CSDN
不 止 于 代 码

3.1 决策树

CSDN学院

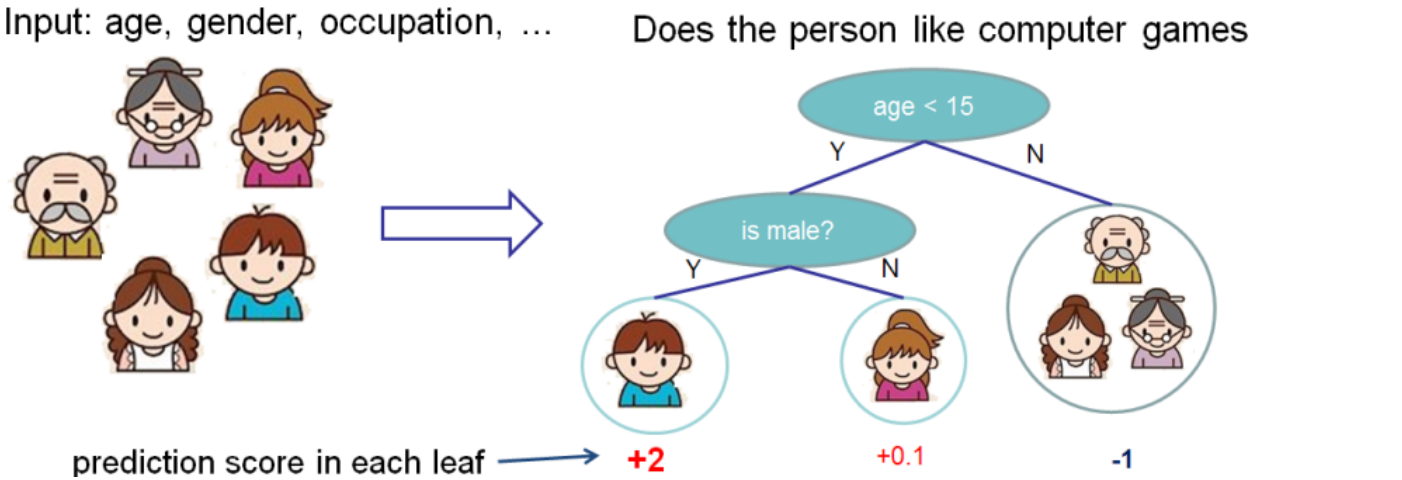
2017年11月

大纲

- 分类回归树（Classification And Regression Trees ，CART）
- Scikit learn中决策树的实现
- 案例

分类回归树

- Classification And Regression Tree（CART）：机器学习十大算法之一，是一个用于监督学习的非参数模型
  - 二分递归分割：将当前样本集合划分为两个子样本集合，使得生成的每个非叶子结点都有两个分支→ 生成的树是二叉树



例：回归树

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

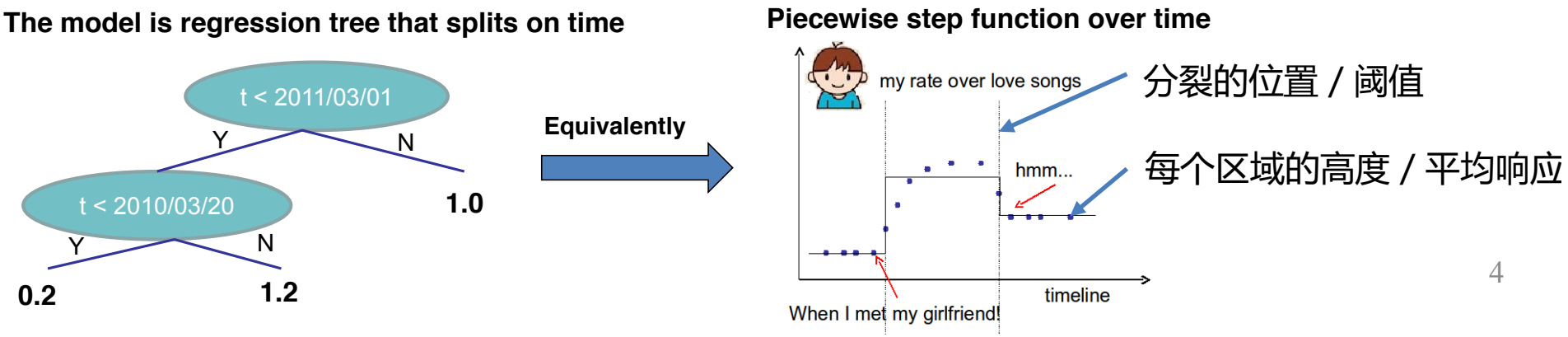
  第m个区域的平均响应

  第m个区域

  选择的分裂特征阈值

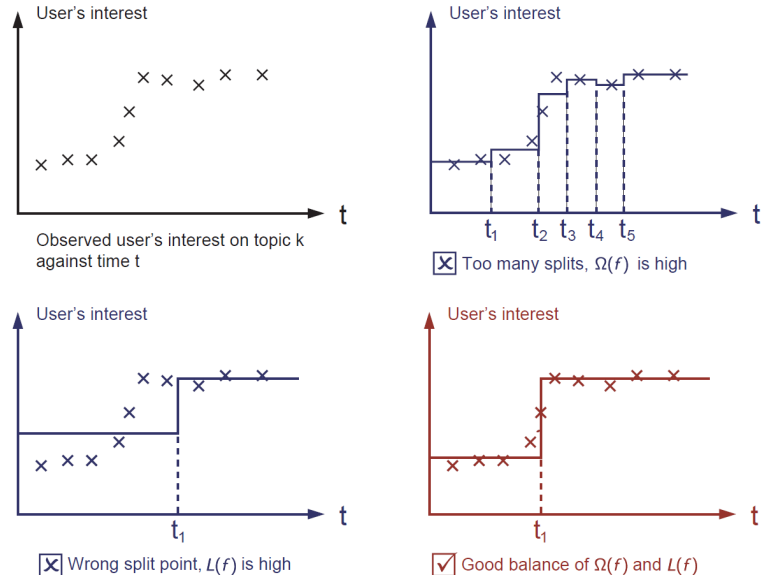
  根节点到第m个叶子结点的路径

- 参数：每次分裂的特征及阈值
- 例：预测t时刻我是否喜欢Romantic Music



例：回归树（cont.）

- 例：预测t时刻我是否喜欢Romantic Music
- 回归问题的目标函数
  - 训练误差 / 损失：函数与样本点的匹配程度
  - 正则：分裂点的数目？每一段的高度？



树模型很容易过拟合
所以很多策略都是在防止模型过拟合，如
提前终止、剪枝、Bagging…

CART

- 在CART算法中主要分为两个步骤
- （1）将样本递归划分进行建树
  - 损失最小
- （2）用验证数据进行剪枝
  - 正则：减小模型复杂度 (节点数目)

建树

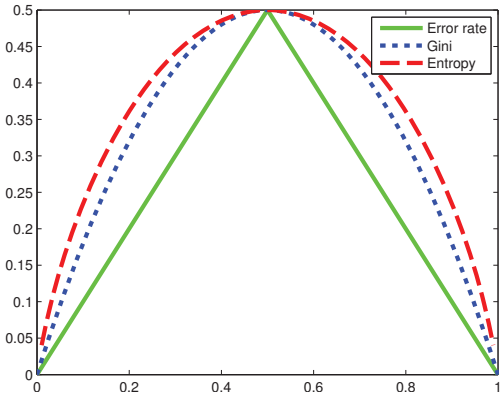
- 令节点的样本集合为 $\mathcal{D}$, 对候选分裂$\boldsymbol{\theta} = (j, t_m)$，选择特征$j$，分裂阈值为$t_m$，将样本分裂成左右两个分支$\mathcal{D}_L$和$\mathcal{D}_R$
- $\mathcal{D}_L(\boldsymbol{\theta}) = \{(\boldsymbol{x}_i, y_i) | x_{ij} \leq t_m\}$
- $\mathcal{D}_R(\boldsymbol{\theta}) = \{(\boldsymbol{x}_i, y_i) | x_{ij} > t_m\}$
- 分裂原则：分裂后两个分支的样本越纯净越好
  - $G(\mathcal{D}, \boldsymbol{\theta}) = \frac{N_{left}}{N_m} H(\mathcal{D}_L(\boldsymbol{\theta})) + \frac{N_{right}}{N_m} H(\mathcal{D}_R(\boldsymbol{\theta}))$
  - 不纯净性$G(\mathcal{D}, \boldsymbol{\theta})$最小: $\boldsymbol{\theta}^* = argmin_{\boldsymbol{\theta}} G(\mathcal{D}, \boldsymbol{\theta})$
  - 不纯净性度量函数H与任务有关

建树——回归

- 对回归问题, 集合$\mathcal{D}$的不纯净性为集合中样本的y值的差异
  - $H(\mathcal{D}) = \sum_{i\in\mathcal{D}}(\bar{y} - y_i)^2$
  - 其中 $\bar{y} = \frac{1}{|\mathcal{D}|}\sum_{i\in\mathcal{D}} y_i$ 为集合中样本的y的均值
  - 集合中样本的 y 值越接近越纯净
  - 相当于损失函数取L2损失，选择最小L2损失的分裂
    - L2损失：$L(\hat{y}(\boldsymbol{\theta}), y) = (\hat{y}(\boldsymbol{\theta}) - y)^2 = (\bar{y} - y)^2$
    - 预测值为样本均值 $\bar{y} = \frac{1}{|\mathcal{D}|}\sum_{i\in\mathcal{D}} y_i$ 时L2损失最小

建树——分类

- 分类问题中，分布的估计值取 $\hat{\pi}_c = \frac{1}{|\mathcal{D}|} \sum_{i \in \mathcal{D}} \mathbb{I}(y_i = c)$
  - 即集合中每类样本的比例
- 常用的不纯净度量：
  - 错分率： $H(\mathcal{D}) = \frac{1}{|\mathcal{D}|} \sum_{i \in \mathcal{D}} \mathbb{I}(y_i \neq \hat{y}) = 1 - \hat{\pi}_{\hat{y}}$
  - 熵： $H(\mathcal{D}) = -\sum_{c=1}^{C} \hat{\pi}_c \log \hat{\pi}_c$
  - Gini系数 C
  - $H(\mathcal{D}) = \sum_{c=1} \hat{\pi}_c (1 - \hat{\pi}_c) = \sum_{c} \hat{\pi}_c - \sum_{c} \hat{\pi}_c^2 = 1 - \sum_{c} \hat{\pi}_c^2$



两类分类的不纯度度量

建树——停止条件

- 建树过程是一个自顶向下的递归过程
- 递归的停止条件：
  - 分裂带来的损失的减小太小：损失的减少量可定义为

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

  - 树的深度超过了最大深度
  - 左 / 右分支的样本分布足够纯净
  - 左 / 右分支中样本数目足够少

剪枝

- 分类回归树算法容易过拟合，通过剪枝去除部分分支，降低模型复杂度
- 剪枝：给定一个完全树，自底向上进行剪枝，直到根节点
- 剪枝准则：Cost complexity pruning

$$CC(T) = Err(T) + \alpha|T|$$

树的错误率　　正则因子　树的节点数目

  - 形式同机器学习模型的目标函数：$J(\boldsymbol{\theta}) = \sum_{i=1}^{N} L(f(\mathbf{x}_i; \boldsymbol{\theta}), y_i) + \lambda\Omega(\boldsymbol{\theta})$
  - 当α从0开始增大，树的一些分支被剪掉，得到不同α对应的树
  - 采用交叉验证得到最佳的α

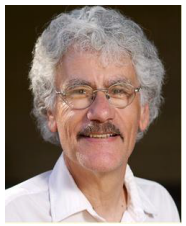
树模型的优点

- 容易解释
- 不要求对特征做预处理
  - 能处理离散值和连续值混合的输入
  - 对特征的单调变换不敏感 (只与数据的排序有关)
  - 能自动进行特征选择
  - 可处理缺失数据
- 可扩展到大数据规模

树模型的缺点

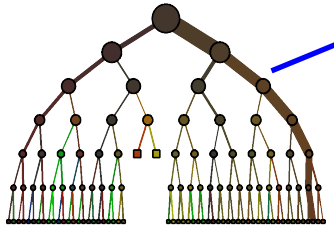
- 正确率不高：建树过程过于贪心
  - 可作为Boosting的弱学习器（深度不太深）
- 模型不稳定（方差大）：输入数据小的变化会带来树结构的变化
  - Bagging：随机森林
- 当特征数目相对样本数目太多时，容易过拟合

A brief history of forests

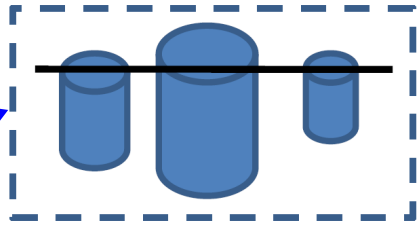


Jerome H. Friedman

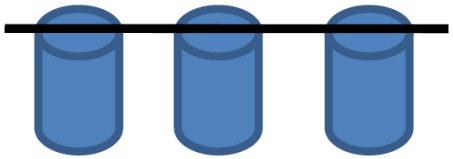
Leo Breiman



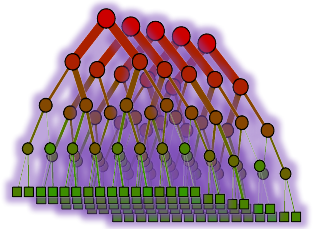
CART



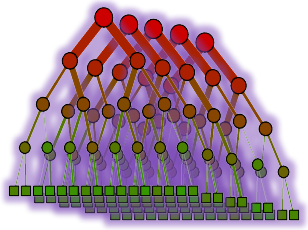
Boosting



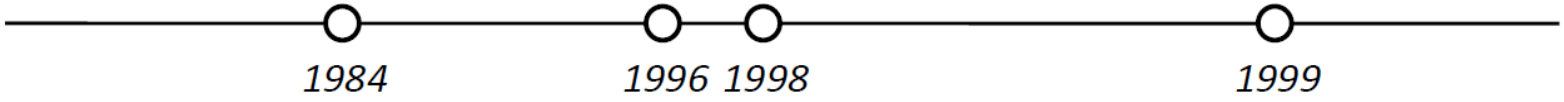
Bagging



GBDT



Random Forest



► 小结：分类回归树

- 模型：树（非参数模型）
- 参数：分裂的特征及阈值
- 目标函数
  - 损失函数：L2损失 / GINI指数
  - 正则项：树的节点数目（L0/L1）、叶子结点分数平方和（L2）
- 优化
  - 建树
  - 剪枝

Scikit-learn中的Tree

- CART算法的优化实现
  - 建树：穷举搜索所有特征所有可能取值
  - 没有实现剪枝（采用交叉验证选择最佳的树的参数）
  - 分类树： DecisionTreeClassifier
  - 回归树： DecisionTreeRegressor

DecisionTreeClassifier

- 构造时参数传递参数
- *class* sklearn.tree.**DecisionTreeClassifier**(*criterion='gini', splitter='best', max_depth=None, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features=None, random_state=None, max_leaf_nodes=None, min_impurity_split=1e-07, class_weight=None, presort=False*)

- 决策树算法特有的参数：*criterion*、 *splitter*、*max_depth*、*min_samples_split*、*min_samples_leaf*、*min_weight_fraction_leaf*、*max_features*、*max_leaf_nodes*、*min_impurity_split*

参数	说明
criterion	用来权衡划分的质量。缺省 ‘gini’： 即 Gini impurity。 或者 ‘entropy’
splitter	划分方式有三种：best, presort-best, random. 缺省：best
max_features	当进行best划分时，考虑的特征数目（个 / 比例）. 缺省:None
max_depth	树的最大深度。缺省为：None
min_samples_split	对于一个中间节点（internal node），必须有min个samples才对它进行分割。缺省为：2
min_samples_leaf	每个叶子节点（left node），至少需要有min_samples_leaf个样本。缺省为：1
min_weight_fraction_leaf	叶子节点的样本权重占总权重的比例。缺省为：0
max_leaf_nodes	以最好优先（best-first）的方式使用该值生成树。如果为None:不限制叶子节点的数目。如果不为None，则忽略max_depth。缺省为：None
class_weight	每个类别的权重：{class_label: weight}。如果不给定，所有类别的权重均1. “balanced”模式：自动调整权重。n_samples / (n_classes * np.bincount(y))
random_state	随机种子
presort	是.否对数据进行预先排序，以便在fitting时加快最优划分。对于大数据集，使用False，对于小数据集，使用True

DecisionTreeRegressor

- *class* sklearn.tree.**DecisionTreeRegressor**(*criterion='mse', splitter='best', max_depth=None, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features=None, random_state=None, max_leaf_nodes=None, min_impurity_split=1e-07, presort=False*)

- 与分类树的参数基本相同，*criterion*的缺省值为*'mse'*，*mean squared error,* 即残差平方和的均值（L2损失）

feature_importances_

- 特征重要性亦被称为Gini重要性，即增加该特征对Gini指标的（归一化的）减少量。

- 可以做特征选择…

CSDN
不 止 于 代 码

THANK YOU

AI100