机器学习
第6节

涉及知识点：
决策树、ID3决策树、CART决策树、集成学习概念及应用、集成学习的组合模型、Bagging算法
Bagging算法有效性分析

树模型基础

张伟楠 – 上海交通大学

决策树

讲师：张伟楠 – 上海交通大学

目录

Contents

01

泛函空间优化

泛函空间优化

函数逼近

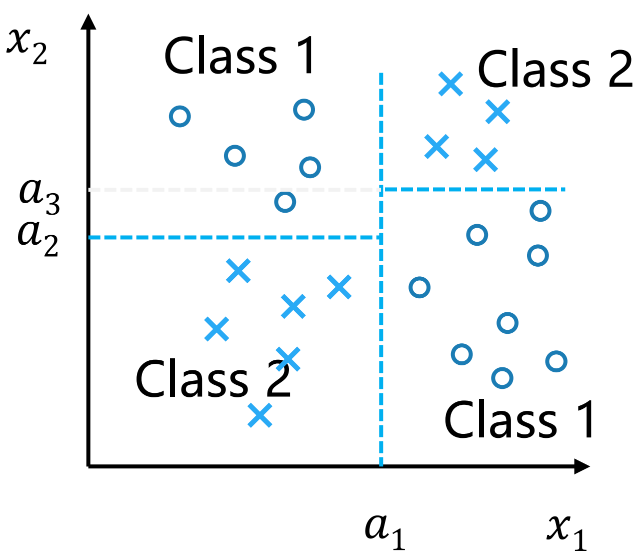
- 问题设定:
  - 样例特征空间 $\mathcal{X}$
  - 样例标签空间 $\mathcal{Y}$
  - 内在的未知映射函数（目标函数）$f: \mathcal{X} \mapsto \mathcal{Y}$
  - 函数假设集合$H = \{h | h: \mathcal{X} \mapsto \mathcal{Y}\}$
- 输入：由未知函数生成的训练数据

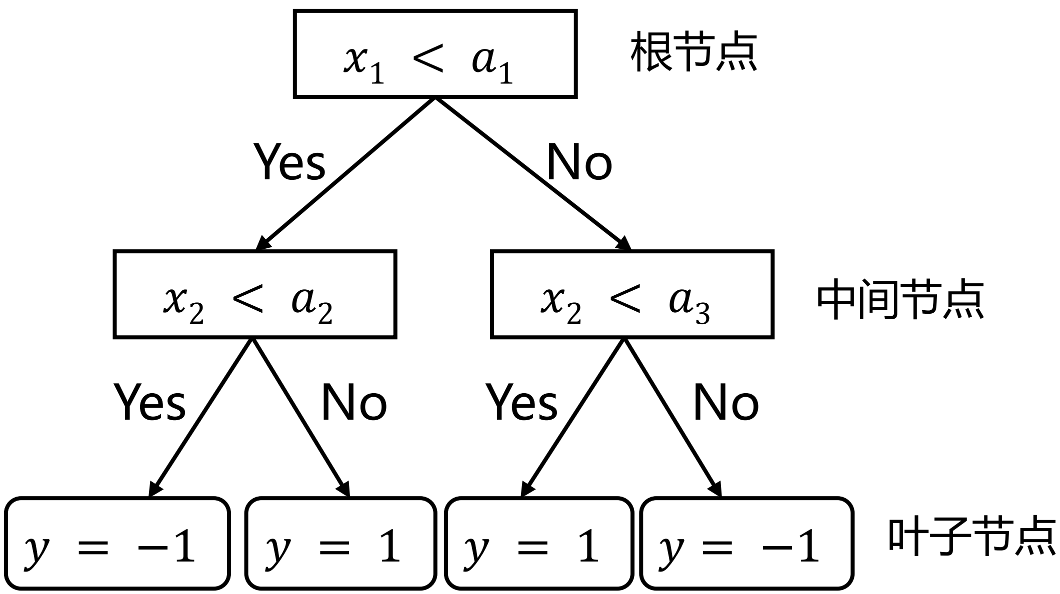
$$\left\{\left(x^{(i)}, y^{(i)}\right)\right\} = \left\{\left(x^{(1)}, y^{(1)}\right), \ldots, \left(x^{(n)}, y^{(n)}\right)\right\}$$

- 输出：对f的最佳逼近假设$h \in H$
- 优化：在泛函空间中进行，而不仅仅着眼于参数空间

泛函空间优化

- 树模型：
  - 中间节点用于分割数据
  - 叶子节点用于标签预测
- 连续数据举例

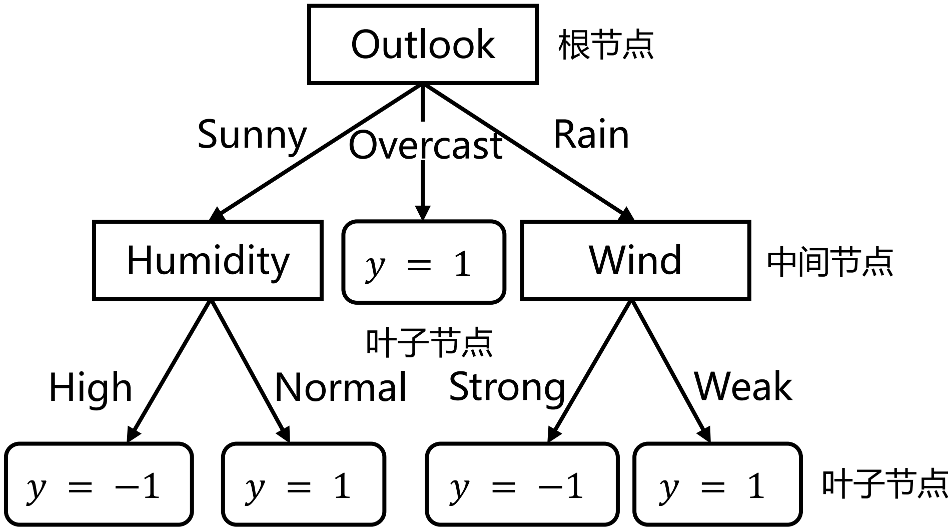




泛函空间优化

- 树模型：
  - 中间节点用于分割数据
  - 叶子节点用于标签预测
- 离散/类别数据举例

Predictors				Response
Outlook	**Temperature**	**Humidity**	**Wind**	**Class** **Play=Yes** **Play=No**
Sunny	Hot	High	Weak	No
Sunny	Hot	High	Strong	No
Overcast	Hot	High	Weak	Yes
Rain	Mild	High	Weak	Yes
Rain	Cool	Normal	Weak	Yes
Rain	Cool	Normal	Strong	No
Overcast	Cool	Normal	Strong	Yes
Sunny	Mild	High	Weak	No
Sunny	Cool	Normal	Weak	Yes
Rain	Mild	Normal	Weak	Yes
Sunny	Mild	Normal	Strong	Yes
Overcast	Mild	High	Strong	Yes
Overcast	Hot	Normal	Weak	Yes
Rain	Mild	High	Strong	No



泛函空间优化

函数逼近

- 问题设定:
  - 样例特征空间 $\mathcal{X}$
  - 样例标签空间 $\mathcal{Y}$
  - 内在的未知映射函数（目标函数）$f: \mathcal{X} \mapsto \mathcal{Y}$
  - 函数假设集合$H = \{h | h: \mathcal{X} \mapsto \mathcal{Y}\}$
- 输入：由未知函数生成的训练数据

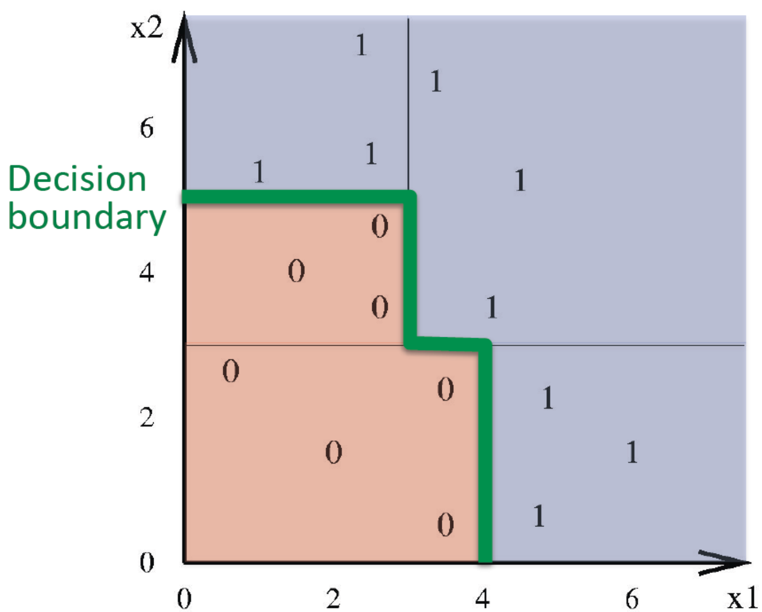
$$\{(x^{(i)}, y^{(i)})\} = \{(x^{(1)}, y^{(1)}), \dots, (x^{(n)}, y^{(n)})\}$$

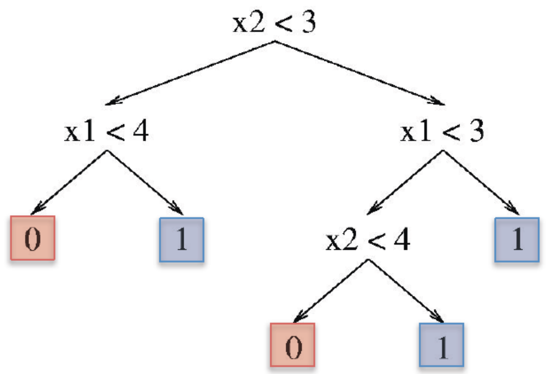
- 输出：对f的最佳逼近假设$h \in H$
- 在这里，每一个假设h为一棵决策树

泛函空间优化

决策边界

- 决策树将特征空间分割成与坐标轴平行的（超）矩形
- 每一个矩形区域由一个标签，或者不同标签的概率分布所标记





泛函空间优化

决策树研究历史

- 二十世纪六十年代，Hunt和同事们使用穷举搜索树（CLS）对人类的认知学习进行建模
- 七十年代末期，Quinlan发明了根据信息增益启发式搜索的ID3，从样例中学习专家系统
- 同时，Breiman、Friedman以及同事们发明了与ID3类似的分类回归树（CART）
- 二十世纪八十年代，许多关于决策树的改进被引入，用于处理噪声、连续特征、丢失特征以及分裂标准等，多种专家系统都得到了发展
- 在1993年， Quinlan更新并且发布了决策树程序包C4.5
- 程序包Sklearn（Python）Weka（Java）现在都包含了ID3以及C4.5

02

决策树

决策树

决策树

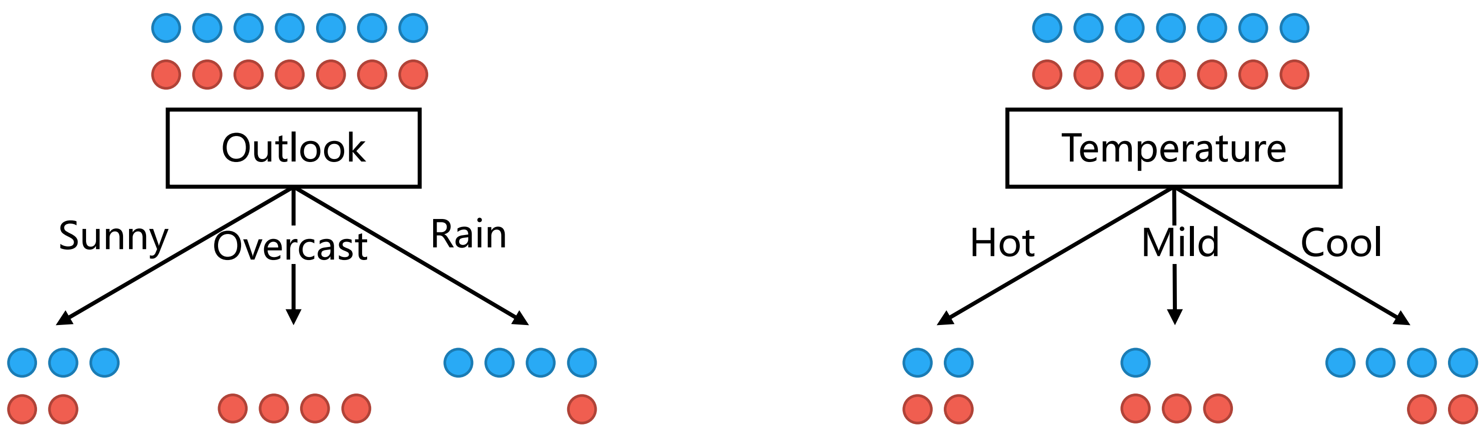
- 树模型：
  - 中间节点用于分割数据
  - 叶子节点用于标签预测

- 树模型的核心问题：
  - 如何选择分裂节点的条件?
  - 如何做出预测?
  - 如何决定树结构?

决策树

节点分裂

- 如何选择分裂节点的条件?



- 选择具有更强分类能力的特征
  - 量化地说，即具有更高的信息增益的特征

决策树

信息论基础

- （香农）熵（entropy）的含义是：接收的每条消息中包含的信息的平均量
  - 令X为包含n个离散取值的随机变量

$$P(X = x_i) = p_i$$

  - 其熵$H(X)$为

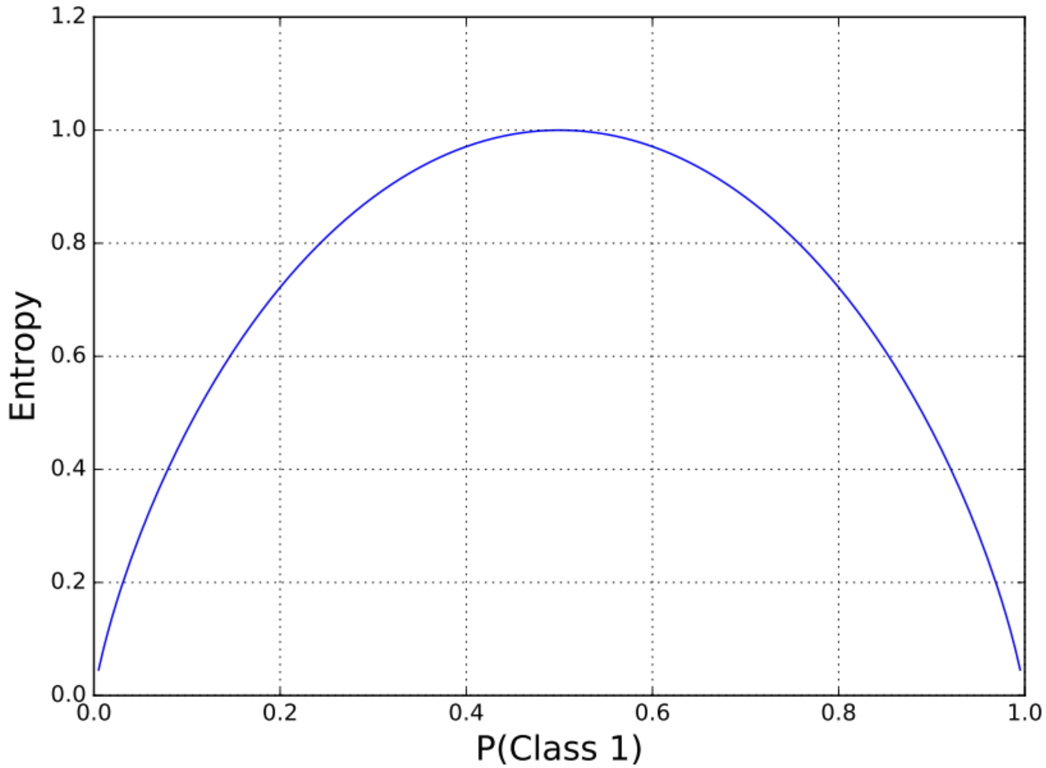
$$H(X) = -\sum_{i=1}^{n} p_i \log p_i$$

  - 容易证明

$$H(X) = -\sum_{i=1}^{n} p_i \log p_i \leq -\sum_{i=1}^{n} \frac{1}{n} \log \frac{1}{n} = \log n$$

决策树

熵的图示



- 二项分布的熵

$$H(X) = -p_1 \log p_1 - (1 - p_1)\log(1 - p_1)$$

决策树

交叉熵

- 交叉熵（cross entropy）用来度量两个随机变量分布之间的差异

$$H(X,Y) = -\sum_{i=1}^{n} P(X=i)\log P(Y=i)$$

- 连续情况下公式为

$$H(p,q) = -\int p(x)\log q(x)dx$$

- 与KL散度比较

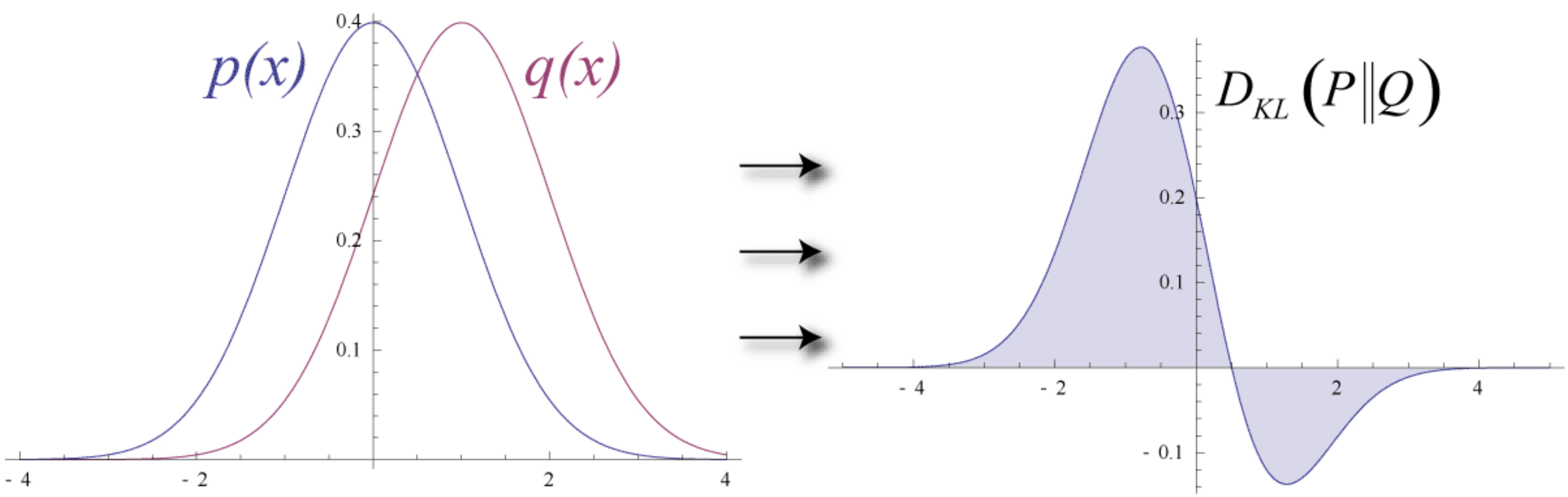
$$D_{KL}(p\|q) = \int p(x)\log\frac{p(x)}{q(x)}dx = H(p,q) - H(p)$$

决策树

KL散度

- KL散度（Kullback–Leibler divergence）也称相对熵（relative entropy），是两个概率分布之间差别的非对称性的度量

$$D_{KL}(p\|q)=\int p\,(x)\log\frac{p(x)}{q(x)}\,dx=H(p,q)-H(p)$$



Original Gaussian PDF's

KL Area to be Integrated

决策树

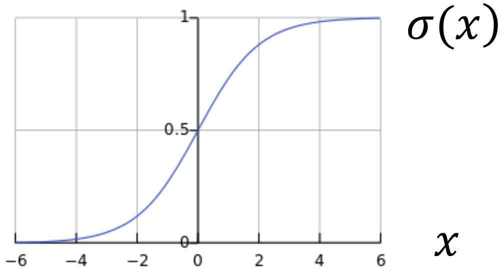
回顾逻辑回归中的交叉熵

$$\frac{\partial \sigma(z)}{\partial z} = \sigma(z)(1-\sigma(z))$$

- 逻辑回归是一种二分类模型

$$p_\theta(y=1|x) = \sigma(\theta^T x) = \frac{1}{1+e^{-\theta^T x}}$$

$$p_\theta(y=0|x) = \frac{e^{-\theta^T x}}{1+e^{-\theta^T x}}$$



- 交叉熵损失函数

$$\mathcal{L}(y,x,p_\theta) = -y\log\sigma(\theta^T x) - (1-y)\log\big(1-\sigma(\theta^T x)\big)$$

- 梯度函数

$$\frac{\partial \mathcal{L}(y,x,p_\theta)}{\partial \theta} = -y\frac{1}{\sigma(\theta^T x)}\sigma(z)(1-\sigma(z))x - (1-y)\frac{-1}{1-\sigma(\theta^T x)}\sigma(z)(1-\sigma(z))x$$

$$= (\sigma(\theta^T x) - y)x$$

$$\theta \leftarrow \theta + \eta\big(y - \sigma(\theta^T x)\big)x$$

决策树

条件熵

- 熵

$$H(X) = -\sum_{i=1}^{n} P(X=i)\log P(X=i)$$

- 在给定条件$Y=v$下，计算X的熵

$$H(X|Y=v) = -\sum_{i=1}^{n} P(X=i|Y=v)\log P(X=i|Y=v)$$

- 在给定条件Y下，计算X的熵

$$H(X|Y) = \sum_{v\in \text{values}(Y)} P(Y=v)\,H(X|Y=v)$$

- 给定条件Y下， X的信息增益（Information Gain)

$$\begin{aligned} I(X,Y) &= H(X) - H(X|Y) = H(Y) - H(Y|X) \\ &= H(X) + H(Y) - H(X,Y) \end{aligned}$$

此为(X,Y)的联合分布的熵，非X,Y的交叉熵

决策树

信息增益

- 给定条件Y下， X的信息增益

$$I(X,Y) = H(X) - H(X|Y)$$

$$= -\sum_v P(X=v)\log P(X=v) + \sum_u P(Y=u)\sum_v P(X=v|Y=u)\log P(X=v|Y=u)$$

$$= -\sum_v P(X=v)\log P(X=v) + \sum_u \sum_v P(X=v,Y=u)\log P(X=v|Y=u)$$

$$= -\sum_v P(X=v)\log P(X=v) + \sum_u \sum_v P(X=v,Y=u)[\log P(X=v,Y=u) - \log P(Y=u)]$$

$$= -\sum_v P(X=v)\log P(X=v) - \sum_u P(Y=u)\log P(Y=u) + \sum_{u,v} P(X=v,Y=u)\log P(X=v,Y=u)]$$

$$= H(X) + H(Y) - H(X,Y)$$

此为(X,Y)的联合分布的熵，非X,Y的交叉熵

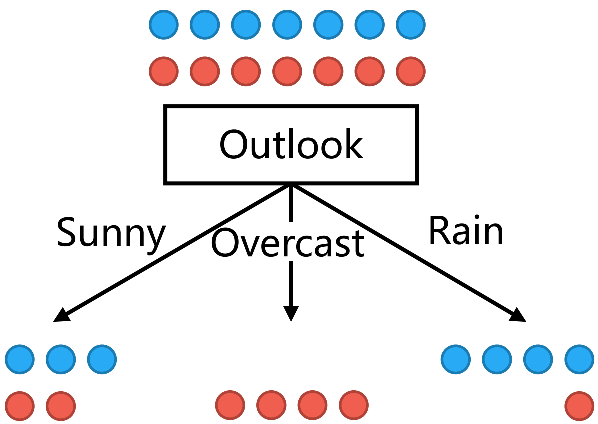
决策树

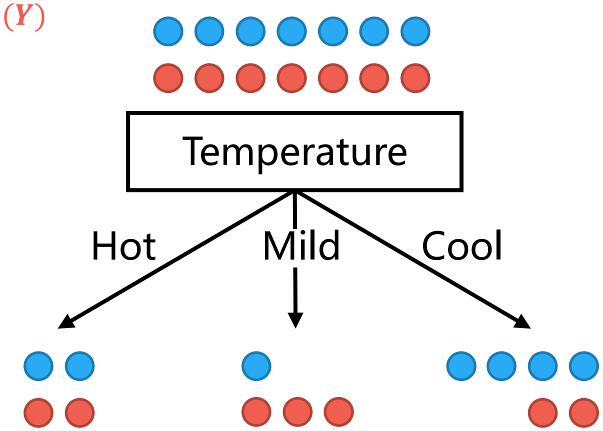
节点分裂

$$H(X|Y=v) = -\sum_{i=1}^{n} P(X=i|Y=v)\log P(X=i|Y=v)$$

- 信息增益

$$H(X|Y) = -\sum_{v\in \text{values}(Y)} P(Y=v)\, H(X|Y=v)$$





$$H(X|Y=S) = -\frac{3}{5}\log\frac{3}{5} - \frac{2}{5}\log\frac{2}{5} = 0.9710$$

$$H(X|Y=O) = -\frac{4}{4}\log\frac{4}{4} = 0$$

$$H(X|Y=R) = -\frac{4}{5}\log\frac{4}{5} - \frac{1}{5}\log\frac{1}{5} = 0.7219$$

$$H(X|Y) = \frac{5}{14}\times 0.9710 + \frac{4}{14}\times 0 + \frac{5}{14}\times 0.7219 = 0.6046$$

$$I(X,Y) = H(X) - H(X|Y) = 1 - 0.6046 = 0.3954$$

$$H(X|Y=H) = -\frac{2}{4}\log\frac{2}{4} - \frac{2}{4}\log\frac{2}{4} = 1$$

$$H(X|Y=M) = -\frac{1}{4}\log\frac{1}{4} - \frac{3}{4}\log\frac{3}{4} = 0.8118$$

$$H(X|Y=C) = -\frac{4}{6}\log\frac{4}{6} - \frac{2}{6}\log\frac{2}{6} = 0.9183$$

$$H(X|Y) = \frac{4}{14}\times 1 + \frac{4}{14}\times 0.8113 + \frac{5}{14}\times 0.9183 = 0.9111$$

$$I(X,Y) = H(X) - H(X|Y) = 1 - 0.9111 = 0.0889$$

决策树

信息增益率

- 信息增益与熵之间的比值

$$I_R(X,Y)=\frac{I(X,Y)}{H_Y(X)}=\frac{H(X)-H(X|Y)}{H_Y(X)}$$

- 在条件Y之下X被分开的熵为

$$H_Y(X)=-\sum_{v\in \text{values}(Y)}\frac{|X_{y=v}|}{|X|}\log\frac{|X_{y=v}|}{|X|}$$

  - 其中， $|X_{y=v}|$为给定条件$y=v$时，得到观测值 X的数目

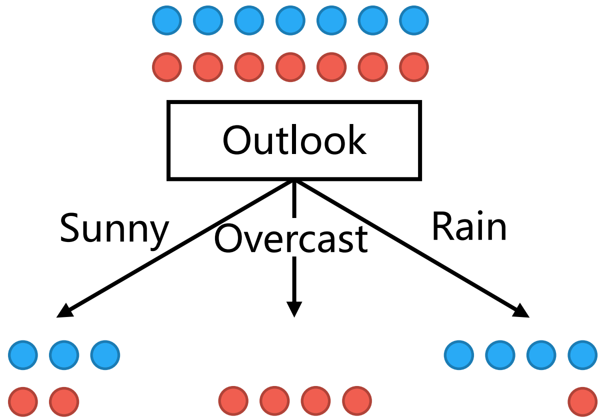
- 如此，避免Y仅仅是把数据分得很细但对X的区分模式并不有效的情况

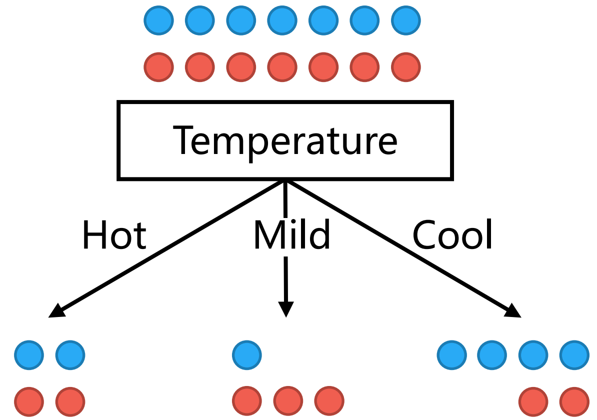
决策树

节点分裂

- 信息增益率

$$I_R(X,Y)=\frac{I(X,Y)}{H_Y(X)}=\frac{H(X)-H(X|Y)}{H_Y(X)}$$





$$I(X,Y)=H(X)-H(X|Y)=1-0.6046=0.3954$$

$$H_Y(X)=-\frac{5}{14}\log\frac{5}{14}-\frac{4}{14}\log\frac{4}{14}-\frac{5}{14}\log\frac{5}{14}=1.5774$$

$$I_R(X,Y)=\frac{0.3954}{1.5774}=0.2507$$

$$I(X,Y)=H(X)-H(X|Y)=1-0.9111=0.0889$$

$$H_Y(X)=-\frac{4}{14}\log\frac{4}{14}-\frac{4}{14}\log\frac{4}{14}-\frac{6}{14}\log\frac{6}{14}=1.5567$$

$$I_R(X,Y)=\frac{0.0889}{1.5567}=0.0571$$

ID3决策树

讲师：张伟楠 – 上海交通大学

ID3算法

决策树的构建

算法框架

从包含所有数据的根节点开始

- 对于每一个节点，计算所有可能特征带来的信息增益
- 选择具有最大信息增益的特征
- 根据选择的特征分割这一节点上的数据

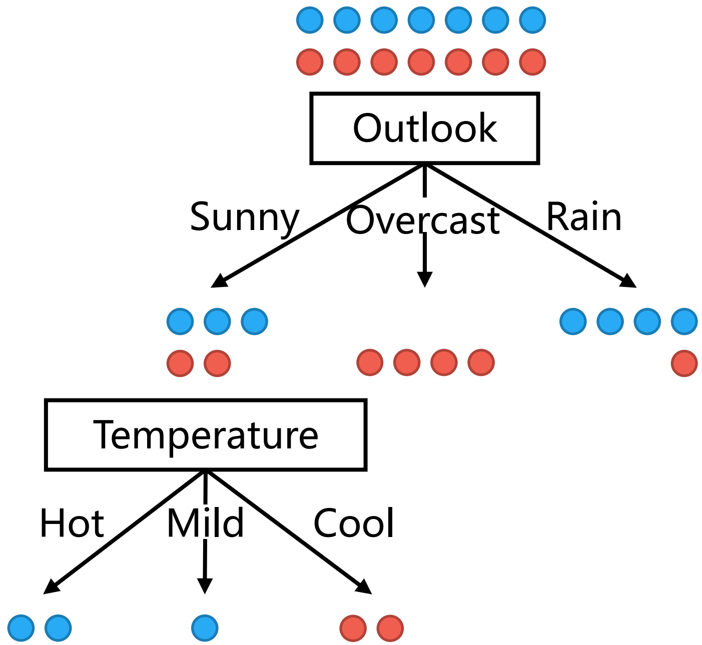
对每个叶子节点递归进行上述步骤，直至

1. 叶子节点不再有信息增益
2. 没有更多的特征可以被选取

ID3算法

决策树的构建

- ID3算法构建的决策树

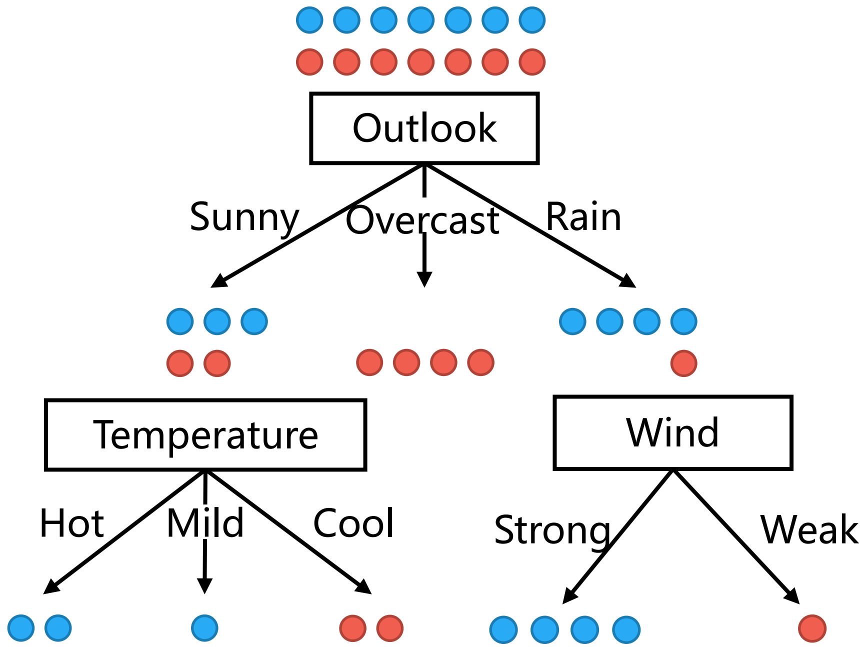


- 一个特征最多会出现在一条路径中一次

ID3算法

决策树的构建

- ID3算法构建的决策树

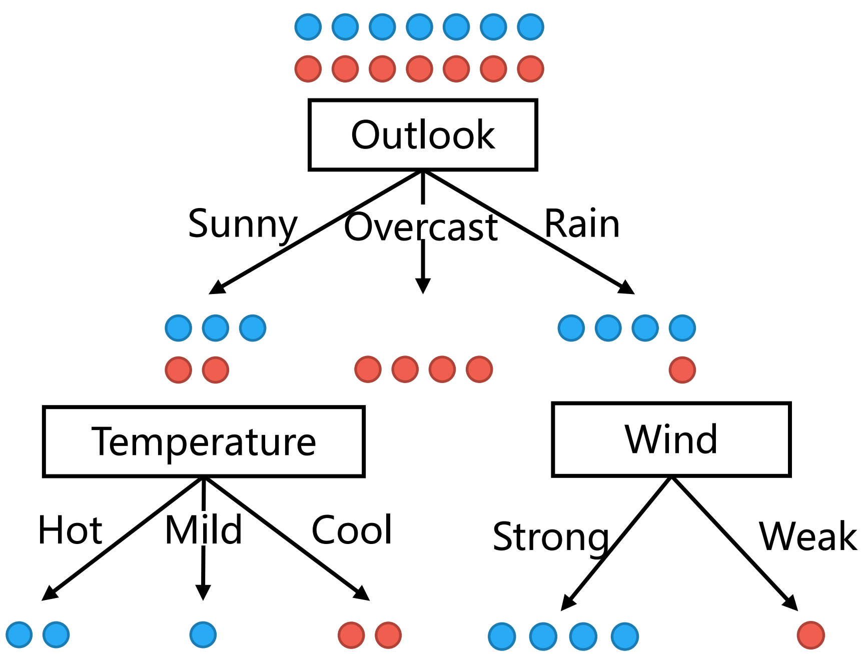


- 这棵树的划分是最优的么?

ID3算法

过拟合

- 通过对每一个样例生成一个叶子节点，树模型可以近似任何有限数据



ID3算法

目标函数

- 利用训练数据构建树T的损失函数

$$C(T) = \sum_{t=1}^{|T|} N_t H_t(T)$$

  - 对于叶子节点t
    - $H_t(T)$为经验熵
    - N_t为节点样例总数目，N_{tk}为类别k中的样例数目

$$H_t(T) = -\sum_k \frac{N_{tk}}{N_t} \log \frac{N_{tk}}{N_t}$$

- 训练目标函数：寻找一棵树能够最小化损失函数

$$\min_T C(T) = \sum_{t=1}^{|T|} N_t H_t(T)$$

ID3算法

决策树的正则化

- 利用训练数据构建树T的损失函数

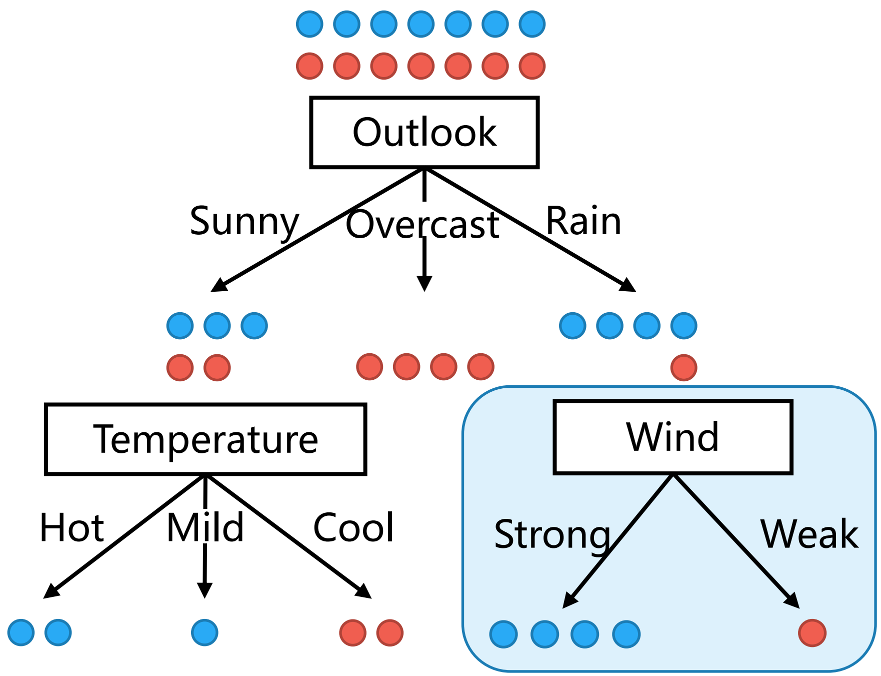
$$C(T)=\sum_{t=1}^{|T|}N_t\,H_t(T)+\lambda|T|$$

- 其中
  - $|T|$为树T的叶子节点数目
  - λ为正则化的超参数

ID3算法

决策树的构建

- ID3算法构建的决策树



是否对该节点进行分裂?

- 计算并且比较损失函数的大小

$$C(T)=\sum_{t=1}^{|T|}N_t H_t(T)+\lambda|T|$$

ID3算法

ID3总结

- ID3算法是一种传统并且直观的决策树训练算法
  - 针对离散的类别型数据
  - 一个特征数值或类别构成一个分支
- C4.5算法与ID3算法相似但是C4.5算法做了改进
  - 根据信息增益率进行节点分裂
- 分裂出来的树枝数目取决于特征中不同类别取值的数量
  - 可能会产生很宽的树结构

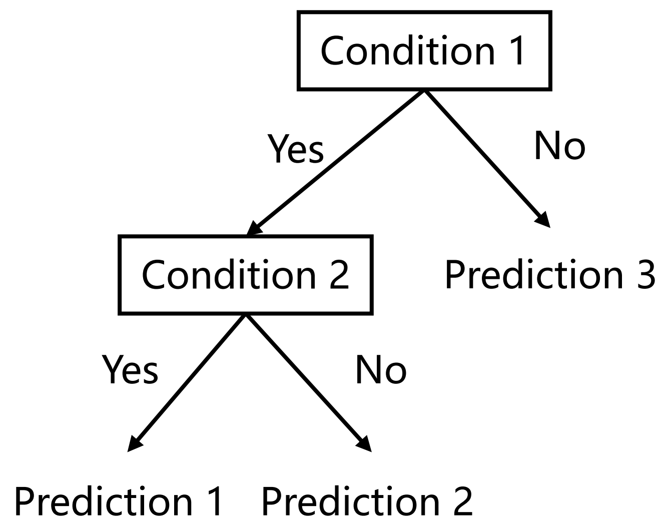
CART决策树

讲师：张伟楠 – 上海交通大学

CART算法

算法简介

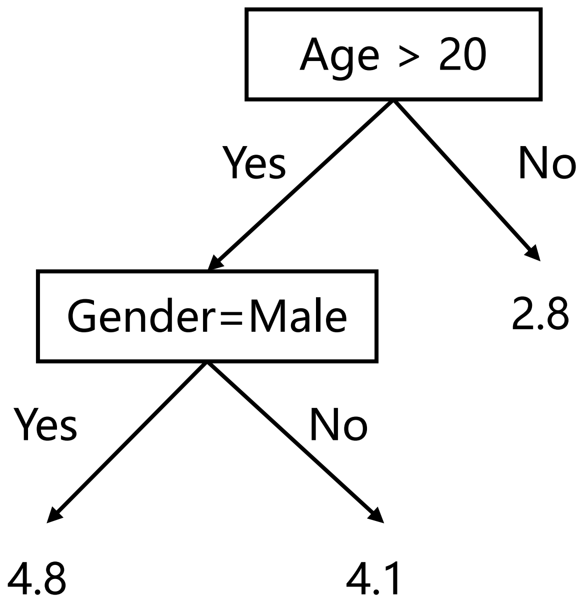
- 分类和回归树（CART）：
  - 在1984年，由Leo Breiman提出
  - 二值分裂，对于分裂条件仅取yes或者no
  - 对于连续数值特征同样有效
  - 当采取不同分裂条件时，可以重复使用相同的特征



CART算法

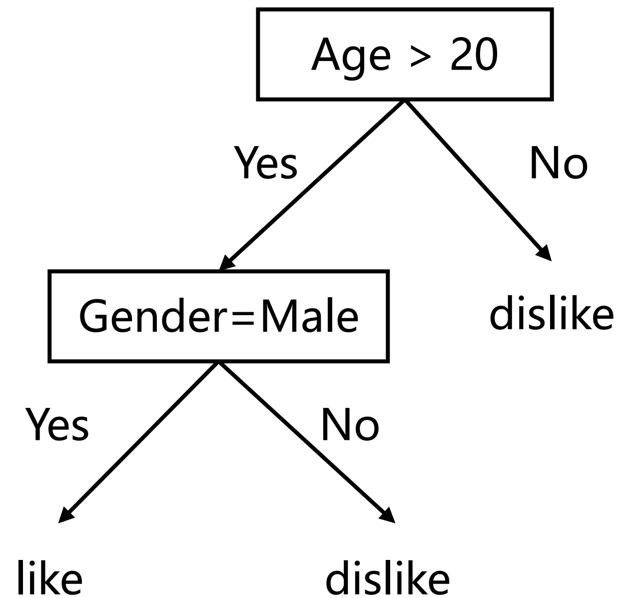
算法简介

- 回归树：
  - 输出预测数值



举例：预测用户对于电影的打分

- 分类树：
  - 输出预测类别



举例：预测用户是否喜欢一部电影

CART算法

回归树

- 对于目标为连续值y的训练数据集

$$D = \{(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)\}$$

- 假设回归树对于空间进行了 M 个区域的划分，记为 $R_1, R_2, \dots, R_M$ ，其中c_m为区域R_m的预测值

$$f(x) = \sum_{m=1}^{M} c_m I(x \in R_m)$$

- (x_i, y_i)的损失函数

$$\frac{1}{2}\left(y_i - f(x_i)\right)^2$$

- 容易得出，区域m的最优预测为

$$\hat{c}_m = \mathrm{avg}(y_i | x_i \in R_m)$$

CART算法

回归树

- 如何得到最优的分割区域?
- 如何寻找最优的分割条件?
  - 在变量j上，定义一个阈值s
  - 即可分割出两个区域

$$R_1(j,s)=\{x|x^{(j)}\leq s\}\quad R_2(j,s)=\{x|x^{(j)}>s\}$$

- 基于当前分割进行训练

$$\min_{j,s}[\min_{c_1}\sum_{x_i\in R_1(j,s)}(y_i-c_1)^2+\min_{c_2}\sum_{x_i\in R_2(j,s)}(y_i-c_2)^2]$$

$$\hat{c}_m=\mathrm{avg}(y_i|x_i\in R_m)$$

CART算法

回归树算法

- 输入：训练数据D
- 输出：回归树$f(x)$
- 重复至满足停止条件：
  - 找到最优分割(j,s)

$$\min_{j,s}\left[\min_{c_1}\sum_{x_i\in R_1(j,s)}(y_i-c_1)^2+\min_{c_2}\sum_{x_i\in R_2(j,s)}(y_i-c_2)^2\right]$$

  - 计算新的区域R_1, R_2的预测值

$$\hat{c}_m=\operatorname{avg}(y_i|x_i\in R_m)$$

- 返回回归树

$$f(x)=\sum_{m=1}^{M}c_m I(x\in R_m)$$

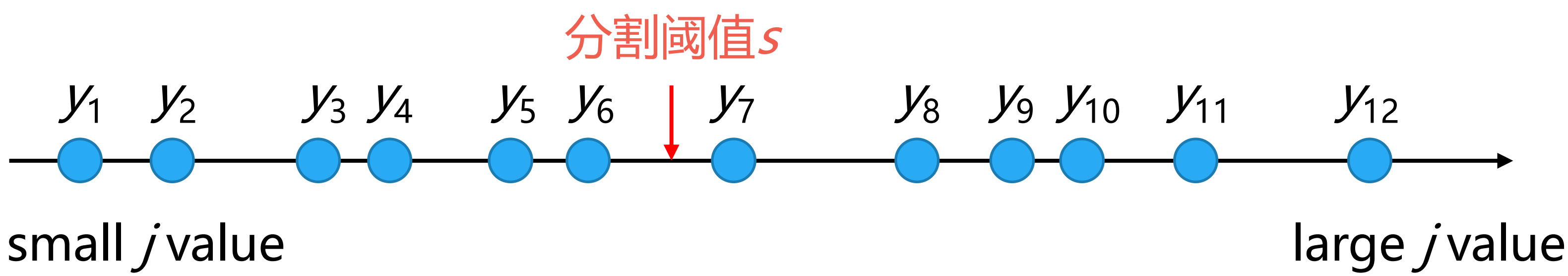
CART算法

回归树算法

- 如何高效寻找最优分割(j,s)?

$$\min_{j,s}\left[\min_{c_1}\sum_{x\in R_1(j,s)}(y_i-c_1)^2+\min_{c_2}\sum_{x\in R_2(j,s)}(y_i-c_2)^2\right]$$

- 根据特征j的数值对数据进行升序的排序



在线更新

$$\text{loss}=\sum_{i=1}^{6}(y_i-c_1)^2+\sum_{i=7}^{12}(y_i-c_2)^2$$

$$=\sum_{i=1}^{6}y_i^2-\frac{1}{6}\left(\sum_{i=1}^{6}y_i\right)^2+\sum_{i=7}^{12}y_i^2-\frac{1}{6}\left(\sum_{i=7}^{12}y_i\right)^2=-\frac{1}{6}\left(\sum_{i=1}^{6}y_i\right)^2-\frac{1}{6}\left(\sum_{i=7}^{12}y_i\right)^2+C$$

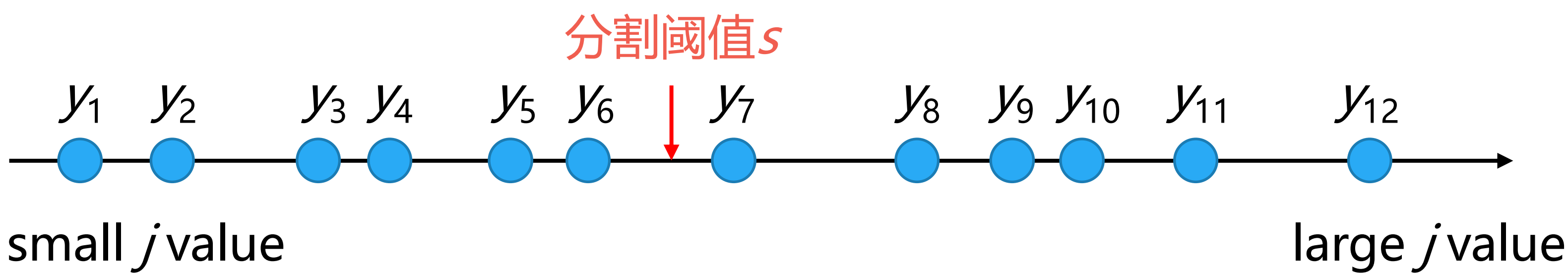
CART算法

回归树算法

- 如何高效寻找最优分割(j,s)?

$$\min_{j,s}\left[\min_{c_1}\sum_{x\in R_1(j,s)}(y_i-c_1)^2+\min_{c_2}\sum_{x\in R_2(j,s)}(y_i-c_2)^2\right]$$

- 根据特征j的数值对数据进行升序的排序



$$\text{loss}_{6,7}=-\frac{1}{6}\left(\sum_{i=1}^{6}y_i\right)^2-\frac{1}{6}\left(\sum_{i=7}^{12}y_i\right)^2+C$$

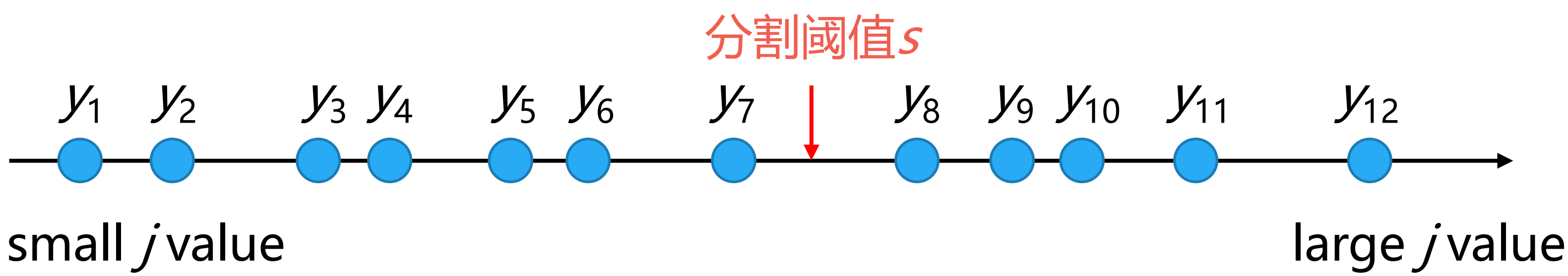
CART算法

回归树算法

- 如何高效寻找最优分割(j, s)?

$$\min_{j,s}\left[\min_{c_1}\sum_{x\in R_1(j,s)}(y_i-c_1)^2+\min_{c_2}\sum_{x\in R_2(j,s)}(y_i-c_2)^2\right]$$

- 根据特征j的数值对数据进行升序的排序



$$\text{loss}_{6,7}=-\frac{1}{6}\left(\sum_{i=1}^{6}y_i\right)^2-\frac{1}{6}\left(\sum_{i=7}^{12}y_i\right)^2+C$$

$$\text{loss}_{7,8}=-\frac{1}{7}\left(\sum_{i=1}^{7}y_i\right)^2-\frac{1}{5}\left(\sum_{i=8}^{12}y_i\right)^2+C$$

- 维护以及在线更新仅需$O(1)$时间

$$\text{Sum}(R_1)=\sum_{i=1}^{k}y_i\quad \text{Sum}(R_2)=\sum_{i=k+1}^{n}y_i$$

- 对于整个特征的检验需要$O(n)$时间

CART算法

分类树

- 对于目标为类别值y的训练数据集

$$D = \{(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)\}$$

- 假设分类树对于空间进行了M个区域的划分，记为$R_1, R_2, \dots, R_M$，其中c_m为区域R_m的预测

$$f(x) = \sum_{m=1}^{M} c_m I(x \in R_m)$$

- 这里叶子节点的预测c_m是类别的分布

$$\hat{c}_m = \{P(y_k | x_i \in R_m)\}_{k=1\dots K}$$

- c_m可以由类别计数进行求解

$$P(y_k | x_i \in R_m) = \frac{C_m^k}{C_m}$$

叶子m上k类别的样例数目

叶子m上的样例数目

CART算法

分类树

- 如何得到最优的分割区域?
- 如何寻找最优的分割条件?
  - 对于连续特征j，定义一个阈值s
  - 即可分割出两个区域

$$R_1(j,s)=\{x|x^{(j)}\le s\}\quad R_2(j,s)=\{x|x^{(j)}>s\}$$

  - 对于类别特征j，选择一个类别a
  - 即可分割出两个区域

$$R_1(j,a)=\{x|x^{(j)}=a\}\quad R_2(j,a)=\{x|x^{(j)}\ne a\}$$

  - 如何选择？基于基尼不纯度（Gini Impurity）

CART算法

基尼不纯度

- 在分类问题中
  - 假设共有K个类别
  - 令p_k为一个样例是第k类的概率
  - 基尼不纯度为

$$\mathrm{Gini}(p)=\sum_{k=1}^{K}p_k\left(1-p_k\right)=1-\sum_{k=1}^{K}p_k^2$$

- 对于给定的训练数据集D，基尼不纯度为

$$\mathrm{Gini}(D)=1-\sum_{k=1}^{K}\left(\frac{|D^k|}{|D|}\right)^2$$

数据集D上k类别的样例数目

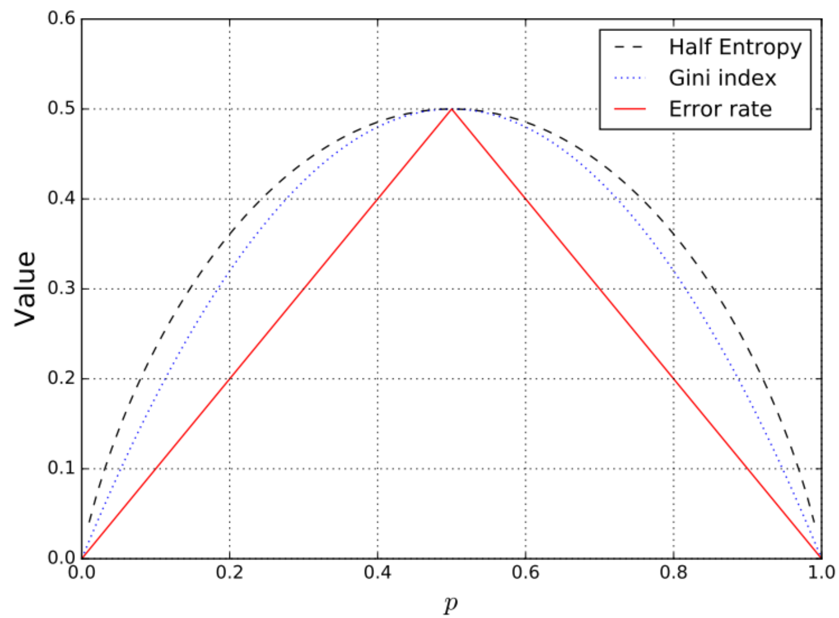
数据集D上的样例数目

CART算法

基尼不纯度

- 在二分类问题中
  - 令p为一个样例是第1类的概率
  - 基尼不纯度为 $\mathrm{Gini}(p) = 2p(1-p)$
  - 熵为 $H(X) = -p\log p - (1-p)\log(1-p)$

基尼不纯度与熵在图示的分类错误率上十分近似



CART算法

基尼不纯度

- 对于类别特征j以及其中一个类别a
  - 两个分割区域R_1, R_2

$$R_1(j,a) = \{x | x^{(j)} = a\} \quad R_2(j,a) = \{x | x^{(j)} \neq a\}$$

$$D_j^1(j,a) = \{(x,y) | x^{(j)} = a\} \quad D_j^2(j,a) = \{(x,y) | x^{(j)} \neq a\}$$

  - 在类别特征j中，选择类别a的基尼不纯度为

$$\mathrm{Gini}(D_j, j = a) = \frac{|D_j^1|}{|D_j|}\mathrm{Gini}(D_j^1) + \frac{|D_j^2|}{|D_j|}\mathrm{Gini}(D_j^2)$$

CART算法

分类树算法

- 输入：训练数据D
- 输出：分类树$f(x)$
- 重复至满足停止条件：
  - 找到最优分割(j, a)

$$\min_{j,a} \text{Gini}\left(D_j, j = a\right)$$

  - 计算新的区域R_1, R_2的预测分布

$$\hat{c}_m = \{P(y_k | x_i \in R_m)\}_{k=1\ldots K}$$

- 返回分类树

$$f(x) = \sum_{m=1}^{M} \hat{c}_m I(x \in R_m)$$

1. 节点样例数很少
2. 基尼不纯度很小
3. 没有多余的特征

CART算法

分类树输出

- 类别标签的输出
  - 输出具有最大条件概率的类别

$$f(x) = \underset{y_k}{\operatorname{argmax}} \sum_{m=1}^{M} I\,(x \in R_m)P(y_k|x_i \in R_m)$$

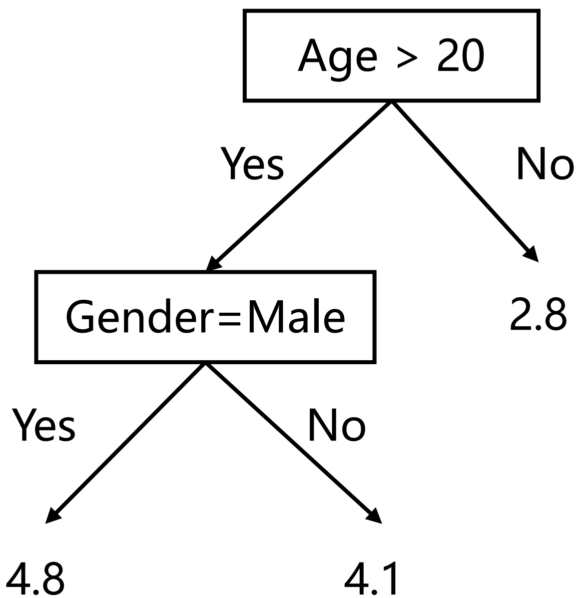
- 概率分布输出

$$f(x) = \sum_{m=1}^{M} \hat{c}_m\, I(x \in R_m)$$

$$\hat{c}_m = \{P(y_k|x_i \in R_m)\}_{k=1\ldots K}$$

CART算法

树与规则的转换




```
IF Age > 20:
   IF Gender == Male:
      return 4.8
   ELSE:
      return 4.1
ELSE:
   return 2.8
```

举例：预测用户对于电影的打分

决策树模型易于可视化、原理解释以及错误调试

CART算法

学习模型比较

Characteristic	Neural Nets	SVM	Trees	MARS	k-NN, Kernels
Natural handling of data of "mixed" type	▼	▼	▲	▲	▼
Handling of missing values	▼	▼	▲	▲	▲
Robustness to outliers in input space	▼	▼	▲	▼	▲
Insensitive to monotone transformations of inputs	▼	▼	▲	▼	▼
Computational scalability (large N)	▼	▼	▲	▲	▼
Ability to deal with irrelevant inputs	▼	▼	▲	▲	▼
Ability to extract linear combinations of features	▲	▲	▼	▼	◆
Interpretability	▼	▼	◆	▲	▼
Predictive power	▲	▲	▼	◆	▲

[Table 10.3 from Hastie et al. Elements of Statistical Learning, 2nd Edition]

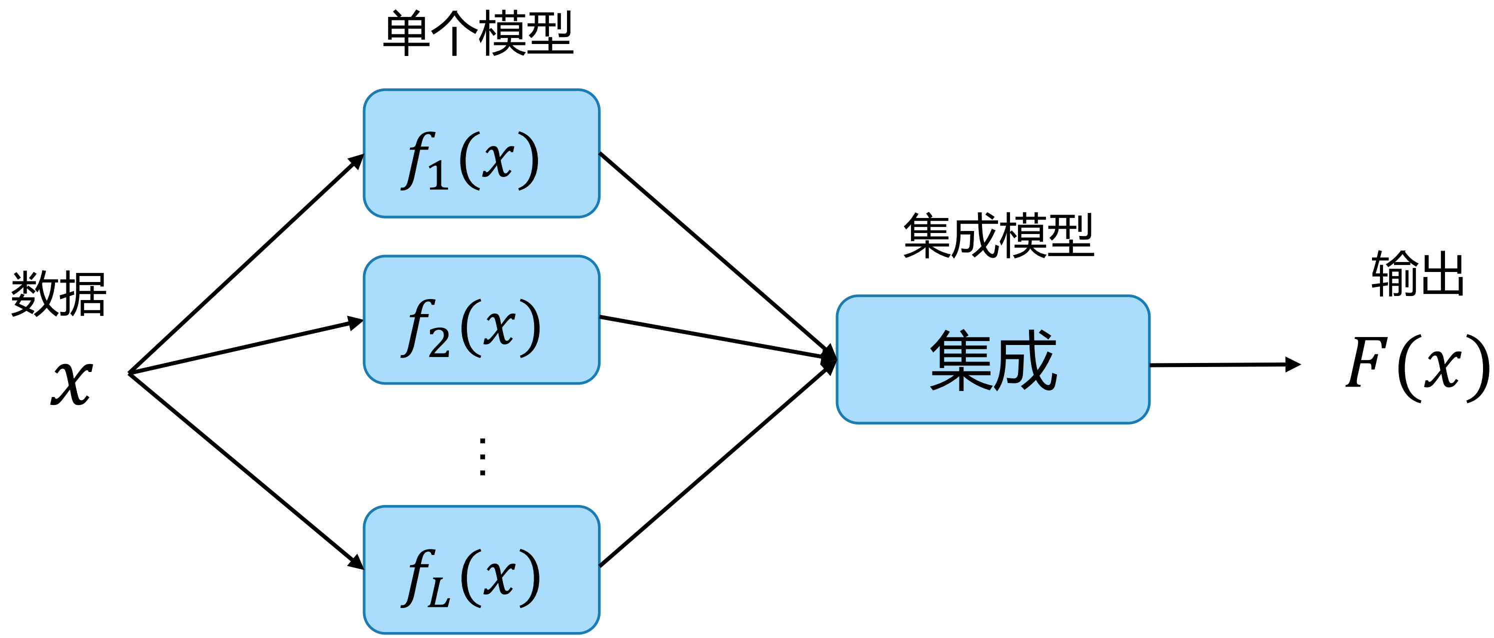
集成学习概念及应用

张伟楠 – 上海交通大学

集成学习

- 考虑一组预测模型 $f_1,\ldots,f_L$
  - 不同的预测模型在数据上具有不同的表现
- 想法：构造预测模型 $F(x)$ 将 $f_1,\ldots,f_L$ 的单独预测结合起来
  - 例如，可以对集合中的预测模型的预测进行投票
  - 例如，可以为数据空间的不同区域使用不同的预测模型
  - 如果每个预测模型的错误率都很低，那么效果会很好
- 成功的集成需要多样性
  - 预测模型应该犯的错误不相同
  - 鼓励把不同类型的预测模型集成在一起

集成学习



虽然复杂，集成学习可能会提供最可靠的输出结果和最佳的经验表现

在竞赛中的实际应用

- Netflix奖竞赛
  - 任务：根据某些用户对某些影片的评分来预测用户对影片的评分
  - “协同过滤”

- 优胜者的解决方案
  - 超过800个预测模型进行集成

Yehuda Koren

[Yehuda Koren. The BellKor Solution to the Netflix Grand Prize. 2009.]

在竞赛中的实际应用

- KDD-Cup 2011年雅虎音乐推荐
  - 任务：根据某些用户对某些音乐的评分来预测用户对音乐的评分
    - 包含音乐信息,比如专辑、艺术家、流派ID
- 优胜者的解决方案
  - 台湾国立大学的研究生课程：将221个预测模型进行集成

A Linear Ensemble of Individual and Blended Models for Music Rating Prediction

Po-Lung Chen, Chen-Tse Tsai, Yao-Nan Chen, Ku-Chun Chou, Chun-Liang Li, Cheng-Hao Tsai, Kuan-Wei Wu, Yu-Cheng Chou, Chung-Yi Li, Wei-Shih Lin, Shu-Hao Yu, Rong-Bing Chiu, Chieh-Yen Lin, Chien-Chih Wang, Po-Wei Wang, Wei-Lun Su, Chen-Hung Wu, Tsung-Ting Kuo, Todd G. McKenzie, Ya-Hsuan Chang, Chun-Sung Ferng, Chia-Mau Ni, Hsuan-Tien Lin, Chih-Jen Lin, Shou-De Lin

{R99922038, R98922028, R99922008, R99922095, B97018, B97705004, B96018, B96115, B96069, B96113, B95076, B97114, B97042, D98922007, B97058, B96110, B96055, D97944007, D97041, B96025, R99922054, B96092, HTLIN, CJLIN, SDLIN}@CSIE.NTU.EDU.TW

Department of Computer Science and Information Engineering,
National Taiwan University

在竞赛中的实际应用

- KDD-Cup 2011年雅虎音乐推荐
  - 任务：根据某些用户对某些音乐的评分来预测用户对音乐的评分
    - 包含音乐信息,比如专辑、艺术家、流派ID
- 第三名的解决方案：
  - SJTU-HKUST联合团队：16个预测模型进行集成

Informative Ensemble of Multi-Resolution Dynamic Factorization Models

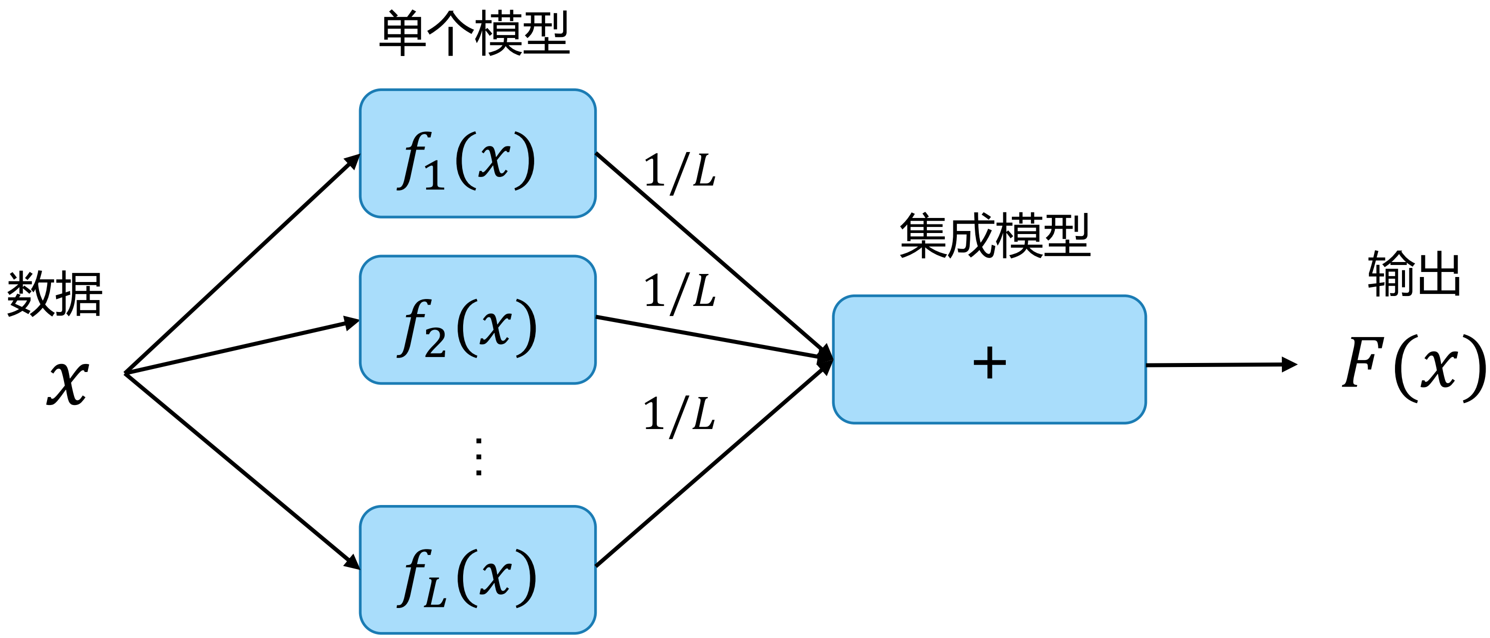
Tianqi Chen*, Zhao Zheng, Qiuxia Lu, Xiao Jiang, Yuqiang Chen, Weinan Zhang
Kailong Chen and Yong Yu
Shanghai Jiao Tong University
800 Dongchuan Road, Shanghai 200240 China
{tqchen, zhengzhao,luqiuxia,jiangxiao,yuqiangchen,wnzhang,chenkl,yyu}@apex.sjtu.edu.cn

Nathan N. Liu†, Bin Cao, Luheng He and Qiang Yang
Hong Kong University of Science and Technology
Clear Water Bay, Hong Kong
{nliu,caobin,luhenghe,qyang}@cse.ust.hk

集成学习的组合模型

张伟楠 – 上海交通大学

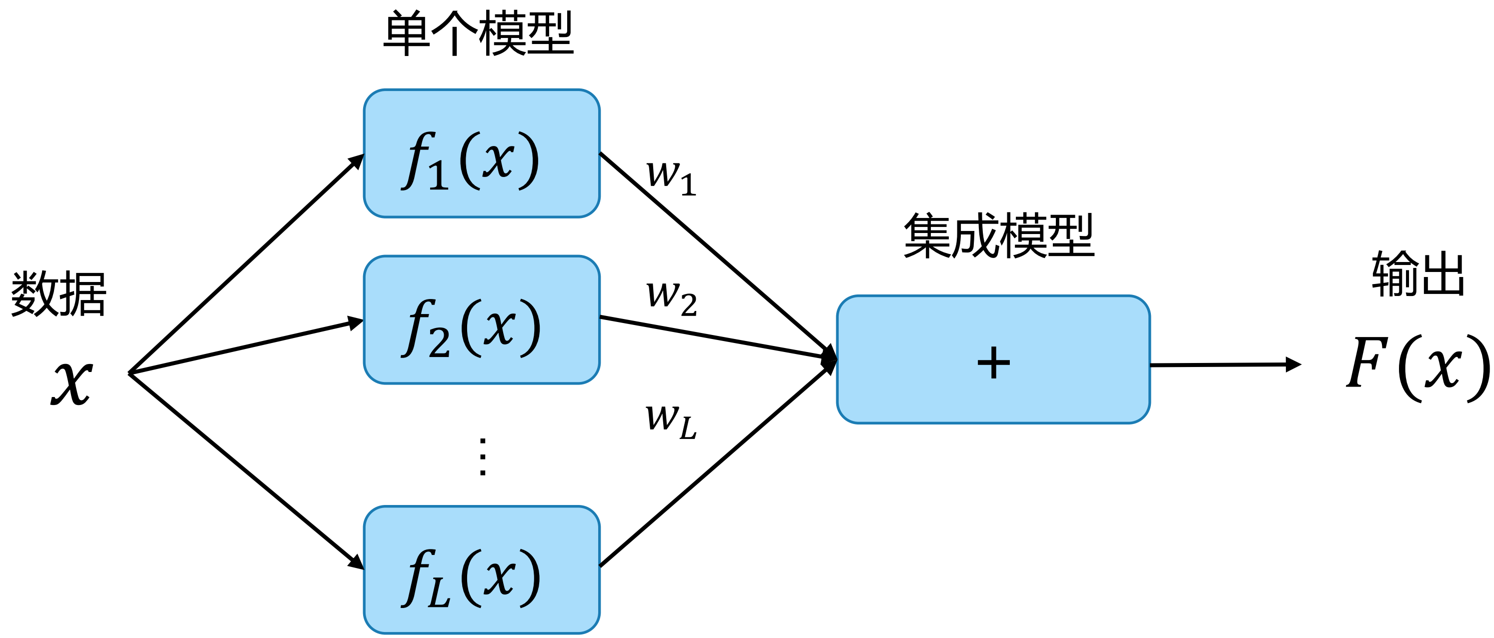
组合模型：平均



$$F(x) = \frac{1}{L}\sum_{i=1}^{L} f_i(x)$$

- 回归问题求平均值，分类问题进行投票

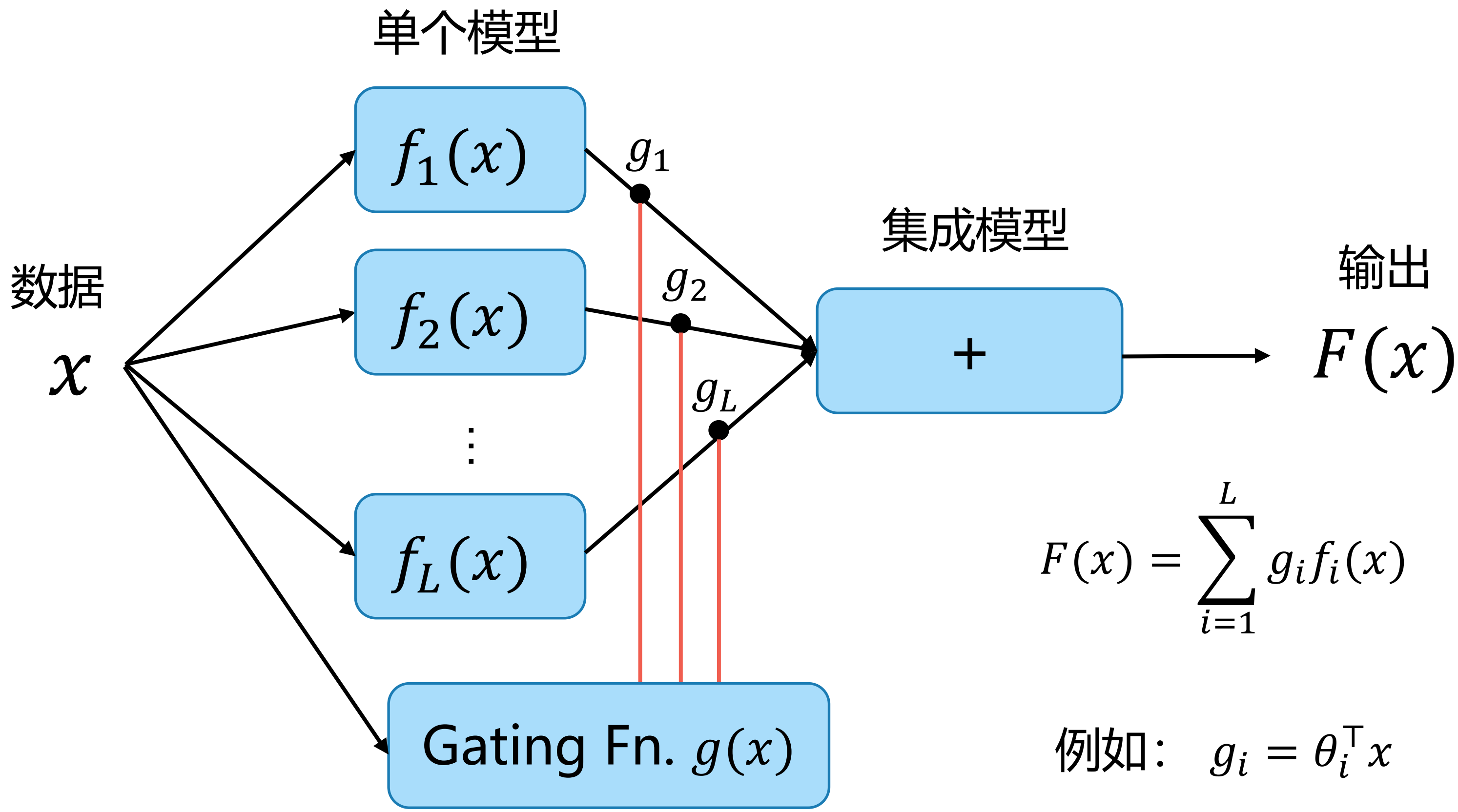
组合模型：带权平均



$$F(x) = \sum_{i=1}^{L} w_i f_i(x)$$

- 像线性回归或分类
- 注意：训练集成模型时不会更新单个模型

组合模型：门控



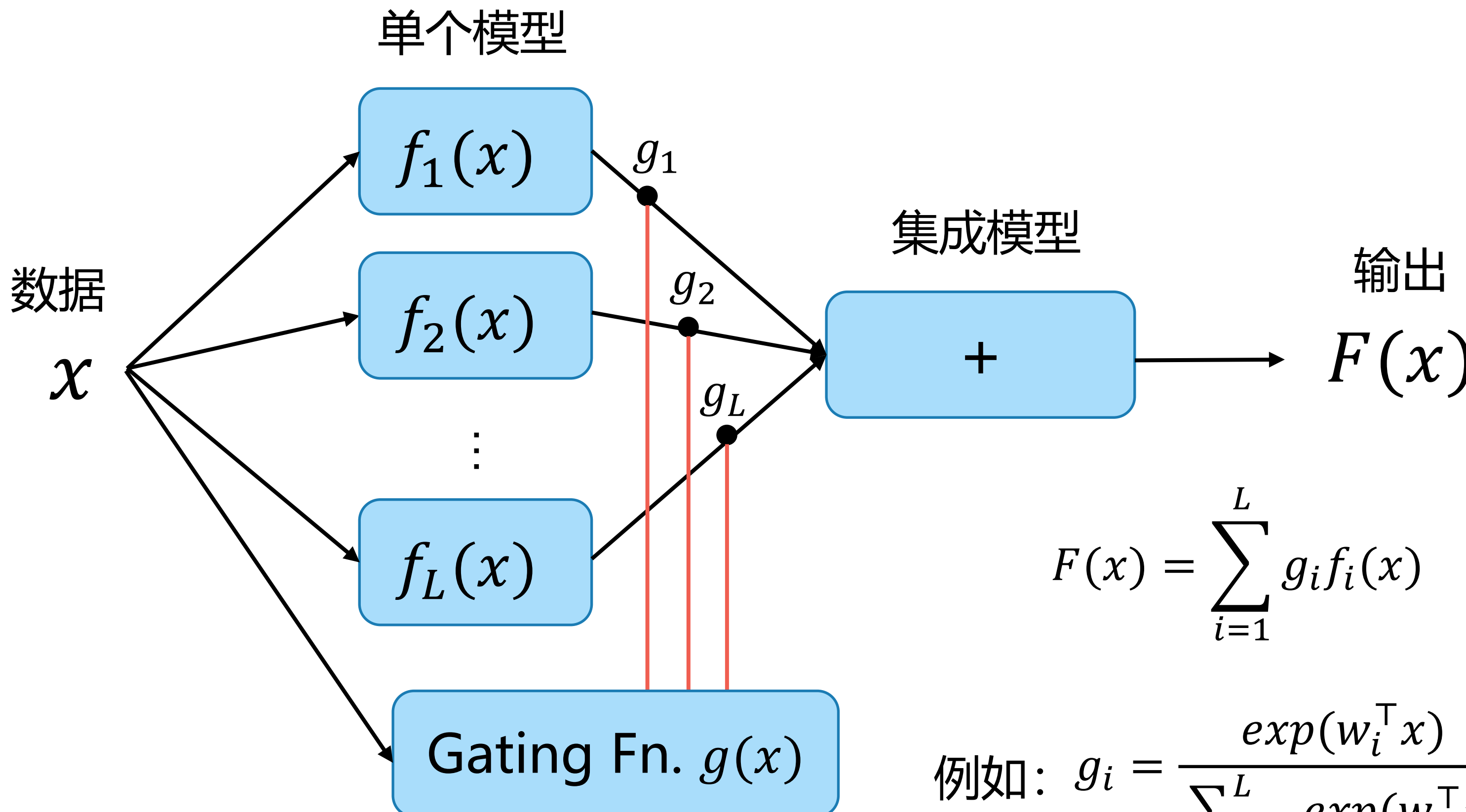
$$F(x) = \sum_{i=1}^{L} g_i f_i(x)$$

例如：$g_i = \theta_i^\top x$

设计不同的可学习门控功能

- 像线性回归或分类
- 注意：训练集成模型时不会更新单个模型

组合模型：门控



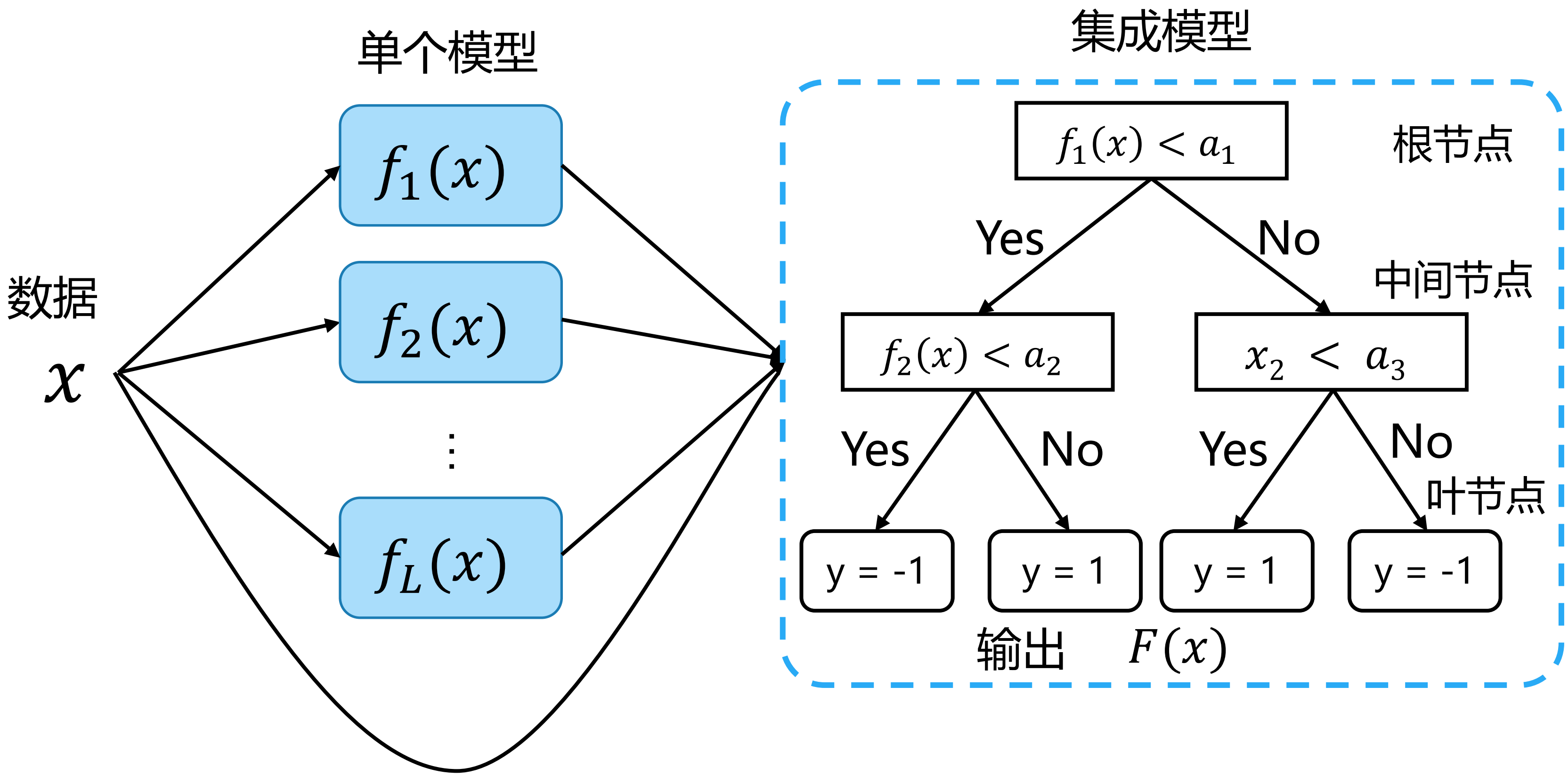
设计不同的可学习门控功能

$$F(x) = \sum_{i=1}^{L} g_i f_i(x)$$

例如：$g_i = \dfrac{exp(w_i^{\top} x)}{\sum_{j=1}^{L} exp(w_i^{\top} x)}$

- 像线性回归或分类
- 注意：训练集成模型时不会更新单个模型

组合模型：树模型



- 用决策树作为集成模型
- 根据 x 和 f 函数的值进行节点拆分

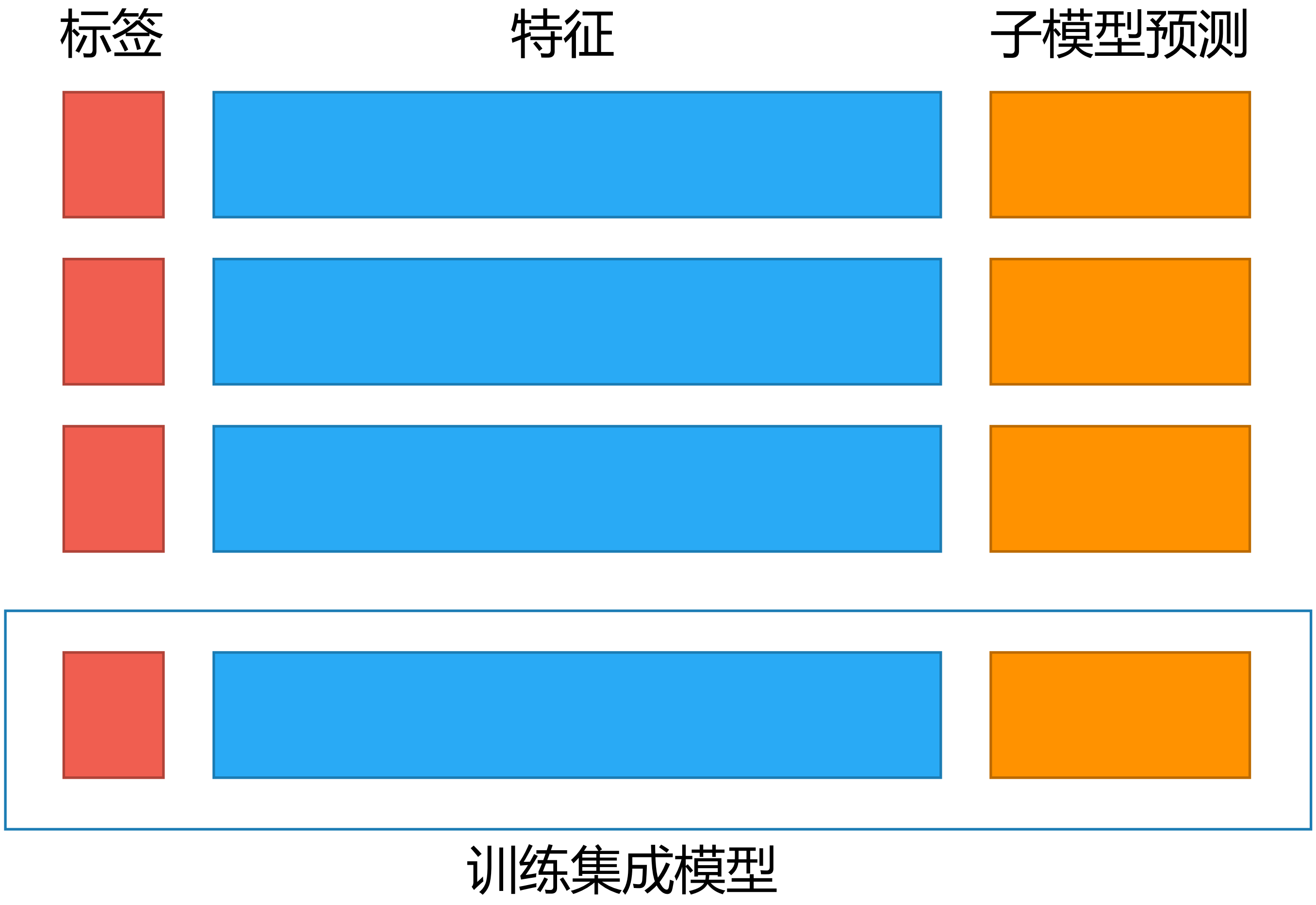
集成输入的多样性

- 成功的集成学习需要多样性
  - 预测模型可能会犯不同的错误
  - 一些多样性策略
    - 包含不同类型的预测模型
    - 改变训练集
    - 改变特征集

原因错误	多元化策略
识别pattern比较困难	尝试不同的模型
数据过拟合	改变训练集
有些特征会有噪声	改变特征集

集成学习的数据处理

- 为了避免学习到的集成模型过拟合



Bagging算法

张伟楠- 上海交通大学

目录

Contents

01

自举法

操作训练数据

自举复制

- 对于一个有 n 个训练样本的训练集 Z，通过有放回地采样 n 个样本得到一个新的训练集 Z^*
- 不包括大约37%的训练实例

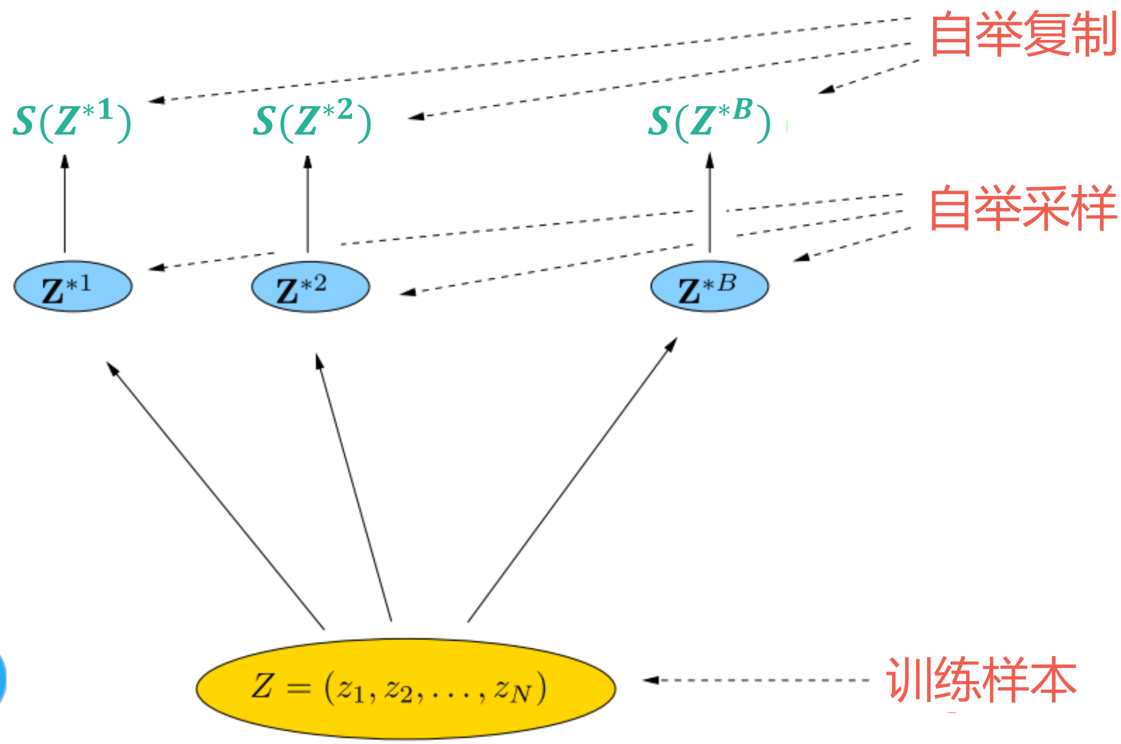
$$P\{observation\ i \in bootstrap\ samples\} \quad = 1 - \left(1 - \frac{1}{N}\right)^N$$
$$\simeq 1 - e^{-1} = 0.632$$

Bagging算法

自举聚合算法（Bootstrap aggregating，Bagging）

- 创建训练集的自举复制训练集（Bootstrap Replicate）
- 为每个复制集训练预测模型
- 使用非自举采样的数据验证预测模型
- 对所有预测模型的输出取平均

自举（Bootstrap）

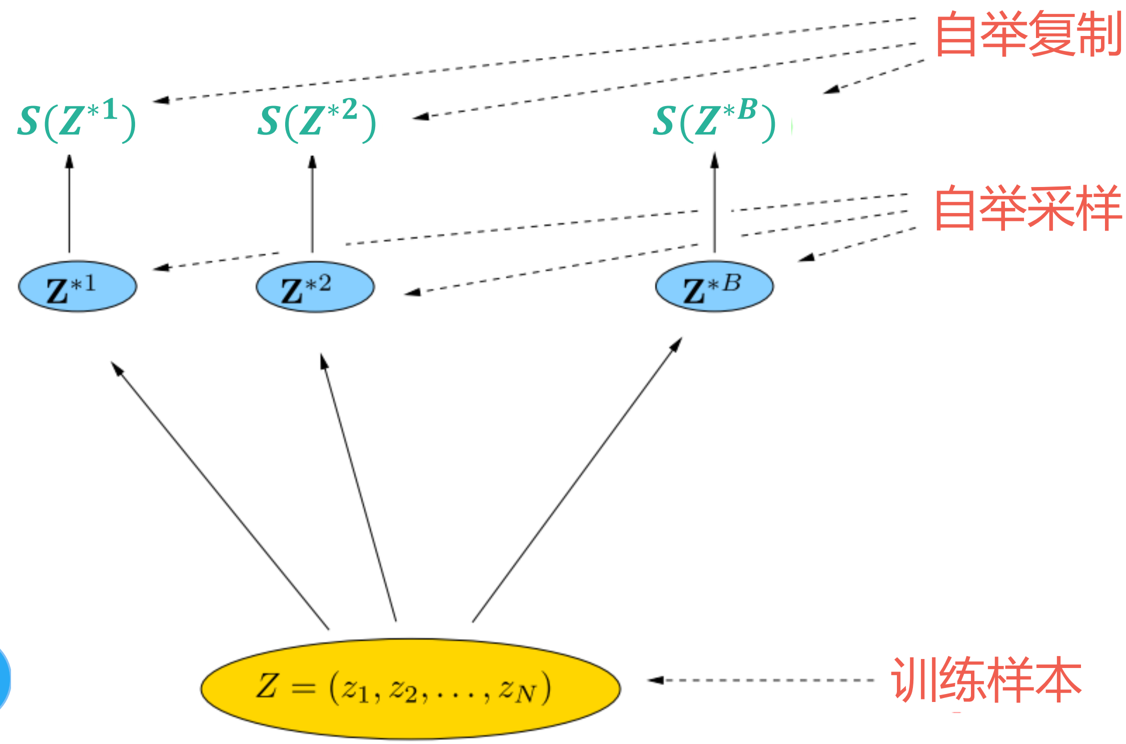


基本思想

- 从训练数据中有放回地采样生成一些数据集
- 每一个复制数据集的大小都和训练集的大小相同
- 在复制数据集上进行统计数值的评估分析
  - 例如，方差

$$\widehat{\mathrm{Var}}[S(Z)] = \frac{1}{B-1}\sum_{b=1}^{B}\left(S(Z^{*b}) - \bar{S}^{*}\right)^2$$

自举（Bootstrap）



基本思想

- 从训练数据中有放回地采样生成一些数据集
- 每一个复制数据集的大小都和训练集的大小相同
- 在复制数据集上进行统计数值的评估分析
  - 例如，模型误差

$$\widehat{\mathrm{Err}}_{\mathrm{boot}} = \frac{1}{B}\frac{1}{N}\sum_{b=1}^{B}\sum_{i=1}^{N} L\left(y_i, \hat{f}^{*b}(x_i)\right)$$

利用自举法进行模型评估

- 如果我们直接使用整个训练数据来评估模型

$$\widehat{\mathrm{Err}}_{\mathrm{boot}} = \frac{1}{B}\frac{1}{N}\sum_{b=1}^{B}\sum_{i=1}^{N} L\left(y_i, \hat{f}^{*b}(x_i)\right)$$

- 由于数据实例在自举采样数据集中的概率为

$$P\{observation\ i \in bootstrap\ samples\} \quad = 1 - \left(1 - \frac{1}{N}\right)^N$$
$$\simeq 1 - e^{-1} = 0.632$$

- 如果直接在训练集上验证，大概率会过拟合
  - 例如，在二值分类问题中 y 与 x 独立
    - 正确的错误率：0.5
    - 利用自举法错误率：0.632*0 + (1-0.632)*0.5=0.184

留一自举法（Leave-One-Out Bootstrap）

- 当构建自举复制集时不采样实例 i，然后利用实例 i 来评估模型

$$\widehat{\mathrm{Err}}^{(1)} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{|C^{-i}|}\sum_{b\in C^{-i}} L\left(y_i, \hat{f}^{*b}(x_i)\right)$$

  - C^{-i}是那些没有包含实例 i 的自举复制集 b 的索引集合
  - 对一些实例 i，集合 C^{-i}可能是空集，这些情况忽略即可

- 在后面的章节中会详细介绍模型评估
- 更多细节请参阅 Sec 8.4 of Hastie et al. The elements of statistical learning. 2008.

02

Bagging算法

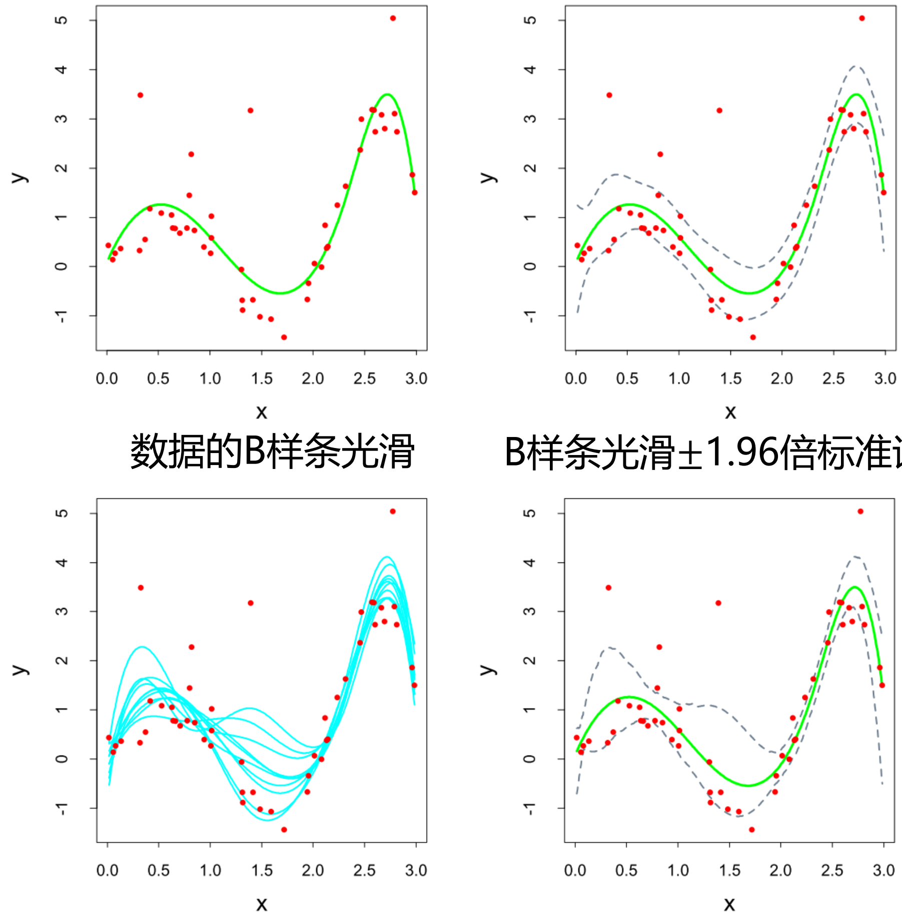
Bagging: Bootstrap Aggregating

算法步骤

- 自举复制
  - 对一个有n个训练样本的数据集 $Z = \{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$，通过有放回地采样 n 个实例构造得到新的训练集 Z^*
  - 构造 B个自举采样集 Z^{*b}， $b = 1,2,\dots,B$
  - 训练一个预测模型集合 $\hat{f}^{*1}(x), \hat{f}^{*2}(x), \dots, \hat{f}^{*B}(x)$
- 在预测结果上求平均值

$$\hat{f}_{bag}(x) = \frac{1}{B}\sum_{b=1}^{B}\hat{f}^{*b}(x)$$

例子



数据的B样条光滑

B样条光滑±1.96倍标准误差带

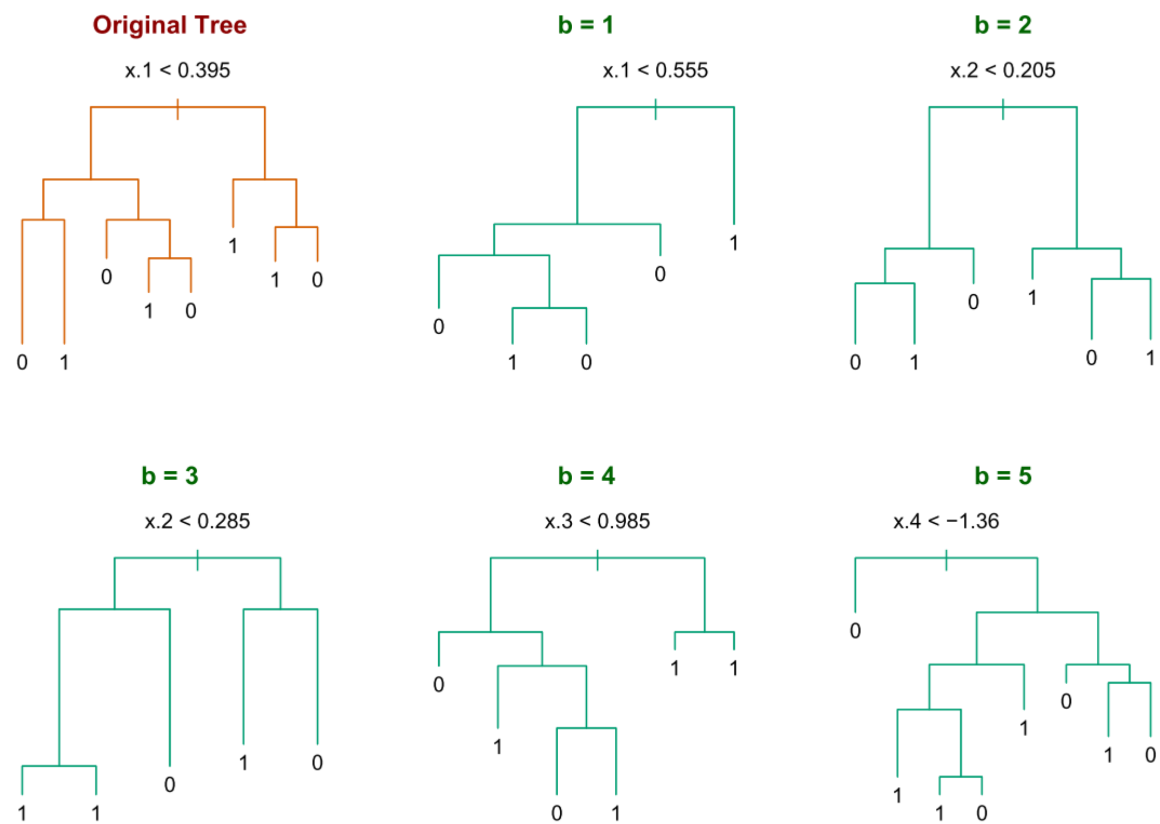
B样条光滑的10个自举法重复实验

从自举法分布计算的有±1.96倍标准误差带的B样条光滑

Fig 8.2 of Hastie et al. The elements of statistical learning.

注：±1.96倍标准误差带为95%置信区间

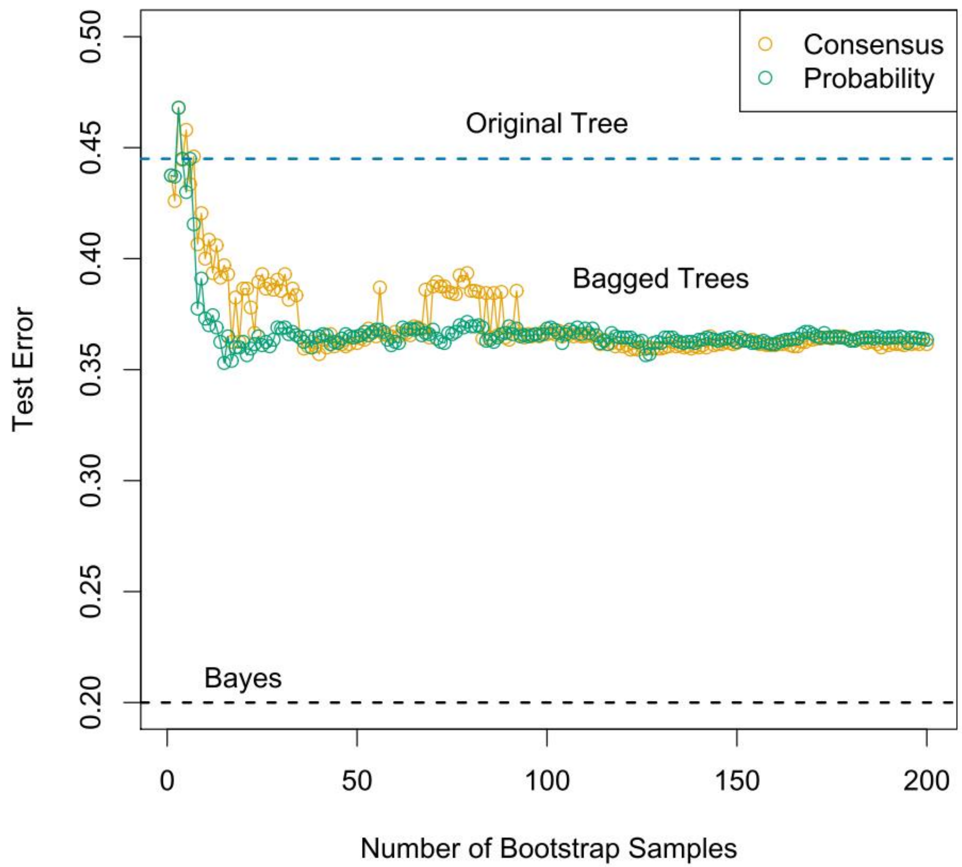
例子



模拟数据集上的Bagging树，左上图显示的是原始树，
其余的图显示了自举法样本上的5棵树

Fig 8.9 of Hastie et al. The elements of statistical learning.

例子



Bagging分类，多数表决vs求概率的平均值

Fig 8.10 of Hastie et al. The elements of statistical learning.

为什么Bagging算法有效?

- 偏差-方差分解（Bias-Variance Decomposition）
  - 假设$Y = f(X) + \epsilon$，其中$\mathbb{E}[\epsilon] = 0 \;\; Var[\epsilon] = \sigma_\epsilon^2$
  - 在输入点x_0的期望预测误差为

$$
\begin{aligned}
Err(x_0) &= \mathbb{E}\left[\left(Y - \hat{f}(x_0)\right)^2 \middle| X = x_0\right] \\
&= \sigma_\epsilon^2 + \left[\mathbb{E}[\hat{f}(x_0)] - f(x_0)\right]^2 + \mathbb{E}\left[\hat{f}(x_0) - \mathbb{E}[\hat{f}(x_0)]\right]^2 \\
&= \sigma_\epsilon^2 + Bias^2\left(\hat{f}(x_0)\right) + Var\left(\hat{f}(x_0)\right)
\end{aligned}
$$

- Bagging有效的原因是与原始模型相比偏差相同但是降低了方差(在整个数据集上进行训练)
  - 对低偏差和高方差的预测模型尤其有效果

总结树模型

- 和线性回归、逻辑回归、支持向量机、神经网络等参数化模型不同，决策树模型不用定义参数化的假设空间，而是在树结构函数空间中直接搜索较为优质的模型实例

- 针对一个训练数据集，搜索到最优树模型结构在时间上是NP-hard，往往用贪心算法选择当前最佳分裂特征和分裂点

- 当用森林模型时，其实不用纠结单科树是否为最优，多棵树集成的效果在实际应用中往往效果拔群

- 思考：既然Bagging的作用是降低森林的variance，那如何通过解耦合每棵树的学习，从而更加有效地降低variance?

THANK YOU