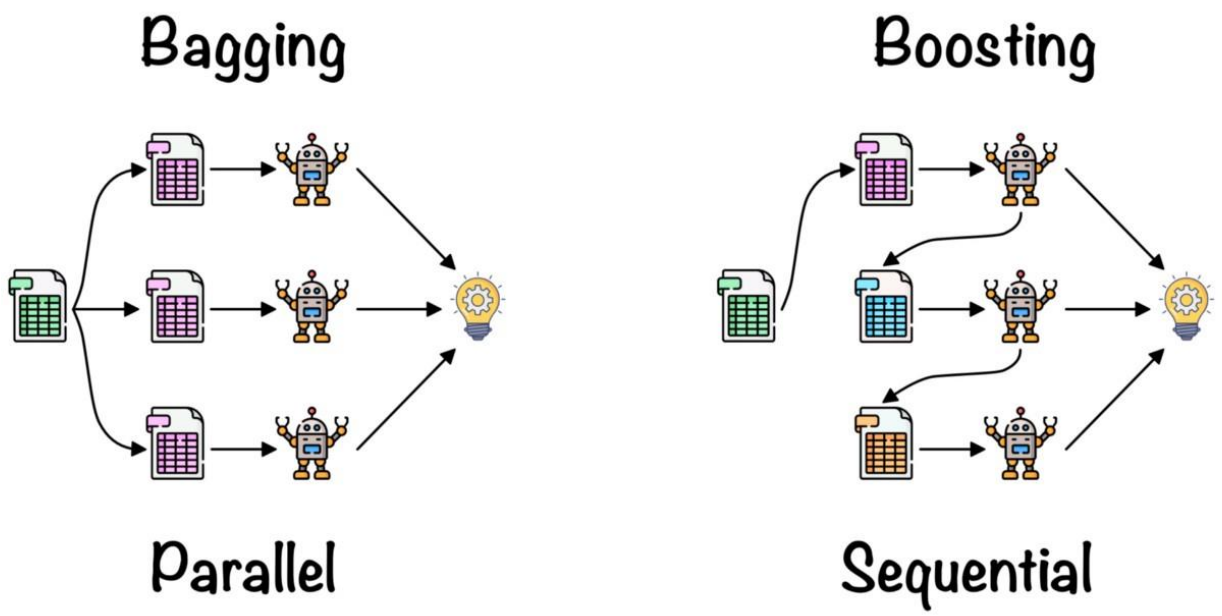
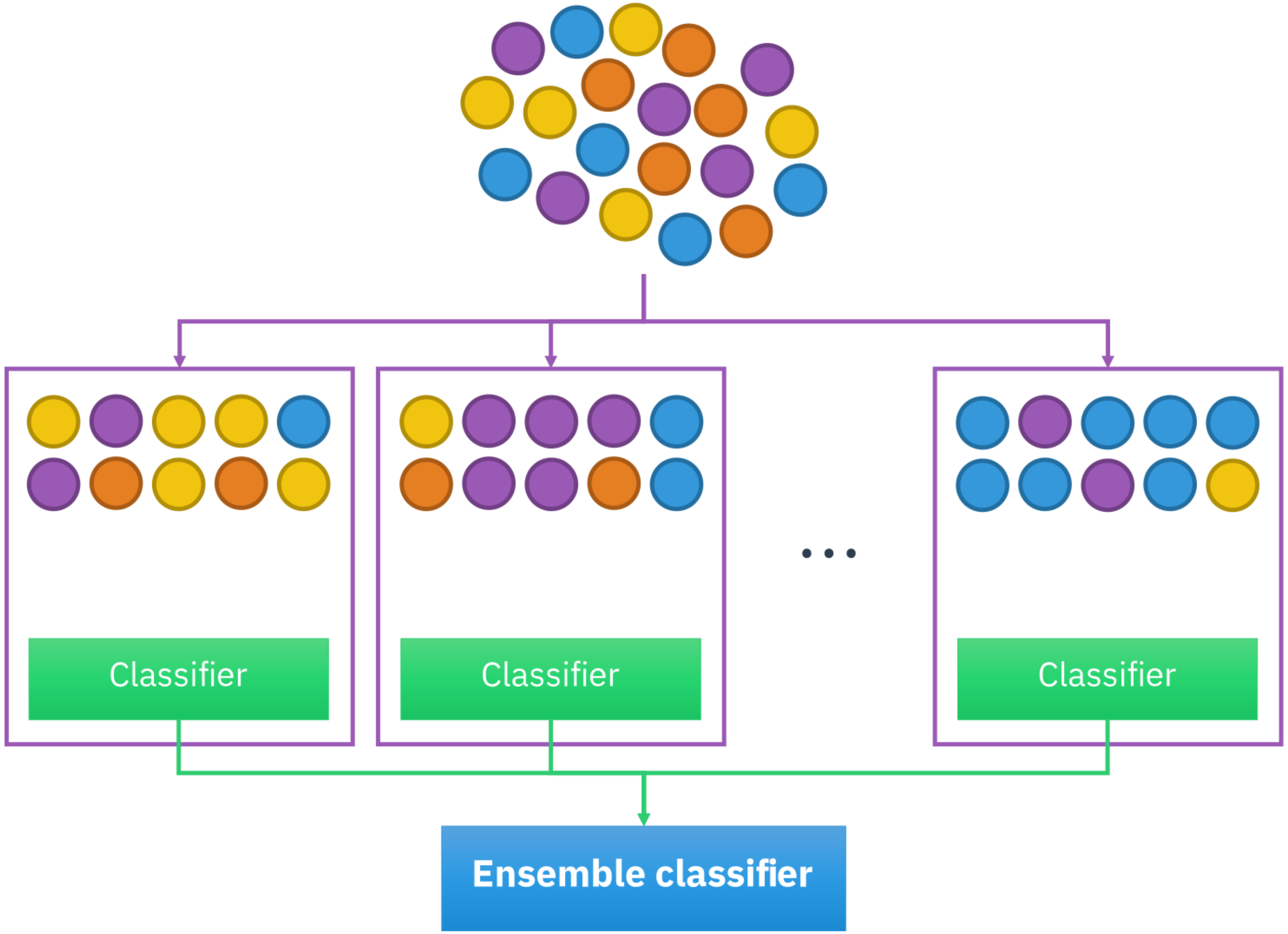
Bagging and Boosting



赵海涛

haitaozhao@ecust.edu.cn

开课一张图



Original Data

Bootstrapping

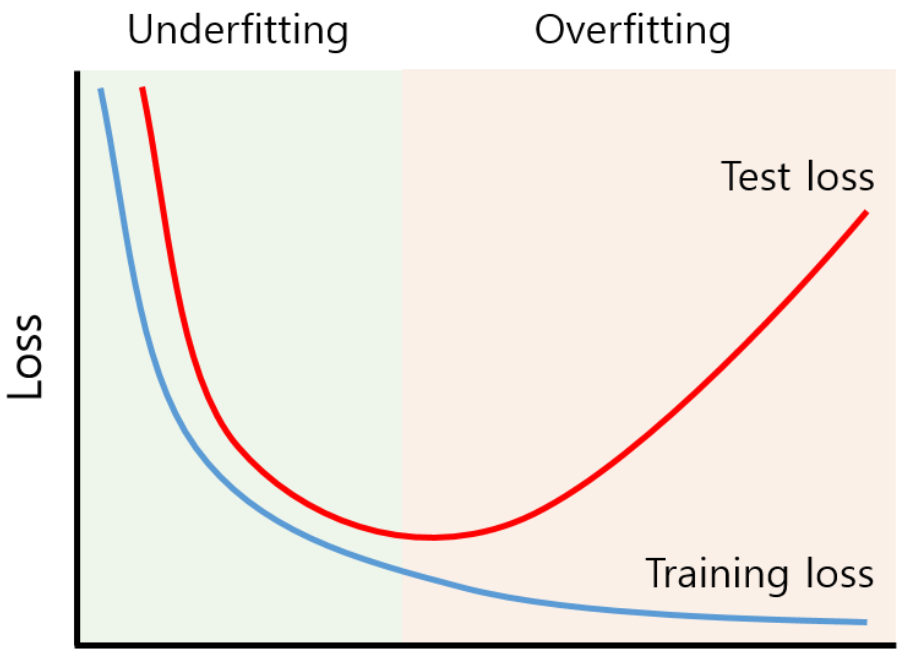
Aggregating

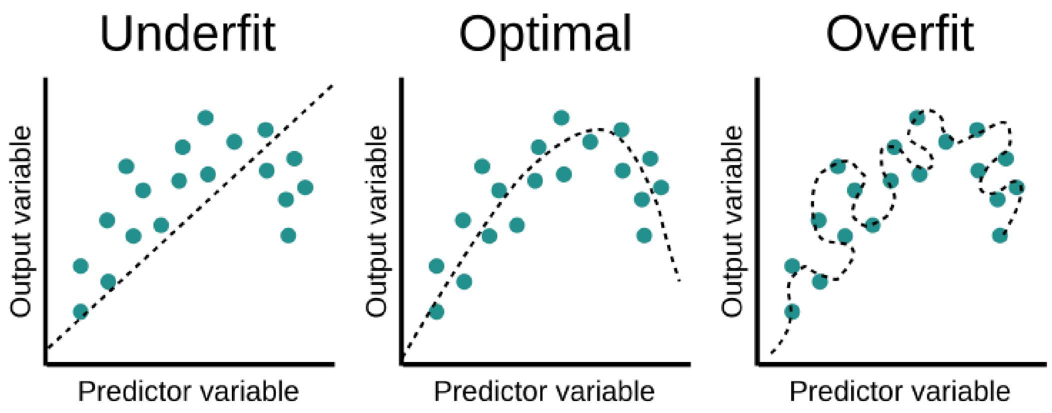
Bagging

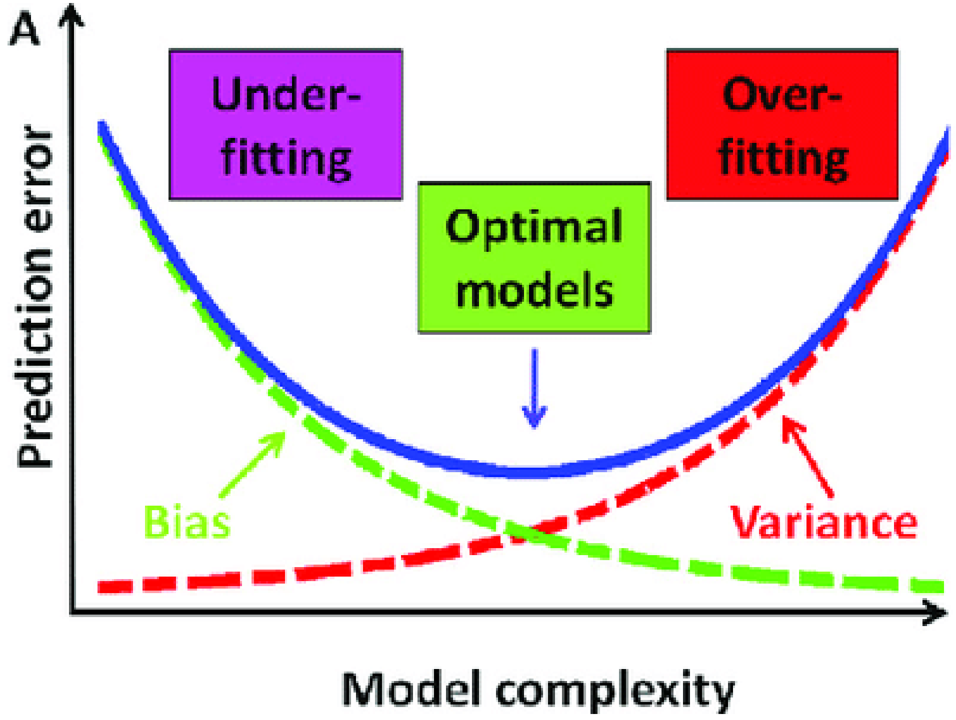
大纲

- 偏差与方差
- Bagging
- Boosting

回归的偏差和方差


Underfitting
Overfitting
Loss
Test loss
Training loss


Underfit
Optimal
Overfit
Output variable
Predictor variable


A
Prediction error
Under-fitting
Optimal models
Over-fitting
Bias
Variance
Model complexity

回归的偏差和方差

- 假定$\phi(x)$是一个未知函数，希望通过$\phi(x)$产生的N个样本的数据集$\mathcal{D}$来估计$\phi(x)$. 假设候选函数属于集合$\mathcal{F}$. 令$f(x;\mathcal{D})$是被估计函数，则$f(x;\mathcal{D}) \in \mathcal{F}$.
- 针对固定的N，对数据集$\mathcal{D}$求平均

$$E_{\mathcal{D}}[(f(x;\mathcal{D}) - \phi(x))^2] = \underbrace{(E_{\mathcal{D}}[f(x;\mathcal{D})] - \phi(x))^2}_{bias^2} + \underbrace{E_{\mathcal{D}}[(f(x;\mathcal{D}) - E_{\mathcal{D}}[f(x;\mathcal{D})])^2]}_{variance}.$$

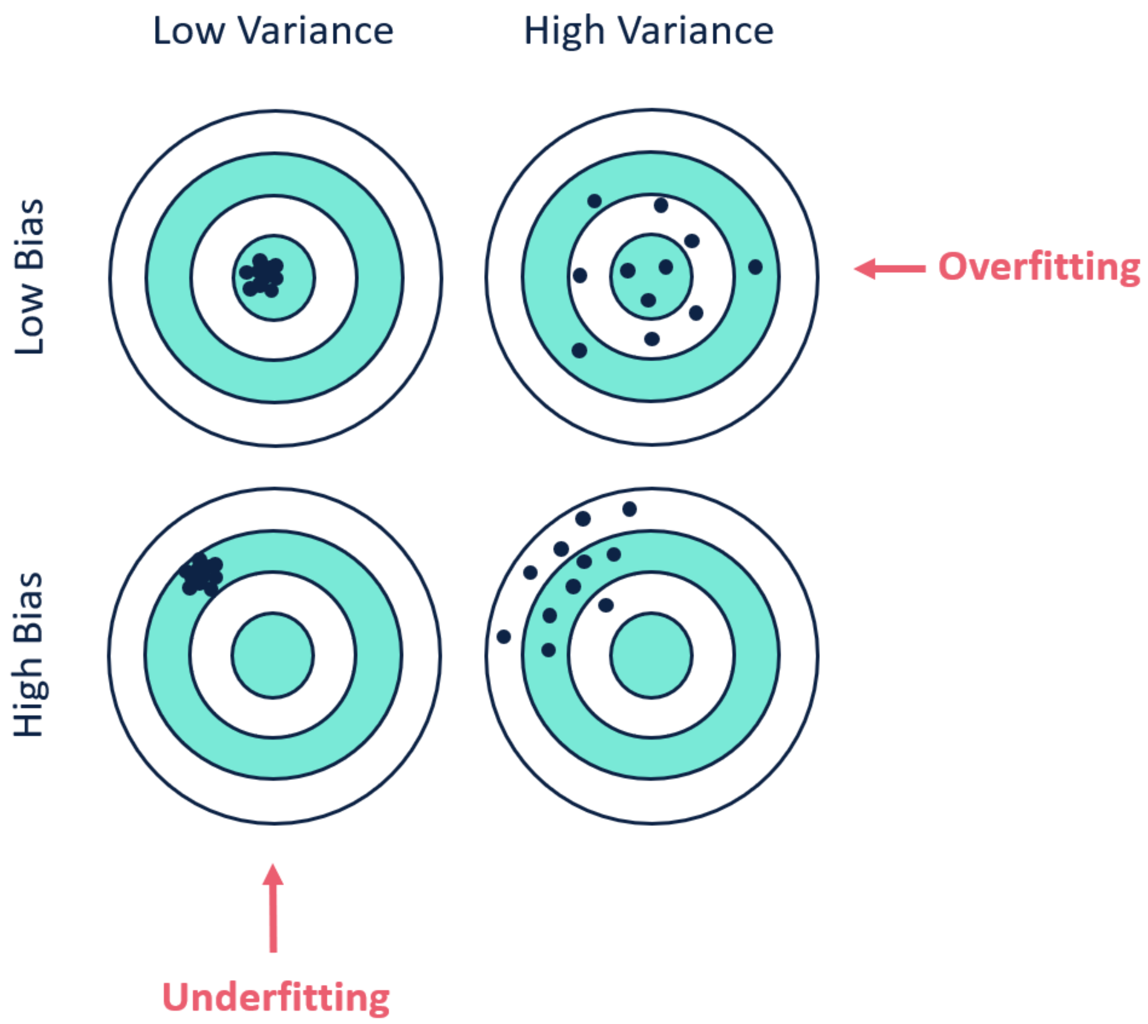
- 令$y = \phi(x) + \varepsilon$, 其中$E(\varepsilon) = 0, \mathrm{var}(\varepsilon) = \sigma_\varepsilon^2$. $\phi(x)$和噪声ε独立， 则

$$\begin{aligned} \mathrm{Error}\,(x) \quad &\triangleq E_{\mathcal{D}}[(y - f(x;\mathcal{D}))^2] \\ &= \sigma_\varepsilon^2 + bias^2 + variance. \end{aligned}$$

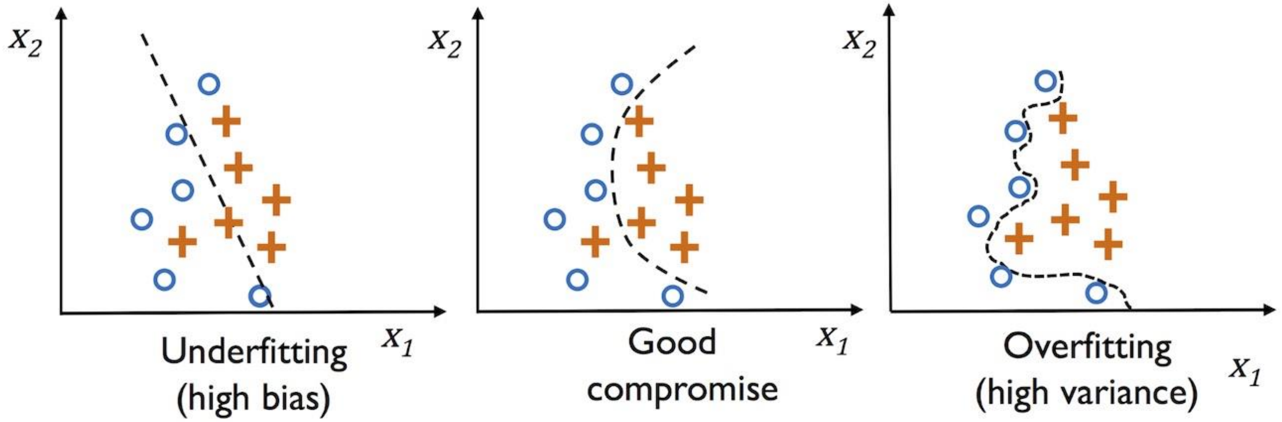
回归的偏差和方差

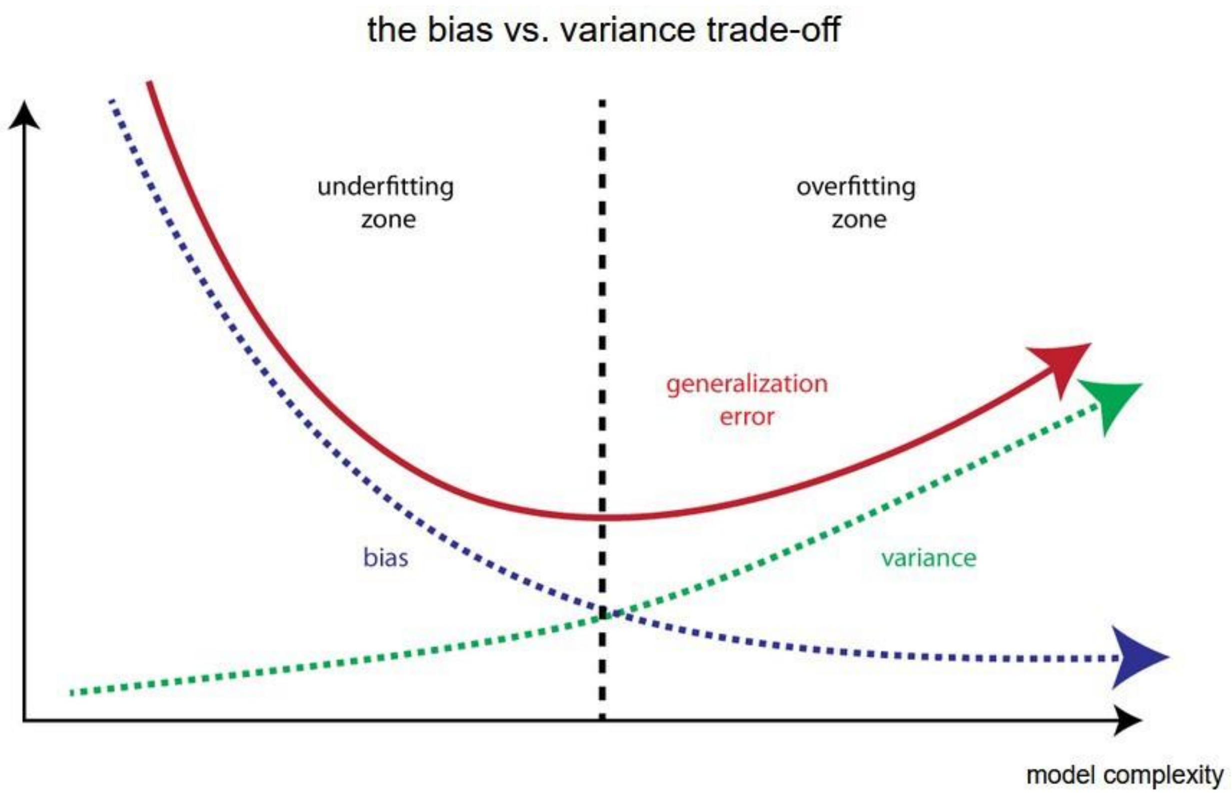
- 大体来说，被估计函数的泛化误差 是其在测试集上的误差期望
- 对一个固定的回归算法，针对一个新采样的训练集，被估计的函数f一般不相同。方差（*variance*） 表示 f 对训练集改变的敏感性
- 如果 $\mathcal{F}$ 中的函数都过于简单，则候选函数不能很好的拟合数据，产生偏差（*bias*）. 偏差可以是：
  - 确定性的 (hard): 没有 $f \in \mathcal{F}$ 能够拟合数据；
  - 随机的 (soft): $f \in \mathcal{F}$ 能够拟合数据的先验概率非常小

回归的偏差和方差


Low Variance
High Variance
Low Bias
High Bias
Overfitting
Underfitting

回归的偏差和方差


x_2
x_1
Underfitting
(high bias)
Good
compromise
Overfitting
(high variance)


the bias vs. variance trade-off
underfitting
zone
overfitting
zone
generalization
error
bias
variance
model complexity

如何估计方差

$$E_{\mathcal{D}}[(f(x;\mathcal{D}) - E_{\mathcal{D}}[f(x;\mathcal{D})])^2]$$

$variance$

- 数据独立同分布
- 如果可以得到大量不同的训练集将是非常好的。这种情况下，可以针对每个训练集进行估计，并对被估计函数的分布进行推断
- 通常额外的训练集并不容易获得

Bootstrap

- 通过从原始样本中重新抽样来估计估计量的抽样分布。[Efron, The Annals of Statistics, 1979]
- 抽样是根据经验分布进行的

$$
P \Rightarrow x_1, x_2, \cdots, x_n \xrightarrow{\hat{p}}
\begin{array}{cc}
\text{Samples} & \text{Estimator} \\
z_{11}, z_{12}, z_{13}, \cdots, z_{1n} & e_1 \\
z_{21}, z_{22}, z_{23}, \cdots, z_{2n} & e_2 \\
\vdots & \vdots \\
z_{m1}, z_{m2}, z_{m3}, \cdots, z_{mn} & e_m
\end{array}
$$

主要思想

- $(x_1, x_2, \cdots, x_n) \sim P$. 注意: 分布函数$P$ 是未知的
- 采样 m 个数据集 $Y_1, Y_2, \cdots, Y_m$. $Y_i = (z_{i1}, z_{i2}, \cdots, z_{in})$ 包含 n 个从经验分布中抽取的样本，训练集的经验分布为:

$$\Pr[z_{jk} = x_i] = \frac{\#x_i}{n}$$

其中 $\#x_i$ 表示x_i 在原始训练集中出现的次数

- $Y_i \sim \hat{P}$
- $\hat{P} = P$ 在理论上并不能保证。但 $\hat{P}$ 是 P 的一个近似。

Bagging

- Bagging 可以减少方差
- 假设我们要估计的是某个参数，参数为 θ，并假定 $f_{ag}=E_P\hat{f}^*(x)$.

  估计函数 $\hat{f}^*(x)$ 是基于样本集 $Y^*=(z_1^*,z_2^*,\cdots,z_n^*)\sim P$，则

$$
\begin{aligned}
E_P[\theta-\hat{f}^*(x)]^2 &= E_P[\theta-f_{\mathrm{ag}}+f_{\mathrm{ag}}-\hat{f}^*(x)]^2\\
&= E_P[\theta-f_{\mathrm{ag}}]^2+E_P[f_{\mathrm{ag}}-\hat{f}^*(x)]^2\\
&\geq E_P[\theta-f_{\mathrm{ag}}]^2
\end{aligned}
$$

Bagging Averages (Bootstrap aggregation)

- 假设我们有独立的训练集 $\mathcal{D}_1, \mathcal{D}_2, \cdots, \mathcal{D}_M$. g_i 是在 $\mathcal{D}_i$ $(i = 1,2,\cdots,M)$ 上得到的分类器，令

$$f = E[g] = \frac{1}{M}\sum_{k=1}^{M} g_i\,.$$

- 平均可以得到更小的方差
- Bagging的步骤（考虑二分类）：
  - 针对训练集$\mathcal{D}$，独立采样 $\mathcal{D}_1, \cdots, \mathcal{D}_M$
  - 在 $\mathcal{D}_i$ $(i = 1,2,\cdots,M)$上分别估计g_i
  - 得到最终的分类器

$$f(x) = \operatorname{sign}\frac{1}{M}\sum_{k=1}^{M} g_i(x)$$

Example: 随机森林

Algorithm 15.1 *Random Forest for Regression or Classification.*

1. For $b = 1$ to B:

   (a) Draw a bootstrap sample $\mathbf{Z}^*$ of size N from the training data.

   (b) Grow a random-forest tree T_b to the bootstrapped data, by recursively repeating the following steps for each terminal node of the tree, until the minimum node size n_{min} is reached.

      i. Select m variables at random from the p variables.

      ii. Pick the best variable/split-point among the m.

      iii. Split the node into two daughter nodes.

2. Output the ensemble of trees $\{T_b\}_1^B$.

To make a prediction at a new point x:

Regression: $\hat{f}_{\text{rf}}^B(x) = \frac{1}{B}\sum_{b=1}^B T_b(x)$.

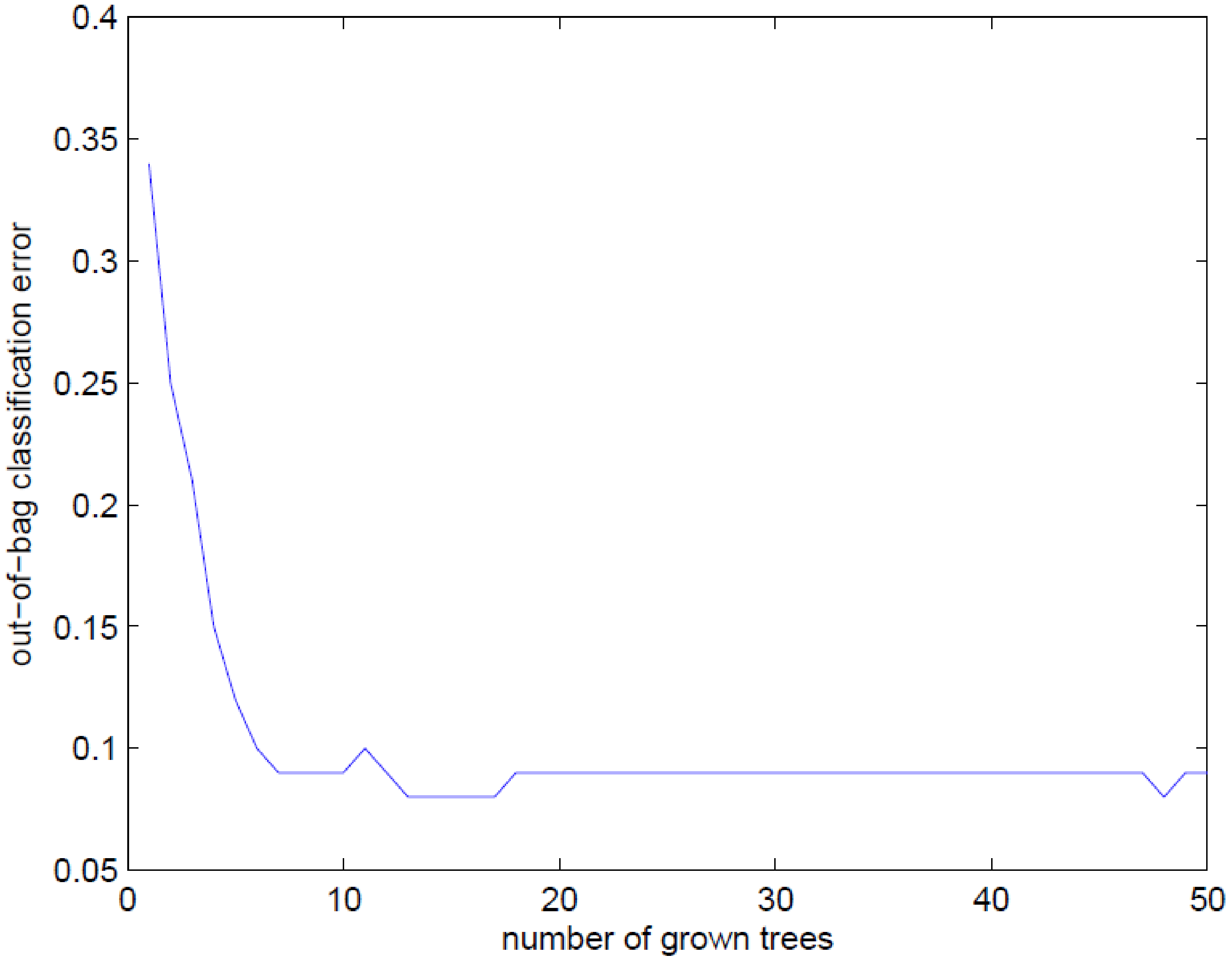
Classification: Let $\hat{C}_b(x)$ be the class prediction of the bth random-forest tree. Then $\hat{C}_{\text{rf}}^B(x) = \textit{majority vote } \{\hat{C}_b(x)\}_1^B$.

- The Elements of Statistical Learning. T. Hastie, R. Tibshirani and J. H. Friedman.

随机森林 (MATLAB Code)

```
%% bagged decision tree
load iris; Data = X;
[Projection, Projection2, rs] = PCA_f(3, 50, Data, ...
Data);
trn.X = Projection(1:2,51:150);
trn.y =[ones(1,50) 2*ones(1,50)];
% random forest
b = TreeBagger(50,trn.X',trn.y,'OOBPred','on')
plot(oobError(b))
% evaluate the tree classifier
b = TreeBagger(15,trn.X',trn.y,'OOBPred','on')
[sfit,scores] = predict(b,trn.X'); % Find ...
assigned class numbers
y_pred = str2num(cell2mat(sfit));
crosstab(y(51:150),y_pred)
```

随机森林 (MATLAB Plot)



怎样组合弱分类器

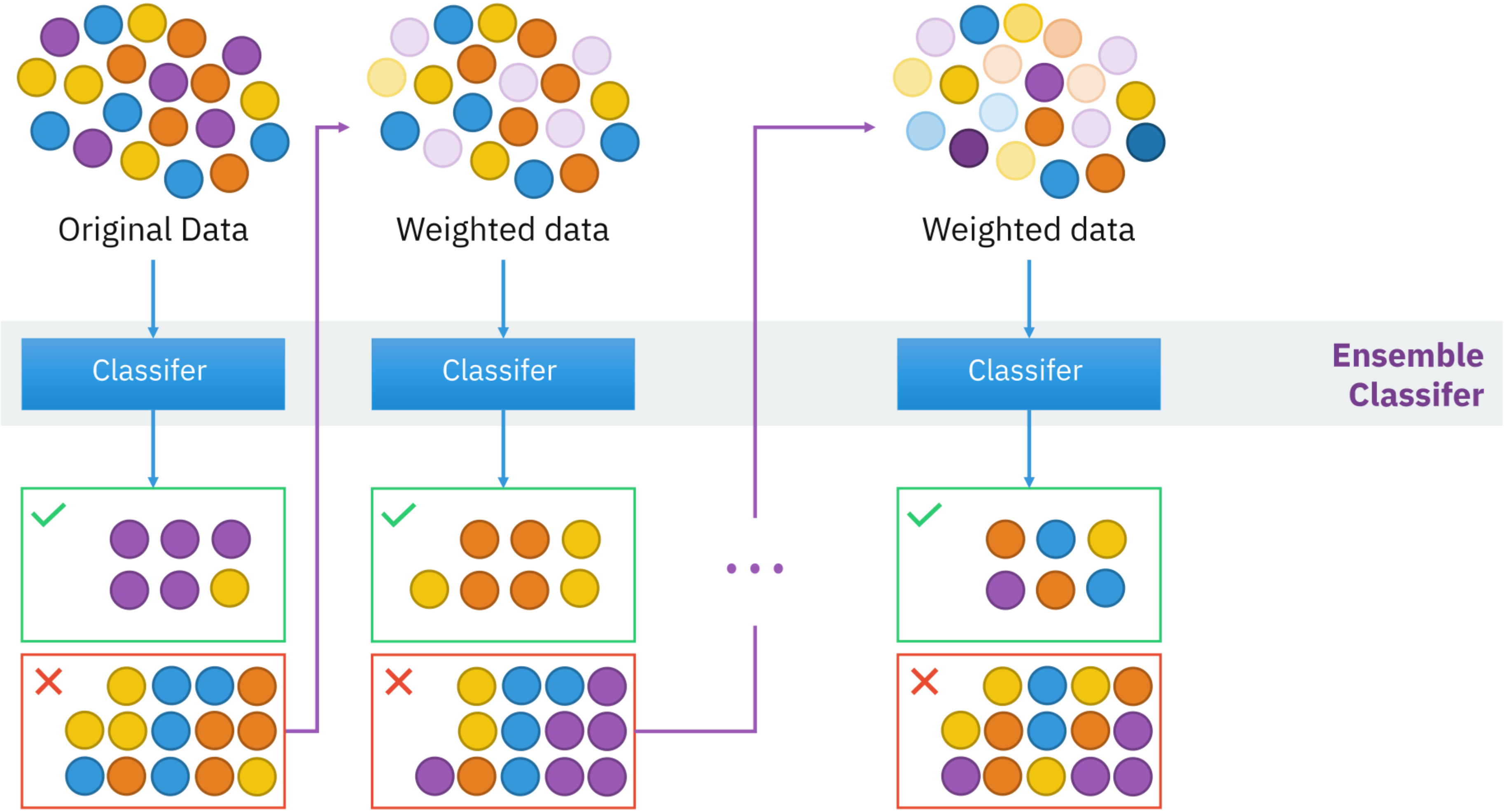
- 多专家组合

一种并行结构，所有的弱分类器都给出各自的预测结果，通过“组合器”把这些预测结果转换为最终结果。 eg. 投票（voting）及其变种、混合专家模型

- 多级组合

一种串行结构，其中下一个分类器只在前一个分类器预测不够准（不够自信）的实例上进行训练或检测。 eg. 级联算法（cascading）

另一张图



Original Data
Weighted data
Weighted data
Classifer
Classifer
Classifer
Ensemble Classifer

提升（Boosting）

$$(E_{\mathcal{D}}[f(x;\mathcal{D})] - \phi(x))^2$$

$bias^2$

- Boosting 可以减小偏差.
- 基分类器族$\mathcal{G}$ 有很大的偏差（例如线性分类器、决策树桩），但学习总是产生分类器 g 比随机猜测更好（在训练集上）。
- Boosting的先决条件:
  - 对于$g \in \mathcal{G}$,，g的训练错误率有上界，且$0 < \widehat{\mathrm{Error}}(g) \leq \delta < \frac{1}{2}$
  - 可以在加权的数据上进行训练

Boosting先决条件

$$\left.\begin{aligned} h_1(x) &\in \{-1,+1\} \\ h_2(x) &\in \{-1,+1\} \\ &\vdots \\ h_T(x) &\in \{-1,+1\} \end{aligned}\right\}$$

Weak classifiers

slightly better than random

$$H_T(x) = sign\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$

strong classifier

Adaboost基本概念

两个问题如何解决：

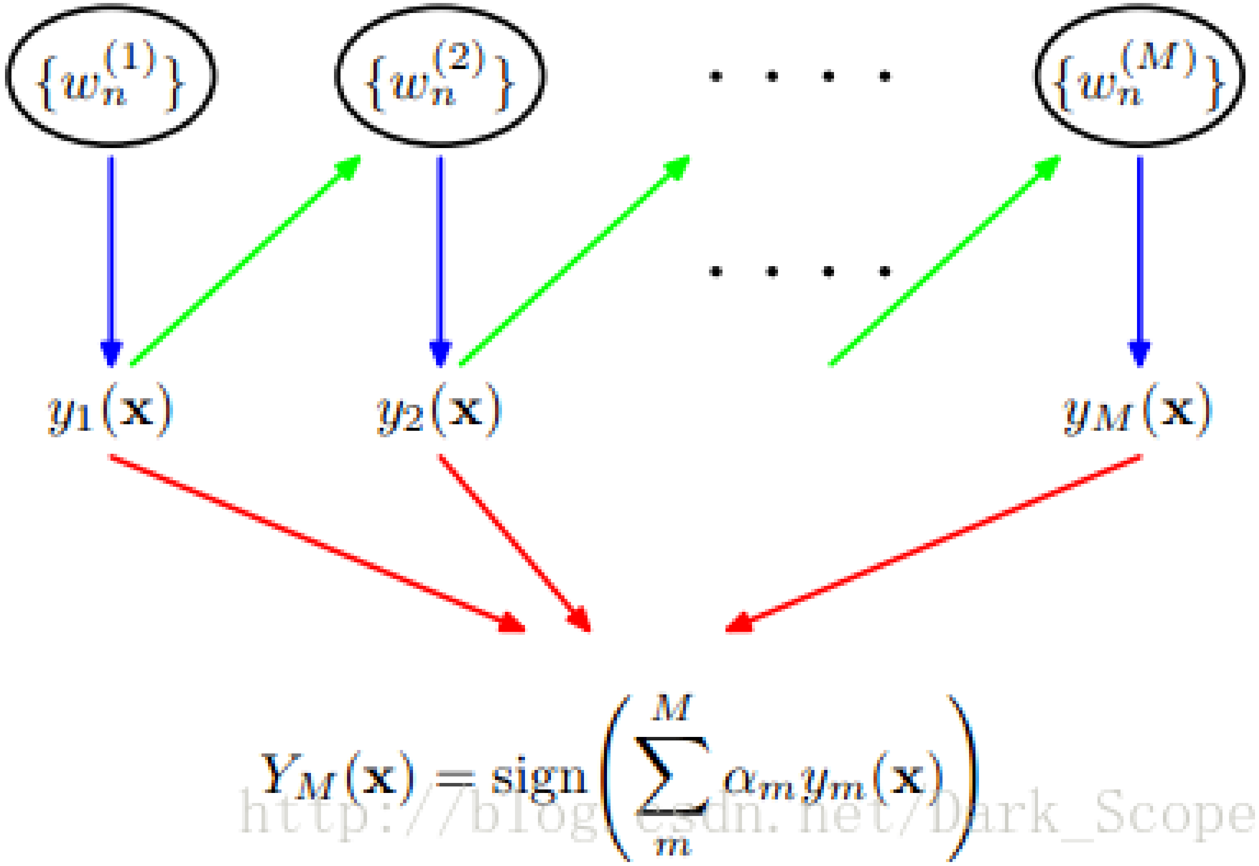
每一轮如何改变训练数据的权值或概率分布？

AdaBoost：提高那些被前一轮弱分类器错误分类样本的权值，降低那些被正确分类样本的权值

如何将弱分类器组合成一个强分类器？

AdaBoost：加权多数表决，加大分类误差率小的弱分类器的权值，使其在表决中起较大的作用，减小分类误差率大的弱分类器的权值，使其在表决中起较小的作用。

Adaboost基本概念



Adaboost.M1 (Freund and Schapire)

- Initialize the observation weights $w_i = \frac{1}{N}, i = 1,2, \ldots, N$
- For $m = 1 \ to\ M$:
- Fit a classifier $g_m(x)$ to the training data using weights w_i .
- Compute

$$\text{err}_m = \sum_{i=1}^{N} w_i I(y_i \neq g_m(x_i))$$

- Compute

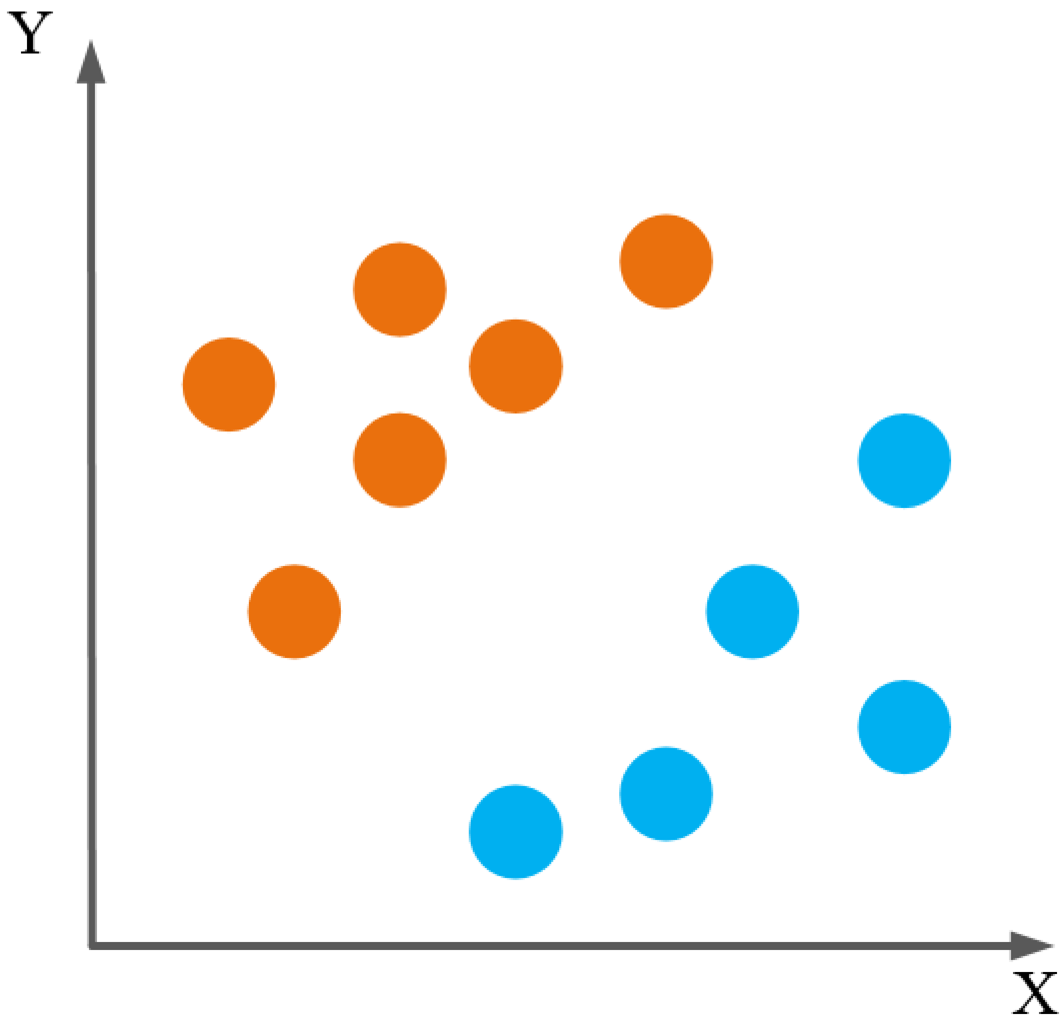
$$\boxed{\alpha_m} = \frac{1}{2}\ln[\frac{1 - err_m}{err_m}]$$

- Set $\boxed{w_i} \leftarrow \frac{w_i \exp[-\alpha_m y_i g_m(x_i)]}{\sum_{i=1}^{N} w_i \exp[-\alpha_m y_i g_m(x_i)]}$
- Output

$$H_g(x) = \text{sign}[\sum_{m=1}^{M} \alpha_m g_m(x)]$$

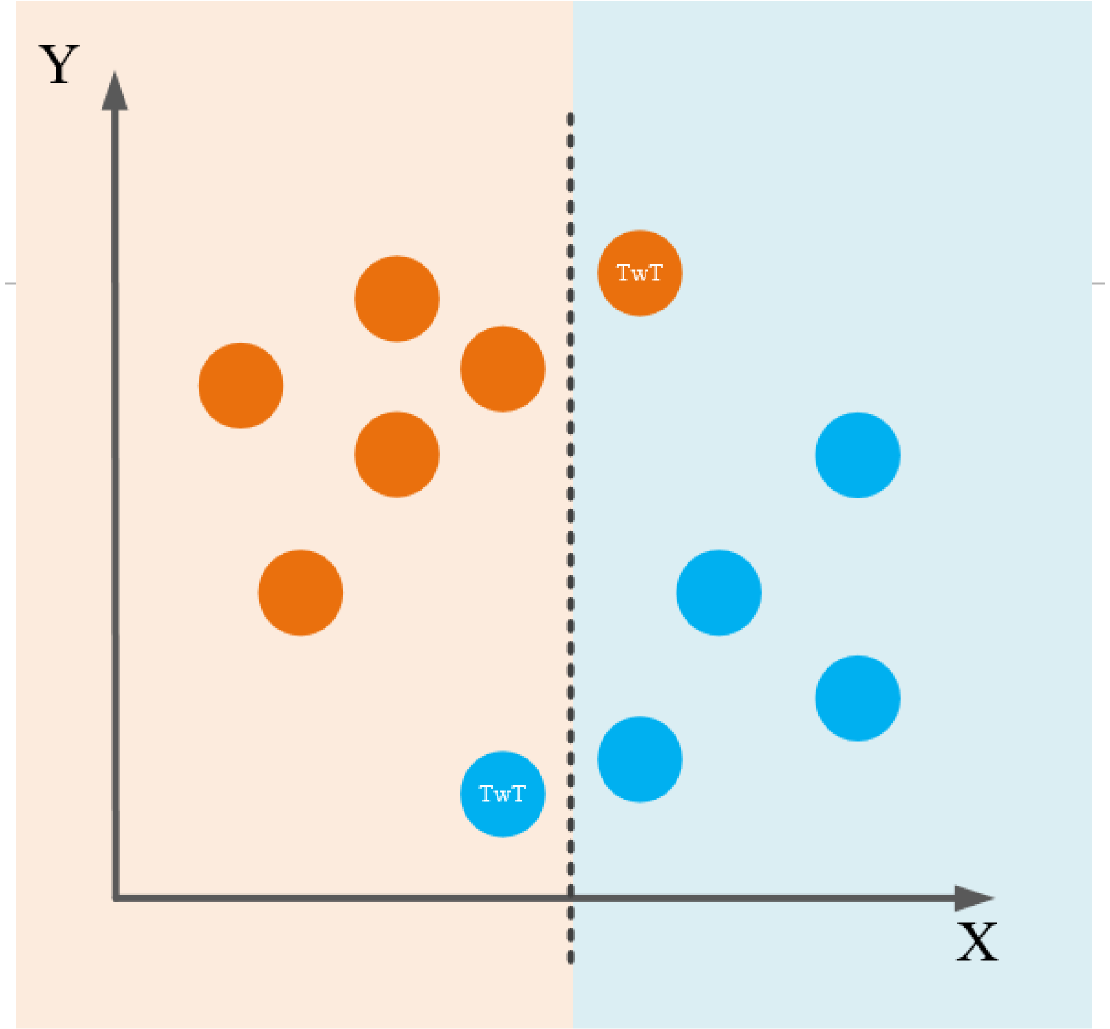
Adaboost图解

用决策桩（就是阈值判别器）去划分平面上红色、蓝色小球，初始状态是这样的：

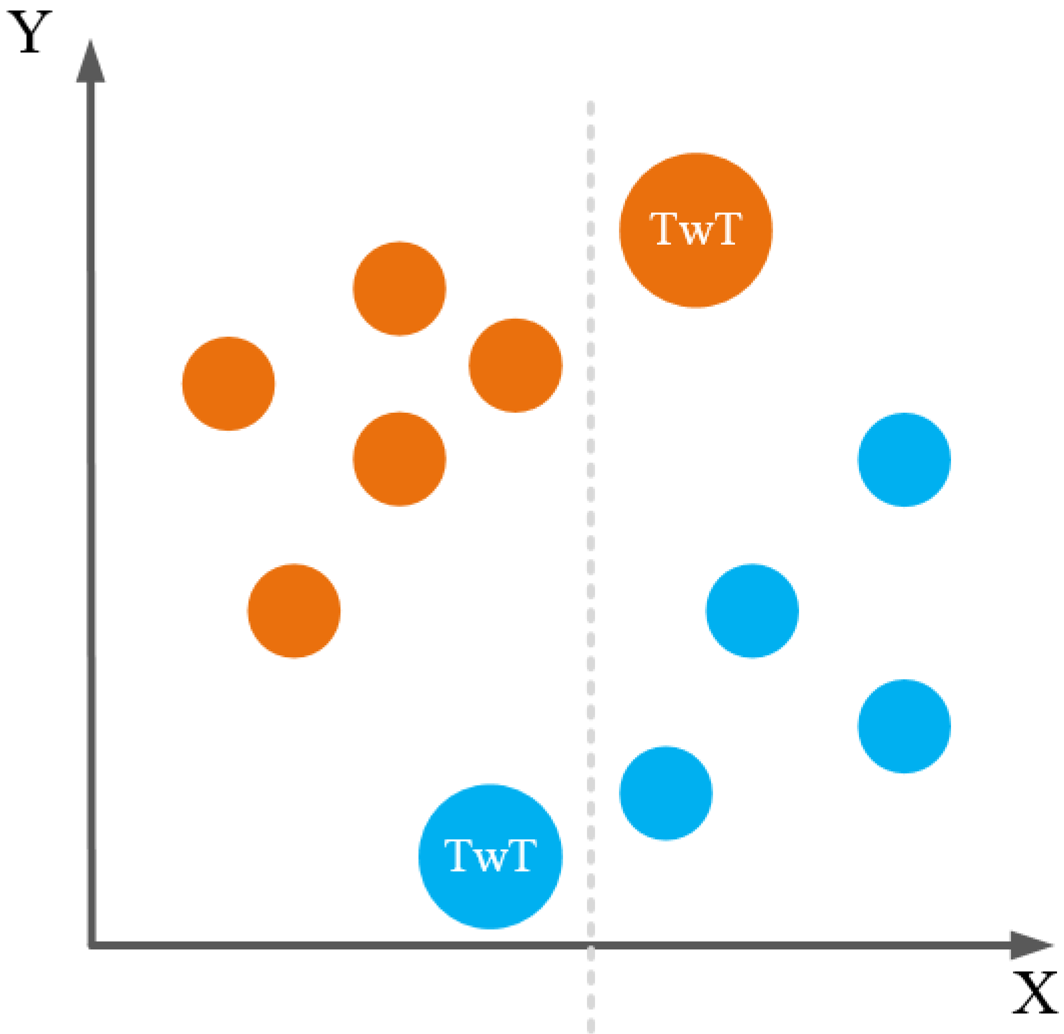


Adaboost

然后第一个桩分类器竖着划分，错分了两个：

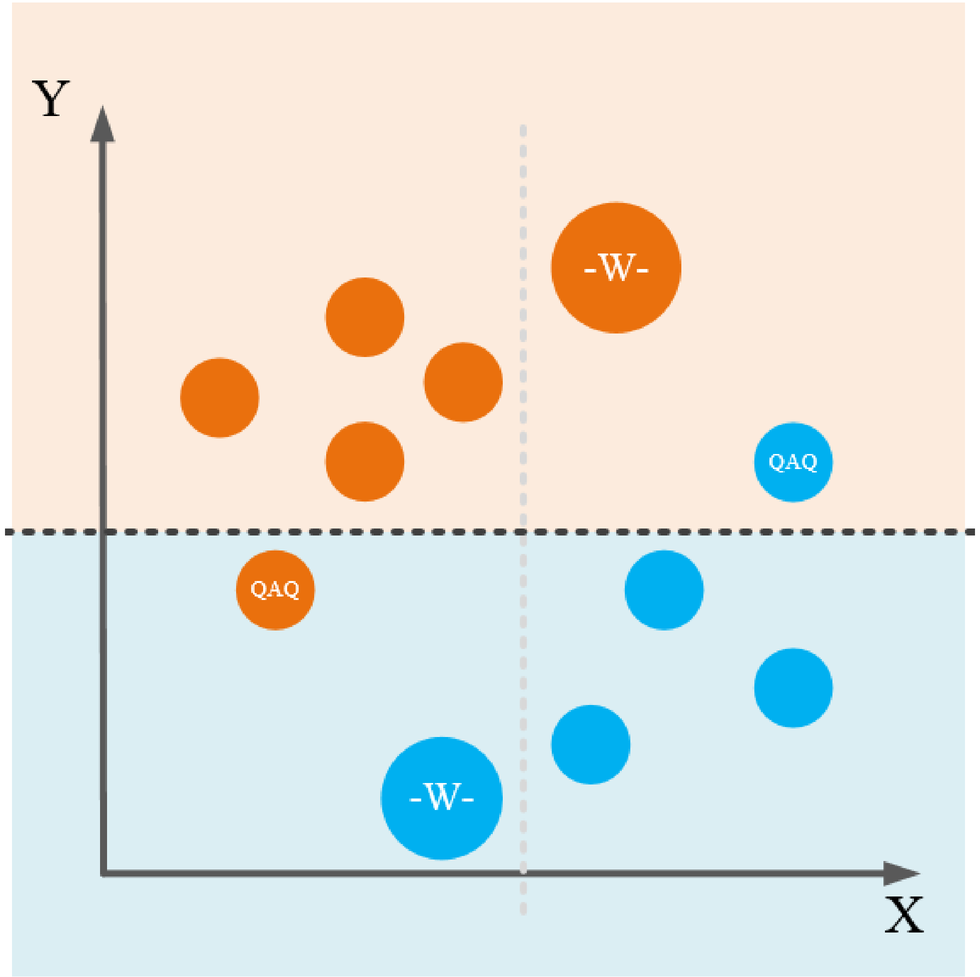


于是在重新计算分布后，它们的权重变大了：

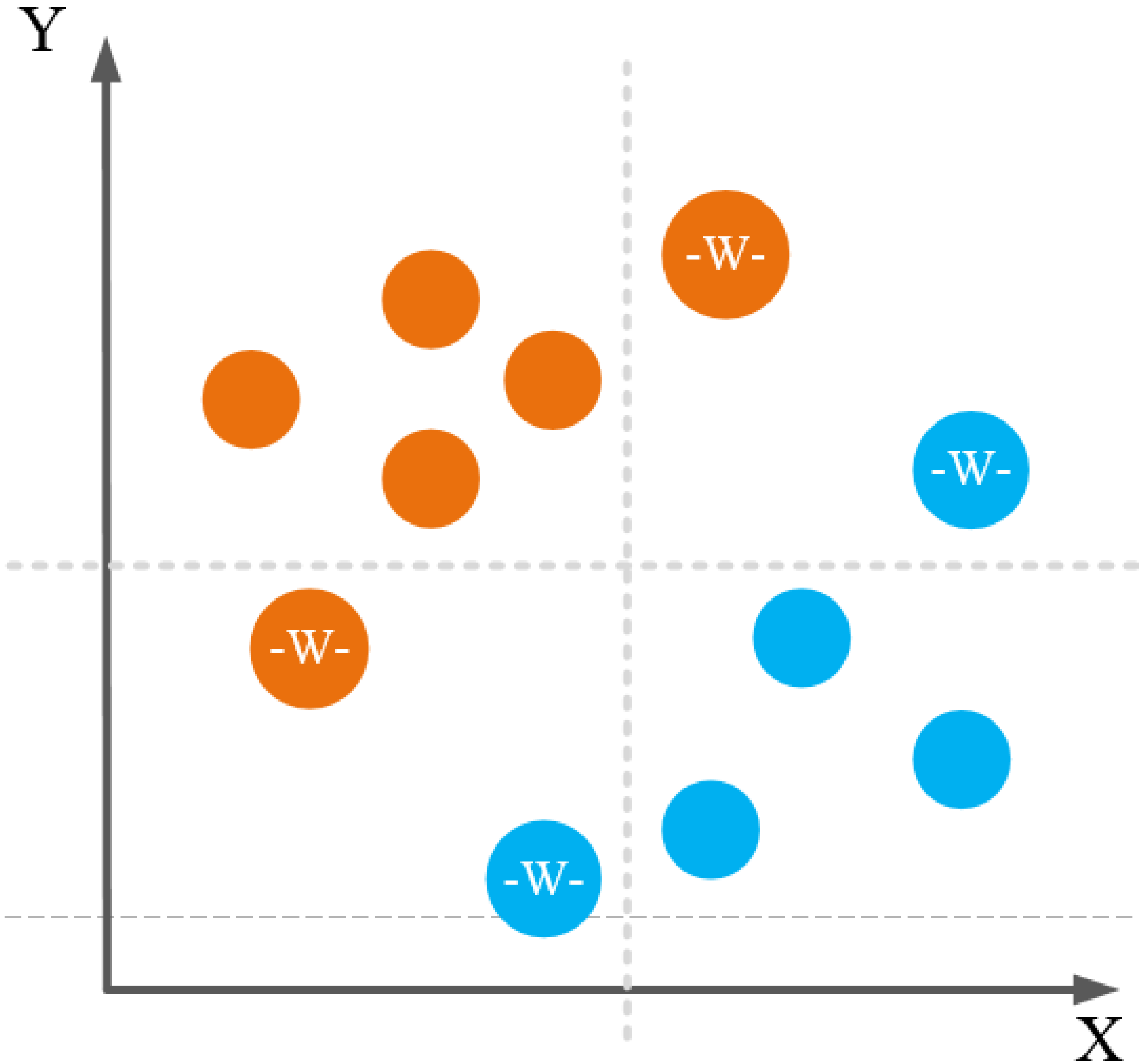


Adaboost

第二个分类器更多的考虑了被错分的样本，
然而第一次被分对的样本又被分错了两个：

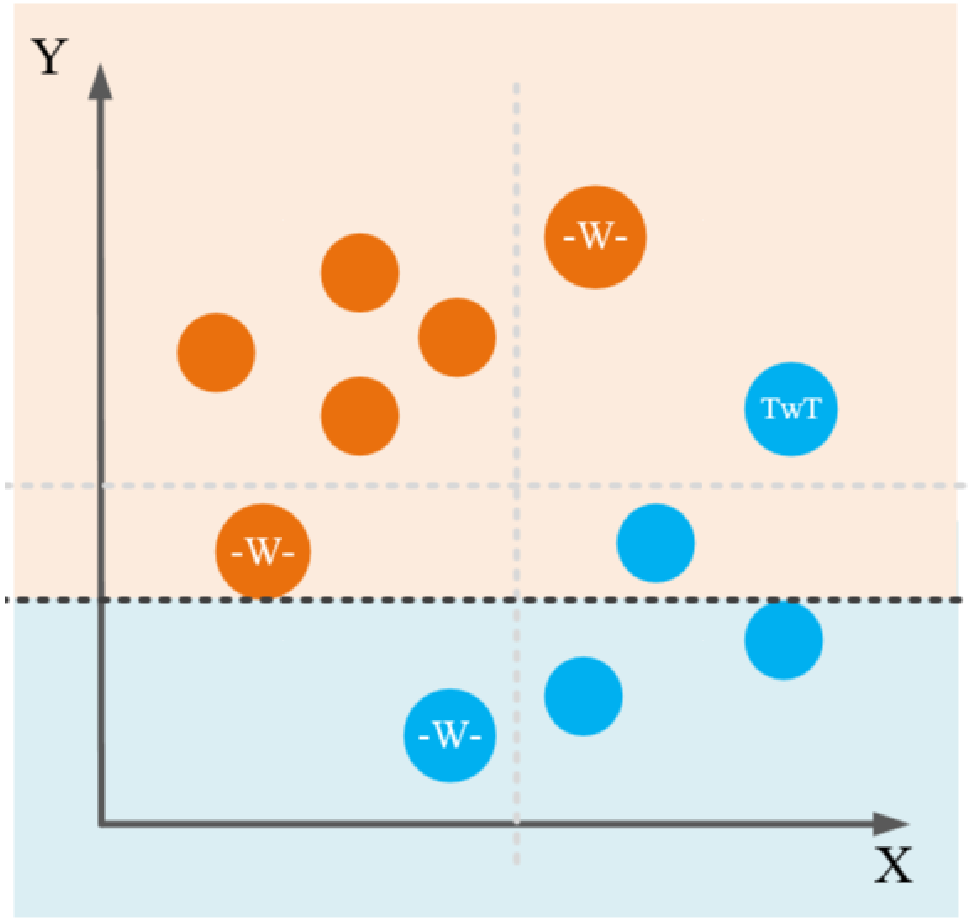


于是再次进行权重调整：

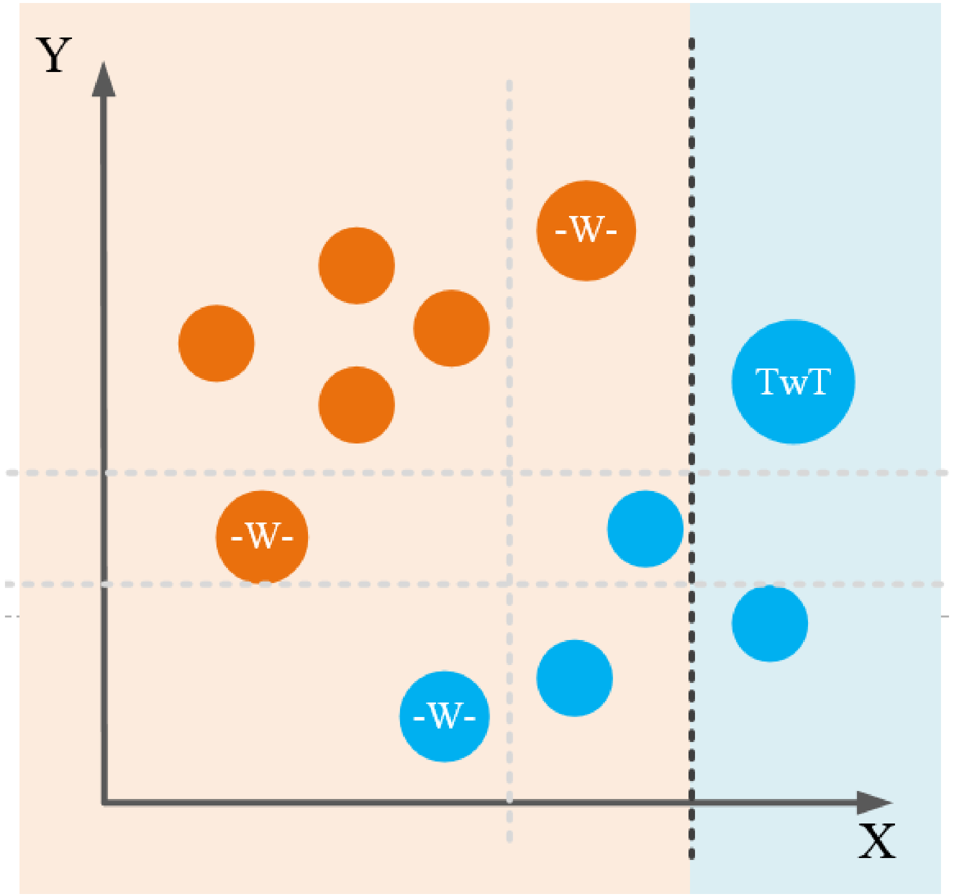


Adaboost

第三次继续划分：

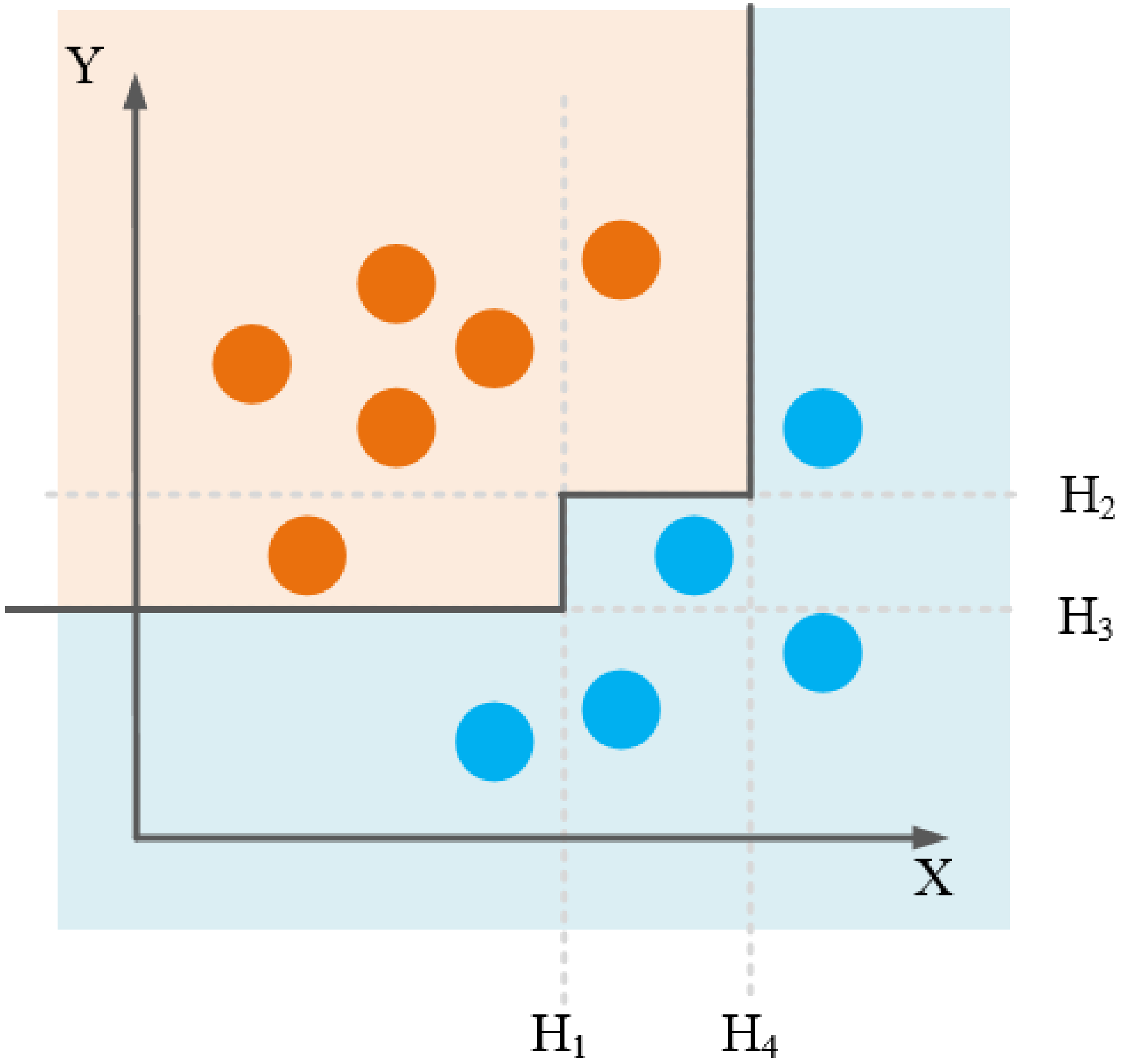


第三次权重调整+第四次继续划分：

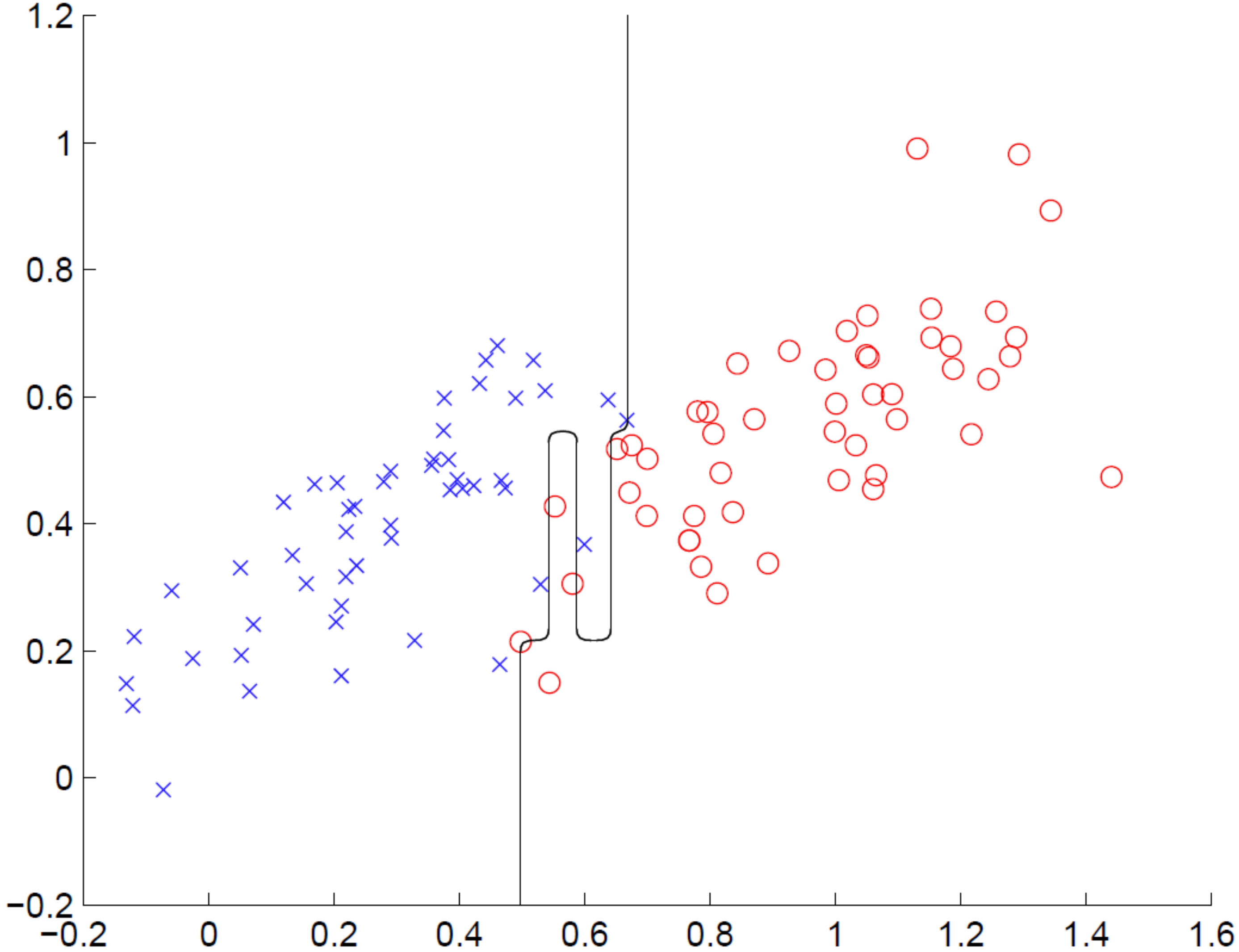


Adaboost

最后得到强分类器的决策平面：



Adaboost (MATLAB plot)



Justification

- 令 $Z_m = \sum_{i=1}^{N} w_i^{(m-1)} \exp[-\alpha_m y_i g_m(x_i)], w_i^{(0)} = \frac{1}{N} (i = 1,2,\cdots,N)$,

  $w_i^{(m)} = \frac{w_i^{(m-1)} \exp[-\alpha_m y_i g_m(x_i)]}{Z_m}, G(x) = \sum_{m=1}^{M} \alpha_m\, g_m(x)$.

- 对于训练集上的错误率 $R_{\mathrm{tr}}(H_G) = \frac{1}{N}\sum_{i=1}^{N} |\{i: y_i \neq H_G(x_i)\}|$, 有

$$
\begin{aligned}
R_{\mathrm{tr}}(H_G) &= \frac{1}{N}\sum_{i=1}^{N} |\{i: y_i \neq H_G(x_i)\}| = \frac{1}{N}\sum_{i=1}^{N} \begin{cases} 1 & \text{if } y_i \neq H_G(x_i) \\ 0 & \text{else} \end{cases} \\
&= \frac{1}{N}\sum_{i=1}^{N} \begin{cases} 1 & \text{if } y_i G(x_i) \leq 0 \\ 0 & \text{else} \end{cases} \\
&\leq \frac{1}{N}\sum_{i=1}^{N} \exp[-y_i G(x_i)] \\
&= \frac{1}{N}\sum_{i=1}^{N} \exp[-y_i \sum_{j=1}^{m} \alpha_j g_j(x_i)] = \prod_{j=1}^{m} Z_j
\end{aligned}
$$

Justification

- The training error $R_{tr}(H_G)$ is bounded by $\prod_{j=1}^{m} Z_j$. In order to minimize the training error $R_{tr}(H_G)$, we can try to minimize Z_m.

$$
\begin{aligned}
\frac{\partial Z_m}{\partial \alpha_m} &= \frac{\partial \sum_{i=1}^{N} w_i^{(m-1)} \exp[-\alpha_m y_i g_m(x_i)]}{\partial \alpha_m} \\
&= -\sum_{i=1}^{N} w_i^{(m-1)} y_i g_m(x_i) \exp[-\alpha_m y_i g_m(x_i)] \\
&= -\exp(-\alpha_m) \sum_{y_i = g_m(x_i)} w_i^{(m-1)} + \exp(\alpha_m) \sum_{y_i \neq g_m(x_i)} w_i^{(m-1)}.
\end{aligned}
$$

Justification

- From $\frac{\partial Z_m}{\partial \alpha_m} = 0$, we have

$$\exp(-\alpha_m) \sum_{y_i = g_m(x_i)} w_i^{(m-1)} = \exp(\alpha_m) \sum_{y_i \neq g_m(x_i)} w_i^{(m-1)}$$

$$\alpha_m = \frac{1}{2} \ln \frac{\sum_{y_i = g_m(x_i)} w_i^{(m-1)}}{\sum_{y_i \neq g_m(x_i)} w_i^{(m-1)}}$$

- Let $\mathrm{err}_m = \sum_{y_i \neq g_m(x_i)} w_i^{(m-1)}$. Then $\sum_{y_i = g_m(x_i)} w_i^{(m-1)} = 1 - \mathrm{err}_m$

And

$$\alpha_m = \frac{1}{2} \ln \frac{1 - \mathrm{err}_m}{\mathrm{err}_m}$$

Justification

- With $\alpha_m = \frac{1}{2}\ln\frac{1-\mathrm{err}_m}{\mathrm{err}_m}$ in the following we show that the training error $R_{tr}(H_G)$ decreases exponentially with M.

$$
\begin{aligned}
Z_m &= \sum_{i=1}^{N} w_i^{(m-1)} \exp[-\alpha_m y_i g_m(x_i)] \\
&= \exp(-\alpha_m) \sum_{y_i = g_m(x_i)} w_i^{(m-1)} + \exp(\alpha_m) \sum_{y_i \neq g_m(x_i)} w_i^{(m-1)} \\
&= (1-\mathrm{err}_{m-1})\sqrt{\frac{\mathrm{err}_m}{1-\mathrm{err}_m}} + \mathrm{err}_{m-1}\sqrt{\frac{1-\mathrm{err}_m}{\mathrm{err}_m}} \\
&= 2\sqrt{(1-\mathrm{err}_m)\mathrm{err}_m}.
\end{aligned}
$$

Justification

- Let $\text{err}_m = \frac{1}{2} - \gamma_m$. Since the training error of g on $\mathcal{G}$ is bounded below one half, then $0 < \gamma_m < \frac{1}{2}$. Then we have

$$
\begin{aligned}
Z_m &= 2\sqrt{(1-\text{err}_m)\text{err}_m} \\
&= 2\sqrt{(\frac{1}{2}-\gamma_m)(1-\frac{1}{2}+\gamma_m)} \\
&= \sqrt{1-4\gamma_m^2} \\
&\leq \sqrt{\exp(-4\gamma_m^2)} \\
&= \exp(-2\gamma_m^2)
\end{aligned}
$$

Consider the Taylor expansion of $\exp(-x^2)$ at $x = 0$.

Justification

- Let $\gamma = \min\{\gamma_1, \gamma_2, \cdots, \gamma_M\}$.The bound of the training error

$$R_{\mathrm{tr}}(H_G) \le \prod_{m=1}^{M} Z_m = \prod_{m=1}^{M} \exp(-2\gamma_m^2) \le \prod_{m=1}^{M} \exp(-2\gamma^2) = \exp(-2\gamma^2 M).$$

- In other words, the training error $R_{\mathrm{tr}}(H_G)$ is bounded by a decaying exponential. Moreover, since $R_{\mathrm{tr}}(H_G) \in \{0, \frac{1}{N}, \frac{2}{N}, \cdots, 1\}$. it follows that after a finite number of steps, when $\exp(-2\gamma^2 M_0) < \frac{1}{N}$, the training error will become 0 and the training data will be perfectly classified!

谢谢各位同学！