林之阳 3118311071

1. **在高斯密度的情况下，似然比$p\left(x|c_1\right)/p\left(x|c_2\right)$是什么？**

$$\begin{aligned}\frac{p(x|C_1)}{p(x|C_2)} &= \frac{\frac{1}{\sqrt{2\pi}\sigma_1}\mathrm{e}^{-\frac{(x-\mu_1)^2}{2{\sigma_1}^2}}}{\frac{1}{\sqrt{2\pi}\sigma_2}\mathrm{e}^{-\frac{(x-\mu_2)^2}{2{\sigma_2}^2}}} \\ &= e^{-\frac{(x-\mu_1)^2}{2\sigma^2}+\frac{(x-\mu_2)^2}{2\sigma^2}} \\ &= e^{\frac{2(\mu_1-\mu_2)x-\left(\mu_2^2-\mu_1^2\right)}{2\sigma^2}}\end{aligned}$$

1. **在回归中，我们看到拟合一个二次模型等价于用于对应于输入的平方的附加输入拟合一个线性模型。对于分类，我们也能这样做吗？**

   不太理解题意，没有好的思路，抱歉

2. **用极大似然估计法推出朴素贝叶斯法中的先验概率公式**

   令参数$P(Y=c_k)=\theta_k$，其中$k\in\{1,2..K\}$。

   那么随机变量Y的概率可以用参数来表示为一个紧凑的形式$P(Y)=\sum_{k=1}^{K}\theta_k I(Y=c_k)$，I是指示函数$Y=c_k$成立时，I=1；否则I=0。 极大似然函数

   $L(\theta_k;y_1,y_2..y_N)=\prod_{i=1}^{N}P(y_i)=\prod_{k=1}^{K}\theta_k^{N_k}$，其中N为样本总数，$N_k$为样本中$Y=c_k$的样本数目，取对数得到$l(\theta_k)=ln(L(\theta))=\sum_{k=1}^{K}N_k ln\theta_k$，要求该函数的最大值，注意到约束条件$\sum_{k=1}^{K}\theta_k=1$可以用拉格朗日乘子法，即$l(\theta_k,\lambda)=\sum_{k=1}^{K}N_k ln\theta_k+\lambda(\sum_{k=1}^{K}\theta_k-1)$，求导就可以得到：$\frac{N_k}{\theta_k}+\lambda=0$联立所有的k以及约束条件得到$\theta_k=\frac{N_k}{N}$

3. **有一种肿瘤，得肿瘤的人被检测出为“+性”的几率为90%，未得这种肿瘤的人被检测出“-性”的几率为90%，而人群中得这种肿瘤的几率为1%，一个人被检测出“+性”，问这个人得肿瘤的几率为多少？**

$$\begin{aligned}p\left(Y|+\right) &= \frac{P(+|Y)\times P(Y)}{P(+)} \\ &= \frac{P(+|Y)\times P(Y)}{P(+|Y)\times P(Y)+P(+|N)\times P(N)} \\ &= 8.3\%\end{aligned}$$

4. **实现基于贝叶斯估计的文本分类。测试数据在data文件中，data文件下有女性、体育、文学出版、校园四类文件。程序读取data各个分类下的txt文件，并且预测文件类别，对比看是否与实际类别一致。**

```
# -*- coding: utf-8 -*-

# __author__='vickylzy'

import jieba
import jieba.analyse
import os
from sklearn.feature_extraction.text import CountVectorizer, TfidfTransformer
from sklearn.naive_bayes import MultinomialNB
from sklearn.model_selection import train_test_split

#提取关键词
def extractDirWords(path):
    # path='./hw2data/女性/'
    print('loading '+ path )
    textNum=len(os.listdir(path))
    wordslist=list()
    for i in range(1,textNum+1):  #textNum
        with open(path+str(i)+'.txt','r',errors='ignore') as txt:
            currline=txt.readline()
            # print(currline)
            keyWordList=jieba.analyse.extract_tags(currline,topK=10,allowPOS=
['a','c','d','e','f','i','n','nr','ns','nt','v','vn'])# nz:其他专有名称 p:介词 r:代词])
            line=str()
            for keyw in keyWordList:
                line=line+' '+keyw
            wordslist.append(line)
    return wordslist


if __name__ == '__main__':
    # print(os.path.abspath('.')+'\data')
    girlText=extractDirWords('./data/女性/')
    sportText=extractDirWords('./data/体育/')
    collageText=extractDirWords('./data/校园/')
    labelGirl=['女性']*len(girlText)
    labelSport=['体育']*len(sportText)
    labelCollage=['校园']*len(collageText)

    #构造数据集
    dataSet=girlText+sportText+collageText
    labelSet=labelGirl+labelSport+labelCollage
    print('dataset generated completed!' )

    #分选测试集训练集
    data_train, data_test, target_train, target_test = train_test_split(dataSet, labelSet,
test_size=0.2, random_state=0)
```

```
    #汉语词组特征化
    #maxDF去除过高词频单词“个，评论等...”
    count_ver= CountVectorizer( max_df = 0.8, decode_error = 'ignore')
    trainVector=count_ver.fit_transform(data_train)
    tfidTer=TfidfTransformer()
    tfidVector=tfidTer.fit(trainVector).transform(trainVector)

    #采用训练集生成的词典
    count_ver_t= CountVectorizer(vocabulary=count_ver.vocabulary_, max_df = 0.8,
decode_error = 'ignore')
    trainVector_t=count_ver_t.fit_transform(data_test)
    tfidVector_t=tfidTer.fit(trainVector_t).transform(trainVector_t)
    print('vectorizering completed!' )

    #train
    muNB=MultinomialNB(alpha =0.01)  #0.01?
    muNB.fit(tfidVector,target_train)
    print('training completed!' )

    #predict
    print('predicting result...' )
    predict_result=muNB.predict(tfidVector_t)

    #展示结果
    correct = [target_test[i]==predict_result[i] for i in range(len(predict_result))]
    r = len(predict_result)
    t = correct.count(True)
    f = correct.count(False)
    print('predict times: '+str(r)+'\ncorrect times: '+str(t)+'\ncorr pencentage:
'+str(float(t/r)))


    >>>dataset generated completed!
    >>>vectorizering completed!
    >>>training completed!
    >>>predicting result...
    >>>predict times: 508
    >>>correct times: 478
    >>>corr pencentage: 0.9409448818897638
```