华东师范大学数据科学与工程学院实验报告

课程名称：统计学习与机器方法	年级：2019	实践成绩：
指导教师：董启文	**姓名**：周辛娜	**学号**：10195501442
上机实践： Project2三种分布贝叶斯解决文本分类	**上机实践时间**：	2021.12.20

任务：

1000个文档分成20类，五重交叉验证结果。

实验数据集：

从(https://www.cs.cmu.edu/afs/cs.cmu.edu/project/theo-11/www/naive-bayes.html)下载数据集，数据集包含来自 20 个新闻组中每个组的 1000 个文档。由于下载下来是文本的形式，并不是现成的.csv的格式，需要进行整理。

一、探索文本数据

经过整理后的数据全貌：

	Unnamed: 0	data	target
0	0	Xref: cantaloupe.srv.cs.cmu.edu alt.atheism:51...	alt.atheism
1	1	Path: cantaloupe.srv.cs.cmu.edu!crabapple.srv....	alt.atheism
2	2	Path: cantaloupe.srv.cs.cmu.edu!crabapple.srv....	alt.atheism
3	3	Path: cantaloupe.srv.cs.cmu.edu!crabapple.srv....	alt.atheism
4	4	Path: cantaloupe.srv.cs.cmu.edu!das-news.harva...	alt.atheism
...	...	...	...
1995	1995	Xref: cantaloupe.srv.cs.cmu.edu alt.atheism:54...	talk.religion.misc
1996	1996	Xref: cantaloupe.srv.cs.cmu.edu alt.atheism:54...	talk.religion.misc
1997	1997	Path: cantaloupe.srv.cs.cmu.edu!magnesium.club...	talk.religion.misc
1998	1998	Path: cantaloupe.srv.cs.cmu.edu!magnesium.club...	talk.religion.misc
1999	1999	Xref: cantaloupe.srv.cs.cmu.edu alt.atheism:54...	talk.religion.misc

2000 rows × 3 columns

数据集有2000行，3列，每行表示一个新闻，第一列不需要，第二列是data，对应的新闻的文本内容，第三列是新闻所属的类别。

查看第一条新闻的内容：

```
df['data'][0]
```

```
'Xref: cantaloupe.srv.cs.cmu.edu alt.atheism:51121 soc.motss:139944 rec.scouting:5318\nNewsgroups: alt.atheism,soc.motss,rec.scout
ing\nPath: cantaloupe.srv.cs.cmu.edu!crabapple.srv.cs.cmu.edu!fs7.ece.cmu.edu!europa.eng.gtefsd.com!howland.reston.ans.net!wupost!
uunet!newsgate.watson.ibm.com!yktnews.watson.ibm.com!watson!Watson.Ibm.Com!strom\nFrom: strom@Watson.Ibm.Com (Rob Strom)\nSubject:
Re: [soc.motss, et al.] "Princeton axes matching funds for Boy Scouts"\nSender: @watson.ibm.com\nMessage-ID: <1993Apr05.180116.433
46@watson.ibm.com>\nDate: Mon, 05 Apr 93 18:01:16 GMT\nDistribution: usa\nReferences: <C47EFs.3q47@austin.ibm.com> <1993Mar22.0331
50.17345@cbnewsl.cb.att.com> <N4HY.93Apr5120934@harder.ccr-p.ida.org>\nOrganization: IBM Research\nLines: 15\n\nIn article <N4HY.9
3Apr5120934@harder.ccr-p.ida.org>, n4hy@harder.ccr-p.ida.org (Bob McGwier) writes:\n\n|> [1] HOWEVER, I hate economic terrorism an
d political correctness\n|> worse than I hate this policy.  \n\n\n|> [2] A more effective approach is to stop donating\n|> to ANY
organizating that directly or indirectly supports gay rights issues\n|> until they end the boycott on funding of scouts.  \n\nCan
somebody reconcile the apparent contradiction between [1] and [2]?\n\n-- \nRob Strom, strom@watson.ibm.com, (914) 784-7641\nIBM Re
search, 30 Saw Mill River Road, P.O. Box 704, Yorktown Heights, NY  10598\n'
```

查看一共20个类别：

```
np.unique(df.target)
```

```
array(['alt.atheism', 'comp.graphics', 'comp.os.ms-windows.misc',
       'comp.sys.ibm.pc.hardware', 'comp.sys.mac.hardware',
       'comp.windows.x', 'misc.forsale', 'rec.autos', 'rec.motorcycles',
       'rec.sport.baseball', 'rec.sport.hockey', 'sci.crypt',
       'sci.electronics', 'sci.med', 'sci.space',
       'soc.religion.christian', 'talk.politics.guns',
       'talk.politics.mideast', 'talk.politics.misc',
       'talk.religion.misc'], dtype=object)
```

接下来看样本均不均衡，这是要代入朴素贝叶斯模型之前需要关注的：

```
alt.atheism 0.05
comp.graphics 0.05
comp.os.ms-windows.misc 0.05
comp.sys.ibm.pc.hardware 0.05
comp.sys.mac.hardware 0.05
comp.windows.x 0.05
misc.forsale 0.05
rec.autos 0.05
rec.motorcycles 0.05
rec.sport.baseball 0.05
rec.sport.hockey 0.05
sci.crypt 0.05
sci.electronics 0.05
sci.med 0.05
sci.space 0.05
soc.religion.christian 0.05
talk.politics.guns 0.05
talk.politics.mideast 0.05
talk.politics.misc 0.05
talk.religion.misc 0.05
```

可以看到总体是比较均衡

二、文本特征提取

使用TF-IDF向量计数

在开始分类之前，必须先将文本编码成数字，一般常用的方法是单词计数向量计数和TF-IDF向量计数方法。

1. 单词计数向量，在这种技术中，一个样本可以包含一段话或者一篇文章，这个样本如果出现了10个单词，就会有10个特征，每个特征代表一个单词，特征的取值代表这个单词在这个样本中总共出现了几次，是一个离散的，代表次数的整数。
   在sklearn中，单词计数向量计数可以通过feature_extraction.text模块中的CountVectorizer类实现.
2. 可以预见，如果使用单词计数向量，可能会导致一部分常用词频繁出现在矩阵中并且占有很高的权重，对分类来说，这明显是对算法的一种误导。为了解决这个问题，比起使用次数，使用单词在句子中所占的比例来编码单词，这就是TF-IDF方法，词频逆文档频率，是通过单词在文档中出现的频率来衡量其权重，IDF的大小与一个词的常见程度成反比，这个词越常见，编码后为它设置的权重

就会倾向于越小，从而来压制频繁出现的一些无意义的词。
在sklearn中，使用feature_extraction.text中类TfidVectorizer来执行这种编码

在实验初期，两种方法都用了，但是TF-IDF向量计数效果要好一些，所以就使用TF-IDF向量计数进行文本特征提取。

三、划分训练集测试集

```
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,random_state=0)
```

四、归一化

在本实验集上，归一化并没有什么实质的效果。

五、构建模型：

1. 先使用三种不同分布的朴素贝叶斯模型先进行预测，分别是Multinomial，Complement，Bernuolli

模型介绍：

朴素贝叶斯有各种各样的假设，除了“朴素”的假设即假设变量之间相互独立的假设，还有对于概率分布的假设，包括多项式朴素贝叶斯multinomialNB、伯努利朴素贝叶斯BernuolliNB（二项分布，数据集中存在多个特征，但每个特征都是二分类的，可以用布尔变量表示，由于本数据集中有20个类，那么可以使用类中专门用来二值化的参数binarize来改变数据）、高斯朴素贝叶斯和补集朴素贝叶斯。

1. 多项式朴素贝叶斯适用于二项分布、多项分布，擅长分类型变量，多项式朴素贝叶斯的特征矩阵经常是稀疏矩阵，所以经常被用于文本分类。
2. 伯努利朴素贝叶斯和多项式朴素贝叶斯非常相似，常用于处理文本分类数据，但由于伯努利朴素贝叶斯处理二项分布，所以更在意存在与否，而不是出现多少次，这是两者的根本不同。在文本分类的情况下，伯努利朴素贝叶斯可以使用单词出现向量而不是单词计数向量来训练分类器，在文档较短的数据集上，伯努利朴素贝叶斯效果更好。它适用于二项分布，数据集中存在多个特征，但每个特征都是二分类的，可以用布尔变量表示，由于本数据集中有20个类，那么可以使用类中专门用来二值化的参数binarize来改变数据，在本实验中也试着采用binarize，但效果反而更差，所以就不进行该项操作了。
3. 高斯朴素贝叶斯是不接受稀疏矩阵的，而这里恰恰就是稀疏矩阵，所以在本次实验中就无法使用高斯朴素贝叶斯。
4. 补集朴素贝叶斯是多项式朴素贝叶斯的改进，它能够解决样本不平衡的问题，并且能够一定程度上忽略朴素假设。

超参数：平滑系数 alpha，是为了防止训练数据中出现过的一些词汇没有出现在测试集中导致0概率，之后会采用五折交叉验证得到这三种模型当超参数取哪个值的时候表现最优。

当平滑系数 alpha默认为1时的实验结果：

Multinomial
Accuracy:0.692

	precision	recall	f1-score	support
alt.atheism	0.79	0.66	0.72	29
comp.graphics	1.00	0.23	0.37	31
comp.os.ms-windows.misc	0.83	0.66	0.73	29
comp.sys.ibm.pc.hardware	0.64	0.56	0.60	25
comp.sys.mac.hardware	0.49	0.95	0.64	20
comp.windows.x	0.72	0.59	0.65	22
misc.forsale	0.80	0.70	0.74	23
rec.autos	0.90	0.75	0.82	24
rec.motorcycles	1.00	0.42	0.59	31
rec.sport.baseball	0.82	0.78	0.80	23
rec.sport.hockey	0.85	0.96	0.90	23
sci.crypt	0.86	0.96	0.91	26
sci.electronics	0.71	0.55	0.62	22
sci.med	0.71	0.89	0.79	19
sci.space	0.88	0.88	0.88	24
soc.religion.christian	0.40	1.00	0.57	24
talk.politics.guns	0.47	0.96	0.63	23
talk.politics.mideast	0.96	0.80	0.87	30
talk.politics.misc	0.71	0.36	0.48	28
talk.religion.misc	0.50	0.54	0.52	24
accuracy			0.69	500
macro avg	0.75	0.71	0.69	500
weighted avg	0.76	0.69	0.69	500

00:00:071227
Complement
Accuracy:0.808

	precision	recall	f1-score	support
alt.atheism	0.74	0.79	0.77	29
comp.graphics	0.85	0.71	0.77	31
comp.os.ms-windows.misc	0.78	0.86	0.82	29
comp.sys.ibm.pc.hardware	0.77	0.68	0.72	25
comp.sys.mac.hardware	0.84	0.80	0.82	20
comp.windows.x	0.94	0.68	0.79	22
misc.forsale	0.90	0.83	0.86	23
rec.autos	1.00	0.92	0.96	24
rec.motorcycles	0.87	0.87	0.87	31
rec.sport.baseball	0.79	0.83	0.81	23
rec.sport.hockey	0.77	1.00	0.87	23
sci.crypt	0.87	1.00	0.93	26
sci.electronics	0.88	0.68	0.77	22
sci.med	0.86	0.95	0.90	19
sci.space	0.92	0.92	0.92	24
soc.religion.christian	0.80	1.00	0.89	24
talk.politics.guns	0.70	0.91	0.79	23
talk.politics.mideast	0.93	0.93	0.93	30
talk.politics.misc	0.65	0.39	0.49	28
talk.religion.misc	0.41	0.46	0.43	24
accuracy			0.81	500
macro avg	0.81	0.81	0.81	500
weighted avg	0.81	0.81	0.80	500

00:00:042216
Bernuolli
Accuracy:0.476

	precision	recall	f1-score	support
alt.atheism	0.87	0.45	0.59	29
comp.graphics	0.50	0.03	0.06	31
comp.os.ms-windows.misc	0.00	0.00	0.00	29
comp.sys.ibm.pc.hardware	1.00	0.12	0.21	25
comp.sys.mac.hardware	0.14	1.00	0.25	20
comp.windows.x	0.35	0.73	0.47	22
misc.forsale	0.26	0.91	0.41	23
rec.autos	0.67	0.67	0.67	24
rec.motorcycles	1.00	0.06	0.12	31
rec.sport.baseball	0.74	0.87	0.80	23
rec.sport.hockey	0.88	0.91	0.89	23
sci.crypt	0.92	0.42	0.58	26
sci.electronics	1.00	0.50	0.67	22
sci.med	0.92	0.58	0.71	19

```
                  sci.space       1.00      0.38      0.55        24
soc.religion.christian       0.94      0.71      0.81        24
    talk.politics.guns       0.60      0.78      0.68        23
 talk.politics.mideast       1.00      0.27      0.42        30
    talk.politics.misc       0.75      0.11      0.19        28
    talk.religion.misc       0.52      0.71      0.60        24

              accuracy                           0.48       500
             macro avg       0.70      0.51      0.48       500
          weighted avg       0.71      0.48      0.46       500

00:00:101940
```

为了对比更加清晰，用表格显示上面的结果：

模型	用时	accuracy
Multinomial	7ms	0.692
Complement	4ms	0.808
Bernuolli	10ms	0.476

在平滑系数 alpha默认为1时，Complement不仅Accuracy最高而且用时最短，表现最好

2. 分别在这三种模型下通过交叉验证，选择最优超参数

交叉验证介绍：

本次实验采用五折交叉验证，5-fold cross validation，将数据集分为3部分，training set、validation set和test set，一般划分的比例是3:1:1，三者之间没有交集。

超参数alpha的取值范围：0.00001，0.0001，0.0005，0.001，0.005，0.01，0.05，0.1，0.2，0.5，0.7，0.9，1。由于有三个模型，所以先要画出随着alpha的变化，三种不同模型的平均accuracy的变化，进而选择较优的模型，然后针对该模型，画出对应的随着alpha变化，对应的交叉验证图，选择平均准确率最高且方差较小的alpha。也就是不断增大alpha，对于每一个alpha，计算验证集上的平均accuracy和standard deviation，选取accuracy较大且表现稳定的alpha。

由于参数的搜索空间比较大，而且又是五折交叉验证，比较耗时，对于更大型的数据集合深度学习来说，较少使用，本实验的数据集并不大，所以耗时还能接受。

交叉验证过程：

以下为交叉验证的过程。

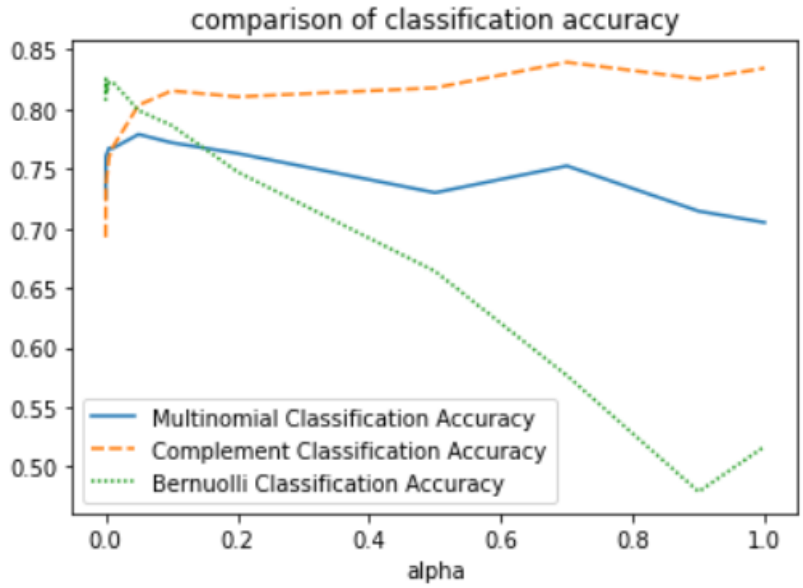
```
alpha: 1e-05
-------------fold0.0------------
1 val_accuracy1: 0.7375
1 val_accuracy2: 0.7
1 val_accuracy3: 0.8
-------------fold1.0------------
2 val_accuracy1: 0.73
2 val_accuracy2: 0.7
2 val_accuracy3: 0.81
-------------fold2.0------------
3 val_accuracy1: 0.7375
3 val_accuracy2: 0.7125
3 val_accuracy3: 0.8125
-------------fold3.0------------
4 val_accuracy1: 0.7575
4 val_accuracy2: 0.7075
4 val_accuracy3: 0.81
-------------fold4.0------------
5 val_accuracy1: 0.735
5 val_accuracy2: 0.6725
5 val_accuracy3: 0.81
alpha: 1e-05 accuracy1: 0.7394999999999999
alpha: 1e-05 accuracy2: 0.6984999999999999
alpha: 1e-05 accuracy3: 0.8085000000000001


alpha: 0.0001
-------------fold0.0------------
1 val_accuracy1: 0.7375
1 val_accuracy2: 0.7125
1 val_accuracy3: 0.825
-------------fold1.0------------
2 val_accuracy1: 0.73
2 val_accuracy2: 0.68
2 val_accuracy3: 0.82
-------------fold2.0------------
3 val_accuracy1: 0.76
3 val_accuracy2: 0.75
3 val_accuracy3: 0.8
-------------fold3.0------------
4 val_accuracy1: 0.7425
4 val_accuracy2: 0.6975
4 val_accuracy3: 0.805
-------------fold4.0------------
5 val_accuracy1: 0.745
5 val_accuracy2: 0.725
5 val_accuracy3: 0.81
alpha: 0.0001 accuracy1: 0.7430000000000001
alpha: 0.0001 accuracy2: 0.713
alpha: 0.0001 accuracy3: 0.812


alpha: 0.0005
-------------fold0.0------------
1 val_accuracy1: 0.7975
1 val_accuracy2: 0.785
1 val_accuracy3: 0.82
-------------fold1.0------------
2 val_accuracy1: 0.77
2 val_accuracy2: 0.72
2 val_accuracy3: 0.835
-------------fold2.0------------
3 val_accuracy1: 0.75
3 val_accuracy2: 0.7225
3 val_accuracy3: 0.8025
-------------fold3.0------------
4 val_accuracy1: 0.7525
4 val_accuracy2: 0.7075
4 val_accuracy3: 0.7925
-------------fold4.0------------
5 val_accuracy1: 0.7525
5 val_accuracy2: 0.7275
5 val_accuracy3: 0.8325
alpha: 0.0005 accuracy1: 0.7645
alpha: 0.0005 accuracy2: 0.7325
alpha: 0.0005 accuracy3: 0.8164999999999999


alpha: 0.001
-------------fold0.0------------
1 val_accuracy1: 0.7675
1 val_accuracy2: 0.7275
1 val_accuracy3: 0.8125
-------------fold1.0------------
2 val_accuracy1: 0.76
2 val_accuracy2: 0.725
2 val_accuracy3: 0.81
-------------fold2.0------------
3 val_accuracy1: 0.8075
3 val_accuracy2: 0.7475
3 val_accuracy3: 0.8575
-------------fold3.0------------
4 val_accuracy1: 0.76
4 val_accuracy2: 0.72
4 val_accuracy3: 0.8425
-------------fold4.0------------
5 val_accuracy1: 0.7725
5 val_accuracy2: 0.7375
5 val_accuracy3: 0.835
alpha: 0.001 accuracy1: 0.7735
alpha: 0.001 accuracy2: 0.7314999999999999
alpha: 0.001 accuracy3: 0.8314999999999999


alpha: 0.005
```

```
-------------fold0.0------------
1 val_accuracy1: 0.8
1 val_accuracy2: 0.7625
1 val_accuracy3: 0.835
-------------fold1.0------------
2 val_accuracy1: 0.7875
2 val_accuracy2: 0.7675
2 val_accuracy3: 0.855
-------------fold2.0------------
3 val_accuracy1: 0.765
3 val_accuracy2: 0.7675
3 val_accuracy3: 0.8375
-------------fold3.0------------
4 val_accuracy1: 0.7675
4 val_accuracy2: 0.745
4 val_accuracy3: 0.82
-------------fold4.0------------
5 val_accuracy1: 0.7575
5 val_accuracy2: 0.75
5 val_accuracy3: 0.8025
alpha: 0.005 accuracy1: 0.7755
alpha: 0.005 accuracy2: 0.7585
alpha: 0.005 accuracy3: 0.8299999999999998


alpha: 0.01
-------------fold0.0------------
1 val_accuracy1: 0.7875
1 val_accuracy2: 0.8125
1 val_accuracy3: 0.81
-------------fold1.0------------
2 val_accuracy1: 0.8
2 val_accuracy2: 0.78
2 val_accuracy3: 0.855
-------------fold2.0------------
3 val_accuracy1: 0.7875
3 val_accuracy2: 0.7775
3 val_accuracy3: 0.8275
-------------fold3.0------------
4 val_accuracy1: 0.765
4 val_accuracy2: 0.7675
4 val_accuracy3: 0.8375
-------------fold4.0------------
5 val_accuracy1: 0.7425
5 val_accuracy2: 0.745
5 val_accuracy3: 0.7775
alpha: 0.01 accuracy1: 0.7765000000000001
alpha: 0.01 accuracy2: 0.7765000000000001
alpha: 0.01 accuracy3: 0.8215


alpha: 0.05
-------------fold0.0------------
1 val_accuracy1: 0.7725
1 val_accuracy2: 0.79
1 val_accuracy3: 0.8075
-------------fold1.0------------
2 val_accuracy1: 0.7775
2 val_accuracy2: 0.8175
2 val_accuracy3: 0.8225
-------------fold2.0------------
3 val_accuracy1: 0.7525
3 val_accuracy2: 0.8175
3 val_accuracy3: 0.79
-------------fold3.0------------
4 val_accuracy1: 0.7675
4 val_accuracy2: 0.7825
4 val_accuracy3: 0.815
-------------fold4.0------------
5 val_accuracy1: 0.79
5 val_accuracy2: 0.83
5 val_accuracy3: 0.795
alpha: 0.05 accuracy1: 0.772
alpha: 0.05 accuracy2: 0.8074999999999999
alpha: 0.05 accuracy3: 0.806


alpha: 0.1
-------------fold0.0------------
1 val_accuracy1: 0.785
1 val_accuracy2: 0.82
1 val_accuracy3: 0.795
-------------fold1.0------------
2 val_accuracy1: 0.81
2 val_accuracy2: 0.83
2 val_accuracy3: 0.785
-------------fold2.0------------
3 val_accuracy1: 0.7575
3 val_accuracy2: 0.7925
3 val_accuracy3: 0.7825
-------------fold3.0------------
4 val_accuracy1: 0.755
4 val_accuracy2: 0.795
4 val_accuracy3: 0.8025
-------------fold4.0------------
5 val_accuracy1: 0.7875
5 val_accuracy2: 0.8125
5 val_accuracy3: 0.8375
alpha: 0.1 accuracy1: 0.779
alpha: 0.1 accuracy2: 0.8099999999999999
alpha: 0.1 accuracy3: 0.8005000000000001


alpha: 0.2
```

```
-------------fold0.0------------
1 val_accuracy1: 0.7525
1 val_accuracy2: 0.825
1 val_accuracy3: 0.7575
-------------fold1.0------------
2 val_accuracy1: 0.7325
2 val_accuracy2: 0.82
2 val_accuracy3: 0.7075
-------------fold2.0------------
3 val_accuracy1: 0.745
3 val_accuracy2: 0.8175
3 val_accuracy3: 0.75
-------------fold3.0------------
4 val_accuracy1: 0.7925
4 val_accuracy2: 0.8475
4 val_accuracy3: 0.785
-------------fold4.0------------
5 val_accuracy1: 0.745
5 val_accuracy2: 0.825
5 val_accuracy3: 0.745
alpha: 0.2 accuracy1: 0.7535000000000001
alpha: 0.2 accuracy2: 0.827
alpha: 0.2 accuracy3: 0.749


alpha: 0.5
-------------fold0.0------------
1 val_accuracy1: 0.735
1 val_accuracy2: 0.8175
1 val_accuracy3: 0.63
-------------fold1.0------------
2 val_accuracy1: 0.7225
2 val_accuracy2: 0.825
2 val_accuracy3: 0.635
-------------fold2.0------------
3 val_accuracy1: 0.7725
3 val_accuracy2: 0.8625
3 val_accuracy3: 0.7425
-------------fold3.0------------
4 val_accuracy1: 0.7625
4 val_accuracy2: 0.8525
4 val_accuracy3: 0.7175
-------------fold4.0------------
5 val_accuracy1: 0.71
5 val_accuracy2: 0.8275
5 val_accuracy3: 0.595
alpha: 0.5 accuracy1: 0.7404999999999999
alpha: 0.5 accuracy2: 0.837
alpha: 0.5 accuracy3: 0.664


alpha: 0.7
-------------fold0.0------------
1 val_accuracy1: 0.72
1 val_accuracy2: 0.835
1 val_accuracy3: 0.6425
-------------fold1.0------------
2 val_accuracy1: 0.7425
2 val_accuracy2: 0.8275
2 val_accuracy3: 0.66
-------------fold2.0------------
3 val_accuracy1: 0.7125
3 val_accuracy2: 0.8675
3 val_accuracy3: 0.5225
-------------fold3.0------------
4 val_accuracy1: 0.7325
4 val_accuracy2: 0.85
4 val_accuracy3: 0.49
-------------fold4.0------------
5 val_accuracy1: 0.7075
5 val_accuracy2: 0.835
5 val_accuracy3: 0.5575
alpha: 0.7 accuracy1: 0.723
alpha: 0.7 accuracy2: 0.843
alpha: 0.7 accuracy3: 0.5745


alpha: 0.9
-------------fold0.0------------
1 val_accuracy1: 0.695
1 val_accuracy2: 0.8175
1 val_accuracy3: 0.61
-------------fold1.0------------
2 val_accuracy1: 0.7225
2 val_accuracy2: 0.8125
2 val_accuracy3: 0.38
-------------fold2.0------------
3 val_accuracy1: 0.6725
3 val_accuracy2: 0.8025
3 val_accuracy3: 0.495
-------------fold3.0------------
4 val_accuracy1: 0.7075
4 val_accuracy2: 0.82
4 val_accuracy3: 0.4775
-------------fold4.0------------
5 val_accuracy1: 0.715
5 val_accuracy2: 0.85
5 val_accuracy3: 0.345
alpha: 0.9 accuracy1: 0.7024999999999999
alpha: 0.9 accuracy2: 0.8205
alpha: 0.9 accuracy3: 0.4615


alpha: 1
-------------fold0.0------------
```

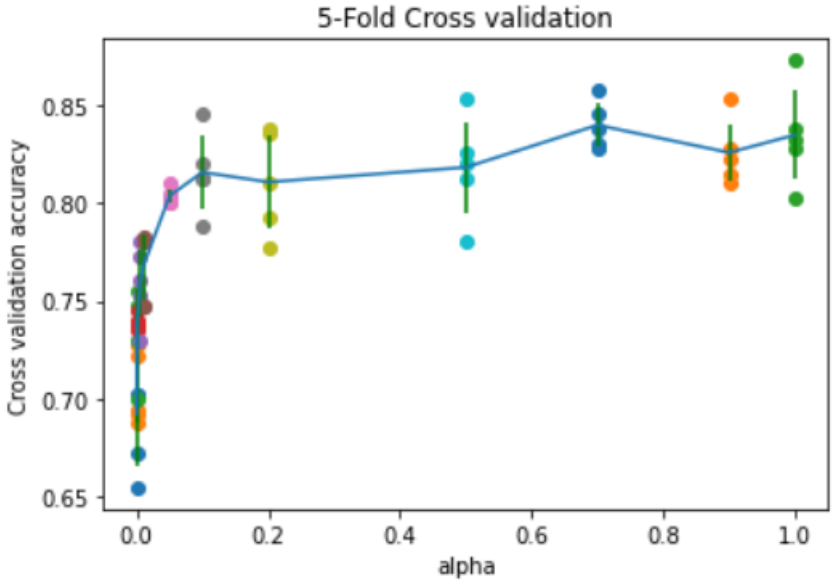
```
1 val_accuracy1: 0.695
1 val_accuracy2: 0.835
1 val_accuracy3: 0.3825
------------fold1.0------------
2 val_accuracy1: 0.745
2 val_accuracy2: 0.84
2 val_accuracy3: 0.42
------------fold2.0------------
3 val_accuracy1: 0.6675
3 val_accuracy2: 0.7825
3 val_accuracy3: 0.5925
------------fold3.0------------
4 val_accuracy1: 0.7425
4 val_accuracy2: 0.8325
4 val_accuracy3: 0.365
------------fold4.0------------
5 val_accuracy1: 0.675
5 val_accuracy2: 0.805
5 val_accuracy3: 0.45
alpha: 1 accuracy1: 0.7050000000000001
alpha: 1 accuracy2: 0.819
alpha: 1 accuracy3: 0.442
```

结果：



由此可以看到，Multinomial和Bernuolli的classification accuracy均是先升高再下降，但是这两种方法的最大值要小于complement的最大值，complement的classification accuracy是明显要高于Multinomial和Bernuolli，所以接下来用complement最好的模型参数来看看最后的结果

选取平均准确率最高且方差较小的alpha作为最后模型的参数：



六、最后达到的效果

	precision	recall	f1-score	support
alt.atheism	0.85	0.65	0.73	17
comp.graphics	0.85	0.77	0.81	22
comp.os.ms-windows.misc	0.78	0.91	0.84	23
comp.sys.ibm.pc.hardware	0.88	0.75	0.81	20
comp.sys.mac.hardware	0.81	0.94	0.87	18
comp.windows.x	0.93	0.65	0.76	20
misc.forsale	0.82	0.90	0.86	20
rec.autos	0.96	0.92	0.94	24
rec.motorcycles	0.94	1.00	0.97	15
rec.sport.baseball	1.00	0.82	0.90	17
rec.sport.hockey	0.89	0.96	0.93	26
sci.crypt	0.91	1.00	0.95	20
sci.electronics	0.93	0.87	0.90	15
sci.med	0.90	1.00	0.95	19
sci.space	0.82	0.82	0.82	17
soc.religion.christian	0.74	0.94	0.83	18
talk.politics.guns	0.61	0.89	0.72	19
talk.politics.mideast	0.92	1.00	0.96	22
talk.politics.misc	0.75	0.46	0.57	26
talk.religion.misc	0.55	0.50	0.52	22
accuracy			0.83	400
macro avg	0.84	0.84	0.83	400
weighted avg	0.84	0.83	0.83	400

最后达到83%的准确率，效果还不错。

七、总结

本次实验采用用三种不同分布朴素贝叶斯解决文本分类，通过整理数据、探索文本数据、文本特征提取、划分训练集测试集、归一化、构建模型、先使用三种不同分布的朴素贝叶斯模型先进行预测，分别是Multinomial，Complement，Bernuolli，分别在这三种模型下通过交叉验证，选择最优超参数，最后用complement最好的模型参数来看最后的结果。最终达到了83%的准确率。

对于本次实验的数据集来说，Complement是最适合的贝叶斯模型，不仅运行速度是最快的而且准确率是最高的，但是对于不同的数据集，可能Bernuolli、Multinomial的表现会更优秀，这个要根据具体情况确定。