5.决策树

决策树（decision tree）是一种基本的分类与回归方法，本文主要讨论分类的决策树。

通常决策树包含：**特征选择**，**决策树生成**，**决策树的修剪。**

特征选择

这里主要是通过递归来选择最优的特征，这里首先要了解 **熵**，**经验熵**，**经验条件熵**，**信息增益** 的概念及相关关系。

> 熵 H(X)：表示随机变量不确定性的变量，不确定性越大，熵越大。

设 X 是有限个的离散随机变量，X的熵定义为：

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

其中 p_i 为 X_i 的概率分布。 这里信息熵对数以 2 为底或以 e 为底，熵的单位分别为比特(bit) 或 纳特(nat)。

> 条件熵 H(Y|X): 表示已知随机变量 X 的条件下随机变量 Y 的不确定性。

定义为 X 给定条件下 Y 的条件概率分布的熵对 X 的数学期望：

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

其中 p_i 为 X_i 的概率分布。

当熵与条件熵中的概率由数据估计（尤其是极大似然估计）得到时，对应的熵和条件熵就是**经验熵**(empirical entropy) 和**经验条件熵**(empirical conditional emtropy)。

相关代码：(详见info_gain.py)

```
# 经验熵
def entropy(datasets):
    data_length = len(datasets)
    label_count = {}
    for i in range(data_length):
        label = datasets[i][-1]
        if label not in label_count:
            label_count[label] = 0
        label_count[label]+=1
    ent = -sum([( p/data_length) * log(p/data_length,2)
                for p in label_count.values()])
    return ent
```

```
# 经验条件熵
def condition_entropy(datasets,axis = 0):
    data_length = len(datasets)
    feature_sets = {}
```

```
    for i in range(data_length):
        feature = datasets[i][axis]
        if feature not in feature_sets:
            feature_sets[feature] = []
        feature_sets[feature].append(datasets[i])
    condi_ent = sum([ (len(p) / data_length)*entropy(p)
                for p in feature_sets.values()])
    return condi_ent
```

> 信息增益：经验熵与条件经验熵之差

代码：

```
# 信息增益
def info_gain(ent,condi_entropy):
    return ent - condi_entropy
```

训特征 A 在训练集 D 上的信息增益定义式为：

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

下面给出例子帮助理解上述的公式。

例5.1

表 5.1 贷款申请样本数据表

ID	年龄	有工作	有自己的房子	信贷情况	类别
1	青年	否	否	一般	否
2	青年	否	否	好	否
3	青年	是	否	好	是
4	青年	是	是	一般	是
5	青年	否	否	一般	否
6	中年	否	否	一般	否
7	中年	否	否	好	否
8	中年	是	是	好	是
9	中年	否	是	非常好	是
10	中年	否	是	非常好	是
11	老年	否	是	非常好	是
12	老年	否	是	好	是
13	老年	是	否	好	是
14	老年	是	否	非常好	是
15	老年	否	否	一般	否

这里将上述表格的信息进行一个归纳总结，总共有 4 个特征，然后每个特征对应最后的类别为 `是/否`：

特征	分布
A^1 年龄	{ 5青年 : [2是 3否] } { 5中年 : [3是 2否] } { 5老年 : [4是 1否] }
A^2 工作	{ 10没有工作 : [4是 6否] } { 5有工作 : [5是] }
A^3 有房子	{9没有房子 : [3是 6否] } { 6有房子 : [6是] }
A^4 信贷	{ 6好 : [4是 2否] } { 5一般 : [1是 4否] } { 4非常好 : [4是] }
类别 D	9 是 6 否

以其中一个归纳举例，{ 5青年 : [2是 3否] } 表示 A^1年龄 这个特征中 有 5 个是青年，5个青年中有 2 个批准申请贷款，3个不批准申请贷款，依次类推。

根据上面的定义式：

经验熵 H(D)：

$$H(D)=-\frac{9}{15}\log_2\frac{9}{15}-\frac{6}{15}\log_2\frac{6}{15}=0.971$$

因为其中 类别 D 有 9 个是批准，6个不批准。

条件经验熵 $H(D|A^1)$：

$$H\left(D|A^1\right)=\frac{5}{15}H\left(D_1\right)+\frac{5}{15}H\left(D_2\right)+\frac{5}{15}H\left(D_3\right),\text{将参数代入得：}$$

$$H\left(D|A^1\right)=\frac{5}{15}\left(-\frac{2}{15}\log_2\frac{2}{5}-\frac{3}{15}\log_2\frac{3}{5}\right)+\frac{5}{15}\left(-\frac{3}{15}\log_2\frac{3}{5}-\frac{2}{15}\log_2\frac{2}{5}\right)+\frac{5}{15}\left(-\frac{4}{15}\log_2\frac{4}{5}-\frac{1}{15}\log_2\frac{1}{5}\right)$$

最后求得$H(D|A^1) = 0.888$ 。
这里得 D_1，D_2，D_3，分别对应 青年，中年，老年，括号里的概率分别对应青年，中年，老年中的 `是` 和 `否` 的个数。

依次类推可以计算得到 $H(D|A^2)$， $H(D|A^3)$， $H(D|A^4)$。

信息增益$g(D，A^1)$：

根据定义式可得：
$g(D，A^1) = H(D) - H(D|A^1) = 0.971 - 0.888 = 0.083$

同理可得：
$g(D，A^2) = H(D) - H(D|A^2)= 0.971 - 0.647 = 0.324$
$g(D，A^3) = H(D) - H(D|A^3)= 0.971 - 0.551 = 0.420$
$g(D，A^4) = H(D) - H(D|A^4)= 0.971 - 0.608 = 0.363$

代码运行结果：

```
特征（年龄）的信息增益为： 0.083
特征（有工作）的信息增益为： 0.324
特征（有自己的房子）的信息增益为： 0.420
特征（信贷情况）的信息增益为： 0.363
特征(有自己的房子)的信息增益最大，选择为根节点特征
```

这里新增一个概念 **信息增益比**:

> 信息增益比： 信息增益与经验熵之比

特征 A 对训练集 D 的信息增益为 g(D|A),经验熵为 H(D):

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

决策树的生成

主要介绍两种算法，

- ID3 算法 （见 DecisionTree.py）

> ID3 算法：在各个节点上应用 **信息增益** 准则来选择特征，递归地构建决策树。

而 ID3 算法只有树的生成，所以产生的树容易过拟合。因此 C4.5 可以看作是对 ID3 算法的改进。

- C4.5 算法

> C4.5 算法： 在各个节点上应用**信息增益比** 来选择特征，递归地构建决策树，并进行了剪枝。

决策树剪枝

主要是在决策树学习的损失函数中加入了 **叶节点个数** 的因素。 将 |T|*α 作为惩罚项，自下而上递归地将叶节点回缩以后再计算损失函数的值，如果损失函数的值有减小，就删除当前节点，其中 α 为系数，|T| 为当前决策树 T 的叶节点个数，这里 α 越大,模型（树）越简单，α 越小,模型（树）越复杂。

决策树生成只考虑通过提高信息增益（或信息增益比）对训练集进行拟合，剪枝也是在损失函数中加入了模型复杂度（叶节点的个数）的因素。所以生成时学习的是局部的模型，剪枝学习的是整体的模型。

CART 算法

CART （classification and regression tree，分类与回归树）模型是在给定输入随机变量 X 下输出随机变量 Y 的条件概率分布的学习方法。

分成两步构成:

- **决策树生成**：对回归树用 **平方误差** 最小化准则，对分类树用 **基尼指数** 最小化准则，进行特征选择生成决策树。
- **决策树剪枝**： 首先从生成算法生成的树的底端不断剪枝，直到根节点，形成一个子树序列，然后通过交叉验证法在独立的验证数据集上进行测试，从中选择最优子树。

决策树生成

这里主要介绍**分类树**的生成，及基尼指数。

基尼指数：分类问题中，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定义为：

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

对于二分类问题，第一类的概率为 p，则概率分布的基尼指数为：

$$Gini(p) = 2p(1-p)$$

如果样本集合 D 根绝特征 A 是否取某一个可能值 a 被分割成 D_1，和D_2 两部分，则在特征 A 的条件下，集合 D 的基尼指数定义为：

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

以上述例5.1为例，年龄特征 A_1 ： `青年，中年，老年` 会分别划分为：

> Gini(D,A^1 = 1)　对应 D_1 = 青年，D_2 = 中年，老年
> Gini(D,A^1 = 2)　对应 D_1 = 中年，D_2 = 青年，老年
> Gini(D,A^1 = 3)　对应 D_1 = 老年，D_2 = 青年，中年

依次类推。 $|D_i|$ 指对应类别的样本总数，|D| 表示集合总样本数。

$$H\left(D|A^1\right) = \frac{5}{15}\left(2\times\frac{2}{5}\times\left(1-\frac{2}{5}\right)\right) + \frac{10}{15}\left(2\times\frac{7}{10}\times\left(1-\frac{7}{10}\right)\right) = 0.44$$

基尼指数 Gini(D) 表示集合 D 的不确定性，Gini(D,A) 表示经 A=a 分割后集合 D 的不确定性。基尼指数值越大，样本集合的不确定性就越大，与熵相似。

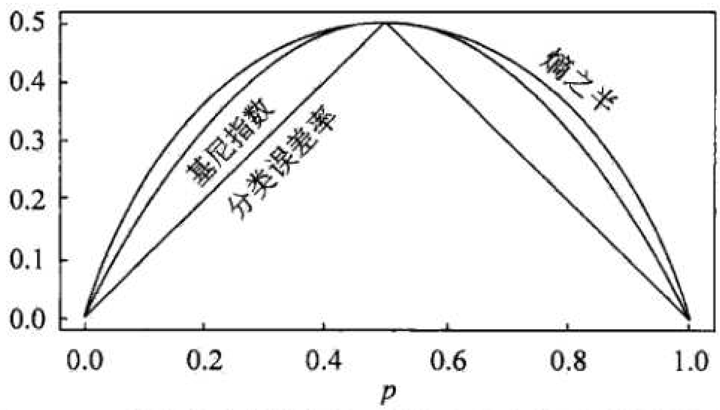

图 5.7　二类分类中基尼指数、熵之半和分类误差率的关系

基尼指数的作用

基尼指数主要是用来选择 **最优特征** 和 **最优切分点** 的。

仍然以例5.1为例，对特征 A_1，A_2，A_3，A_4，分别求对应的：

> **Gini(D,A^1)**
>
> Gini(D,A^1 = 1) = 0.44
>
> Gini(D,A^1 = 2) = 0.48
>
> Gini(D,A^1 = 3) = 0.44
>
> **Gini(D,A^2)**
>
> Gini(D,A^2 = 1) = 0.32
>
> **Gini(D,A^3)**
>
> Gini(D,A^3 = 1) = 0.27 # 所有基尼指数中的最小值
>
> **Gini(D,A^4)**
>
> Gini(D,A^4 = 1) = 0.36
>
> Gini(D,A^4 = 2) = 0.47
>
> Gini(D,A^4 = 3) = 0.32

由上面可以看到，Gini(D,A^3 = 1)=0.27 最小，所以选择特征 A^3 为最优特征，A^3 = 1 为最优分割点。 对另一个节点 A_1，A_2，A_4，继续使用这个方法计算最优特征及最优切分点，依次类推，构建决策树。