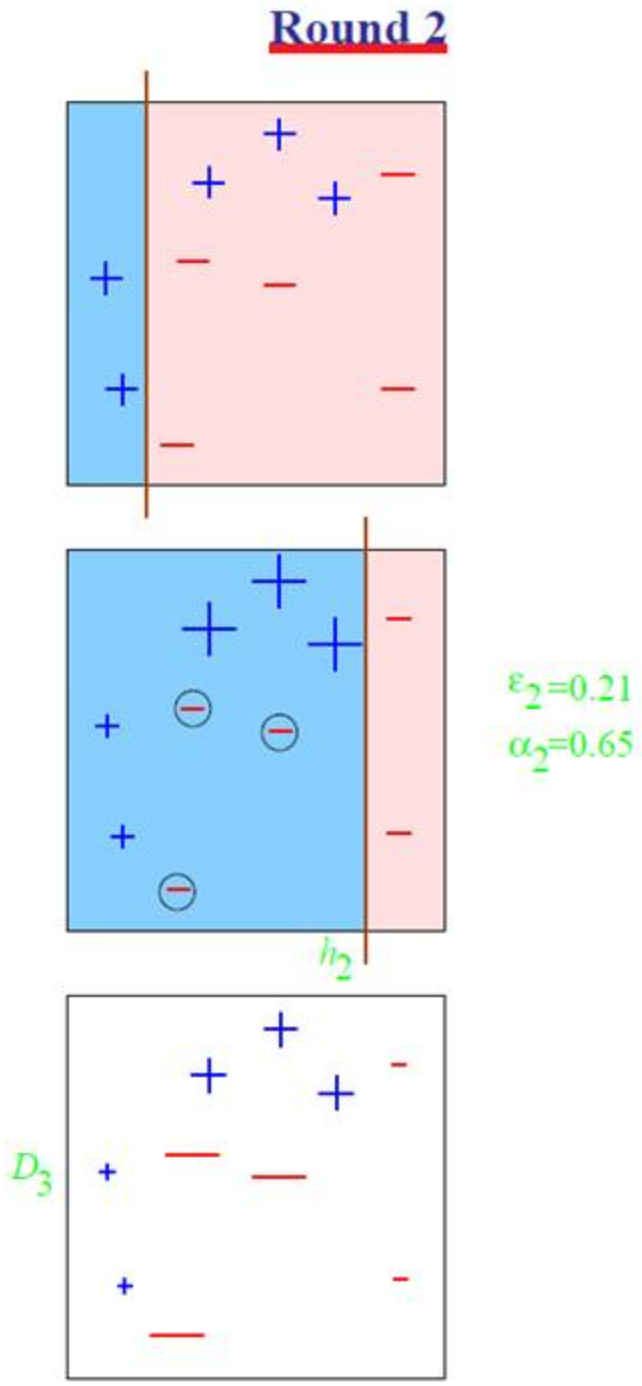


$\varepsilon_1 = 0.30$

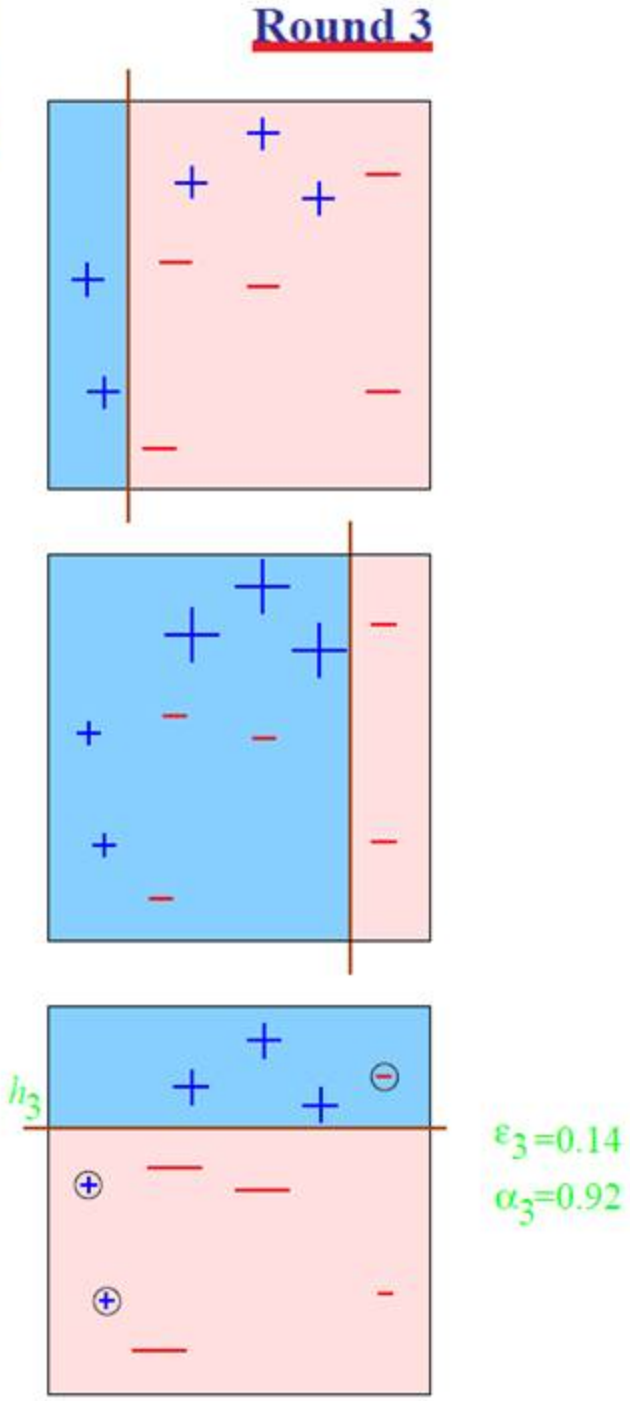
$\alpha_1 = 0.42$



A Toy Exan



A Toy Exan

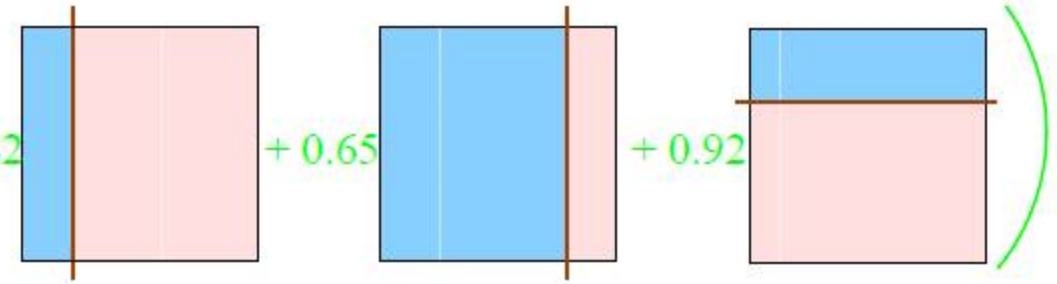


A Toy Example

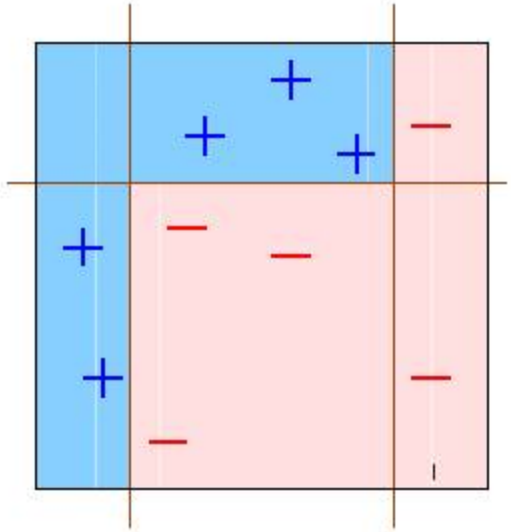
A cool and important note about the final hypothesis: it is possible that the combined hypothesis makes no mistakes on the training data, but boosting can still learn, by adding more weak hypotheses.

Final Hypothesis

$$H_{\text{final}} = \text{sign}\left(0.42 + 0.65 + 0.92\right)$$



$=$



2.2.11Boosting小结

- Performance of AdaBoost depends on data and weak learner
- Consistent with theory, AdaBoost can fail if
  - weak classifier too complex – overfitting
  - weak classifier too weak -- underfitting
- Empirically, AdaBoost seems especially susceptible to uniform noise

多选题 1分

设置

下列关于AdaBoost算法，说法正确的是

A 比较适合于关注疑难数据的分类

B 弱分类器太复杂容易过拟合

C 弱分类器太简单容易欠拟合

D 对噪音数据敏感

提交

2.2.12Other Boosting Method

- 不同的损失函数和极小化损失函数方法决定了boosting的最终效果

Name	Loss	Derivative	f^*	Algorithm
Squared error	$\frac{1}{2}(y_i - f(\mathbf{x}_i))^2$	$y_i - f(\mathbf{x}_i)$	$\mathbb{E}[y\|\mathbf{x}_i]$	L2Boosting
Absolute error	$\|y_i - f(\mathbf{x}_i)\|$	$\text{sgn}(y_i - f(\mathbf{x}_i))$	$\text{median}(y\|\mathbf{x}_i)$	Gradient boosting
Exponential loss	$\exp(-\tilde{y}_i f(\mathbf{x}_i))$	$-\tilde{y}_i \exp(-\tilde{y}_i f(\mathbf{x}_i))$	$\frac{1}{2}\log\frac{\pi_i}{1-\pi_i}$	AdaBoost
Logloss	$\log(1 + e^{-\tilde{y}_i f_i})$	$y_i - \pi_i$	$\frac{1}{2}\log\frac{\pi_i}{1-\pi_i}$	LogitBoost

(图自 Machine Learning A Probabilistic Perspective)

2.2.13算法对比

Items	Bagging	Boosting
采样算法	均匀取样	根据错误率来取样
各轮训练集选取	随机的	与前面各轮的学习结果有关
预测函数权重	没有权重	有权重
并行性	各个预测函数可以并行生成	只能顺序生成
准确性	没有boosting高	在大多数数据集中，高
过拟合	不会	在有些数据集中，会

2.2.14Gradient Boost Decision Tree

- GBDT、Treelink、 GBRT(Gradient Boost Regression Tree)、Tree Net、MART(Multiple Additive Regression Tree)
  - 由多棵决策树构成，通常都是上百棵树，而且每棵树规模都较小（即树的深度会比较浅）
  - 模型预测的时候，对于输入的一个样本实例，然后会遍历每一棵决策树，每棵树都会对预测值进行调整修正，最后得到预测的结果

$$F(X) = F_0 + \beta_1 T_1(X) + \beta_2 T_2(X) + \ldots + \beta_M T_M(X)$$

2.2.14Gradient Boost Decision Tree

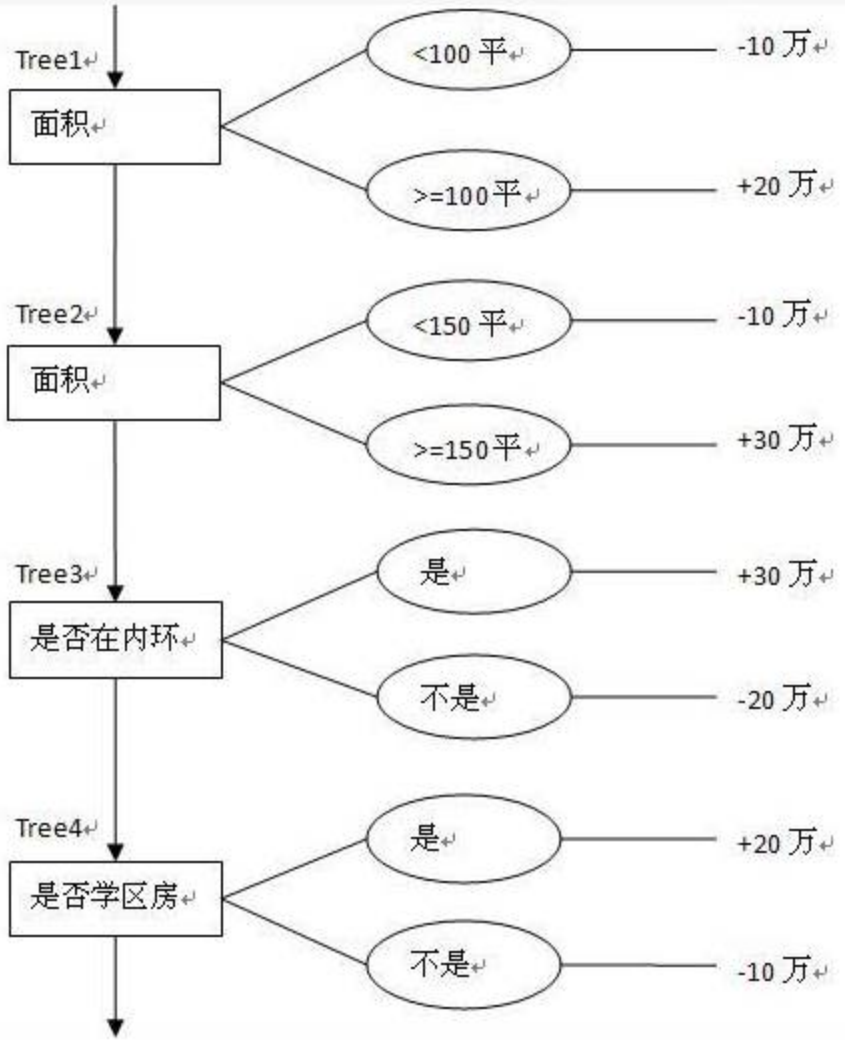
$$F(X) = F_0 + \beta_1 T_1(X) + \beta_2 T_2(X) + \ldots + \beta_M T_M(X)$$

- 对于不同的问题和选择不同的损失函数，初值的设定是不同的
- 比如回归问题并且选择高斯损失函数，那么这个初值就是训练样本的目标的均值

填空题 1分

设置

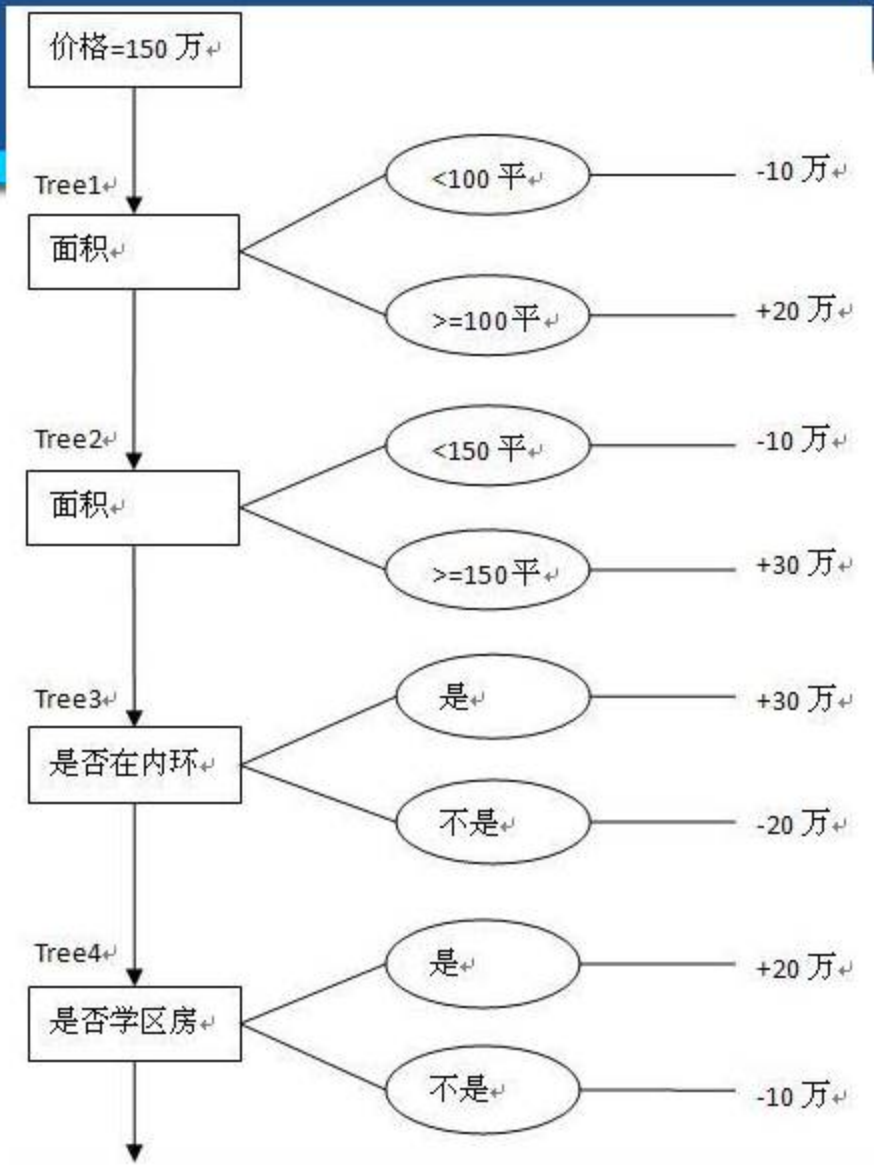
- 一套房子的价格
- 特征有三个
  - 房子的面积
  - 是否在内环
  - 是否学区房
- 由四棵决策树构成
  - Decision Stump
- 初值设为价格的均值150万
- 一个面积为120平的内环非学区房的价格预测值为150+20-10+30-10=[填空1]万



作答

2.2.14举个栗子

- 一套房子的价格
- 特征有三个
  - 房子的面积
  - 是否在内环
  - 是否学区房
- 由四棵决策树构成
  - Decision Stump
- 初值设为价格的均值150万
- 一个面积为120平的内环非学区房的价格预测值为150+20-10+30-10=180万

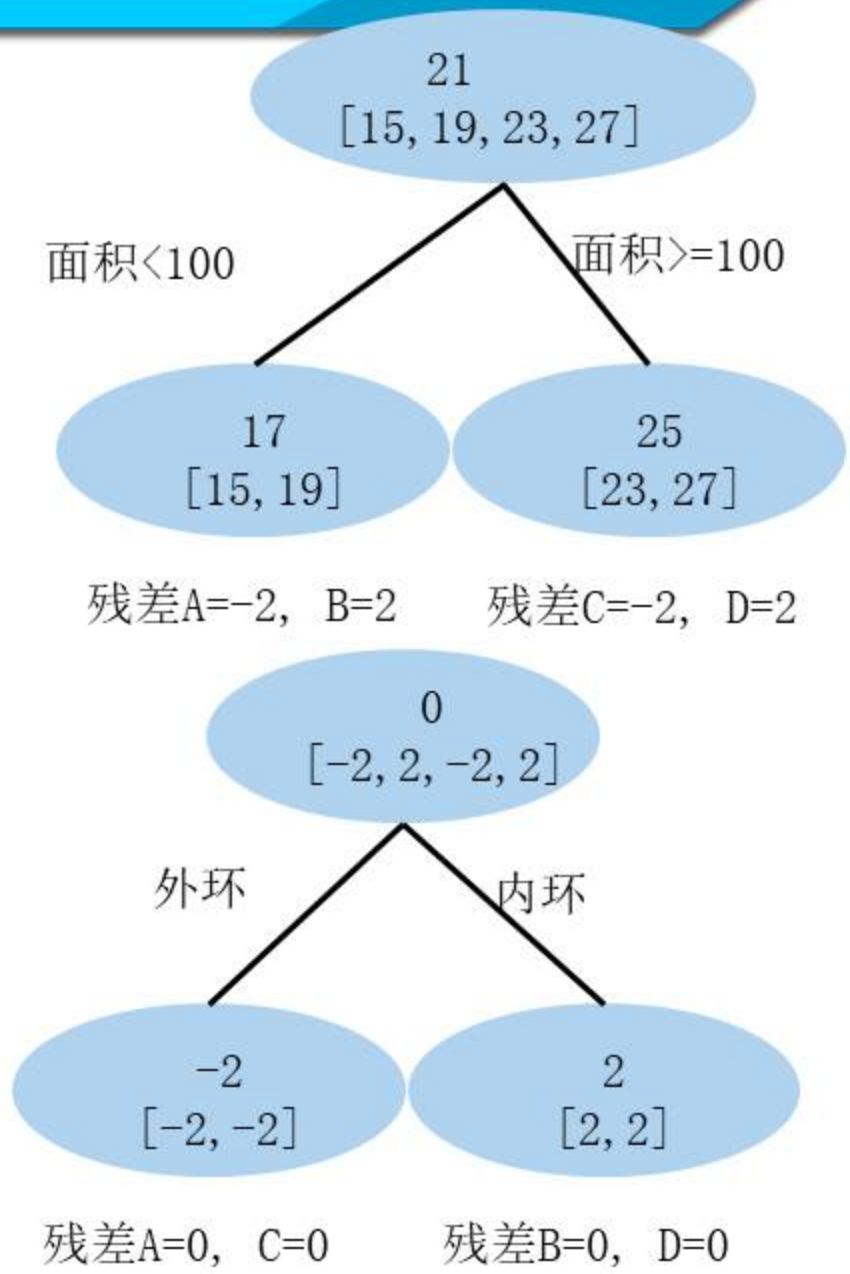


2.2.15GBDT学习过程

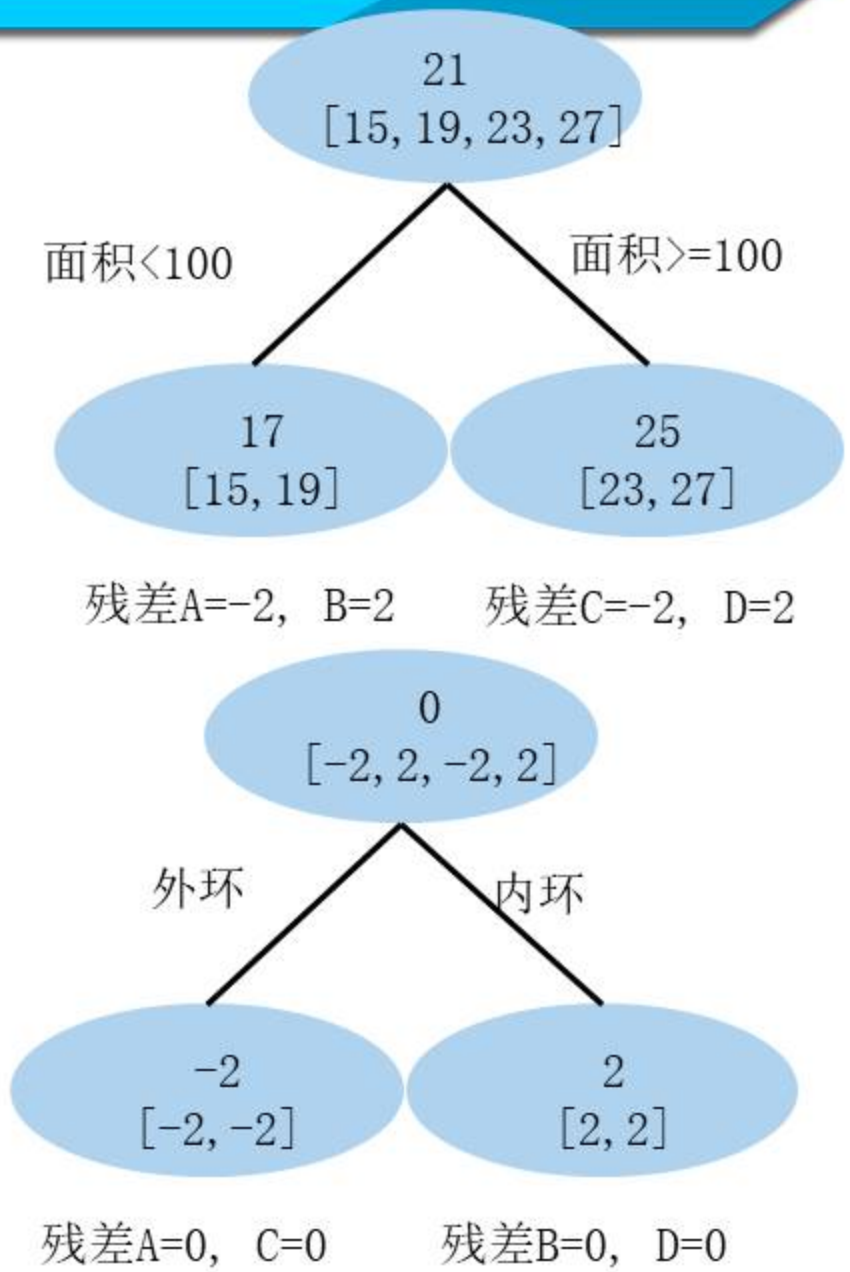
- Boosting，迭代，即通过迭代多棵树来共同决策
- GBDT是把所有树的结论累加起来做最终结论的，所以可以想到每棵树的结论并不是房价本身，而是房价的一个累加量
- 每一棵树学的是之前所有树结论和的残差，这个残差就是一个加预测值后能得真实值的累加量

2.2.16GBDT学习过程——例子引入

面积	是否内环	房价
92	否	15
90	是	19
120	否	23
130	是	27



- 预测
  - 125平方米，内环


21
[15, 19, 23, 27]
面积<100
面积>=100
17
[15, 19]
25
[23, 27]
残差A=-2， B=2
残差C=-2， D=2
0
[-2, 2, -2, 2]
外环
内环
-2
[-2, -2]
2
[2, 2]
残差A=0， C=0
残差B=0， D=0

2.2.17GBDT例子的几点问题

- GBDT有何优点?
  - 防止过拟合
  - 每一步的残差计算其实变相地增大了分错instance的权重，而已经分对的instance则都趋向于0
- 其他类似算法
  - xgboost

多选题 1分

设置

GBDT算法特点

A 防止过拟合

B 通过残差来构造每个子树

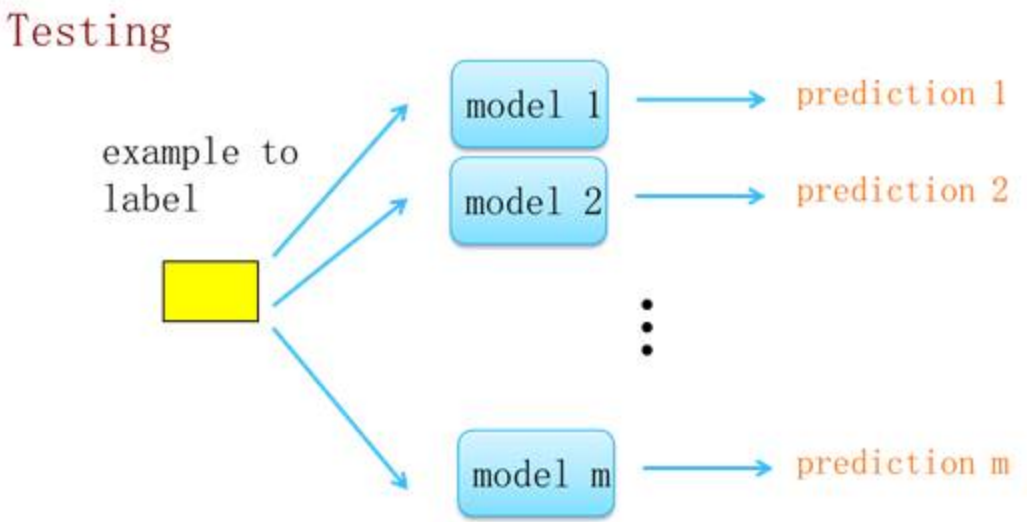
提交

主要内容

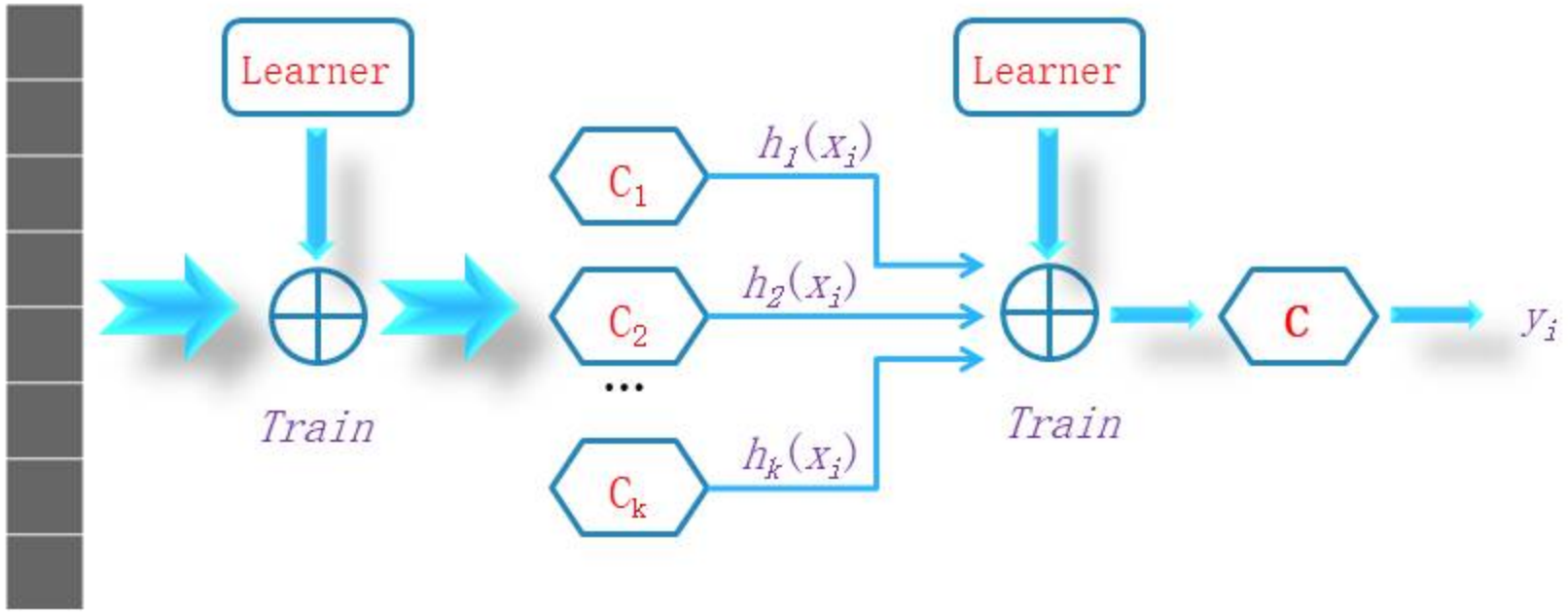
- 集成学习简介
- 集成学习算法
  - Bagging
  - Boosting
  - Stacking
- 集成学习应用

2.3.1如何组合多个分类器?

- 平均
- 投票
  - 多数投票：Bagging
  - 带权重的多数投票：Adaboost
- 学习组合器
  - Stacking
  - RegionBoost



2.3.2Stacking



D

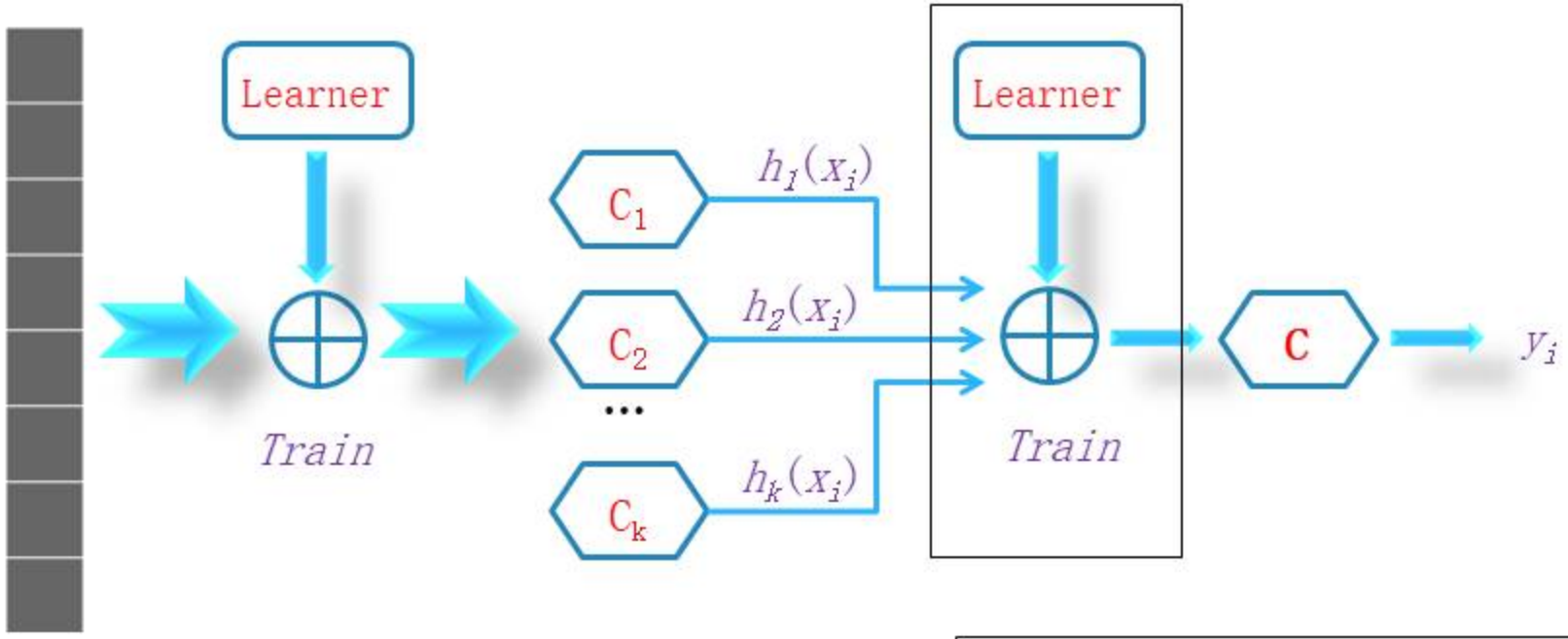
Base Classifiers

$\{(x_1, y_1) \dots (x_n, y_n)\}$

Meta Classifier

$\{(h_1(x_i), h_2(x_i), \dots, h_k(x_i), y_i)\}$

2.3.2Stacking



D

Base Classifiers

$\{(x_1, y_1) \dots (x_n, y_n)\}$

Meta Classifier

$\{(h_1(x_i), h_2(x_i), \dots, h_k(x_i), y_i)\}$

基分类器的结果再
训练出一个分类器

主要内容

- 集成学习简介
- 集成学习算法
  - Bagging
  - Boosting
  - Stacking
- 集成学习应用

集成学习应用

- 竞赛的冠军
  - KDDCUP
  - Netflix Prize
- 视觉领域
  - 动作检测、人物追踪、物体识别
- 生物领域
  - 基因组功能预测、癌症预测

集成学习编程实践

- https://scikit-learn.org/stable/modules/ensemble.html
- 同学们可以尝试利用python读入iris，来完成集成学习，分析其分类效果

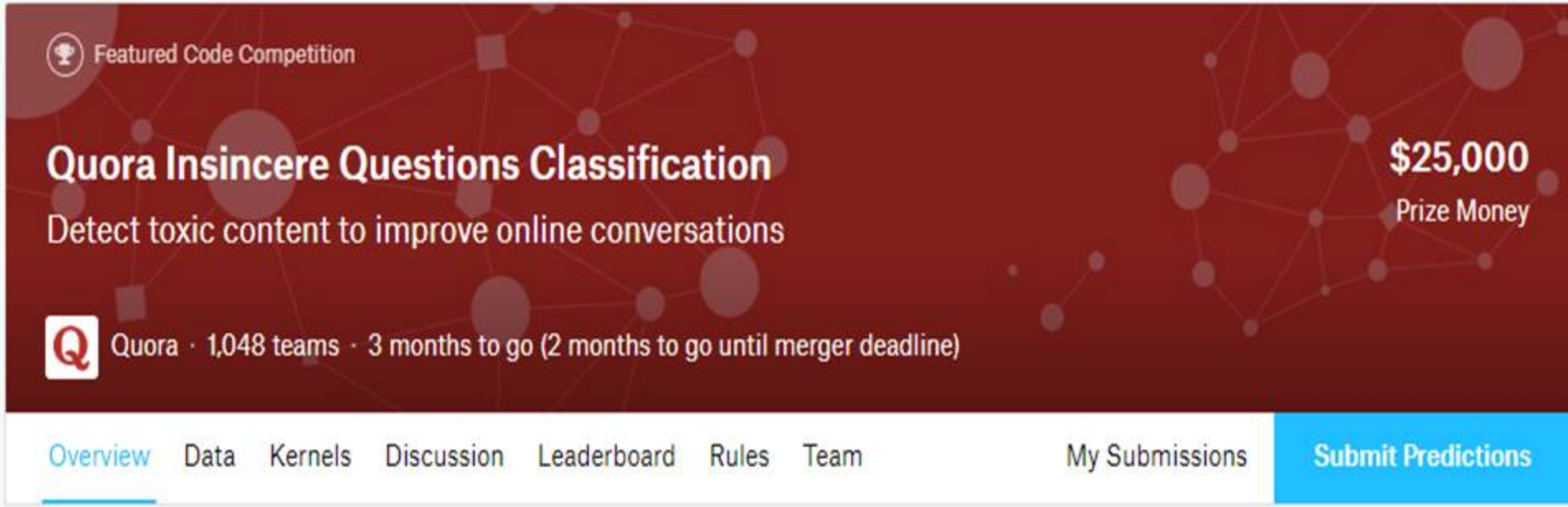
第14次课后作业

- 第十四次课后作业-在educoder平台上完成作业
- https://www.educoder.net/shixuns/xkv4b6yc/challenges

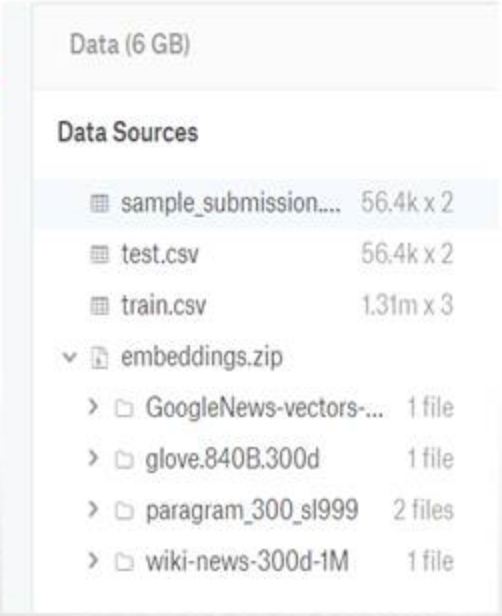
提交作业截至时间：2020年4月5日

1.问题描述

- 分析对象：Quora（问答网站挖掘网络上未有的维基知识）上被提出的问题
- 目的：判断问题是否真诚（而非有害信息、反动言论、有感表情色彩、荒谬、毁三观）
- 问题归纳：真诚的问题=0； 不真诚的问题=1 >>>二分类问题



1.问题描述



- 只能在kaggle服务器上运行
- 不能使用外部数据
- 官方提供部分embedding数据
- 根据F1-socre排名

Embeddings

External data sources are not allowed for this competition. We are, though, providing a number of word embeddings along with the dataset that can be used in the models. These are as follows:

- **GoogleNews-vectors-negative300** - https://code.google.com/archive/p/word2vec/
- **glove.840B.300d** - https://nlp.stanford.edu/projects/glove/
- **paragram_300_sl999** - https://cogcomp.org/page/resource_view/106
- **wiki-news-300d-1M** - https://fasttext.cc/docs/en/english-vectors.html

2.数据预处理

[32]:
```
train_df.head()
```

	qid	question_text	target
0	00002165364db923c7e6	How did Quebec nationalists see their province...	0
1	000032939017120e6e44	Do you have an adopted dog, how would you enco...	0
2	0000412ca6e4628ce2cf	Why does velocity affect time? Does velocity a...	0
3	000042bf85aa498cd78e	How did Otto von Guericke used the Magdeburg h...	0
4	0000455dfa3e01eae3af	Can I convert montra helicon D to a mountain b...	0

```
train_df = pd.read_csv("../input/train.csv")
test_df = pd.read_csv("../input/test.csv")
print("Train shape : ",train_df.shape)
print("Test shape : ",test_df.shape)
```

```
Train shape :  (1306122, 3)
Test shape :  (56370, 2)
```

[33]:
```
test_df.head()
```

	qid	question_text
0	00014894849d00ba98a9	My voice range is A2-C5. My chest voice goes u...
1	000156468431f09b3cae	How much does a tutor earn in Bangalore?
2	000227734433360e1aae	What are the best made pocket knives under $20...
3	0005e06fbe3045bd2a92	Why would they add a hypothetical scenario tha...
4	00068a0f7f41f50fc399	What is the dresscode for Techmahindra freshers?

```
print(train_df["question_text"].isnull().sum())
print(test_df["question_text"].isnull().sum())
```

```
0
0
```

2.数据预处理

- 将单词对应数字

```
word_index
```

```
{'the': 1,
 'what': 2,
 'is': 3,
 'a': 4,
 'to': 5,
 'in': 6,
 'of': 7,
 'i': 8,
 'how': 9,
 'and': 10,
 'do': 11,
 'are': 12,
 'for': 13,
 'you': 14
```

2.数据预处理

用到三个不同的embedding库进行embedding降维：

① GloVe: Global Vectors for Word Representation（emb1）

② wiki-news-300d-1M（emb2）

③ paragram_300_sl999（emb3）

④ emb_mean = (emb1+emb3)/2

◆ 将每个单词通过embedding拉成300维向量

```
seeEmd
```

```
array([[-0.13971121, -0.42888786,  0.48072671, ...,  0.201047  ,
         0.03283418, -0.39315751],
       [ 0.27204001, -0.06203   , -0.1884    , ...,  0.13015001,
        -0.18317001,  0.1323    ],
       [-0.038548  ,  0.54251999, -0.21843   , ...,  0.11798   ,
         0.24590001,  0.22872999],
       ...,
       [ 0.86807001,  0.069787  , -0.37619001, ..., -0.25782999,
        -0.66478997,  0.36126   ],
       [ 0.43889999,  0.98128003, -0.22822   , ..., -0.060725  ,
        -0.56114   ,  0.15267999],
       [ 0.43902001, -0.0081998 ,  0.51852   , ...,  0.21456   ,
         0.028362  , -0.0031905 ]])
```

```
seeEmd.shape
```

```
(95000, 300)
```

3.实验过程

A. 划分训练集：划分为子训练集和验证集（0.92：0.08）；

B. 用到模型：CNN　LSTM (attention机制) LR（线性回归）

C. 混合模型：相互独立的9个不同的CNN和LSTM（两种CNN和两种LSTM和不同的embedding方法进行混合得到）

D. 线性回归：LR对混合模型的输出与标签在验证集上回归

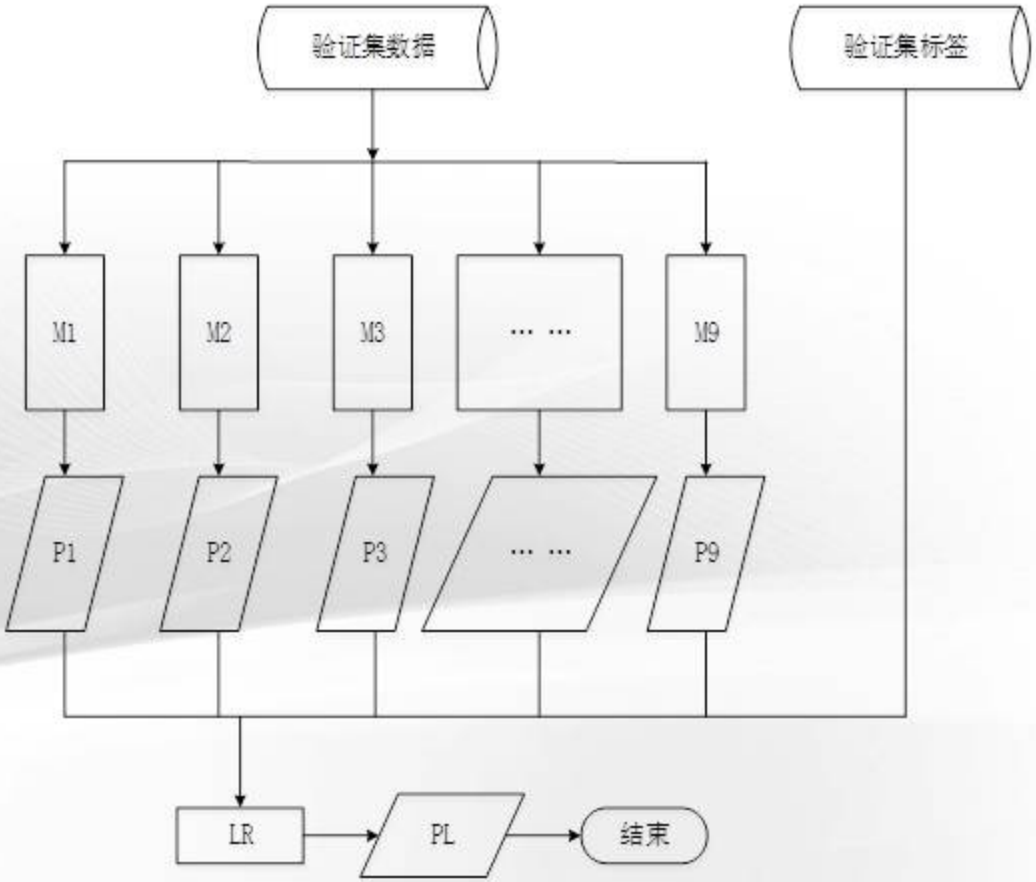
E. 寻找F1-socre最好的划分阈值

F. 对测试集按上述方法进行分类

3.实验过程

- 混合模型：
  输出0~1之间的小数

- 拟合线性回归模型：
  将各模型结果根据标签进行拟合
  $Y=ax_1+bx_2+\ldots+ix_9$



3.实验过程

- 选取最优阈值（0.1-0.501）

```
thresholds = []
for thresh in np.arange(0.1, 0.501, 0.01):
    thresh = np.round(thresh, 2)
    res = metrics.f1_score(val_y, (pred_val_y >
    thresholds.append([thresh, res])
    print("F1 score at threshold {0} is {1}".fo

thresholds.sort(key=lambda x: x[1], reverse=True
best_thresh = thresholds[0][0]
print("Best threshold: ", best_thresh)
```

```
F1 score at threshold 0.29 is 0.6803790803790805
F1 score at threshold 0.3 is 0.6807455176812416
F1 score at threshold 0.31 is 0.6800200300450676
F1 score at threshold 0.32 is 0.6791227817053817
F1 score at threshold 0.33 is 0.6779661016949153
F1 score at threshold 0.34 is 0.6778144481114778
F1 score at threshold 0.35 is 0.6768730123511575
F1 score at threshold 0.36 is 0.676663682482841
F1 score at threshold 0.37 is 0.6760457417995788
F1 score at threshold 0.38 is 0.6738073533880279
F1 score at threshold 0.39 is 0.6739663093415007
F1 score at threshold 0.4 is 0.6729952921200896
F1 score at threshold 0.41 is 0.670614821164186
F1 score at threshold 0.42 is 0.6687116564417178
F1 score at threshold 0.43 is 0.668569616238503
F1 score at threshold 0.44 is 0.6655475619504397
F1 score at threshold 0.45 is 0.6641947444784781
F1 score at threshold 0.46 is 0.6608029969867253
F1 score at threshold 0.47 is 0.6577258515714991
F1 score at threshold 0.48 is 0.6544820137908116
F1 score at threshold 0.49 is 0.652020794901895
F1 score at threshold 0.5 is 0.6471186440677967
Best threshold:  0.3
```