集成学习

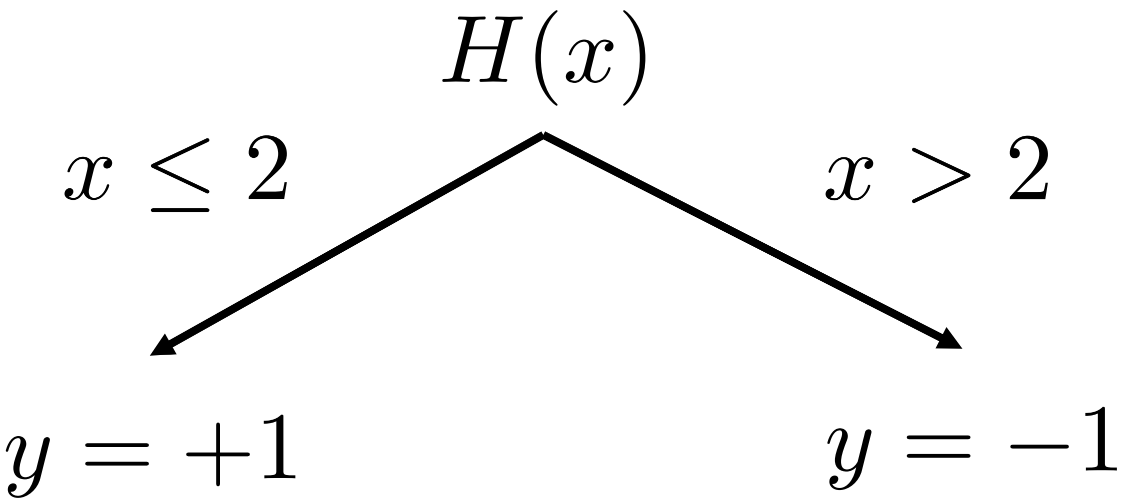
深度学习专业班

欧阳若飞

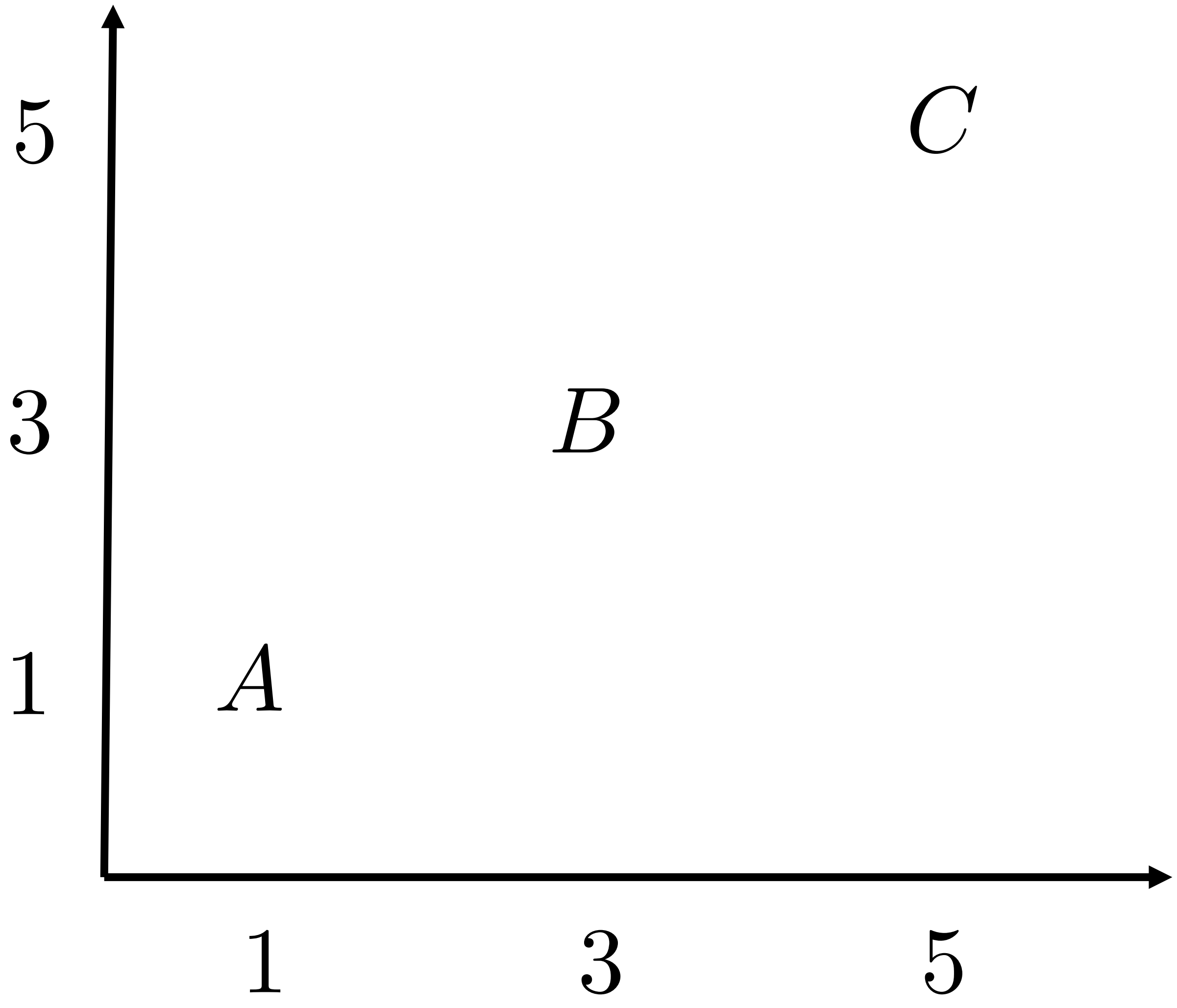
主要内容

- Bagging (简介)
  - 三个学渣一起考试
- Random Forest (简介)
  - 三个偏科学生一起考试
- Boosting (重点)
  - 一个学霸不停的刷错题考试超神
  - AdaBoost
  - XGBoost

决策树(Decision Tree)



决策树(Decision Tree)



左图能构建几棵可能的决策树？

$-1 \quad x \le 2 \quad +1$

$-1 \quad x \le 4 \quad +1$

$-1 \quad x \le 6 \quad +1$

$-1 \quad x > 2 \quad +1$

$-1 \quad x > 4 \quad +1$

$-1 \quad x > 6 \quad +1$

$\cancel{-1 \quad x \le 0 \quad +1}$

决策树的构建

根据信息增益或基尼系数选择一个特征上的一个分割点做分裂

递归的重复直到无法继续分裂

Bagging

把多棵决策树的决策结果做投票 (不一定需要决策树 也可以是其他分类模型)

for k in range(0, K):

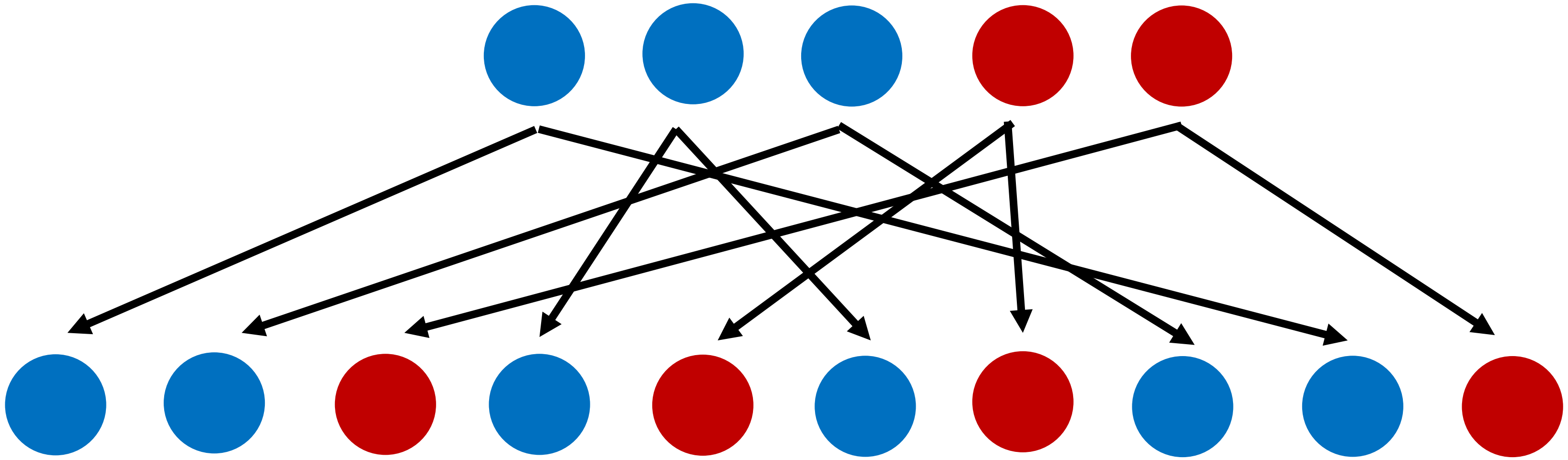
取样一个数据子集 D_k (后面我们说取样细节)

训练分类器 $h_k(x|D_k)$

综合预测结果 $\text{sign}(\sum_{k=1}^{K} h_k(x|D_k))$

Bootstrap

有放回的取样 sample with replacement



Bootstrap

如果N个数据采样N次，有多少数据不在采样集中？

$$(1-\frac{1}{N})^N$$

$$\lim_{n\to\infty}(1-\frac{1}{N})^N=\frac{1}{e}$$

36.788% 约三分之一的数据不在采样集

Bootstrap

有放回的取样 sample with replacement

好处:

1. 数据少的时候靠取样凑成大的数据集
2. 解决模型学习出现的数据不平衡的问题

单看Bagging本身意义不大，但是在数据不平衡时很有实战意义

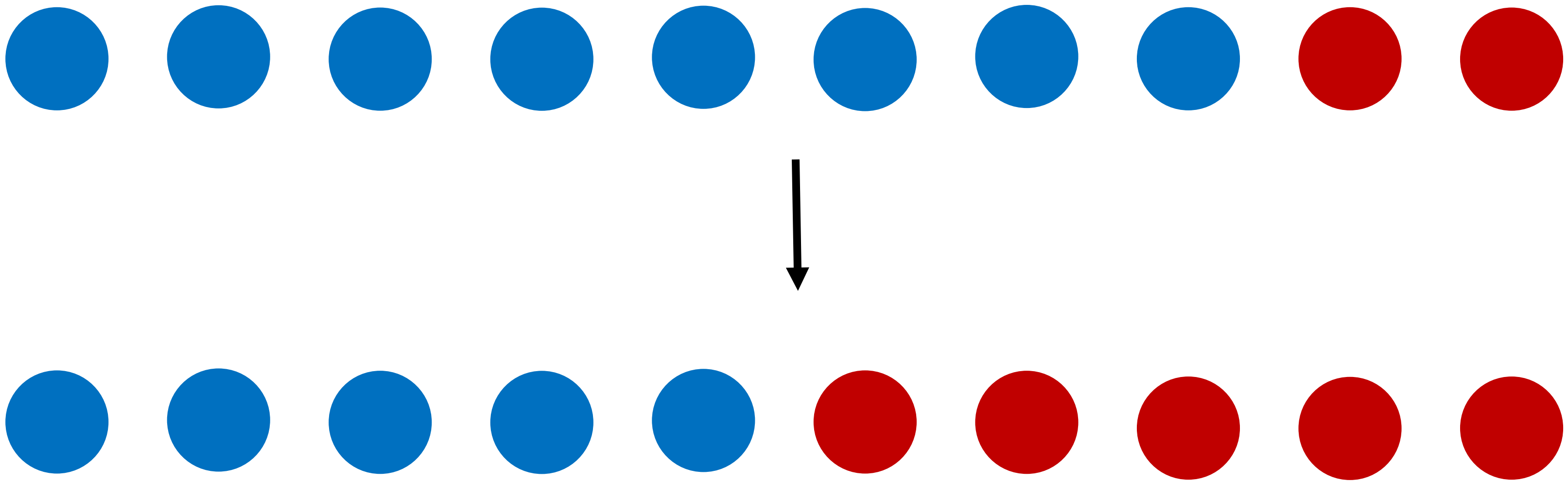
不平衡数据

正样本10 负样本90
无脑全部分类成负样本 准确率90% (有意义吗?)

举例: 癌症检测
正样本: 患有癌症 负样本: 健康
无脑分类成健康。。。
把健康的人错误分类成患有癌症。。。

数据不平衡:
数据的数目不平衡 (使用bootstrap)
分类错误的代价不平衡 (修改cost function)

不平衡数据



不平衡数据

实战Trick:

在深度学习的stochastic gradient descend里
每次选一小批数据进行学习

这时候如果是不平衡数据
我们可以使用bootstrap来取样这个小批次数据

Random Forest

按行取样
Bagging是从同一数据集抽取不同的数据子集训练再集成

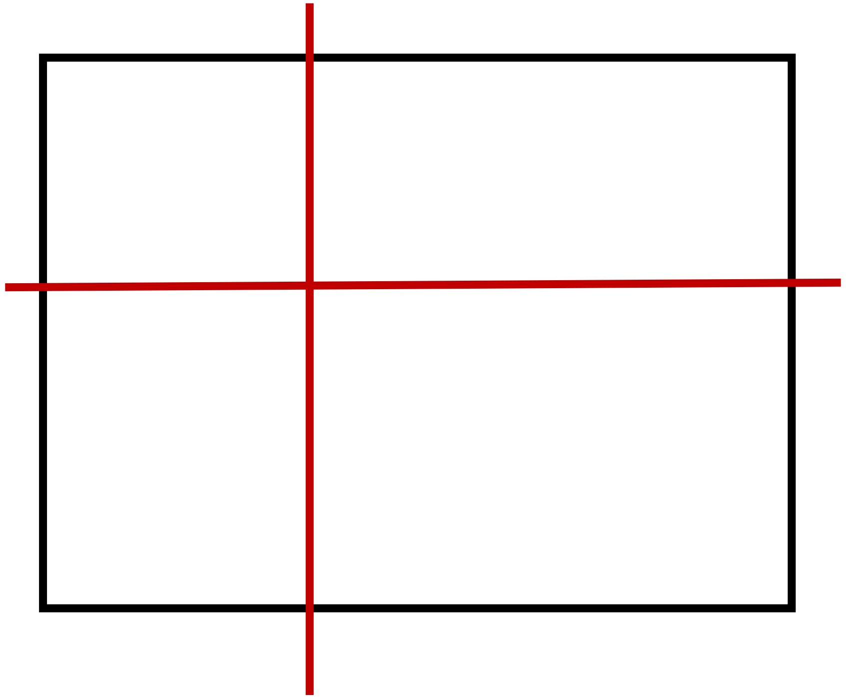
按列取样
Random Forest是在特征维度抽取不同的特征子集训练再集成

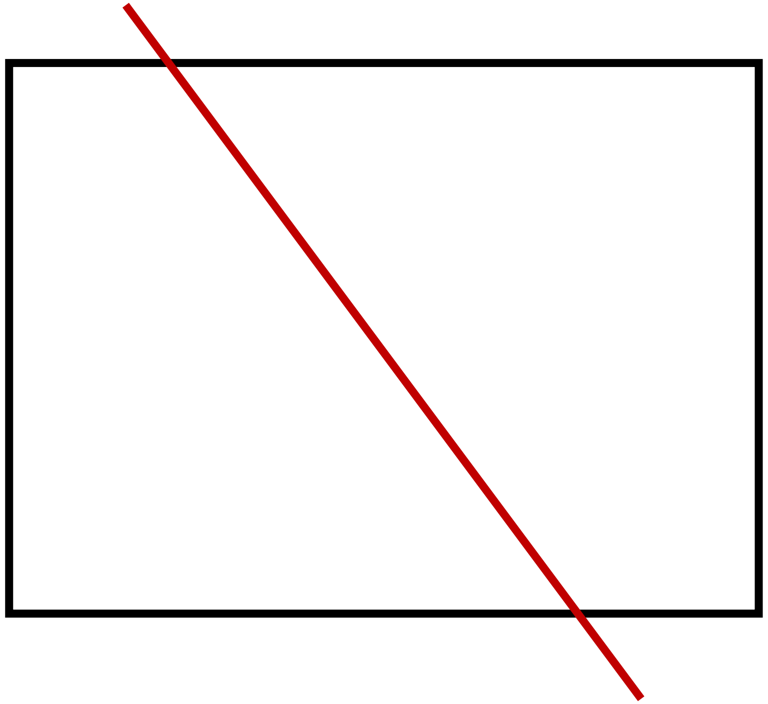
	d1	d2	d3
n1			
n2			
n3			

Random Forest

实际的RF:

按行取样 + 按列取样再投影 + 生成决策树

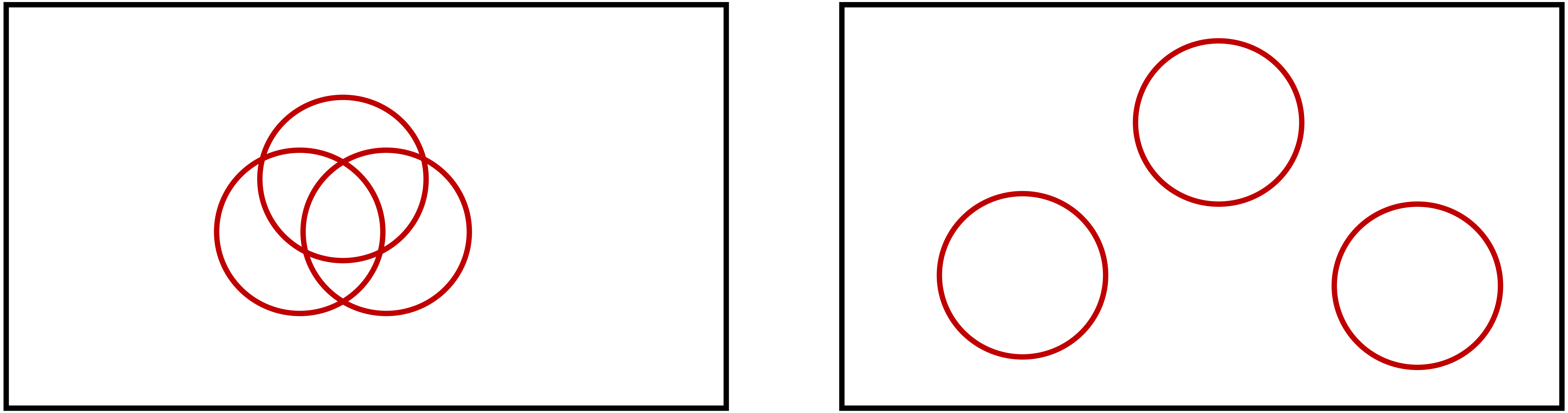




Random Forest

特征空间的不同subspaces之间的相关性较小
构建的决策树的相关性较小
犯同样错误的决策树很少
集成的结果变好

Random Forest



如果决策树相关性大的话 集成的森林不一定比单棵树好

降低相关性很重要！

Random Forest

for k in range(0, K):

取样一个subspace上的数据子集 D_k

训练分类器 $h_k(x|D_k)$

综合预测结果 $\text{sign}(\sum_{k=1}^{K} h_k(x|D_k))$

如果不希望是等权重的投票结果怎样呢?

Boosting

今天的主角登场了

投票不一定需要等权重 $H(x|D) = \mathrm{sign}(\sum_{k=1}^{K} \alpha_k h_k(x|D))$

权重如何设定呢？效果好的权重大

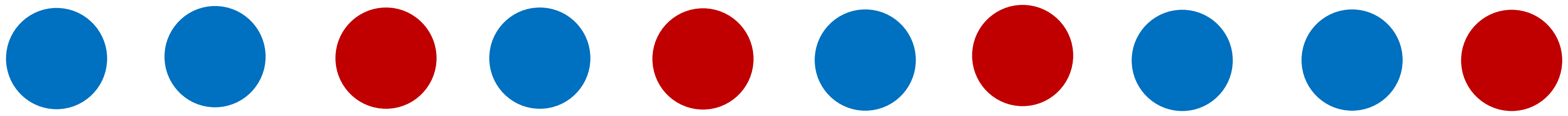
每个数据点分类难度不同，效果需要根据数据分类难度来计算

Boosting

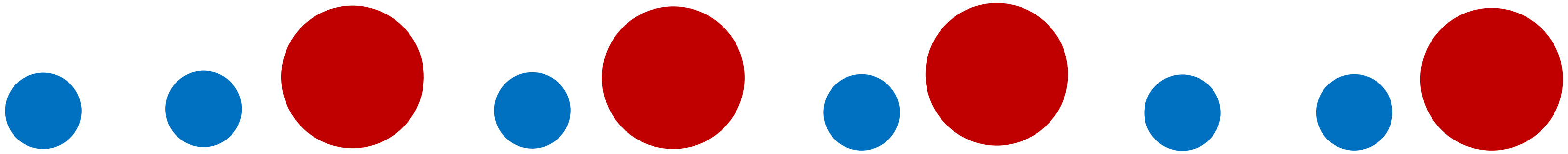
本次课介绍Adaptive Boost (Adaboost)

下次课介绍Gradient Boost

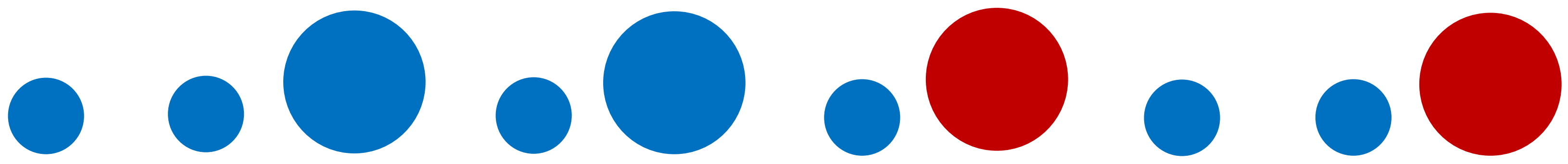
AdaBoost



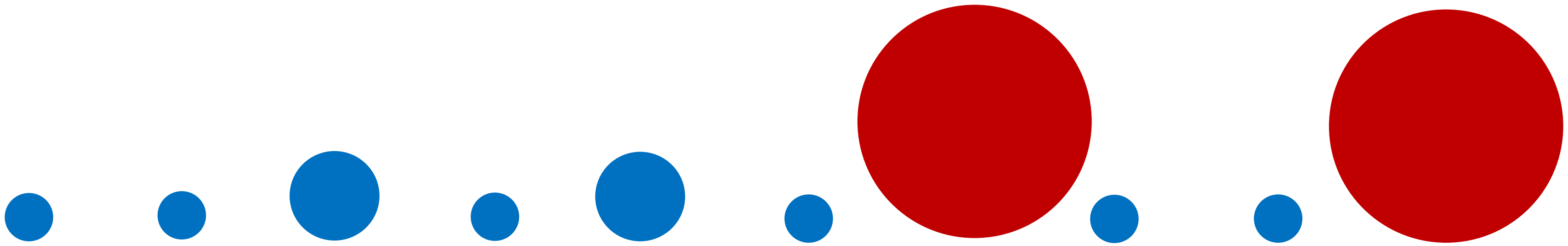
AdaBoost



AdaBoost



AdaBoost



AdaBoost

AdaBoost

重点提高当前被分错的数据上的效果

考试前刷以前做错的习题

AdaBoost

for k in range(0, K):

给每个数据x_i重新计算权重w_i　　如何计算数据权重

训练分类器$h_k(D)$

计算分类器的投票权重α_k　　如何计算投票权重

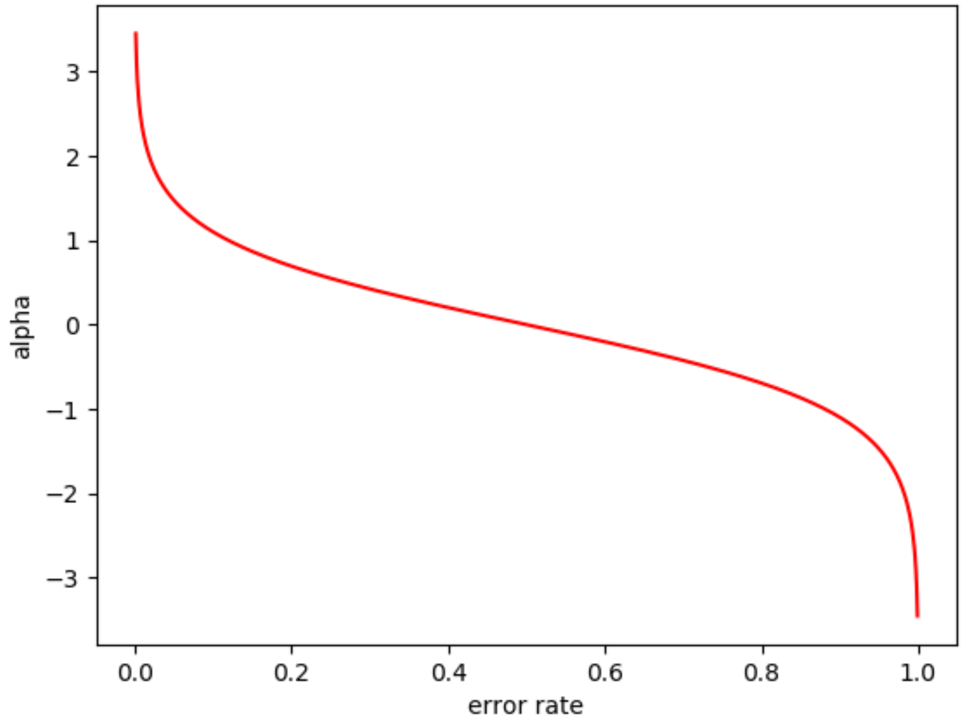
综合预测结果 $\mathrm{sign}(\sum_{k=1}^{K} \alpha_k h_k(x|D))$

AdaBoost

投票权重需要根据分类效果设置的

分类效果评价 错误率： $\epsilon_k = \frac{1}{N}\sum_{i=1}^{N} w_i \mathrm{err}(y_i, h_k(x_i|D))$

投票权重： $\alpha_k = \log\sqrt{\frac{1-\epsilon_k}{\epsilon_k}}$



错误率小权重大

错误率大权重也大！ 考试选择题全部做错也是很难的

AdaBoost

数据权重也需要根据分类效果设置

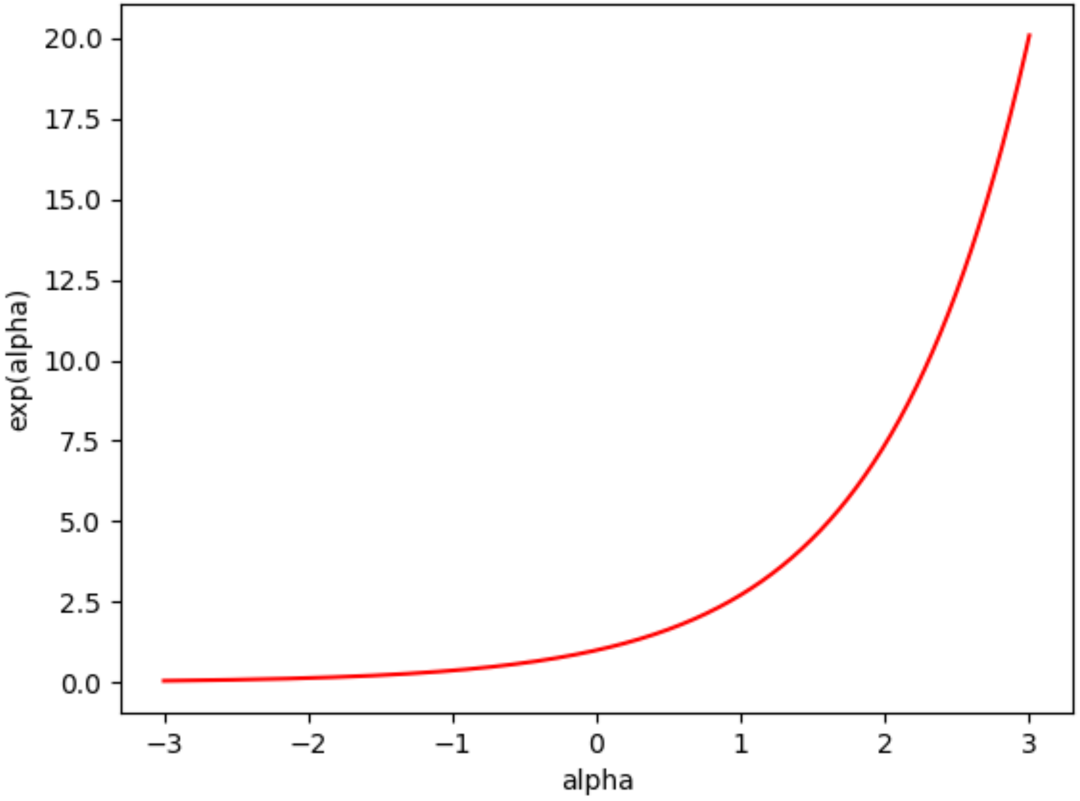
数据权重:

分类正确 权重变小

当 $y_i = h_k(x_i|D)$ $\quad w_i = \frac{w_i}{Z} e^{-\alpha_k}$

分类错误 权重变大

当 $y_i \neq h_k(x_i|D)$ $\quad w_i = \frac{w_i}{Z} e^{\alpha_k}$



AdaBoost

分类正确 权重变小

当 $y_i = h_k(x_i|D)$ $\quad w_i = \frac{w_i}{Z} e^{-\alpha_k} = \frac{w_i}{Z}\sqrt{\frac{1-\epsilon_k}{\epsilon_k}}$

分类错误 权重变大

当 $y_i \neq h_k(x_i|D)$ $\quad w_i = \frac{w_i}{Z} e^{\alpha_k} = \frac{w_i}{Z}\sqrt{\frac{\epsilon_k}{1-\epsilon_k}}$

$$Z = \underbrace{\boxed{\sum_{y_i = h_k(x_i|D)} w_i}}_{1-\epsilon_k} \sqrt{\frac{1-\epsilon_k}{\epsilon_k}} + \underbrace{\boxed{\sum_{y_i \neq h_k(x_i|D)} w_i}}_{\epsilon_k} \sqrt{\frac{\epsilon_k}{1-\epsilon_k}} = 2\sqrt{\epsilon_k(1-\epsilon_k)}$$

AdaBoost

分类正确 权重变小

当 $y_i = h_k(x_i|D)$ $\quad w_i = \frac{w_i}{Z} e^{-\alpha_k} = \frac{w_i}{Z}\sqrt{\frac{1-\epsilon_k}{\epsilon_k}} = \frac{w_i}{2(1-\epsilon_k)}$

分类错误 权重变大

当 $y_i \neq h_k(x_i|D)$ $\quad w_i = \frac{w_i}{Z} e^{\alpha_k} = \frac{w_i}{Z}\sqrt{\frac{\epsilon_k}{1-\epsilon_k}} = \frac{w_i}{2\epsilon_k}$

$$Z = \underbrace{\boxed{\sum_{y_i = h_k(x_i|D)} w_i}}_{1-\epsilon_k} \sqrt{\frac{1-\epsilon_k}{\epsilon_k}} + \underbrace{\boxed{\sum_{y_i \neq h_k(x_i|D)} w_i}}_{\epsilon_k} \sqrt{\frac{\epsilon_k}{1-\epsilon_k}} = 2\sqrt{\epsilon_k(1-\epsilon_k)}$$

AdaBoost

$$\sum_{y_i=h_k(x_i|D)} w_i = \boxed{\sum_{y_i=h_k(x_i|D)} w_i} \frac{1}{2(1-\epsilon_k)} = \frac{1}{2}$$

$$1-\epsilon_k$$

$$\sum_{y_i\neq h_k(x_i|D)} w_i = \boxed{\sum_{y_i\neq h_k(x_i|D)} w_i} \frac{1}{2\epsilon_k} = \frac{1}{2}$$

$$\epsilon_k$$

分类正确的样本和分类错误的样本权重和为1/2
只需要根据本次分类正确与否按照上一次的权重scale一下就好

AdaBoost

$$\underbrace{\boxed{\sum_{y_i=h_k(x_i|D)}}}_{1-\epsilon_k} w_i e^{-\alpha_k} + \underbrace{\boxed{\sum_{y_i\neq h_k(x_i|D)}}}_{\epsilon_k} w_i e^{\alpha_k}$$

$$(1-\epsilon_k)e^{-\alpha_k} + \epsilon_k e^{\alpha_k}$$

求导得： $$\alpha_k = \log\sqrt{\frac{1-\epsilon_k}{\epsilon_k}}$$

AdaBoost

树的深度大好吗?

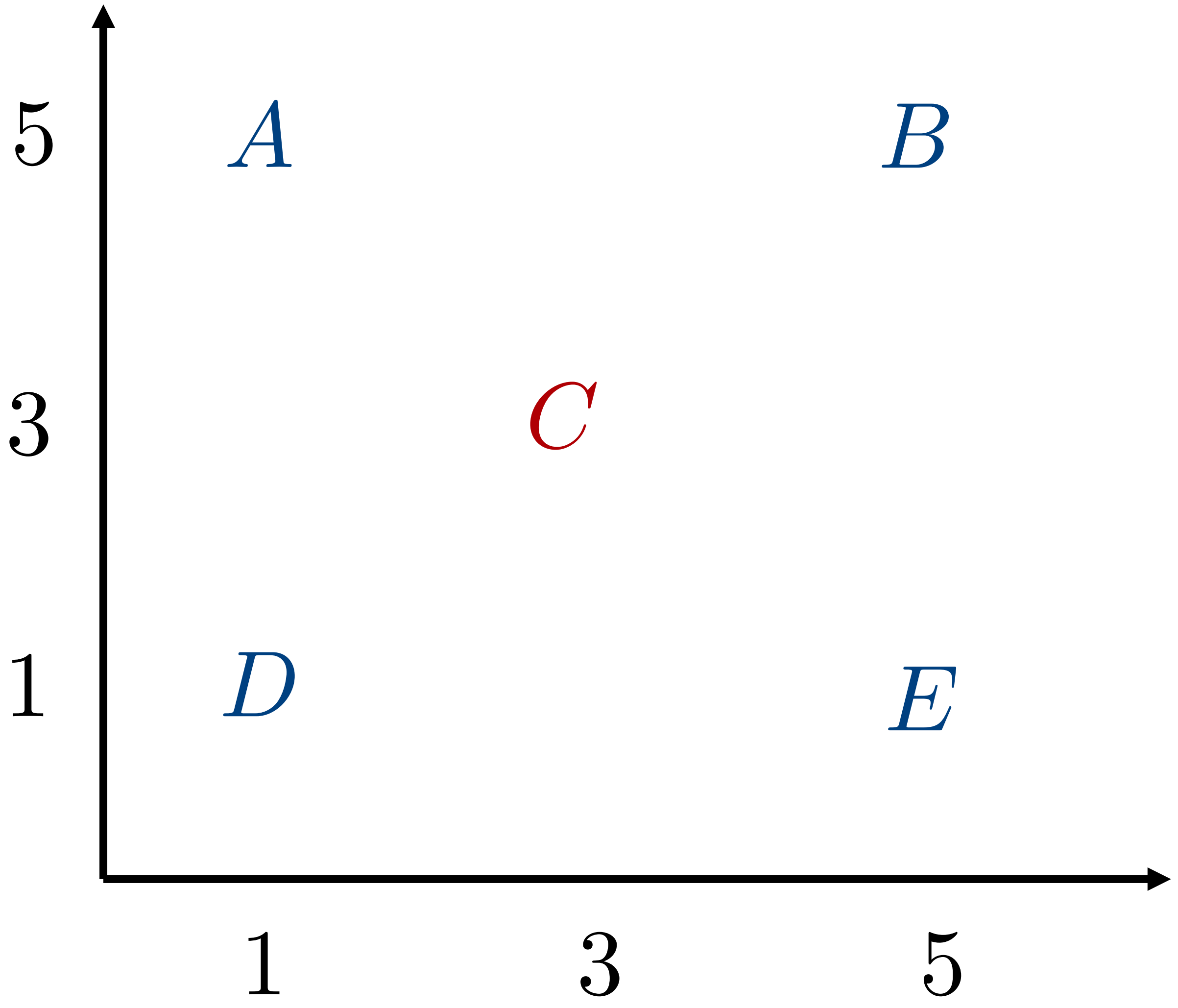
一个学习了所有数据的决策树 $\epsilon_k = 0$

投票权重: $\alpha_k = \log\sqrt{\frac{1-\epsilon_k}{\epsilon_k}} = \infty$

权重无限大说明这个决策树是个独裁者
集成多个模型就没有意义了

控制树的深度不要过大
只用一部分数据学习单棵树

AdaBoost实例

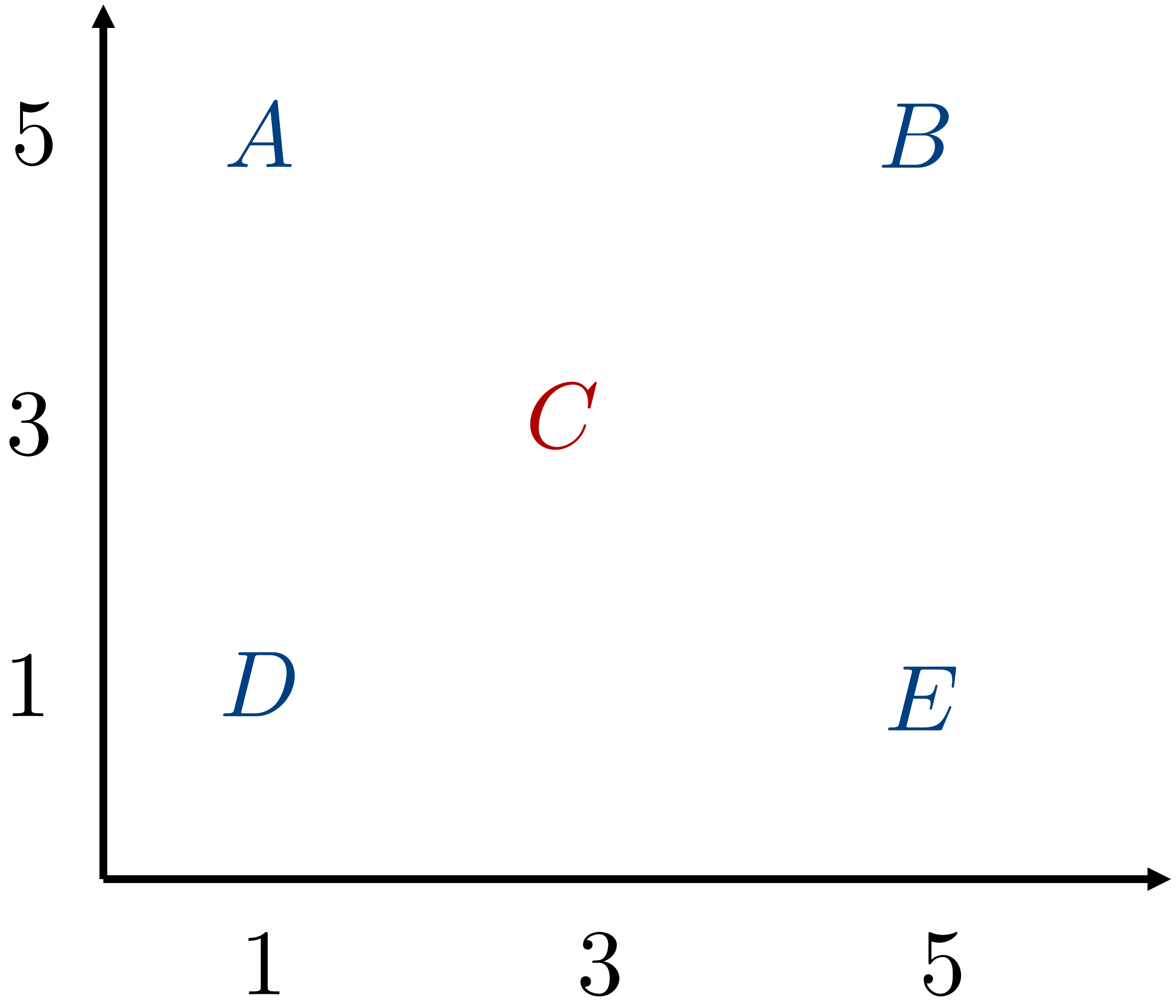


data	weight
A	1/5
B	1/5
C	1/5
D	1/5
E	1/5

tree	wrong	error rate	alpha
x<2	BE	2/5	
x<4	BCE	3/5	
x<6	C	1/5	1/2ln((1-1/5)/(1/5))
x>2	ACD	3/5	
x>4	AD	2/5	
x>6	ABDE	4/5	

if condition blue else red

AdaBoost实例

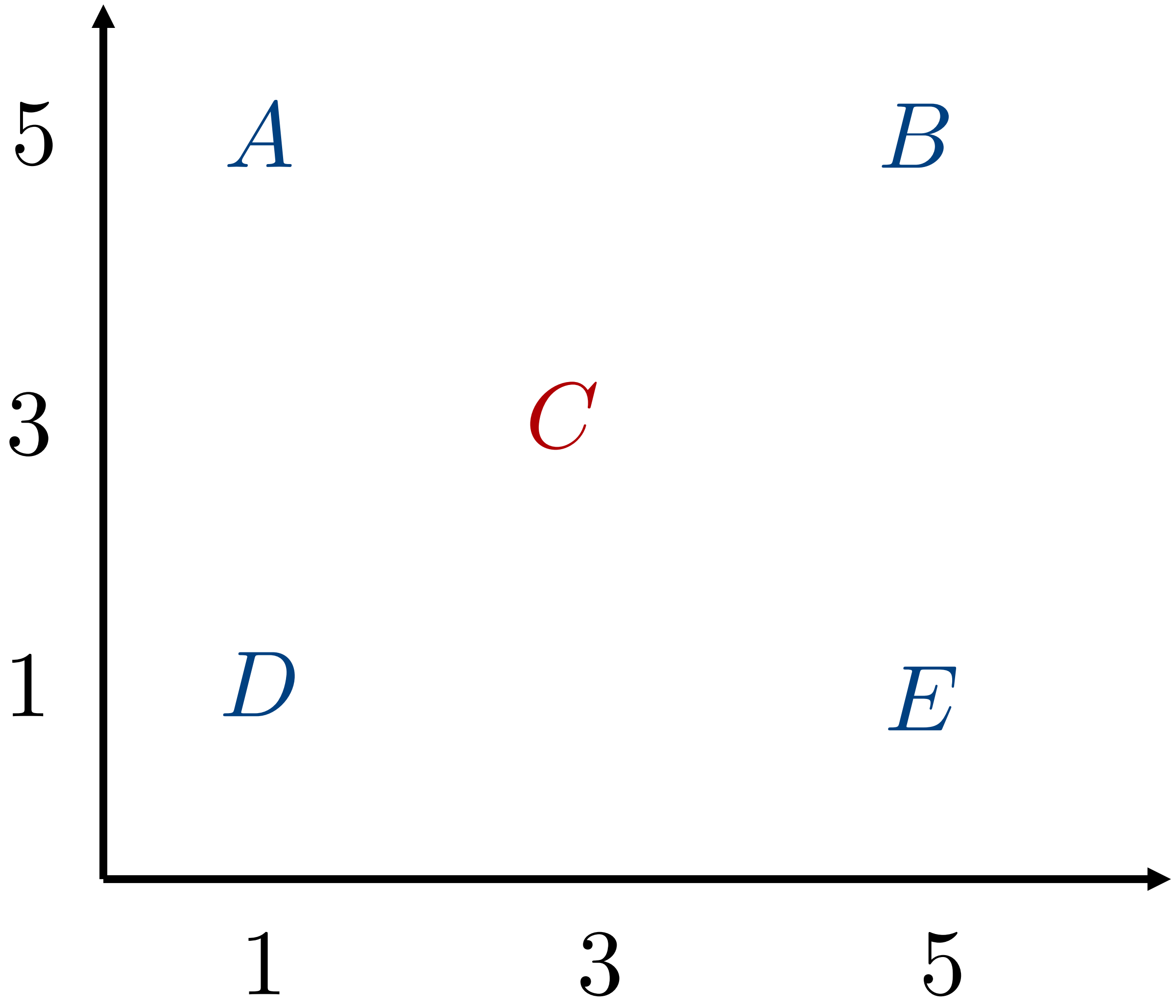


data	weight	Updated weight
A	1/5	1/8
B	1/5	1/8
C	1/5	1/2
D	1/5	1/8
E	1/5	1/8

tree	wrong	error rate	alpha
x<2	BE	2/5	
x<4	BCE	3/5	
x<6	C	1/5	1/2ln((1-1/5)/(1/5))
x>2	ACD	3/5	
x>4	AD	2/5	
x>6	ABDE	4/5	

if condition blue else red

AdaBoost实例



data	weight	Updated weight
A	1/8	1/12
B	1/8	3/12
C	4/8	4/12
D	1/8	1/12
E	1/8	3/12

tree	wrong	error rate	alpha
x<2	BE	2/8	1/2ln((1-2/8)/(2/8))
x<4	BCE	6/8	
x<6	C	4/8	1/2ln((1-1/5)/(1/5))
x>2	ACD	6/8	
x>4	AD	2/8	
x>6	ABDE	4/8	

if condition blue else red

AdaBoost终止条件

1 预测精度满足需要
2 达到最大迭代次数
3 可能的分类器错误率均为1/2　(alpha=0)

$$H(x) = \alpha_1 h_1(x) + \alpha_2 h_2(x) + \alpha_3 h_3(x) \qquad + \alpha_4 h_4(x)$$

Gradient Boost

$$H(x)=\sum_{k=1}^{K}h_k(x)$$

此时为了方便理解 我们讨论回归的情况

$$H_K(x)=\sum_{k=1}^{K-1}h_k(x)+h_K(x)$$

$$\sum_{i=1}^{N}\mathcal{L}(y_i,H_K(x_i))=\sum_{i=1}^{N}(y_i-H_K(x_i))^2$$

$$H_K(x)=H_{K-1}(x)+h_K(x)$$

$$\sum_{i=1}^{N}\mathcal{L}(y_i,H_{K-1}(x_i)+h_K(x_i))=\sum_{i=1}^{N}(y_i-H_{K-1}(x_i)-h_K(x_i))^2$$

每次学习上一次结果的残差 $r_i^K=y_i-H_{K-1}(x_i)$

Gradient Boost

初始化 $H_0(x) = \text{argmax} \sum_{i=1}^{N} \mathcal{L}(y_i, h_0(x))$

for k in range(1, K):

计算残差 $r_i^k = \frac{\partial \mathcal{L}}{\partial f_{k-1}(x_i)}$

训练分类器 $h_k(x)$

$H_k(x) = H_{k-1}(x) + h_k(x)$

XGBoost

泰勒展开

$$f(x+\Delta x) \approx f(x) + f'(x)\Delta x + \frac{1}{2}f''(x)\Delta x^2$$

$$\sum_{i=1}^{N} \mathcal{L}(y_i, \underbrace{H_{k-1}(x_i)}_{x} + \underbrace{h_k(x_i)}_{\Delta x}) = \sum_{i=1}^{N} \mathcal{L}(y_i, H_{k-1}(x_i)) + \underbrace{a_k}_{f'(x)} h_k(x_i) + \frac{1}{2}\underbrace{b_k}_{f''(x)} h_k(x_i)^2$$

最小化该目标函数等价于

$$\min \sum_{i=1}^{N} a_k h_k(x_i) + \frac{1}{2} b_k h_k(x_i)^2$$

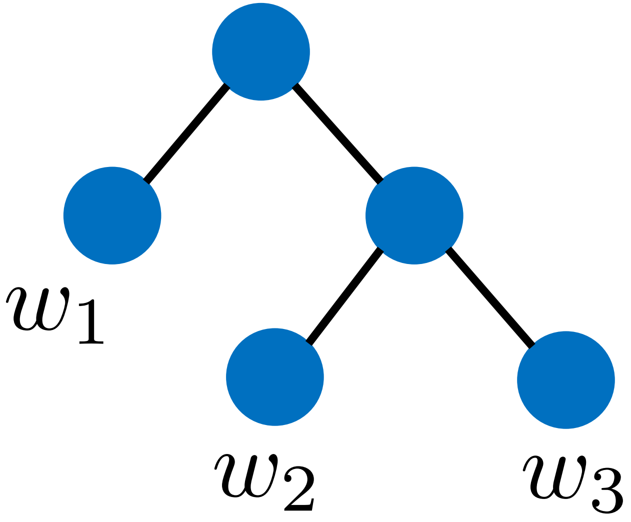
XGBoost

加入正则项

$$\min \sum_{i=1}^{N} a_k h_k(x_i) + \frac{1}{2} b_k h_k(x_i)^2 + \Omega(h_k)$$

$$\Omega(h_k) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2$$

叶子节点个数　叶子节点权重

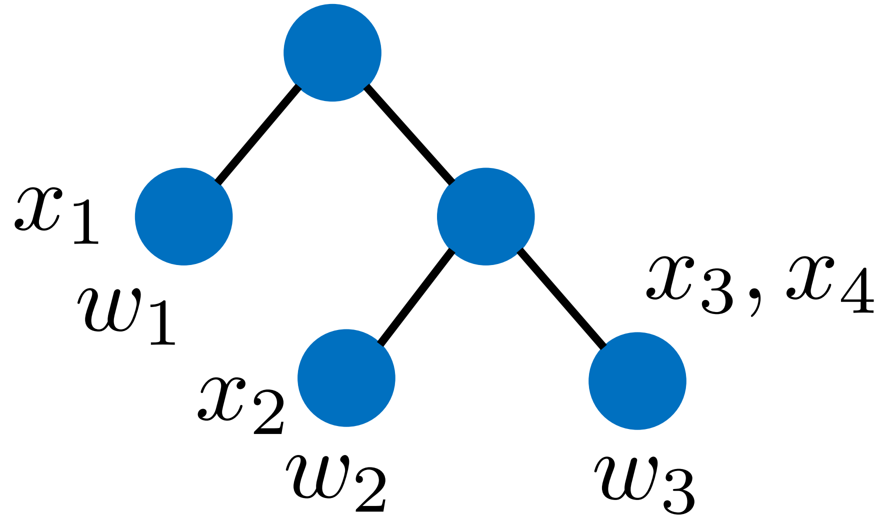


$$T = 3$$

XGBoost

$$\sum_{i=1}^{N} a_k h_k(x_i) + \frac{1}{2} b_k h_k(x_i)^2 + \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2$$

$$\sum_{j=1}^{T}\left(\Big(\sum_{i\in I_j} a_i\Big)w_j + \frac{1}{2}\Big(\sum_{i\in I_j} b_i+\lambda\Big)w_j^2\right)+\gamma T$$



数每个数据变成数每个叶子节点上的数据

对w求导: $$\sum_{j=1}^{T}\left(\Big(\underbrace{\sum_{i\in I_j} a_i}_{A_j}\Big) + \Big(\underbrace{\sum_{i\in I_j} b_i}_{B_j}+\lambda\Big)w_j\right)=0$$

$$w_j = -\frac{A_j}{B_j+\lambda}$$

$$\mathcal{L} = -\frac{1}{2}\sum_{j=1}^{T}\frac{A_j^2}{B_j+\lambda}+\gamma T$$

XGBoost

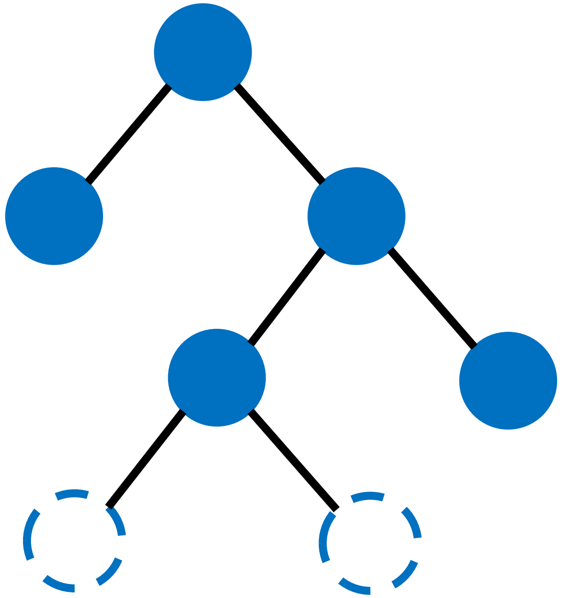
选择下个分裂点:

$$\mathcal{L} = -\frac{1}{2}\sum_{j=1}^{T}\frac{A_j^2}{B_j+\lambda}+\gamma T$$

分裂前 $$\mathcal{L} = -\frac{1}{2}\frac{(A_L+A_R)^2}{(B_L+B_R)+\lambda}+\gamma T+\mathrm{Rest}$$

分裂后 $$\mathcal{L} = -\frac{1}{2}\left(\frac{A_L^2}{B_L+\lambda}+\frac{A_R^2}{B_R+\lambda}\right)+\gamma(T+1)+\mathrm{Rest}$$

$$\Delta\mathcal{L} = -\frac{1}{2}\left(\frac{(A_L+A_R)^2}{(B_L+B_R)+\lambda}-\frac{A_L^2}{B_L+\lambda}-\frac{A_R^2}{B_R+\lambda}\right)-\gamma$$



每次取cost下降最多的点去做分裂

XGBoost

XGBoost的优势

分类回归都能用 很多实际问题可以直接解决

树型模型的可解释型非常强

无需做过多的数据预处理

XGBoost的代码支持多种平台，自带GPU和分布式功能

XGBoost参数详解

eta 学习速率 learning rate $H_k(x) = H_{k-1} + \eta h_k(x)$ α_k in Adaboost

gamma 叶子节点个数的正则项

lambda 叶子节点权重的L2正则项

alpha 叶子节点权重的L1正则项

$$\Omega(h_k) = \gamma T + \frac{1}{2}\lambda\sum_{j=1}^{T} w_j^2$$

max_depth 单棵树的深度 在Boosting模型里树不能大

min_child_weight 叶子节点最小权重

subsample 按行取样的概率 取数据的子集

col_sample_bytree 按列取样的概率 取特征的子集

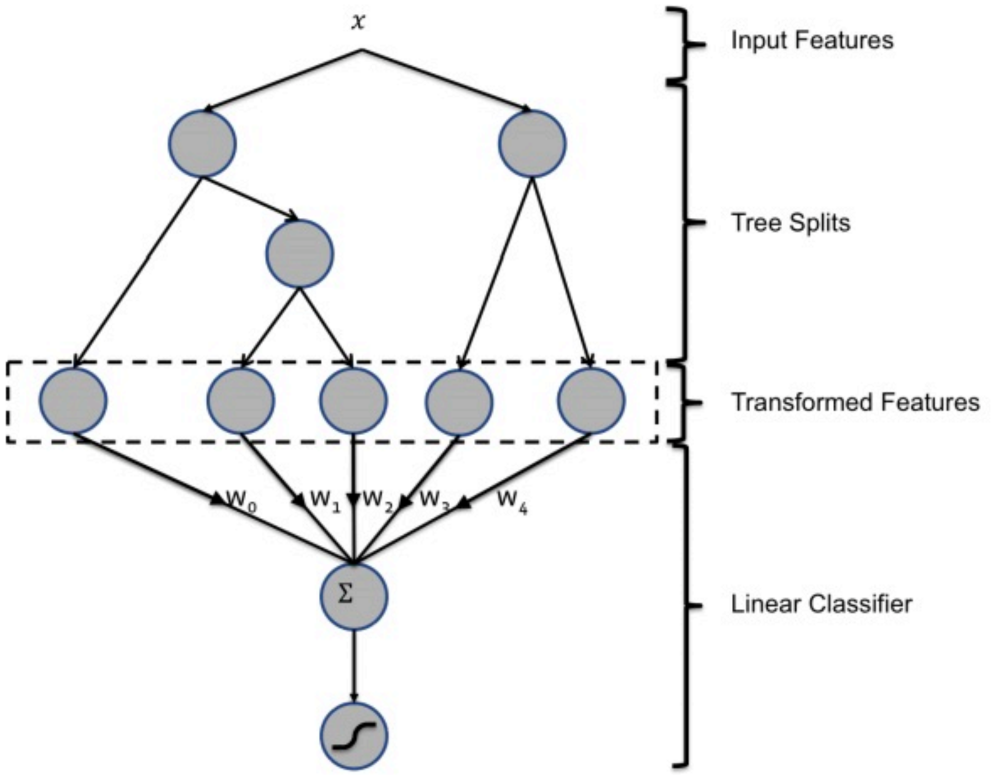
scale_pos_weight 正样本的权重 调节不平衡样本

XGBoost参数详解

tree_method: hist 节点分裂是算法的速度瓶颈 hist相当于LightGBM的实现

在categorical feature上XGboost的计算不如CatBoost

XGBoost生成新特征



Practical Lessons from Predicting Clicks on Ads at Facebook

XGBoost生成新特征

除了可以用逻辑回归在新特征上做分类，也可以使用其他模型

具有实战意义的模型有libfm
因为libfm擅长处理稀疏的特征，而决策树生成的特征比较稀疏

利用XGBoost生成的新特征做聚类分析
Topic Model能把句子按主题聚类
XGBoost对每个数据点生成的特征向量可以看做一个句子
这样可以很容易的获取数据的聚类结果

优势：不用做繁琐的数据预处理，数值型和类别型特征都可以统一处理

XGBoost代码演示

https://github.com/rfouyang/machine-learning.git

小结

集成类别	集成现有模型	边学边集成
Uniform	投票	Bagging, Random Forest
Non-uniform	线性组合	AdaBoost, Gradient Boost
Conditional	stacking	Decision Tree

谢谢大家

助教微信：aischool007