第五章 朴素贝叶斯算法

朴素贝叶斯（Naive Bayes）算法利用概率分布进行分类，有深厚的理论基础，算法简单，分类效率高，在文本处理领域获得了广泛的应用。朴素贝叶斯之所以称为朴素，是因为假定所有特征是条件独立的，当实际问题的特征之间有强的相关性时，分类准确性会降低。

1. 朴素贝叶斯原理

利用朴素贝叶斯分类，对一个给定的样本$x=[x_0,x_1,\cdots,x_{m-1}]$，分别计算它属于类别y的概率，即$P(y|x_0,x_1,\cdots,x_{m-1})$，然后比较哪个类别的概率最大，则将其归为哪一类。

其中$P(y|x_0,x_1,\cdots,x_{m-1})$为条件概率，即在给定特征$x_0,x_1,\cdots,x_{m-1}$的条件下，类别为y的概率。

根据贝叶斯公式：　　$P(A|B)=\frac{P(B|A)P(A)}{P(B)}$

$$P(y|x_0,x_1,\cdots,x_{m-1})=\frac{P(y)P(x_0,x_1,\cdots,x_{m-1}|y)}{P(x_0,x_1,\cdots,x_{m-1})}$$

由于假定各特征之间是条件独立的，则有：

$$P(x_0,x_1,\cdots,x_{m-1}|y)=P(x_0|y)P(x_1|y)\cdots P(x_{m-1}|y)=\prod_{j=0}^{m-1}P(x_j|y)$$

于是：

$$P(y|x_0,x_1,\cdots,x_{m-1})=\frac{P(y)\prod_{j=0}^{m-1}P(x_j|y)}{P(x_0,x_1,\cdots,x_{m-1})}$$

上式中的分母对每一个类别都是相同的，因此只需计算各个类别概率的分子项即可。

$$P(y|x_0, x_1, \cdots, x_{m-1}) \propto P(y) \prod_{j=0}^{m-1} P(x_j|y)$$

确定其类别为：

$$\hat{y} = \underset{y}{\operatorname{argmax}} P(y) \prod_{j=0}^{m-1} P(x_j|y)$$

其中P(y)为先验概率，可通过经验获得，或者利用训练集数据、通过统计所有样本中y类别的分率获得；$P(x_j|y)$通过统计训练集中属于y类别的样本中x_j特征出现的分率获得。

为便于理解，来看一个例子。

某婚恋网站统计了12对速配情侣的情况：

表5-1 婚恋网站速配结果样本

长相	性格	身高	表现	女生决定
帅	不好	矮	不上进	不嫁
不帅	好	矮	上进	不嫁
帅	好	矮	上进	嫁
不帅	好	高	上进	嫁
帅	不好	矮	上进	不嫁
不帅	不好	矮	不上进	不嫁
帅	好	高	不上进	嫁
不帅	好	高	上进	嫁
帅	好	高	上进	嫁
不帅	不好	高	上进	嫁
帅	好	矮	不上进	不嫁
帅	好	矮	不上进	不嫁

现在又有一对速配新人，男生特征是：不帅、性格不好、矮、不上进，请你预测一下女生是嫁还是不嫁。

这个数据集中含有12个样本，每个样本有4个特征（长相、性格、身高、表现），分属于两个类别（嫁、不嫁）。要预测新样本属于哪个类别，按照贝叶斯算法，先要计算两个条件概率：P(嫁|不帅,性格不好,矮,不上进)和P(不嫁|不帅,性格不好,矮,不上进)，取概率大的那个类别作为预测分类。
先来计算"嫁"的概率：

$$P(\text{嫁}|\text{不帅},\text{性格不好},\text{矮},\text{不上进}) \propto$$

$$P(\text{嫁})P(\text{不帅}|\text{嫁})P(\text{性格不好}|\text{嫁})P(\text{矮}|\text{嫁})P(\text{不上进}|\text{嫁})$$

对训练集进行统计。

长相	性格	身高	表现	女生决定
帅	不好	矮	不上进	不嫁
不帅	好	矮	上进	不嫁
帅	好	矮	上进	嫁
不帅	好	高	上进	嫁
帅	不好	矮	上进	不嫁
不帅	不好	矮	不上进	不嫁
帅	好	高	不上进	嫁
不帅	好	高	上进	嫁
帅	好	高	上进	嫁
不帅	不好	高	上进	嫁
帅	好	矮	不上进	不嫁
帅	好	矮	不上进	不嫁

12个样本中有6个“嫁”：

$$P(\text{嫁}) = 6/12$$

6个"嫁"的样本中，有3个具有"不帅"的特征：

$$P(\text{不帅}|\text{嫁}) = 3/6$$

6个"嫁"的样本中，有1个具有"性格不好"的特征：

$$P(\text{性格不好}|\text{嫁}) = 1/6$$

6个"嫁"的样本中，有1个具有"矮"的特征：$P(\text{矮}|\text{嫁}) = 1/6$

6个"嫁"的样本中，有1个具有"不上进"的特征：

$$P(\text{不上进}|\text{嫁}) = 1/6$$

因此：

$$P(\text{嫁}|\text{不帅},\text{性格不好},\text{矮},\text{不上进}) \propto \frac{6}{12}\times\frac{3}{6}\times\frac{1}{6}\times\frac{1}{6}\times\frac{1}{6}=\frac{1}{864}$$

类似地计算：

$$P(\text{不嫁}|\text{不帅},\text{性格不好},\text{矮},\text{不上进}) \propto \frac{6}{12}\times\frac{2}{6}\times\frac{3}{6}\times\frac{6}{6}\times\frac{4}{6}=\frac{1}{18}$$

比较可知，显然“不嫁”的概率更大。

如果要计算具体的概率是多少，由于只有两个结果，其概率相加应该等于1，则"不嫁"的概率为：

$$P(\text{不嫁}|\text{不帅},\text{性格不好},\text{矮},\text{不上进}) = \frac{1/18}{1/18+1/864}=0.98$$

2. 朴素贝叶斯的python实现

下面编写朴素贝叶斯算法的程序，并对上一节的问题进行计算。由于在这个例子中，类别只有两个，为了方便计算，对其进行数值化，'嫁'为0，'不嫁'为1。另外，每个特征的取值也只有两个，同样进行数值化，长相中'帅'为0，'不帅'为1；性格中'好'为0，'不好'为1；身高中'高'为0，'矮'为1；表现中'上进'为0，'不上进'为1。

程序输出如下：

```
不嫁
prob of the new sample: [ 0.02040816  0.97959184]
```

程序计算结果与前面手算结果相同，新样本属于'嫁'与'不嫁'类别的概率分别为0.02, 0.98，预测类别为'不嫁'。

3. 利用scikit-learn进行朴素贝叶斯分类

在scikit-learn的naive_bayes模块，实现了四个朴素贝叶斯分类器：BernoulliNB、MultinomialNB、ComplementNB及GaussianNB类，其中：

- BernoulliNB类适用于数据符合多变量Bernoulli分布的情况，即每个特征都只有两个值的数据；
- MultinomialNB适合于多值分布的数据；
- ComplementNB是MultinomialNB的改编版，特别适合于各类分布不均衡即各个类别数目相差较大的数据集；
- GaussianNB适合于符合高斯分布的数据集，当特征值为实数时可以采用GaussianNB类。

四个朴素贝叶斯分类器的用法相似，下面以BernoulliNB类为例介绍其用法：

BernoulliNB(alpha=1.0, binarize=0.0, fit_prior=True, class_prior=None)

参数意义：

- alpha: 设置Laplace/Lidstone平滑参数。当新样本中有一个特征在训练集中并不存在时，该特征的条件概率将等于0，此时会出现0/0或每个类别的概率都为0，为避免这种情况，可以假设每个特征都至少出现alpha次，常用的alpha = 1即称为Laplace平滑，当0<alpha<1时称为Lidstone平滑。
- binarize: 设置对样本特征二值化的阀值，如为None则表示输入数据已经是二值向量。
- fit_prior: 是否由训练集学习得到先验概率，如果设置为False则采用均匀先验分布。
- class_prior: 每个类别的先验概率，如为None，则从训练集计算该值。

主要属性：

- class_log_prior_: 每一个类别的概率的对数值。
- feature_log_prob_: 在给定类别条件下各特征的概率$P(x_i|y)$的对数值。
- class_count_: 每一个类别在训练集中出现的次数。
- feature_count_: 每一个(类别，特征)在训练中的次数。

主要方法：

- fit(X, y, sample_weight = None): 利用训练数据集X, y拟合分类器。sample_weight可以设置每个样本的权重。
- predict(X): 对测试样本或新样本X，预测分类结果。
- predict_proba(X): 对测试样本或新样本X，计算预测结果的概率值。
- score(X, y, sample_weight = None): 对测试集X, y计算预测的平均准确率。

下面利用BernoulliNB类对上一节的问题进行求解。

输出结果为：

```
Classification result: 不嫁
Probobility: [0.07079646 0.92920354]
```

分类结果与上一节中的结果相同，都是‘不嫁’，但输出的概率值却并不相同，这是由于产生BernoulliNB类时默认进行Laplace平滑。

如果强制不进行平滑，即在生成类的时候令alpha=0，再看一下结果：

```
In [10]: nb = BernoulliNB(alpha = 0)

In [11]: nb.fit(X, y)
D:\...:472: UserWarning: alpha too small will result in numeric
errors, setting alpha = 1.0e-10
'setting alpha = %.1e' % _ALPHA_MIN)
Out[11]: BernoulliNB(alpha=0, binarize=0.0, class_prior=None,
fit_prior=True)

In [12]: nb.predict_proba(X_new)
Out[12]: array([[0.02040816, 0.97959184]])
```

这次输出的概率值与上一节的结果相同。但输出一个警告：alpha的值太小将导致数值错误，建议将alpha设为1.0e-10。alpha值的改变虽然会导致最终的概率值略有不同，但不会改变各个类别概率的相对大小，也不会影响最终的分类结果。

4. 利用朴素贝叶斯进行文本分类

朴素贝叶斯最成功的应用是在文本处理方面，包括文件分类、邮件过滤等。下面以一个简单的例子说明文本分类的过程。假定一个社区留言版的一些文本已经进行了分类，如表5-2所示，对于一些新的留言，利用朴素贝叶斯对其进行分类。

表5-2 社区留言版文本分类

留言内容	分类
my dog has flea problem help please	normal
maybe not take him to dog park stupid	abusive
my dalmation is so cute I love him	normal
stop posting stupid worthless garbage	abusive
mr licks ate ny steak how to stop him	normal
quit buying worthless dog food stupid	abusive
love my dog	?
stupid garbage	?

基本思路是先对留言内容进行分割，得到一个词向量，然后构建一个词汇表并从中去除掉重复的单词，然后每一个词向量都可以转化为一个长度与词汇表相同的数值向量，如果词汇表中的某个词在词向量中有出现则相应位置为1，如果没有则相应位置为0。这样就可以将所有留言内容都转化为一个数值向量。由于分类结果中只有两类，我们用0表示normal，用1表示abusive。对于本例，由于特征只有两个取值，因此也可以用BernoulliNB处理。

程序输出结果为：

```
Classification result for 0 sample: normal
Probobility for 0 sample: [0.99310345 0.00689655]
Classification result for 1 sample: abusive
Probobility for 1 sample: [4.94698153e-04 9.99505302e-01]
```