机器学习笔记 (4): 决策树

https://blog.csdn.net/gaocui883

2021 年 3 月 25 日

1 基本思路与概念

之前没看前边的章节，adaboost 后边有提升树的内容，所以就回去看了下决策树这章，并做下记录。

1.1 决策树模型与学习

1.1.1 决策树模型

Theorem 1.1 一种描述对实例进行分类的树形结构，结点和有向边组成，节点有内部节点和叶节点，内部节点代表特征，叶节点代表类别。

1.1.2 决策树与规则

也可以将其看成一种规则：路径上的内部节点对应规则条件，叶节点对应规则结论。

1.1.3 决策树与条件概率分布

P(Y|X) 中，X 表示特征空间，Y 表示类的取值集合。

1.1.4 决策树学习

包括：
特征选择: 选择对分类最优的特征，去除那些对决策无用的特征。
决策树生成： 生成完整的树。
剪枝： 对完整的树的某些叶节点进行减除，降低树的复杂度。主要用验证

集来上的效果来衡量。

2 特征选择

决策树的内部节点是特征，那么要生成决策树首先就需要选择对决策来说比较重要的特征，如果一个参考一个特征进行决策或者分类的效果仅仅比随机好那么一点点，那么就可以认为这个特征对决策没有意义，可以不考虑。

2.1 信息增益和信息增益比

如果一个特征提供的信息大，那么它就重要，反之就不重要，信息论中的熵的概念就可以用来作为这个评判标准。

2.1.1 熵

$$P(X = x_i) = p_i, i = 1, 2, ..., n$$

为离散所及变量的概率分布，则其熵定义为

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

如果 log 则是以 2 为底，单位比特，如果 ln 则以 e 为底，单位为纳特。

因为熵只是与分布有关，所以也可以写为：

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

熵越大，则其不确定性就越大。当分布为均匀分布时，达到最大：

$$0 \leq H(p) \leq \log n$$

2.1.2 条件熵

表示在已知随机变量 X 的条件下，随机变量 Y 的不确定性。

$$H(Y|X) = \sum_{i=1}^{n} p_i H(Y|X = x_i)$$

其中 p_i 为 x 的分布。

2.1.3 信息增益

表示得知特征 X 的信息而使类 Y 的信息不确定性减少的程度。

Theorem 2.1 特征 A 对训练数据集 D 的信息增益 $g(D, A)$, 定义为集合 D 的经验熵 $H(D)$ 与特征 A 给定条件下 D 的经验条件上 $H(D|A)$ 的差：

$$g(D, A) = H(D) - H(D|A)$$

Algorithm 1 信息增益的算法.

Input:

训练师数据 D 和特征 A

Output:

信息增益 $g(D|A)$

1: 计算经验熵 () 分布式从训练数据估计的

$$H(D) = -\sum_{k=1}^{K} \frac{|C_k|}{|D|} \log \frac{|C_k|}{|D|}$$

其中 K 为类别数，|D| 为样本数，$|C_k|$ 为第 k 类的包含的样本数。

2: 计算条件熵 $H(D|A)$

$$H(D|A) = \sum_{i=1}^{n} \frac{|D_i|}{|D|} H(D_i) = -\sum_{i=1}^{n} \frac{|D_i|}{|D|} \sum_{k=1}^{K} \frac{|D_{ik}|}{|D_i|} \log \frac{|D_{ik}|}{|D_i|}$$

其中 n 为特征 A 将 D 划分的区域数，$|D_i|$ 第 i 个区域样本数，$|D_{ik}|$ 为第 i 个区域，属于 k 类的样本数。

3: 计算信息增益：

$$g(D, A) = H(D) - H(D|A)$$

2.1.4 信息增益比

Theorem 2.2 (信息增益比)

$$g_R(D,A) = \frac{g(D,A)}{H_A(D)}$$

其中，$H_A(D) = -\sum_{i=1}^{n} \frac{|D_i|}{|D|} \log \frac{|D_i|}{|D|}$

3 决策树生成

ID3 算法选择特征为信息增益也就是互信息；C4.5 算法选择特征为信息增益比。

3.1 ID3 算法

该算法的核心是从根节点开始，选择信息增益最大的特征作为该节点的特征，由该特征的不同取值建立子节点，然后重复的对子节点以及子节点的子节点应用以上方法。

4 决策树的剪枝

4.1 损失函数

决策树的整体损失函数，当考虑决策树的复杂度后为：

$$C_\alpha(T) = C(T) + \alpha|T|$$

其中的 $C(T)$ 为模型对训练数据的预测误差，而后边的 $\alpha|T|$ 表示模型的复杂度，当 α 为 0 时，决策树将不考虑复杂度，直接生成最大的树。

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

其中的 $|T|$ 表示节点数量，N_t 表示每个节点的样本数。$H_t(T)$ 表示节点 t 的经验熵。

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

Algorithm 2 ID3 算法.

Input:

训练集 D，特征集 A，阈值 ϵ

Output:

决策树 T

1: 若 D 中的所有实例属于同一类 C_k , 则 T 为单节点树，并将类 C_k 作为结点类的标记，返回 T。

2: 若 $A = \varnothing$ 则 T 为单节点树，并将 D 中实例数最大的类 C_k 作为该节点的类标记，返回 T；

3: 否则，按照之前的信息增益算法计算 A 中各个特征对 D 的信息增益或者信息增益比 A_g;

4: 如果 A_g 的信息增益小于阈值 ϵ, 则 T 为单节点树，并将 D 中实例数最大的类 C_k 作为该结点的类标记，返回 T；

5: 如果大于阈值，那么对 A_g 的每一个可能值 a_i，依据 a_i 的值将 D 分割为若干非空子集 D_i, 将 D_i 中实例数最大的类作为标记，构建子节点，由结点和子节点构成树 T，返回 T；

6: 对第 i 个子节点，以 D_i 为训练集，以 A-A_g 为特征集，递归的调用 1-5，得到子树，返回 T

可以看出，α 一定时，可以找出损失函数最小的树，这样的树在一定的复杂度和准确率之间做了一定的平衡。当 $\aleph$ 增大时，树变得更加简单，当 α 变小时，树更加复杂。

然后就我们就可以设定一些列 α ，然后在测试集上进行验证，选取准确率符合要求并且树的复杂度合适的树。

5 CART 算法

CART : classification and regression tree,CART.

CART 假设决策树是二叉的，节点特征为是或者否，左是右否，递归二分每个特征，最终将输入空间氛围有限个个单元，然后在这些单元上预测概率分布，也就是 y 的条件概率分布。P(Y|X)

Algorithm 3 树的剪枝算法

Input:

生成算法生成的整个树 T，参数 α

Output:

修剪后的子树 T_α

1: 计算每个节点的经验熵 H_t

2: 递归的从树的叶节点向上回缩。

3: 一组叶节点回缩之前和回缩之后整体树为 T_B T_A, 如果

$$C_\alpha(T_A) \leq C_\alpha(T_B)$$

则表示剪枝后，损失函数变小了。

4: 重复 2-3，知道损失函数不能继续变小为止。

两步：

1. 决策树的生成
2. 决策树的减枝

5.1 CART 生成

$D = (x_1, y_1), (x_2, y_2), ..., (x_N, y_N)$

当输入空间被划分为 $R_1, R_2, ..., R_M$ 时，每个划分输出为 c_m, 则决策树表示为

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

其中 $I(x \in R_m)$ 表示在区域内为 1，否则为 0，c_m 可以用样本输出均值

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

代替。

5.1.1 回归树

回归树的误差用均值误差来表示

Algorithm 4 最小二乘回归树

Input: 训练数据集 D

Output: 回归树 $f(x)$

对训练数据, 递归的进行二划分，并决定每个区域上的输出值，构建二叉决策树。

1: 选择最优切分变量 j 与切分点 s, 求：

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

可以看出需要遍历变量 xj, 找到其优切分点 s，选择最小的 j,s。

2: 用选定的 (j,s) 划分区域，并决定相应的输出值：

$$R_1(j,s)=\left\{x|x^{(j)}\le s\right\}$$

$$R_2(j,s)=\left\{x|x^{(j)}>s\right\}$$

$$\hat{c}_m=\frac{1}{N_m(j,s)}y_i, x\in R_m, m=1,2$$

3: 对 $R_1(j,s), R_2(j,s)$ 重复 1，2，直到满足停止条件。

4: 最终输入空间划分为了 M 个区域，生成的决策树为：

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

5.1.2 分类树

分类树用基尼指数选择特征。

Theorem 5.1 (基尼指数) *分类样本中，假设有 K 类，样本点属于第 k 类的概率为 p_k, 则概率分布的基尼指数为*

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

对于给定一个样本集合，可以根据经验估计 p_k

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

如果在特征 A 下，将集合 D 的基尼指数定义为

$$Gini(D, A) = \frac{|D_1|}{|D|}Gini(D_1) + \frac{|D_2|}{|D|}Gini(D_2)$$

$Gini(D)$ 表示集合 D 的不确定性，$Gini(D, A)$ 表示集合经过 A 分割后，D 的不确定性。

Algorithm 5 CART 生成算法

Input: 训练数据集 D

Output: CART 决策树 $f(x)$

对训练数据, 递归的进行二划分，构建二叉决策树。

1: 节点的训练数据为 D, 计算现有特征对该数据集的基尼指数，对每个特征 A 的每个取值 α, 用其将 D 划分为是和否两个区域，计算此时的基尼指数，选择基尼指数最小的那个 α_{min}。

2: 比较每个特征的最小的 α, 选取最小的基尼指数对应的特征以及对应的 α

3: 将数据根据上一步的特征和切分点分到两个子节点中去。

4: 对子节点继续使用 1-3，直到满足停止条件。

5: 生成 CART 决策树。

5.2 CART 减枝

从树的叶节点开始剪，得到一系列的树 { T0,T1,...,TN}, 然后交叉验证选择最优子树。

Algorithm 6 CART 剪枝算法

Input: 生成的树 T_0

Output: 最优决策树 T_α

1: k=0,T=T_k。

2: $\alpha = +\infty$

3: 自上而下的对内部节点 t 计算 $C(T_t), |T_t|$,$C(T_t)$ 为 t 为根节点的子树的误差。

$$g(t) = \frac{C(t) - C(T_t)}{|T_t| - 1}$$

$$\alpha = \min(\alpha, g(t))$$

此时的损失函数

$$C_\alpha(T) = C(T) + \alpha|T|$$

t 节点的子树剪去后，的损失函数更小。

4: 对 t 进行剪枝，k=k+1,$\alpha_k = \alpha, T_k = T$, 如果 T_k 不是根节点，重复

5: 交叉验证选择最优子树。