

Lab9 Naïve Bayes 应用实践

实验简介：利用 python 的文本处理能力将文档切分成词，通过集合元素的唯一性生成词汇列表（不包括重复词汇），进而构建词向量（词集向量或词袋向量），从词向量计算概率，然后构建分类器对邮件文档进行垃圾邮件分类。代码文件：bayes.py

1. 准备数据：从文本中构建词向量

在 email/spam 文件夹中有 25 封垃圾邮件，在 email/ham 中有 25 封正常邮件，将其进行垃圾邮件分类。

(1) 切分文本成词

首先遇到的问题是怎样把一封邮件内容进行分词，即将其切割成一个个单词形式。

利用 python 的 string.split()方法切分文本字符串：

```
<<< mySent = 'This book is the best book on Python or M.L. I have ever laid eyes upon.'
```

```
<<< mySent.split()
```

利用正则表达式切分，其中的分隔符是除单词、数字外的任意字符串：

```
<<< import re
```

```
<<< regEx = re.compile('\W+')
```

```
<<< listOfTokens = regEx.split(mySent)
```

```
<<< listOfTokens
```

列表推导式的应用

```
<<< [tok for tok in listOfTokens if len(tok) > 0]
```

```
<<< [tok.lower() for tok in listOfTokens if len(tok) > 0]
```

```
<<< emailText = open('email/ham/6.txt').read()
```

```
<<< listOfTokens = regEx.split(emailText)
```

函数 textParse()实现将一个长的字符串进行分词的操作。

(2) 生成词汇表

将所有的邮件内容分词后，通过集合元素的唯一性生成一个词汇表，每个单词只出现一次，词汇表形式如：['cute', 'love', 'help', 'garbage', 'quit']。

loadDataSet()函数生成实验样本集。

createVocabList()函数建立词汇表：

```
<<< import bayes
```

```
<<< listOPost, listClasses = bayes.loadDataSet()
```

```
<<< myVocabList = bayes.createVocabList(listOPost)
```

```
<<< myVocabList
```

(3) 生成词向量

每一封邮件的词汇都存在词汇表中，可以将每一封邮件生成一个词向量，例如：['cute', 'love', 'help', 'garbage', 'quit']，则它的词向量为[0, 1, 0, 1, 0]，其位置与词汇表所对应，词向量的维度与词汇表相同，其中的数字为相应位置上的词是否出现，这个也称为词集向量模型，由 setOfWords2Vec 函数实现。

bagOfWords2Vec()函数实现词袋模型，在词袋中每个单词可以出现多次。

Fundamentals of Machine Learning

对 `setOfWords2Vec()` 函数稍加修改：遇到每个单词时，增加词向量中的对应值即可。

调用 `setOfWords2Vec()` 函数生成词集向量：

构建 `listOPost` 列表 0 位置对应的词集向量：

```
<<< setOfWords2Vec0 = byyes.setOfWords2Vec(myVocabList, listOPost[0])
```

```
<<< print(setOfWords2Vec0)
```

构建 `listOPost` 列表 3 位置对应的词集向量：

```
<<< setOfWords2Vec3 = bayes.setOfWords2Vec(myVocabList, listOPost[3])
```

```
<<< print(setOfWords2Vec3)
```

2. 训练算法：计算概率

训练模型：在训练样本中计算先验概率 $p(C_i)$ 和条件概率 $p(X | C_i)$ ，本实例有 0 和 1 两个类别，所以返回 $p(X | 0)$ ， $p(X | 1)$ 和 $p(C_i)$ ：

(1) 若某种特征在某类别中没有出现，其概率为 0，导致连乘结果为零。采取各类别默认 1 次累加，总类别（两类）次数 2，这样不影响相对大小。

(2) 若很小的数字相乘，则结果会更小，再四舍五入存在误差，而且会造成下溢出。采用取 \log ，乘法变为加法，并且相对大小趋势不变。

测试 `train()` 函数，返回两个概率向量和一个概率值

for 循环使用词向量充填 `trainMat` 列表

```
<<< trainMat = []
```

```
<<< for postinDoc in listOPost:
```

```
    trainMat.append(bayes.bagOfWords2Vec (myVocabList, postinDoc))
```

```
<<< p0V, p1V, pAb = bayes.train(trainMat, listClasses)
```

```
<<< pAb
```

```
<<< p0V
```

```
<<< p1V
```

3. 测试过程：根据现实情况修改分类器

首先将 50 封邮件（正常邮件和垃圾邮件各 25 封）读进列表中，生成一个词汇表包含所有的单词，使用交叉验证，随机选择 40 个样本进行训练，10 个样本进行测试。

训练模型：40 封训练样本，计算先验概率和类条件概率

测试模型：遍历 10 个测试样本，计算垃圾邮件分类的正确率

`spamTest()` 函数完成测试：

```
<<< bayes.spamTest()
```

由于随机选择样本，可以运行 10 次取平均值。注意，这里一直出现的是将垃圾邮件误判为正常邮件（False Positive），这会比将正常的误判为垃圾邮件（False Negative）要好。

4. 操作练习

(1) 利用 `sklearn` 中 `BernoulliNB` 分类该邮件数据集。

(2) `bayes.py` 中的语句“`from numpy import *`”用语句“`import numpy as np`”代替，修改其中对应的代码，使其能够正常执行。

(3) 将词集向量用 **TF-IDF** 词向量替代，测试分析结果。

(4) 编程实现 PPT 中的例 1，例 2。

(5) 利用朴素贝叶斯算法实现对 `lab6` 的两个数据集分类。

(6) 基于 `Tensorflow` 实现朴素贝叶斯分类器。（扩展）