A、对于算法的描述

（一）朴素贝叶斯分类器

朴素贝叶斯是一种有监督学习方法，它是一种基于概率的学习方法。文档 d 属于类别 c 的概率计算方法如下：

$$P(c|d) \propto P(c) \prod_{1 \le k \le n_d} P(t_k|c)$$

其中 $P(t_k|c)$是 t_k 出现在类 c 中的条件概率，也可将其视为当正确类为 c 时 t_k 的贡献程度。P(c)是文档出现在类 c 中的先验概率。如果根据文档并不能清晰地区分它属于哪一类，就选择先验概率最大的那个类。

在文本分类中，我们的目标是找出文档最可能属于的类别。对于 NB 分类来说，最可能的类是具有最大后验概率(MAP)估计值的结果 $c$map:

$$c_{\text{map}} = \arg\max_{c \in \mathbb{C}} \hat{P}(c|d) = \arg\max_{c \in \mathbb{C}} \hat{P}(c) \prod_{1 \le k \le n_d} \hat{P}(t_k|c)$$

我们不知道参数的确切值，所以上述公式采用从训练集估计出的参数值代替 p。

由于很多小概率的乘积会导致浮点数下溢出，故通过取对数的方式将原来的成绩计算编程求和计算。由于 **log 是单调函数**，因此得分最高的类别不会发生改变。因此，实际中常使用的是：

$$c_{\text{map}} = \arg\max_{c \in \mathbb{C}} [\log \hat{P}(c) + \sum_{1 \le k \le n_d} \log \hat{P}(t_k|c)]$$

如何从训练数据中估计参数？先使用 MLE 估计类别先验概率：

$$\hat{P}(c) = \frac{N_c}{N}$$

N_c : 类 c 中的文档数目; N: 所有文档的总数

而条件概率的估计值为 t 在 c 类文档中出现的相对频率：

$$\hat{P}(t|c) = \frac{T_{ct}}{\sum_{t' \in V} T_{ct'}}$$

T_{ct} 是训练集中类别 c 中的词条 t 的个数 (多次出现要计算多次)

MLE 估计的一个问题是：对没有在训练集出现的<词项，类别>来说，其 MLE 的估计值为 0，出现零概率的主要原因是数据的稀疏性，即训练集合永远都不可能大到所有罕见的事件都能出现，这样就难以计算这些事件的概率。

为了去掉零概率，一个简单的方法是采用加一平滑，在随机变量上各个取值的频数上赋予一个正数 lambda，如果是拉普拉斯平滑，lambda=1：

$$\hat{P}(t|c) = \frac{T_{ct} + 1}{\sum_{t' \in V}(T_{ct'} + 1)} = \frac{T_{ct} + 1}{(\sum_{t' \in V} T_{ct'}) + B}$$

其中 B=|V|是词汇表中所有词项的数目。加一平滑可以认为是采用均匀分布作为词项的先验分布（每个词项在每个类中出现一次）然后根据训练数据进行更新得到的结果。

然而平滑值 lambda 未必一定要取 1，可以取任意整数，将其视为一个超参数，在交叉验证时调优。

（二）特征选择

特征选择是从训练集合出现的词项中选出一部分子集的过程。在文本分类过程中仅使用这个自己作为特征。特征选择有两个主要目的：一，通过减少有效的词汇空间来提高分类器的训练和应用的效率。二，特征选择能够去除噪声特征，从而提高分类的精度。噪声特征指的是那些加入文本表示后反而会增加新数据上的分类错误率的特征。

文档频数是最为简单的一种特征选择算法，它指的是数据集中有多少个文本包含这个单词。在训练文本集中对每个特征计算它的文档频数，若该项的 DF 值小于某个阈值则将其删除，若其 DF 值大于某个阈值也将其去掉。因为他们分别代表了“没有代表性”和“没有区分度”2 种极端的情况。DF 的优点在于计算量很小，而在实际运用中却有很好的效果。缺点是稀有词可能在某一类文本中并不稀有，也可能包含着重要的判断信息，简单舍弃，可能影响分类器的精度。

设定 up 和 down 两个超参数，表示当一个词项的文档频率大于 up 或小于 down 时，从词表中去掉该词项。通过交叉验证来选取合适的超参数。

（三）评价指标

用 F 值作为评价指标：

$$F_1 = \frac{1}{\frac{1}{2}\frac{1}{P} + \frac{1}{2}\frac{1}{R}} = \frac{2PR}{P+R}$$

对于每一个类都能计算出 F1 值，但我们希望得到在所有类别上的综合性能，有两种计算方法：第一，宏平均，对类别 C 中的每个类都计算一个 F1 值，对 C 个结果求平均。第二，微平均，对类别集合 C 中的每个类都计算 TP、FP 和 FN，将 C 中的这些数字累加，基于累加的 TP、FP、FN 计算 P、R 和 F1。

宏平均对所有类一视同仁，而微平均受返回的 TP、FP 和 FN 比较大的类的影响，我们在此采用宏平均作为评价指标。

如果将每个类都做一次二分类，即将该类视为正类，其它所有类视为父类，那么需要对每个文档 20 次二分类才能计算出所有类对应的 F1 值，如何对所有文档只做一次分类就得到所有类的 F1 值呢？

引入混淆矩阵的概念如下，

		预测		
		类1	类2	类3
实际	类1	43	2	0
	类2	5	45	1
	类3	2	3	49

在混淆矩阵中，每一行之和表示该类别的真实样本数量，每一列之和表示被预测为该类别的样本数量。

按照如上图的混淆矩阵，以类别 1 为例计算类别 1 的 F1 值，类别 1 的精确率：43/50，即第一行第一列元素除以第一列所有元素的和；类别 2 的召回率：43/45，即第一行第一列元素除以第一行所有元素之和。再根据这样计算到的精确率与召回率计算出该类对应的 F1 值，对所有类都如此计算，得到 20 个 F1

值，取平均，得到最终的 F 值。

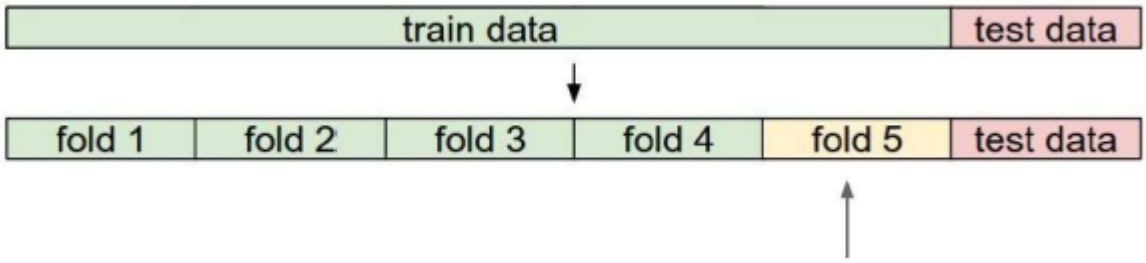
这样对所有文档进行一次分类统计，得到混淆矩阵，就可根据此矩阵计算平均 F 值了。

准确率(accuracy)的概念是在全部预测个体中，有多少个预测准了，注意这不同于精确率的概念，对于每一个类而言，都有一个精确率，而准确率整个分类任务只有一个。

（四）交叉验证

在训练集中划分出验证集，将原始训练集数据分为 k 组（一般是均分），将每个子集数据分别做一次验证集，其余的 k-1 组子集数据作为新训练集，这样会得到 k 个模型，用这 k 个模型的验证集的分类准确率的平均数作为模型的性能指标，进而选择较适合的超参数。

令 k=5，本项目采取这个方法进行交叉验证选取超参数。



B、代码实现

（一）划分数据

由于要进行五重交叉验证，因此我们要将数据随机等分成五份，在 partition 文件夹下创建编号为 0-4 的五个文件夹，每个编号文件夹下都创建 20 个文件夹对应 20 个类，将全部文件依类别随机等分成五份移动到对应文件夹下。

（二）参数估计

1、计算类别的先验概率

统计每一类的文件数目放入 N_c 中，形成 array，然后加总得到文件总数目记为 N，用 N_c array 与 N 作除法，得到类别的先验概率再取对数。

2、得到类别嵌套列表与文件嵌套列表

读取文件，分词，一个文件中的所有词形成一个列表，用 append 方法将此列表添加进 total_df，用 extend 方法将此列表添加到该类的词项列表 l_1 中，读取过该类的全部文件后，将得到的该类的词项列表 l_1 添加到 total 列表中（用 append 方法）。最终得到一个类别嵌套列表 total，total 列表共有 20 个子列表分别对应 20 个类，每个子列表中是该类的所有单词（未去重）。最终也得到一个文件嵌套列表 total_df，total_df 列表中每一个子列表代表一个文件，内含该文件的全部分词。

3、 特征选取

（1）计算每个词的文档频率

遍历 total_df，对于每个文件中的词，用 set 去重，保证重复的词只遍历一次，在字典中添加或更新。用其它方法效率很低。

（2）对频率小于某个阈值或大于某个阈值的词项删除

注意这里不能遍历字典的同时删除字典中的键值对，只有进行这样的遍历，才能同时对字典进行改动：for key in list(df_dic.keys()):

对于阈值的选取见下文描述。

词表 B 的大小为当前 df_dic 的大小。

4、计算词项出现的条件概率

对于每一类的所有词（这个词必须在现在的词表中）统计词在该类中出现的词频，将每个类的这个信息存入字典(词项：词频)，20 个类对应 20 个字典，再将这些字典存入列表。再作计算：

dic_1[key] = np.log((dic_1[key]+k)/(a+b*k))

a 表示词表中的词出现在该类中的词频之和（对于类中的每个词（允许词重复），如果该词出现在了 df_dic 词表中，a 加 1），b 表示词表数目 B。

这里的 k 即为 lambda，表示平滑选取的超参数。

（三）测试

对测试集按类循环，测试每一类的所有文件，对每一文件进行词频统计，存入字典（词项：词频），一共 20 个类，要根据先前存的参数计算该文件属于某一类的概率值。分类器将该文件划分到概率值最高的那一类。如果这一文件的某个词不存在于存储的参数中，使用平滑方法来避免 0 概率问题。还应注意如果该文件中某词多次出现，那么应该多次相加。预测出该文件属于某一类后在结果向量中其对应位置加 1，将 20 个类的结果向量堆叠，最终得到混淆矩阵。

根据混淆矩阵计算每一类的 precision（对角元素除以其所在列）、recall（对角元素除以其所在行）以及 F 值，最后使用宏平均，再计算准确率（accuracy）（对角元素加和除以所有元素之和）。

（四）封装函数

为便于实现循环，将以上代码封装成函数：cal(validation,k,up,down)。

validation：选编号几作为验证集；

k：选几作为平滑参数，一般选 1，即为拉普拉斯平滑；

up：当文档频率大于几时，将其从词表中删除；

down：当文档频率小于几时，将其从词表中删除。

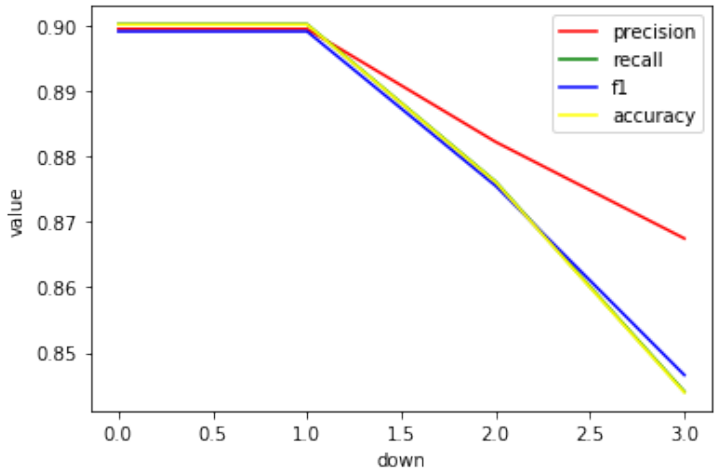
为五重交叉验证封装函数：val(k,up,down)，执行该函数就做了五重交叉验证。

C、交叉验证选取超参数

（一）选取 down

首先固定 k=1 与 up=800，调整 down=0,1,2,3

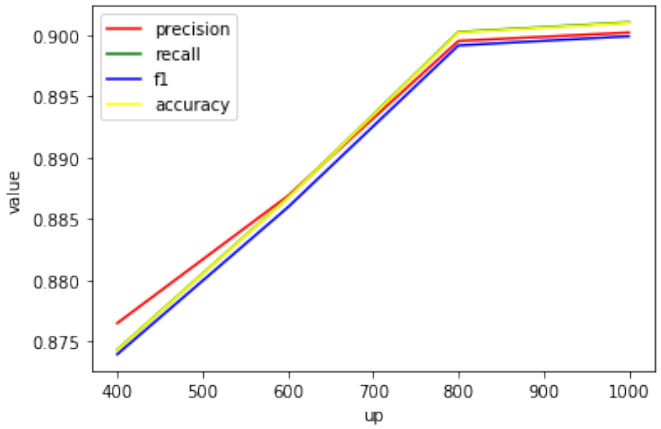
根据指标画图得到下图：



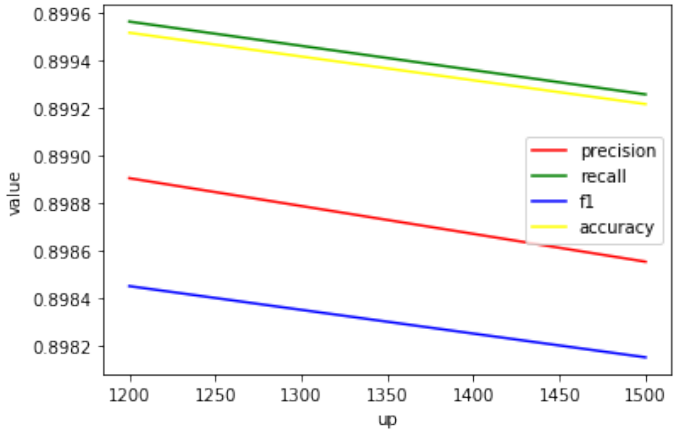
很明显，应该选取 down=0，这里不需要删除 DF 较小的词项。

（二）选取 up

其次固定 k=1 与 down=0 ,调整 up=400,600,800,1000



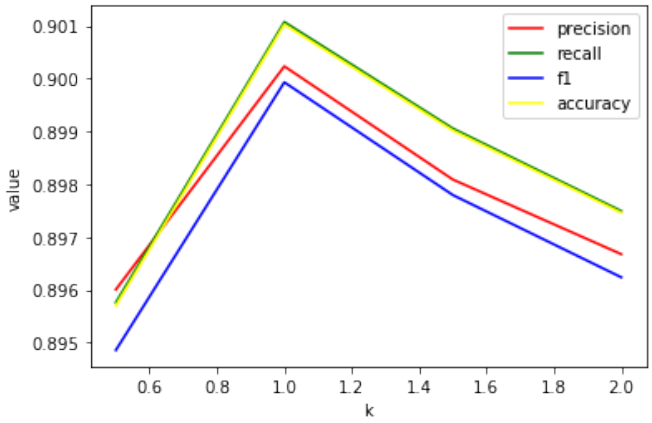
似乎仍然有增长的趋势，故继续调整 up=1200,1500



可以看到指标有所下降，所以选取 up=1000。

（三）选取 k（平滑值）

最后固定 up=1000 与 down= 0，调整 k=0.5，1，1.5，2



很明显选取 k=1

综上所述，应该选择 k=1,up=1000,down=0，也就是将文档频率超过 1000 的词项删除，并且选择拉普拉斯平滑。