Adaboost

Rachel Zhang

大纲

- Adaptive basis function models
- CART
- Boosting
- Adaboost

Adaptive basis function models

- Kernel methods:

所有数据或部分数据

$$f(\mathbf{x}) = \mathbf{w}^T \phi(\mathbf{x})$$

$$\phi(\mathbf{x}) = [\kappa(\mathbf{x}, \boldsymbol{\mu}_1), \ldots, \kappa(\mathbf{x}, \boxed{\boldsymbol{\mu}_N})]$$

- Kernel 是啥? $\kappa(\mathbf{x}, \mathbf{x}')$
  – 距离度量
- 怎样定义好kernel?
- 怎样学kernel?
  – Maximizing likelihood
  – MKL (multiple kernel learning)
  – Adaptive basis function model (ABM)

$$\kappa(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{1}{2}(\mathbf{x} - \mathbf{x}')^T \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \mathbf{x}')\right)$$

$$\kappa(\mathbf{x}, \mathbf{x}') = \theta_0 \exp\left(-\frac{1}{2} \sum_{j=1}^{D} \theta_j (x_j - x_j')^2\right)$$

$$\kappa(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{||\mathbf{x} - \mathbf{x}'||^2}{2\sigma^2}\right)$$

$$\kappa(\mathbf{x}_i, \mathbf{x}_{i'}) = \frac{\mathbf{x}_i^T \mathbf{x}_{i'}}{||\mathbf{x}_i||_2 ||\mathbf{x}_{i'}||_2}$$

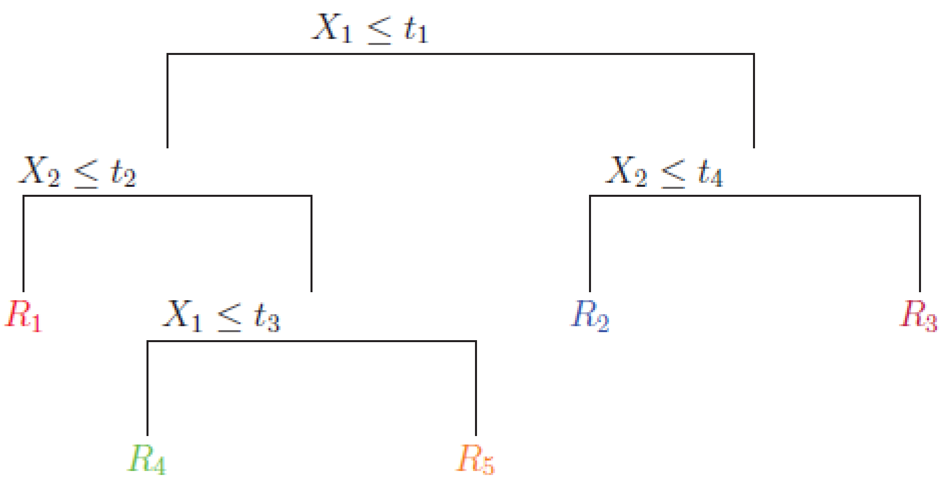
$$\kappa(\mathbf{x}, \mathbf{x}') = \sum_j w_j \kappa_j(\mathbf{x}, \mathbf{x}')$$

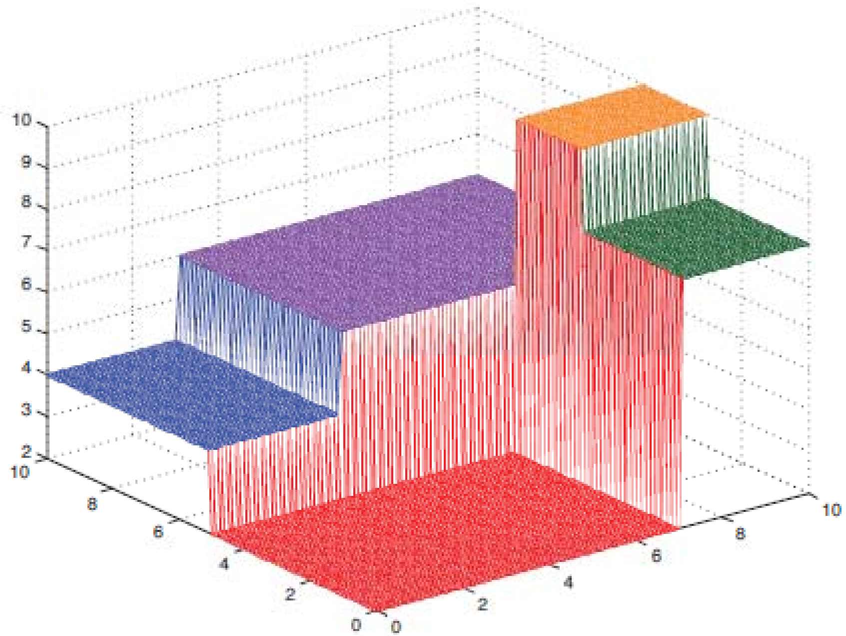
$$f(\mathbf{x}) = w_0 + \sum_{m=1}^{M} w_m \boxed{\phi_m}(\mathbf{x})$$

Basis function,
Learnt from data

CART

$$f(\mathbf{x}) = w_0 + \sum_{m=1}^{M} w_m \phi_m(\mathbf{x})$$





$$f(\mathbf{x}) = \mathbb{E}[y|\mathbf{x}] = \sum_{m=1}^{M} w_m \mathbb{I}(\mathbf{x} \in R_m) = \sum_{m=1}^{M} w_m \phi(\mathbf{x}; \mathbf{v}_m)$$

Rm: region m, 由basis function定义
Wm: mean response
Vm: encodes the variable to split on

$$f(\mathbf{x}) = w_0 + \sum_{m=1}^{M} w_m \phi_m(\mathbf{x})$$

CART

- CART model:

$$f(\mathbf{x}) = \mathbb{E}[y|\mathbf{x}] = \sum_{m=1}^{M} w_m \mathbb{I}(\mathbf{x} \in R_m) = \sum_{m=1}^{M} w_m \phi(\mathbf{x}; \mathbf{v}_m)$$

- Find best split:

$$(j^*, t^*) = \arg \min_{j \in \{1,\ldots,D\}} \min_{t \in \mathcal{T}_j} \text{cost}(\{\mathbf{x}_i, y_i : x_{ij} \le t\}) + \text{cost}(\{\mathbf{x}_i, y_i : x_{ij} > t\})$$

- Algorithm:

function fitTree(node, $\mathcal{D}$, depth) ;
node.prediction = mean($y_i : i \in \mathcal{D}$) // or class label distribution ;
$(j^*, t^*, \mathcal{D}_L, \mathcal{D}_R) = \text{split}(\mathcal{D})$;
if *not worthSplitting(depth, cost, $\mathcal{D}_L$, $\mathcal{D}_R$)* **then**
    return node
else
    node.test = $\lambda \mathbf{x}. x_{j^*} < t^*$ // anonymous function;
    node.left = fitTree(node, $\mathcal{D}_L$, depth+1);
    node.right = fitTree(node, $\mathcal{D}_R$, depth+1);
    return node;

Cost减少太小？
树高超过指定值？
是否所有label分布已经pure了?

$$\Delta \triangleq \text{cost}(\mathcal{D}) - \left(\frac{|\mathcal{D}_L|}{|\mathcal{D}|} \text{cost}(\mathcal{D}_L) + \frac{|\mathcal{D}_R|}{|\mathcal{D}|} \text{cost}(\mathcal{D}_R) \right)$$

CART

- 实战:
  - 加载数据集
  - 计算gini index
  - 根据最佳分割feature进行数据分割
  - 根据最大信息增益选择最佳分割feature
  - 递归构建决策树
  - 样本分类

$$f(\mathbf{x}) = w_0 + \sum_{m=1}^{M} w_m \phi_m(\mathbf{x})$$

CART

- **Misclassification rate**

$$\frac{1}{|\mathcal{D}|} \sum_{i \in \mathcal{D}} \mathbb{I}(y_i \neq \hat{y}) = 1 - \hat{\pi}_{\hat{y}}$$

- **Entropy**

$$\mathbb{H}(\hat{\boldsymbol{\pi}}) = -\sum_{c=1}^{C} \hat{\pi}_c \log \hat{\pi}_c$$

- **Gini index**

$$\sum_{c=1}^{C} \hat{\pi}_c (1 - \hat{\pi}_c) = \sum_c \hat{\pi}_c - \sum_c \hat{\pi}_c^2 = 1 - \sum_c \hat{\pi}_c^2$$

CART

- 实战

Data Set Characteristics:	Multivariate	**Number of Instances:**	150	**Area:**	Life
Attribute Characteristics:	Real	**Number of Attributes:**	4	**Date Donated**	1988-07-01
Associated Tasks:	Classification	**Missing Values?**	No	**Number of Web Hits:**	533125

$$f(\mathbf{x}) = w_0 + \sum_{m=1}^{M} w_m \phi_m(\mathbf{x})$$

CART

缺点：

- 估计不准：
  - Greedy nature难解决，只能local保最佳。
  - PS：Hyafil and Rivest 1976中提出找到最佳分割是NP完全问题，所以只能greedy地去找，也就是只能通过local optimize MLE.
- 树不稳定：
  - Training data小改变，树有可能大不同。

$$f(\mathbf{x}) = w_0 + \sum_{m=1}^{M} w_m \phi_m(\mathbf{x})$$

Boosting

- a greedy algorithm for fitting adaptive basis-function models (Leo Breiman, 1998)
- 以浅层CART作为basis learner (weak learner)
- "best off-the-shelf classifier in the world" (Hastie et al. 2009, p340)
  - Boosting > random forest >> single decision tree
- 成熟应用?

Boosting

- 弱学习机（weak learner)：对一定分布的训练样本给出假设（仅仅强于随机猜测）
  - 根据有云猜测可能会下雨
- 强学习机（strong learner): 根据得到的弱学习机和相应的权重给出假设（最大程度上符合实际情况：almost perfect expert)
  - 综合准确的天气预测

$$\text{弱学习机} \xrightarrow{\text{Boosting}} \text{强学习机}$$

- 目标：

$$\min_f \sum_{i=1}^{N} L(y_i, f(\mathbf{x}_i))$$

Boosting

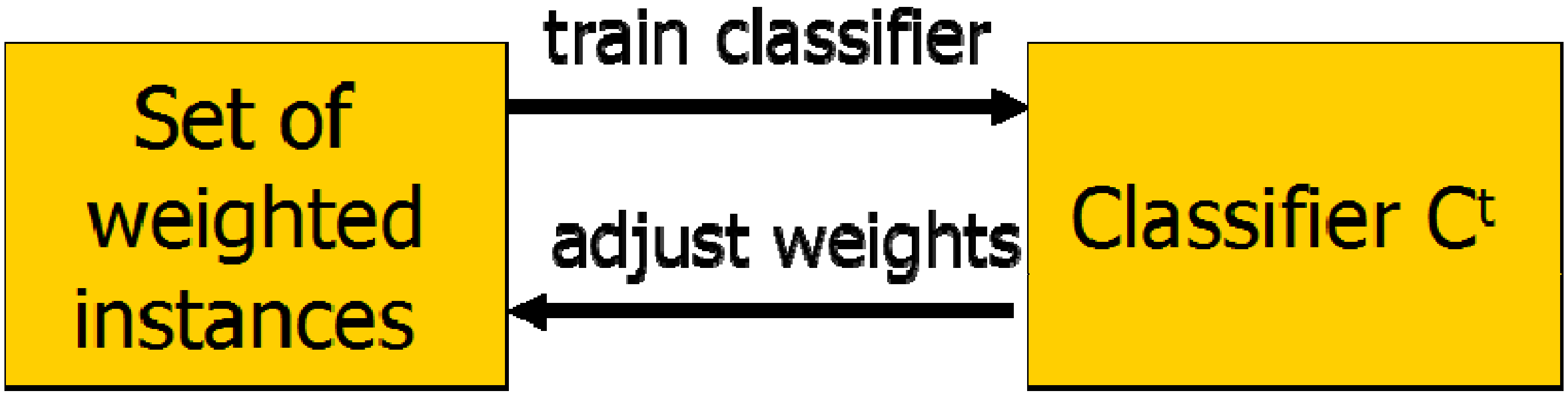
- 最初的boosting算法
- 1989年, Schapire 最先构造出一种多项式级的算法,对该问题做了肯定的证明,这就是最初的Boosting算法。
- 一年后，Freund提出了一种效率更高的boosting算法。

- 缺陷:都要求事先知道弱学习算法学习正确率的下限

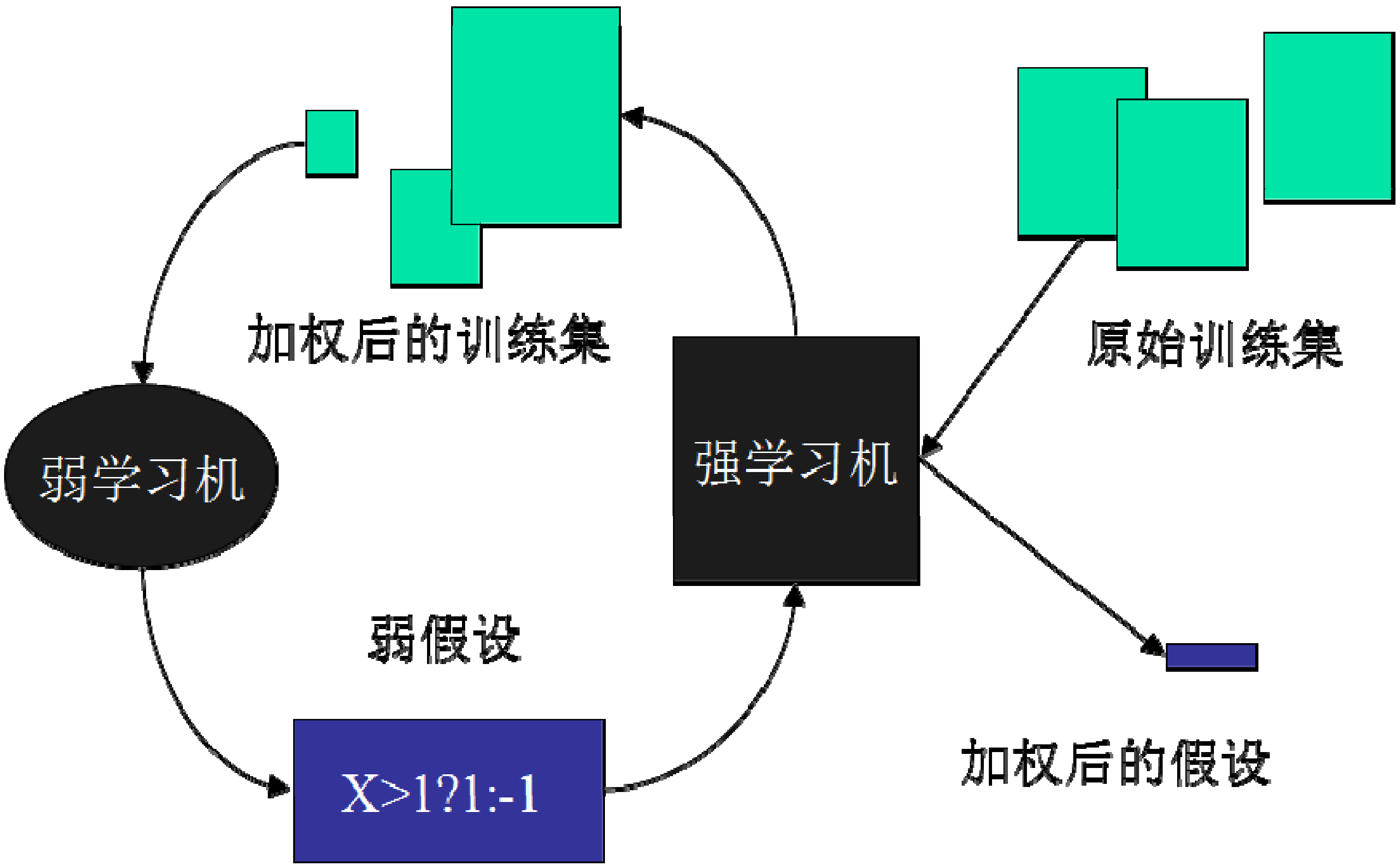
Boosting

- 1995 年, Freund 和schapire 改进了Boosting算法, 提出了AdaBoost (Adaptive Boosting)算法。
- 优点：该算法效率和Freund于1991年提出的Boosting算法几乎相同,但不需要任何关于弱学习器的先验知识,因而更容易应用到实际问题当中。
- 随后,Freund和schapire进一步提出了改变Boosting投票权重的AdaBoost. M1, AdaBoost. M2等算法, 在机器学习领域受到了极大的关注。

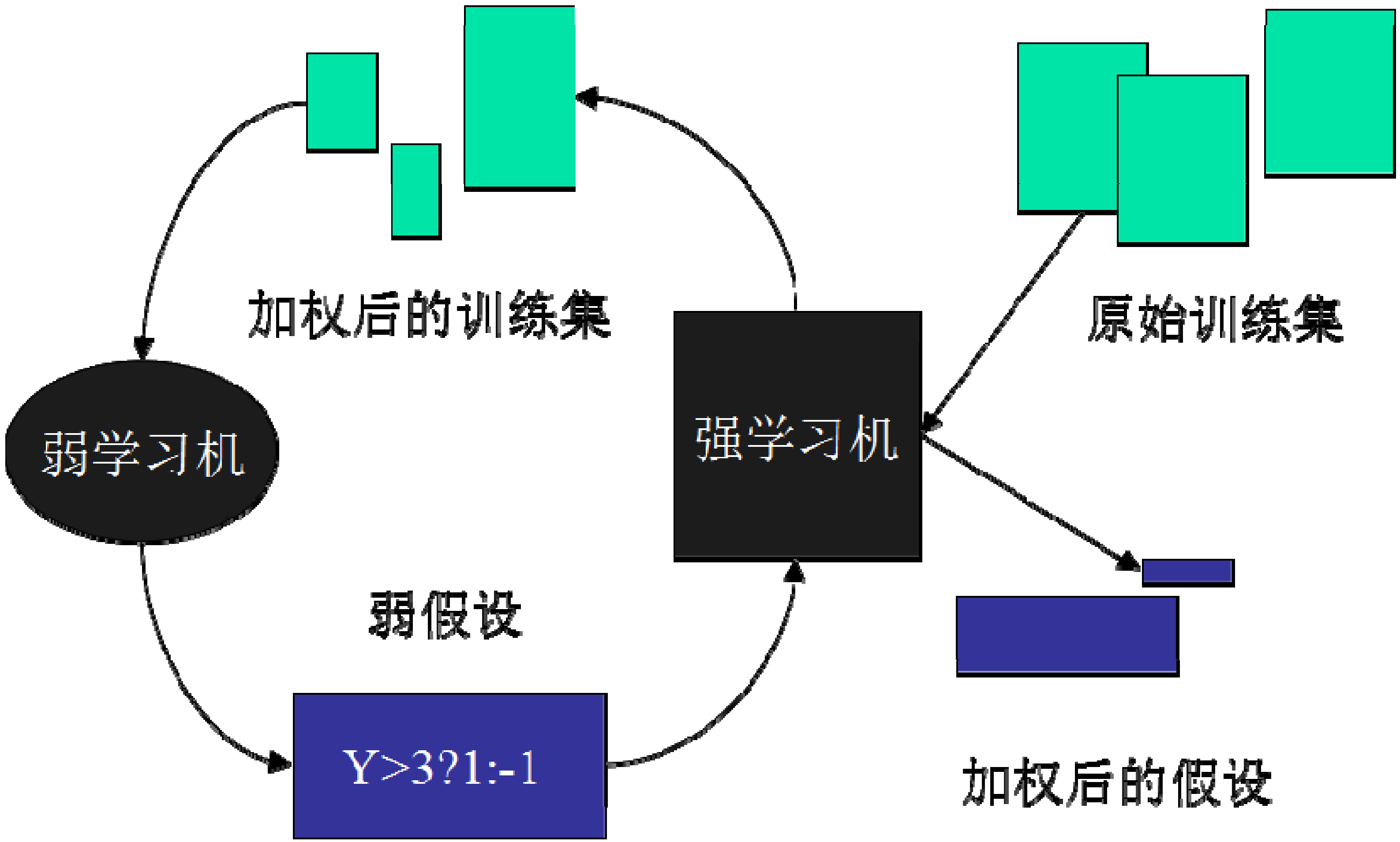
Boosting


Set of weighted instances
train classifier
adjust weights
Classifier Ct

Boosting – Loop 1

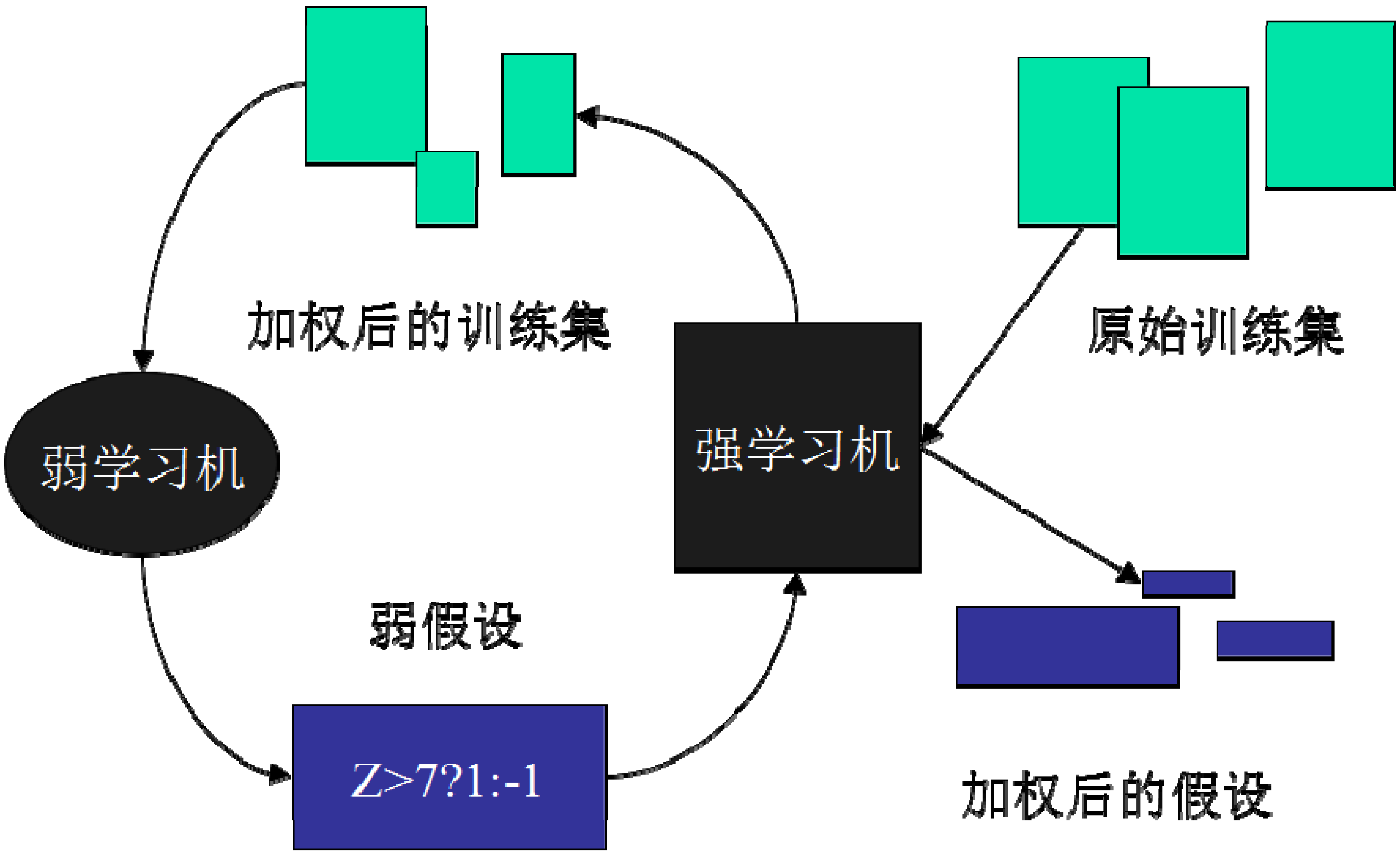


Boosting – Loop 2



Rachel Zhang
http://blog.csdn.net/abcjennifer

Boosting – Loop 3



Rachel Zhang
http://blog.csdn.net/abcjennifer

Boosting

- 首先给出任意一个弱学习算法和训练集 (x1 , y1) , (x2 , y2) , …, (xn , yn) , xi ∈X, X 表示某个实例空间,在分类问题中是一个带类别标志的集合, yi∈Y = { + 1, - 1}。
- 初始化时, Adaboost为训练集指定分布为1 /n, 即每个训练例的权重都相同为1 /n。
- 接着,调用弱学习算法进行T 次迭代,每次迭代后, 按照训练结果更新训练集上的分布,对于训练失败的训练例赋予较大的权重,使得下一次迭代更加关注这些训练例,从而得到一个预测函数序列h1 , h2 , …, ht ,每个预测函数ht 也赋予一个权重, 预测效果好的, 相应的权重越大。
- T 次迭代之后,在分类问题中最终的预测函数 H 采用带权重的投票法产生。
- 单个弱学习器的学习准确率不高,经过运用 Boosting 算法之后,最终结果准确率将得到提高。

$$\min_f \sum_{i=1}^{N} L(y_i, f(\mathbf{x}_i))$$

Boosting

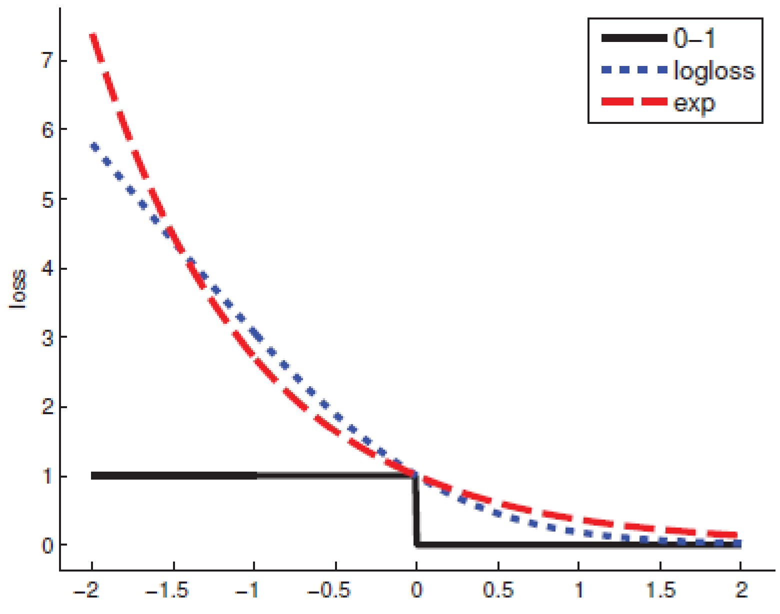
$$f(\mathbf{x}) = w_0 + \sum_{m=1}^{M} w_m \phi_m(\mathbf{x})$$

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	$\|y_i - f(\mathbf{x}_i)\|$	$\mathrm{sgn}(y_i - f(\mathbf{x}_i))$	$\mathrm{median}(y\|\mathbf{x}_i)$	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

$\tilde{y}_i \in \{-1, +1\}$

$y_i \in \{0, 1\}$

$\pi_i = \mathrm{sigm}(2f(\mathbf{x}_i))$.



Rachel Zhang
http://blog.csdn.net/abcjennifer

$$f(\mathbf{x}) = w_0 + \sum_{m=1}^{M} w_m \phi_m(\mathbf{x})$$

Boosting - Adaboost

- 指数loss

$$L(\tilde{y}, f) = \exp(-\tilde{y} f)$$

$$\begin{aligned}
\frac{\partial}{\partial f(\mathbf{x})} \mathbb{E}\left[e^{-\tilde{y} f(\mathbf{x})} | \mathbf{x}\right] &= \frac{\partial}{\partial f(\mathbf{x})} [p(\tilde{y}=1|\mathbf{x}) e^{-f(\mathbf{x})} + p(\tilde{y}=-1|\mathbf{x}) e^{f(\mathbf{x})}] \\
&= -p(\tilde{y}=1|\mathbf{x}) e^{-f(\mathbf{x})} + p(\tilde{y}=-1|\mathbf{x}) e^{f(\mathbf{x})} \\
&= 0 \Rightarrow \boxed{\frac{p(\tilde{y}=1|\mathbf{x})}{p(\tilde{y}=1-|\mathbf{x})}} = e^{2 f(\mathbf{x})}
\end{aligned}$$

$$f(\mathbf{x}) = w_0 + \sum_{m=1}^{M} w_m \phi_m(\mathbf{x})$$

Boosting

- 初始化:

$$f_0(\mathbf{x}) = \arg\min_{\gamma} \sum_{i=1}^{N} L(y_i, f(\mathbf{x}_i; \gamma)) \qquad f_0(\mathbf{x}) = \frac{1}{2} \log \frac{\hat{\pi}}{1-\hat{\pi}}$$

- 迭代求解:

$$(\beta_m, \gamma_m) = \underset{\beta,\gamma}{\operatorname{argmin}} \sum_{i=1}^{N} L(y_i, f_{m-1}(\mathbf{x}_i) + \beta\phi(\mathbf{x}_i; \gamma))$$

$$f_m(\mathbf{x}) = f_{m-1}(\mathbf{x}) + \beta_m \phi(\mathbf{x}; \gamma_m)$$

forward stagewise additive modeling

- In practice: $f_m(\mathbf{x}) = f_{m-1}(\mathbf{x}) + \nu\beta_m \phi(\mathbf{x}; \gamma_m)$

shrinkage

$$f(\mathbf{x}) = w_0 + \sum_{m=1}^{M} w_m \phi_m(\mathbf{x})$$

Adaboost

$$L_m(\phi) = \sum_{i=1}^{N} \exp[-\tilde{y}_i(f_{m-1}(\mathbf{x}_i) + \beta\phi(\mathbf{x}_i))] = \sum_{i=1}^{N} w_{i,m} \exp(-\beta\tilde{y}_i\phi(\mathbf{x}_i))$$

$$w_{i,m} \triangleq \exp(-\tilde{y}_i f_{m-1}(\mathbf{x}_i))$$

$\tilde{y}_i \in \{-1, +1\}$

$$\begin{aligned} L_m &= e^{-\beta} \sum_{\tilde{y}_i=\phi(\mathbf{x}_i)} w_{i,m} + e^{\beta} \sum_{\tilde{y}_i\neq\phi(\mathbf{x}_i)} w_{i,m} \\ &= (e^{\beta} - e^{-\beta}) \sum_{i=1}^{N} w_{i,m}\mathbb{I}(\tilde{y}_i \neq \phi(\mathbf{x}_i)) + e^{-\beta} \sum_{i=1}^{N} w_{i,m} \end{aligned}$$

$$f(\mathbf{x}) = w_0 + \sum_{m=1}^{M} w_m \phi_m(\mathbf{x})$$

Adaboost

$\tilde{y}_i \in \{-1, +1\}$

$$\begin{aligned} L_m &= e^{-\beta} \sum_{\tilde{y}_i = \phi(\mathbf{x}_i)} w_{i,m} + e^{\beta} \sum_{\tilde{y}_i \neq \phi(\mathbf{x}_i)} w_{i,m} \\ &= (e^{\beta} - e^{-\beta}) \sum_{i=1}^{N} w_{i,m} \mathbb{I}(\tilde{y}_i \neq \phi(\mathbf{x}_i)) + e^{-\beta} \sum_{i=1}^{N} w_{i,m} \end{aligned}$$

$$\phi_m = \operatorname*{argmin}_{\phi} w_{i,m} \mathbb{I}(\tilde{y}_i \neq \phi(\mathbf{x}_i))$$

$$\beta_m = \frac{1}{2} \log \frac{1 - \mathrm{err}_m}{\mathrm{err}_m} \qquad \mathrm{err}_m = \frac{\sum_{i=1}^{N} w_i \mathbb{I}(\tilde{y}_i \neq \phi_m(\mathbf{x}_i))}{\sum_{i=1}^{N} w_{i,m}}$$

$$f(\mathbf{x}) = w_0 + \sum_{m=1}^{M} w_m \phi_m(\mathbf{x})$$

Adaboost

- Update

$$f_m(\mathbf{x}) = f_{m-1}(\mathbf{x}) + \beta_m \phi_m(\mathbf{x})$$

$$\phi_m = \operatorname*{argmin}_{\phi} w_{i,m} \mathbb{I}(\tilde{y}_i \neq \phi(\mathbf{x}_i))$$

$$\beta_m = \frac{1}{2} \log \frac{1 - \mathrm{err}_m}{\mathrm{err}_m}$$

$$w_{i,m} \triangleq \exp(-\tilde{y}_i f_{m-1}(\mathbf{x}_i))$$

$$\begin{aligned} w_{i,m+1} &= w_{i,m} e^{-\beta_m \tilde{y}_i \phi_m(\mathbf{x}_i)} \\ &= w_{i,m} e^{\beta_m (2\mathbb{I}(\tilde{y}_i \neq \phi_m(\mathbf{x}_i)) - 1)} \\ &= w_{i,m} e^{2\beta_m \mathbb{I}(\tilde{y}_i \neq \phi_m(\mathbf{x}_i))} e^{-\beta_m} \end{aligned}$$

$\tilde{y}_i \in \{-1, +1\}$

$$\boxed{f(\mathbf{x}) = w_0 + \sum_{m=1}^{M} w_m \phi_m(\mathbf{x})}$$

Adaboost

$w_i = 1/N$;

for $m = 1 : M$ **do**

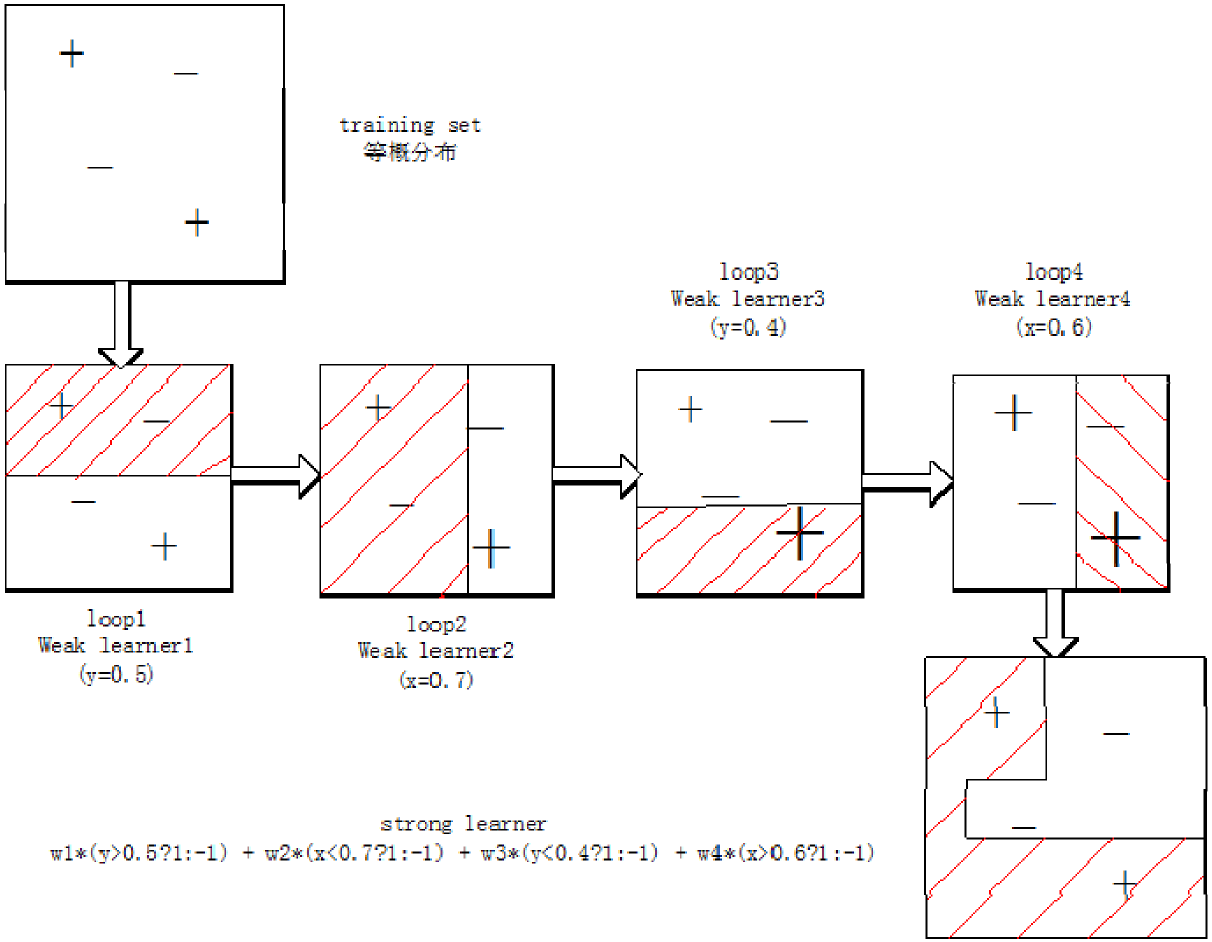
  Fit a classifier $\phi_m(\mathbf{x})$ to the training set using weights $\mathbf{w}$;

  Compute $\text{err}_m = \frac{\sum_{i=1}^{N} w_{i,m} \mathbb{I}(\tilde{y}_i \neq \phi_m(\mathbf{x}_i))}{\sum_{i=1}^{N} w_{i,m})}$;

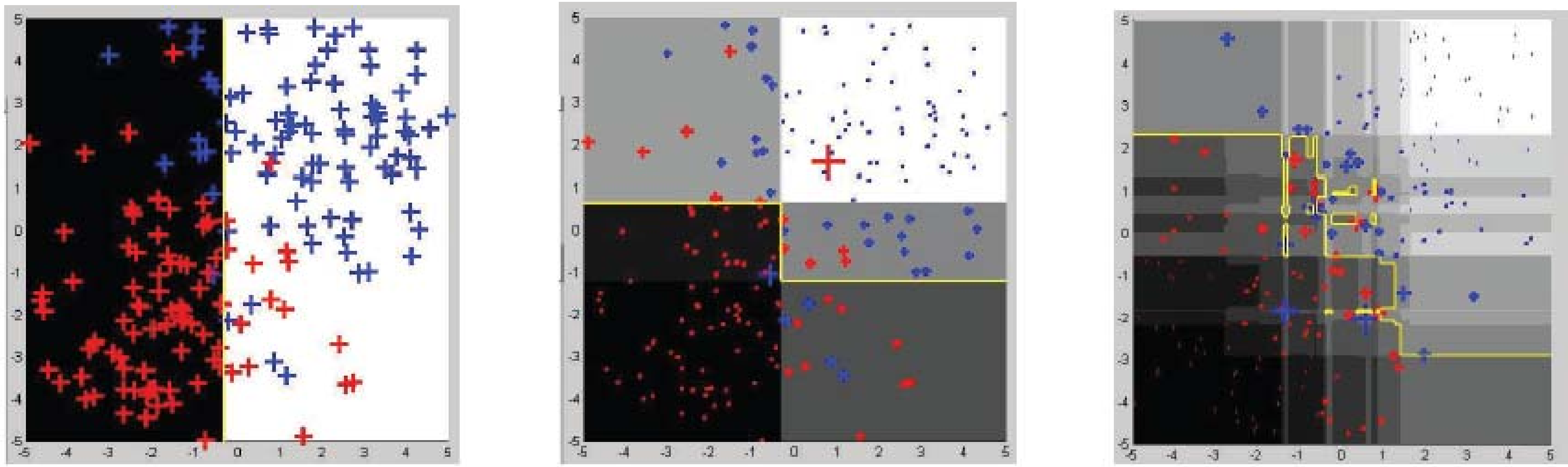
  Compute $\alpha_m = \log[(1 - \text{err}_m)/\text{err}_m]$;

  Set $w_i \leftarrow w_i \exp[\alpha_m \mathbb{I}(\tilde{y}_i \neq \phi_m(\mathbf{x}_i))]$;

Return $f(\mathbf{x}) = \text{sgn}\left[\sum_{m=1}^{M} \alpha_m \phi_m(\mathbf{x})\right]$;


training set
等概分布
loop1
Weak learner1
(y=0.5)
loop2
Weak learner2
(x=0.7)
loop3
Weak learner3
(y=0.4)
loop4
Weak learner4
(x=0.6)
strong learner
w1*(y>0.5?1:-1) + w2*(x<0.7?1:-1) + w3*(y<0.4?1:-1) + w4*(x>0.6?1:-1)

Adaboost



灰度表示属于某一类的置信度：
黑色表示属于红色类置信度，白色表示属于蓝色类置信度。

Rachel Zhang
http://blog.csdn.net/abcjennifer

Adaboost

- 实战
- 总结
  - 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。

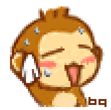
Boosting

- Performance好的原因?
  - 可视作l1-regularization
    - E.g 以计算好一些weak learners, 可以用l1-regularization选出有效feature的子集，也可以用boosting每次选出一个weak learner. 另外L1-Adaboost (Duchi and Singer，2009)提出结合boosting 和 l1-regularization（每次用boosting选择一个best weak learner, 然后用l1剪枝，去掉一些无关feature）。
  - Adaboost最大化了margin(Schapire et al. 1998; Ratsch et al. 2001)

Boosting

- 缺点:
  - 速度慢,在一定程度上依赖于训练数据集合和弱学习器的选择,训练数据不充足或者弱学习器太过“弱”,都将导致其训练精度的下降。
  - Boosting易受到噪声的影响,这是因为它在迭代过程中总是给噪声分配较大的权重,使得这些噪声在以后的迭代中受到更多的关注。

Exercise

- 随机生成一些数据，或者你自己找数据 
- 用RBF kernel去fit 
- 实现ID3, C4.5, CART算法, 分别fit 
- 用Adaboost去fit 
- 用LogitBoost去fit 

Reference

- Breiman L, Friedman J, Stone C J, et al. Classification and regression trees[M]. CRC press, 1984.
- Quinlan J R. Learning with continuous classes[C]//5th Australian joint conference on artificial intelligence. 1992, 92: 343-348.
- Quinlan J R. Induction of decision trees[J]. Machine learning, 1986, 1(1): 81-106.
- Rakotomamonjy A, Bach F, Canu S, et al. SimpleMKL[J]. Journal of Machine Learning Research, 2008, 9: 2491-2521.
- Duchi J, Singer Y. Boosting with structural sparsity[C]//Proceedings of the 26th Annual International Conference on Machine Learning. ACM, 2009: 297-304.
- Bühlmann P, Yu B. Sparse boosting[J]. The Journal of Machine Learning Research, 2006, 7: 1001-1024.
- Schapire R E, Freund Y. Boosting: Foundations and algorithms[M]. MIT press, 2012.
- Caruana R, Niculescu-Mizil A. An empirical comparison of supervised learning algorithms[C]//Proceedings of the 23rd international conference on Machine learning. ACM, 2006: 161-168.