大数据分析

Scalable Machine Learning
decision tree

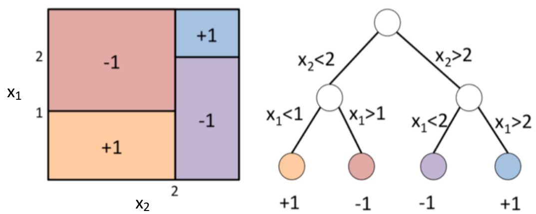
刘盛华

Outline

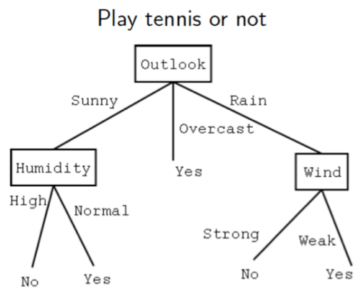
- 决策树（Decision Tree）
- 随机森林（Random Forest）
- 梯度提升树（Gradient Boosted Decision Tree (GBDT)）

Decision Tree

- 每个节点负责检查特征 x_i：
  - 若 $x_i <$ threshold，选择左子树
  - 若 $x_i \geq$ threshold，选择右子树



例子



Decision Tree

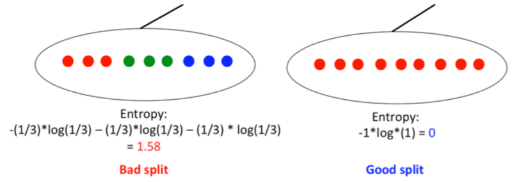
- **计算能力:**
  - 决策树是**非线性**分类器
  - 决策树更具备**可解释性**
  - 决策树天然适合处理**类别型特征**
- **计算效率:**
  - 训练: **较慢**
  - 预测: **较快**
  - 决策树有*h* 步操作operations (h: 树的深度，通常 ≤ 15)

划分节点

- 分类树: 依照使数据的熵最大的准则划分节点
- S表示一个节点中的全部样本,每个样本的标签 $c = 1, \cdots, C$:

$$\text{Entroy}: H(S) = -\sum_{c=1}^{C} p(c) \log p(c),$$

- p(c)表示所有样本中属于类别c的样本所占比例.
  - 若所有样本属于同一类别，Entropy=0
  - 若 p(1) = ··· = p(C)，Entropy最大



信息增益

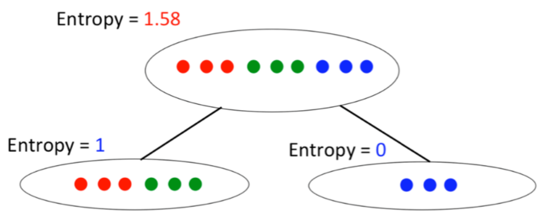
- 划分 S → S1, S2的熵的平均值为：

$$\frac{|S_1|}{|S|} H(S_1) + \frac{|S_2|}{|S|} H(S_2)$$

- 信息增益：评价划分方式的好坏

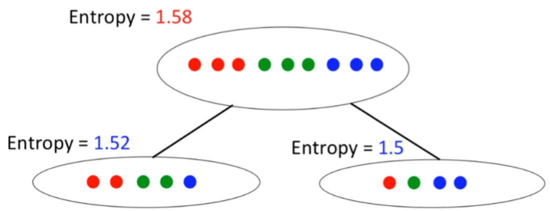
$$H(S) - \Big((|S_1|/|S|) H(S_1) + (|S_2|/|S|) H(S_2) \Big)$$

信息增益



Averaged entropy: 2/3*1 + 1/3*0 = 0.67
Information gain: 1.58 – 0.67 = 0.91

信息增益



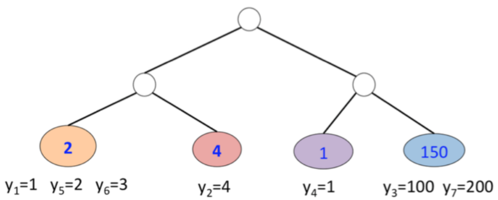
Averaged entropy: 1.51
Information gain: 1.58 – 1.51 = 0.07

节点划分

- 给定当前节点，如何选择**最优划分**方法？
- 对所有特征和阈值：
  - 计算按照每个特征及阈值划分后的信息增益
  - 选择最优划分 (**最大化信息增益**)
- 对n个样本和d个特征: 时间复杂度 $O(nd)$

回归树

- 为每个叶子节点赋一个数值
- 通常每个叶子节点取值为它包含的所有y个样本的平均 (最小化平方误差)



回归树

Objective function:

$$\min_F \frac{1}{n}\sum_{i=1}^{n}(y_i - F(\mathbf{x}_i))^2 + (\text{Regularization})$$

The quality of partition $S = S_1 \cup S_2$ can be computed by the objective function:

$$\sum_{i\in S_1}(y_i - y^{(1)})^2 + \sum_{i\in S_2}(y_i - y^{(2)})^2,$$

where $y^{(1)} = \frac{1}{|S_1|}\sum_{i\in S_1} y_i$, $y^{(2)} = \frac{1}{|S_2|}\sum_{i\in S_2} y_i$

每个划分集的方差最小

Find the best split:

Try all the features & thresholds and find the one with minimal objective function

决策树的超参数

- 最大树深: (通常 ~ 10)
- 分割节点所需的最小样本数: (10, 50, 100)
- 单决策树通常效果有限· · ·
- 是否可以构建多棵决策树，然后集成？

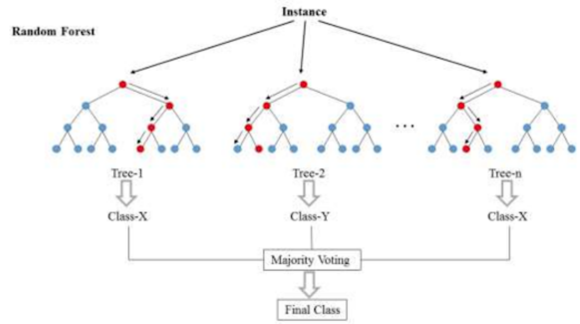
Outline

- 决策树（Decision Tree）
- 随机森林（Random Forest）
- 梯度提升树（Gradient Boosted Decision Tree (GBDT)）

随机森林

- 随机森林 (多决策树通过自助法集成（Bootstrap ensemble）):
  - 创建 T 棵树
  - 对每棵树，采样一个数据子集 S_i 和特征子集 D_i 用来训练
  - 预测: 取 T 棵树的结果的平均
- 随机森林的优点:
  - 避免过拟合
  - 改善稳定性和准确率
- 有很好的软件包:
  - R: “randomForest” package
  - Python: sklearn

例子



通过MapReduce构建决策树

- Parallel Learner for Assembling Numerous Ensemble Trees [Panda et al., VLDB '09]
  - A sequence of MapReduce jobs that builds a decision tree
  - Spark MLlib Decision Trees are based on PLANET

Outline

- 决策树（Decision Tree）
- 随机森林（Random Forest）
- 梯度提升树（Gradient Boosted Decision Tree (GBDT)）

提升树（Boosted Decision Tree）

- Minimize loss $\ell(y, F(x))$ with $F(\cdot)$ being ensemble trees

$$F^* = \operatorname*{argmin}_F \sum_{i=1}^{n} \ell(\mathbf{y}_i, F(\mathbf{x}_i)) \quad \textbf{with} \quad F(\mathbf{x}) = \sum_{m=1}^{T} f_m(\mathbf{x})$$

(each f_m is a decision tree)

- Direct loss minimization: at each stage m, find the best function to minimize loss
  - solve $f_m = \operatorname{argmin}_{f_m} \sum_{i=1}^{N} \ell(y_i, F_{m-1}(\mathbf{x}_i) + f_m(\mathbf{x}_i))$
  - update $F_m \leftarrow F_{m-1} + f_m$

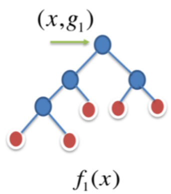
$F_m(\mathbf{x}) = \sum_{j=1}^{m} f_j(\mathbf{x})$ is the prediction of $\mathbf{x}$ after m iterations.

- Two problems:
  - Hard to implement for general loss
  - Tend to overfit training data

梯度提升树 (GBDT)

Key idea:

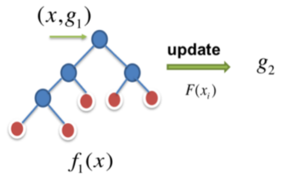
- Each base learner is a decision tree
- Each regression tree approximates the functional gradient $\frac{\partial \ell}{\partial F}$



梯度提升树 (GBDT)

Key idea:

- Each base learner is a decision tree
- Each regression tree approximates the functional gradient $\frac{\partial \ell}{\partial F}$

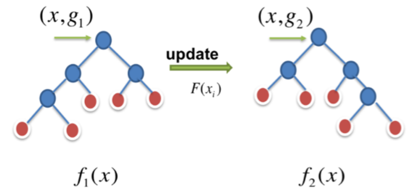


$$F_{m-1}(x_i)=\sum_{j=1}^{m-1} f_j(x_i) \qquad g_m(x_i)=\frac{\partial \ell(y_i,F(x_i))}{\partial F(x_i)}\bigg|_{F(x_i)=F_{m-1}(x_i)}$$

梯度提升树 (GBDT)

Key idea:

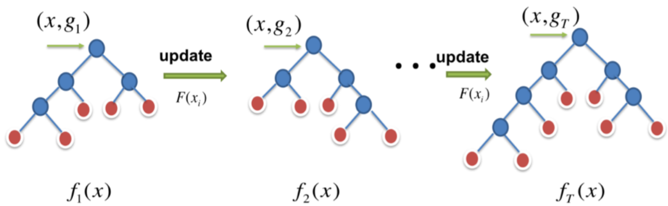
- Each base learner is a decision tree
- Each regression tree approximates the functional gradient $\frac{\partial \ell}{\partial F}$



$$F_{m-1}(x_i)=\sum_{j=1}^{m-1} f_j(x_i) \qquad g_m(x_i)=\frac{\partial \ell(y_i,F(x_i))}{\partial F(x_i)}\bigg|_{F(x_i)=F_{m-1}(x_i)}$$

梯度提升树 (GBDT)

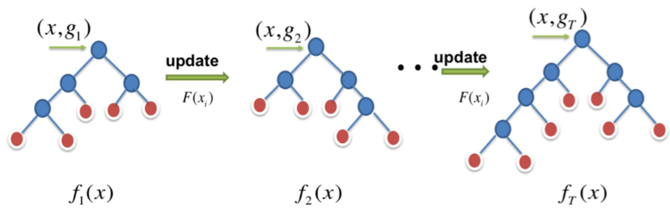
- Key idea:
  - Each base learner is a decision tree
  - Each regression tree approximates the functional gradient $\frac{\partial \ell}{\partial F}$



$$F_{m-1}(x_i)=\sum_{j=1}^{m-1} f_j(x_i) \qquad g_m(x_i)=\frac{\partial \ell(y_i,F(x_i))}{\partial F(x_i)}\bigg|_{F(x_i)=F_{m-1}(x_i)}$$

梯度提升树 (GBDT)

- Key idea:
  - Each base learner is a decision tree
  - Each regression tree approximates the functional gradient $\frac{\partial \ell}{\partial f}$



Final prediction $F(x_i)=\sum_{j=1}^{T} f_j(x_i)$

Questions?