**EE239AS Project 2**

**Classification Analysis**

**Student1**: Xiangrui Liu (004516858) **Student2**: Qi Wang (604522309)

**Student3**: Qinyi Yan (704406413) **Student4**: Ziyin You (404412651)

**Dataset and Problem Statement:**

*Question.a*

There are two ways to solve this problem:

1) Using the function "fetch\_20newsgroups" in "sklearn.datasets" library;

2) Download the data manually, which are automatically divided into "train" and "test" sets.

We choose the second method, with the help of "**os**" library.

Here we summarize the number of files for all 20 topics, where the files include those in "train" set and those in "test" set. The result is plotted as below:

As shown above, those topics which belong to "Computer Technology" and "Recreational Activity" are highlighted, and it can be seen that they are evenly distributed.

Then, the number of documents in these two groups are calculated below (\*Notice again, all the files in "train" and "test" are included):

|  |  |
| --- | --- |
| Computer Technology | Recreational Activity |
| **3903** | **3979** |

**Modeling Text Data and Feature Extraction:**

*Question.b*

In this question, before creating TFxIDF vector representation, several things should be done:

First, the stop words should be chosen. The method is shown below:

from sklearn.feature\_extraction import text

stop\_words = text.ENGLISH\_STOP\_WORDS

After that, in order to remove the different stems for all the words in the documents, we use:

from nltk.stem.snowball import SnowballStemmer

stemmer = SnowballStemmer("english")

Then, all the documents, after remove the headers, footers and quotes, are used to create the TFxIDF vector. The code is shown below:

from sklearn.datasets import fetch\_20newsgroups

all\_data=fetch\_20newsgroups(subset='all',shuffle=True,random\_state=42, remove=('headers','footers','quotes'))

from sklearn.feature\_extraction.text import TfidfVectorizer

TFxIDF = TfidfVectorizer(analyzer='word', tokenizer=tokenizer\_fun, stop\_words=stop\_words, token\_pattern='[a-zA-Z]{2,}',)

where tokenizer\_fun is a function to keep the roots of the words in the input data.

TFxIDF\_data = TFxIDF.fit\_transform(all\_data.data)

The result is shown to be **72399** terms (within all the 18846 documents in "train" and "test" sets).

*Question.c*

The way to solve Question (c) is just like (b), while in (c), we use one of the four topics:

('comp.sys.ibm.pc.hardware', 'comp.sys.mac.hardware', 'misc.forsale', 'soc.religion.christian')

instead of all the documents. Also, for the " TfidfVectorizer" function, in order to find the 10 most significant terms for each of the four topics, we add the sub-function " max\_features=10".

Then, the results are shown below (notice that the results are the **roots** of the terms):

|  |  |
| --- | --- |
| comp.sys.ibm.pc.hardware | 'scsi', 'mb', 'drive', 'control', 'work', 'use', 'problem', 'ani', 'disk', 'card' |
| comp.sys.mac.hardware | 'drive', 'like', 'know', 'mac', 'work', 'use', 'problem', 'ani', 'appl', 'monitor' |
| misc.forsale | 'sell', 'pleas', 'ship', 'offer', 'price', 'drive', 'use', 'includ', 'sale', 'new' |
| soc.religion.christian | 'say', 'god', 'church', 'christian', 'peopl', 'believ', 'think', 'sin', 'jesus', 'know' |

**Feature Selection:**

*Question.d*

With the LSI feature generated in (b)