实验报告一

学号：201814810
姓名：段新朋

一、实验内容

本实验包括两部分内容：

1、预处理文本数据集，并且得到每个文本的 VSM 表示。

2、 实现 Knn 分类器，测试其在 20Newsgroups 上的效果

二、Vsm

在这个部分，需要对数据集进行预处理，并得到每个文本的 vsm 表示，该过程主要包括以下部分：

1、分词：采用正则表达式以及 python 的 split 函数进行分词，以所有的非字母符号作为分割符号('[^a-z]*'，便可得到分词的结果

2、对单词进行处理：用 nltk，textblob 等包对单词进行以下处理，以减小词典大小：去掉单词长度小于 3 的词，去掉停用词，复数变单数，动词形式变为一般形式，去掉文档中出现频率小于某个阈值的单词。

3、统计单词的 tf，idf 等数据，由于用到了倒排索引的思想加速 knn 的运行，所以需要以单词为 key 值，统计 word_idf,word_doc_tf,doc_word_tf 等三个变量。

4、以某个测试数据文本中出现的单词为基础，通过 word_idf,word_doc_tf,doc_word_tf 三个字典，建立该测试文本，以及所有出现该文本中单词的训练文本的 vsm 表示。

三、KNN

按照上面 vsm 过程中建立的测试文本以及相应训练文本的 vsm 表示，即可按照 knn 的思想为测试文本分类，并得到计算准确率，做的工作主要有：

1、写了两种计算向量相似性的方法：欧几里得距离计算相似性，cos 值计算相似性。

2、采用 inverse index 的方式加速了 knn 的过程。

3、实现了 knn 算法。

四、实验结果

通过本次实验，得到了以下的结论：

1、当数据量巨大的时候，knn 方式进行分类的计算速度非常慢，很长时间都无法得到最终结果。

2、使用 inverse index 可以加速该计算过程。

3、最终得到的准确率是：0.79，

实验报告二

姓名：段新朋
学号：201814810

一、实验要求

使用朴素贝叶斯的方法为实验数据集文档分类。

二、朴素贝叶斯

利用朴素贝叶斯进行文档分类的主要思想是：通过比较某个文档属于哪一类的概率的大小，实现文档分类，假设变量 X 表示文档，变量 Y 表示类别，则文档 x 属于类别 y 的概率为：

$$P(Y = y|X = x)$$

若要求出该概率，需要两个关键点：

1、 **贝叶斯定理**

$$P(Y = y|X = x) = \frac{P(X = x, Y = y)}{P(X = x)}$$
$$= [P(X = x|Y = y) * P(Y = y)]/P(X = x)$$

利用贝叶斯定理，将 P(Y=y|X=x)的计算转换为 P(Y=y)和 P(X=x|Y=y)的计算。

2、**朴素**

朴素的意思是向量 x 的每个分量之间相互独立，于是:

$$P(X = x|Y = y)$$
$$= P(X1 = x1|Y = y) * P(X2 = x2|Y = y) \ldots * P(Xn = xn|Y = y)$$

三、朴素贝叶斯的三种模型

朴素贝叶斯主要有三种模型即，伯努利模型，多项式模型，高斯模型；在本次实验中，实现了伯努利模型和多项式模型；其中，伯努利模型不考虑词频，只考虑每个单词是否出现，多项式模型需要考虑词频；高斯模型针对连续的特征变量。

四、拉普拉斯平滑

拉普拉斯平滑指的是在计算某个类下的某个特征的概率的时候，如果测试数据中出现某个特征在训练数据集中没有出现过，那么，该特征的概率会是 0，又因为计算 P(x|y)的时候是将所有的特征的概率乘起来，最后会导致该概率为 0；拉普拉斯平滑就是这种问题的一个解决方法，也就是把分子分母同时加一个数字，使得每个特征的概率都不为 0。伯努利模型在计算概率的时候，分子加 1，分母加上某个类中单词个数。多项式模型计算时，分子加 1，分母加 2。

五、代码实现

代码中主要包括加载数据，计算 P(y)，计算 P(x|y)，得到预测结果，模型评估等五部分；data_namager.py 主要用于加载数据，naïve_bayes.py，naïve_bayes_multipoly.py 分别是伯努利模型，多项式模型的计算 P(y)，计算 P(x|y)，得到预测结果三部分，metrice.py 主要用于模型评估。

在代码实现的过程中遇到两个问题：

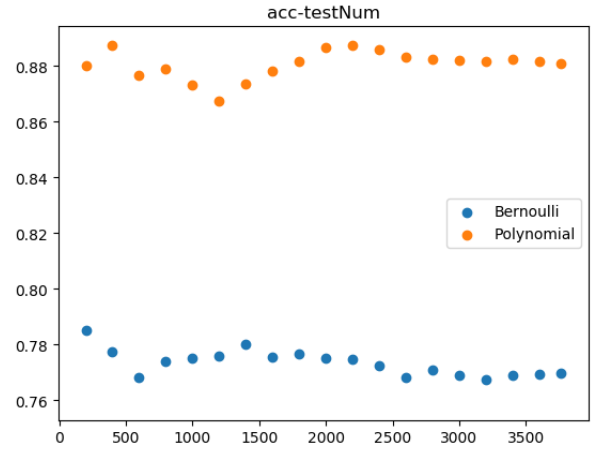
1、 在 python 中，以[[0]]*10 的形式初始化二维数组时，得到的二维数组中的每个一维数组其实是指向的同一个一维数组。

2、 在 python 中概率如何保存的问题；在多项式中由于是每个类中每个单词对应一个概率，所以采用 directory 保存某个类中的所有单词的概率，用 list 保存 20 个类对应的字典。

六、结论

1、伯努利模型和多项式模型的准确率分别为：0.7697822623473181 和 0.8810408921933085。

2、伯努利模型和多项式模型的准确率随着测试文档个数的增多的变化如下图所示：



3、伯努利模型对于每个类的 precision，recall，f1-score 如下图所示：

	precision	recall	f1-score	support
alt.atheism	0.84	0.80	0.82	148
comp.graphics	0.78	0.69	0.74	199
comp.os.ms-windows.misc	0.69	0.74	0.72	195
comp.sys.ibm.pc.hardware	0.71	0.73	0.72	191
comp.sys.mac.hardware	0.54	0.90	0.67	183
comp.windows.x	0.89	0.70	0.79	195
misc.forsale	0.35	0.93	0.51	191
rec.autos	0.91	0.76	0.83	205
rec.motorcycles	0.90	0.93	0.91	222
rec.sport.baseball	0.94	0.83	0.88	209
rec.sport.hockey	0.99	0.89	0.93	193
sci.crypt	0.91	0.80	0.85	191
sci.electronics	0.83	0.67	0.74	221
sci.med	0.95	0.71	0.81	201
sci.space	0.94	0.72	0.81	184
soc.religion.christian	0.91	0.82	0.86	222
talk.politics.guns	0.86	0.80	0.83	158
talk.politics.mideast	0.96	0.73	0.83	194
talk.politics.misc	0.77	0.65	0.70	141
talk.religion.misc	0.88	0.42	0.57	123
avg / total	0.83	0.77	0.78	3766

4、多项式模型对于每个类的 precision，recall，f1-score 如下图所示

	precision	recall	f1-score	support
alt.atheism	0.84	0.93	0.88	148
comp.graphics	0.75	0.82	0.78	199
comp.os.ms-windows.misc	0.94	0.65	0.77	195
comp.sys.ibm.pc.hardware	0.69	0.89	0.74	191
comp.sys.mac.hardware	0.74	0.92	0.82	183
comp.windows.x	0.89	0.84	0.86	195
misc.forsale	0.78	0.84	0.81	191
rec.autos	0.94	0.91	0.93	205
rec.motorcycles	0.94	0.98	0.96	222
rec.sport.baseball	0.96	0.96	0.96	209
rec.sport.hockey	0.97	0.95	0.96	193
sci.crypt	0.95	0.93	0.94	191
sci.electronics	0.85	0.85	0.85	221
sci.med	0.98	0.89	0.93	201
sci.space	0.93	0.93	0.93	184
soc.religion.christian	0.94	0.91	0.92	222
talk.politics.guns	0.87	0.95	0.91	158
talk.politics.mideast	0.98	0.96	0.97	194
talk.politics.misc	0.89	0.86	0.87	141
talk.religion.misc	0.84	0.67	0.75	123
avg / total	0.89	0.88	0.88	3766

七、总结

1、理解了朴素贝叶斯算法，以及其伯努利模式，多项式模式，高斯模式三种模型。

2、理解了在文档分类里，为什么要加拉普拉斯平滑，如何加拉普拉斯平滑；并验证了若不加平滑，对结果的巨大的影响。尤其是在多项式模型中，若不加平滑，准确率只有 0.004，加上平滑之后，准确率可达 0.88。

3、熟悉了 python 中的一些数据结构(directory, generator 等), 更正了对 python 中二维数组初始化的一个误解([[0]*3]*5)。

实验报告三

姓名：段新朋
学号：201814810

一、实验要求

本次实验要求在 tweets 数据集上进行各种聚类算法的实验，一共包括八种聚类方式，然后用 NMI(Normalized Mutual Information)作为评价指标对聚类结果进行评价。

二、数据处理

本实验的数据是用 json 模式存储的，所以在解析的时候，可以用 python 自带的 json 模块进行解析。

三、NMI（Normalized Mutual Information）

熵的定义为：$\mathrm{H(X)} = -\sum_{i=1}^{n} P(x_i) log_b P(x_i)$;

离散变量的互信息定义为：$\mathrm{I(X,Y)} = \sum_{y \in Y} \sum_{x \in X} p(x,y) \log(\frac{p(x,y)}{p(x)p(y)})$,

归一化互信息的定义为：$\mathrm{U(X,Y)} = 2\frac{I(X,Y)}{H(X)+H(Y)}$

四、聚类算法

(一)、K-means

1、算法原理

K-means 算法是硬聚类算法，是典型的基于原型的目标函数聚类方法的代表，它是数据点到原型的某种距离作为优化的目标函数，利用函数求极值的方法得到迭代运算调整规则。

2、算法步骤

a) 随机选择 K 个随机的点，成为聚类质心；
b) 对剩余数据点计算它到每个质心的距离，并把它归到最近的质心的类；
c) 重新计算已经得到的各个类的质心；
d) 重复 a),b),c)直到新的质心与原质心相等或距离小于指定阈值；

3、参数分析

a) K：表示聚类个数，不同的聚类个数会影响聚类效果；

(二)、Affinity Propagation

1、算法原理

AP 算法的基本思想是将全部样本看作网络的节点，然后通过网络中各条边的消息传递计算出个样本的聚类中心。聚类过程中共有两种消息在各节点间传递，分别是吸引度和归属度，直到产生 m 个高质量的 exemplar，同时将剩余的数据点分配到相应的聚类中。

2、算法步骤

a) 计算初始的相似度矩阵，将各点之间的吸引度和归属度初始化为 0；
b) 更新各点之间的吸引度，随之更新各点之间的归属度；
c) 确定当前样本的代表样本点 k；
d) 重复 b),c)，直到所有的样本的所属不再发生变化为止；

3、参数分析

该聚类算法不需要额外设定参数，因为不需要事先指定聚类的数量，而且聚类的结果不会发生变化；

(三)、Spectral Clustering

1、算法原理

谱聚类是一种基于图论的聚类算法，主要思想是把所有的数据看成空间中的点，这些点之间可以用边连接起来。距离较远的两个点之间的边权重值较低，而距离较近的两个点之间的权重值较高，通过对所有数据点组成的图进行切图，让切图后不同的子图间边权重和尽可能的低，而子图内的边权重和尽可能的高，从而达到聚类的目的。

2、算法步骤

a) 对样本构建相似度矩阵 S；
b) 根据相似度矩阵 S 构建邻接矩阵 W、度矩阵 D
c) 计算出拉普拉斯矩阵 L
d) 对拉普拉斯矩阵 L 进行标准化
e) 计算最小的 K 个特征值所对应的特征向量
f) 特征向量标准化，并组成特征矩阵 F
g) 将特征向量按照某种聚类方式聚类

3、参数分析

a) K：聚类个数

(四)、Agglomerative Clustering

1、算法原理

Agglomerative Clustering 是一种自底而上的层次聚类方法，可以在不同的层次上对数据集进行划分，形成树状的聚类结构。

2、算法步骤

a) 将每个样本都作为一个簇；
b) 计算聚类簇之间的距离，找出距离最近的两个簇，将这两个簇合并
c) 重复 b)，直到聚类簇的数量为 K

3、参数分析

a) n_clusters: 整数，代表聚类个数
b) affinity: 一个字符串或者可调用对象，用于计算距离，可以为："euclidean"，"L_1"，"L_2"，"manhattan"，"cosine"，或 'precomputed'。但是作为字符串使用的时候，只有"Euclidean"可用！
c) linkage：一个字符串，用于指定链接算法，有三种方式：
   i. 'ward'：单链接 single-linkage，采用d_{min}；
   ii. 'complete'：全链接 complete-linkage 算法，采用d_{max}；
   iii. 'average'：均连接 average-linkage 算法，采用d_{avg}；

(五)、Mean Shift 聚类算法

1、算法原理

Mean shift 算法是基于核密度估计的爬山算法；简单的说，mean shift 就是沿着密度上升的方向寻找同属一个簇的数据点；具体来说就是要定义一个均值漂移，然后不断迭代这个均值漂移的过程。

a) 均值漂移：给定 d 维空间的 n 个数据点集 X，那么对于空间中的任意点 x 的 mean shift 向量基本形式可以表示为：$M_h = \frac{1}{K}\sum_{x_i \in S_k}(x_i - x)$这个向量就是漂移向量，其中$S_k$表示数据集内到 x 的距离小于圆的半径 h 的数据点。而漂移过程就是利用漂移向量更新球心 x 的位置：$\mathrm{x} = \mathrm{x} + M_h$。

2、聚类流程

a) 在未被标记的数据点中随机选择一个点作为 center；
b) 找出离 center 距离在 bandwidth 之内的所有点，记作集合 M，认为这些点属于族 c，同时把集合 M 内的点属于簇 c 的频率加 1，这个参数将用于最后步骤的分类；
c) 以 center 为中心，计算漂移向量 shift；
d) center= center+ shift，即将 center 按照 shift 进行移动，并将得到的新的 center 作为中心；
e) 重复 b),c),d)，直至收敛，该过程中遇到的所有的点都应被标记为 c；
f) 重复 a),b),c),d),e)直至所有的点都被标记；
g) 将数据点分给被标记频率最高的簇，即完成分类；

3、参数分析

a) bandwidth：浮点数，bandwidth 越小，分类越多；

(六)、DBSCAN

1、算法原理

不同于划分和层次聚类方法，DBSCAN 将密度相连的点的最大集合定义为一个簇，即由密度可达关系导出的最大密度相连的样本集合就可以作为一个簇。

2、算法步骤：

a) 检测数据库中尚未检查过的对象 p，如果 p 未被处理(归为某个簇或者标记为噪声)，则检查其邻域，若包含的对象数不小于 minPts，建立新簇 C，将其中的所有点加入候选集 N；
b) 对候选集 N 中所有尚未被处理的对象 q，检查其邻域，若至少包含 minPts 个对象，则将这些对象加入 N；如果 q 未归入任何一个簇，则将 q 加入 C；
c) 重复步骤 b)，继续检查 N 中未处理的对象，当前候选集 N 为空；
d) 重复步骤 a)~c)，直到所有对象都归入了某个簇或标记为噪声。

3、参数分析

a) Eps: 浮点数，表示两个互为邻居的 samples 之间的最大距离；
b) Min_samples：整数，sample 被视为 core point 的最小邻居数；

(七)、Gaussian_mixture

1、算法原理

利用 E-M 算法，对高斯混合模型进行估计，计算出高斯混合模型中的参数，并利用该高斯混合模型对数据聚类结果进行估计。

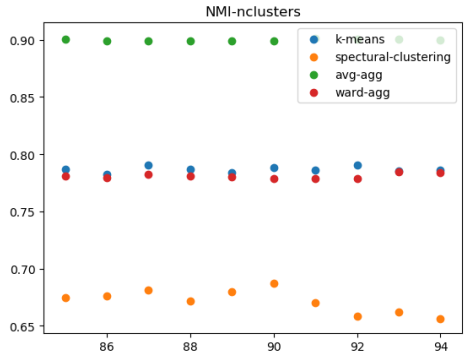
2、参数分析

a) n_components: int 默认值为 1，表示高斯模型的个数

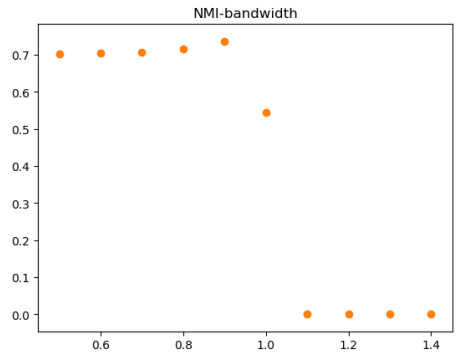
b) covariance_type: string 默认值为'full'，用于描述方差的类型，经过实验，当该参数设为'spherical'的时候，聚类效果最好。

五、聚类效果评估

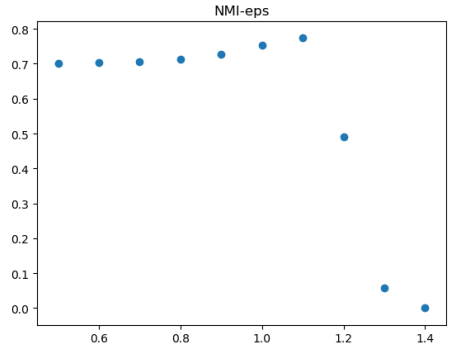
1、k-means、spectral-clustering、avg- Agglomerative、ward- Agglomerative 四种聚类算法的 NMI 随着聚类个数的变化如下图所示：



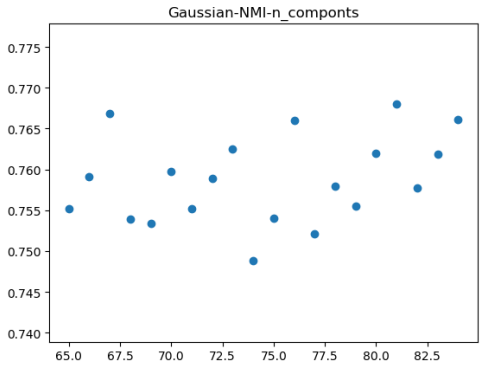
2、Mean-shift 聚类算法的聚类效果随着 bandwidth 的变化如下图所示：



3、DBSCAN 聚类算法的聚类效果随着 eps 的变化情况如下图所示：



4、Gaussian_mixture 模型的 NMI 随着 n_component 的变化如下图所示：



六、结论

本次实验总共实验了七种聚类算法，其中包括划分，层次等多种类型的聚类方式，有些聚类需要设置聚类个数，有些不需要；有些聚类的结果会受到初始值的影响，有些不会；有些聚类则需要设置其他的参数，否则甚至会导致结果完全错误。

通过这次实验，对这七种聚类算法有了大概的了解，熟悉了需要调节的参数的大概范围；了解了对聚类效果的评估手段 NMI。