

### Assignment 3

Describe the steps you have performed for data preprocessing. Provide the size of your resulting vocabulary.

At first, I split the dataset to train dataset and test dataset just by looking `topic=YES` and `lewisplit=YES`. I stored them in two different dictionaries using `news_id` as key and, body concatenated with title as words and labels for that document as class. While separating them I stored the number of each class in training data. So that after finishing this process I can select top 10 classes. I also calculated each classes probabilities in all training dataset for using it later on calculating naïve bayes.

After this step I preprocess data for naïve bayes implementation. First, I preprocessed training data. For each document I look for their classes. If any there is any class that matches with top 10 class, I add that document to real training data and made stopwords removal as in previous assignments. I made punctuation removals and case folding also. I found resulting vocabulary size 35626. I also stored each word count in training set to use it on naïve bayes. For naïve bayes I stored training data in dictionary by dividing them in top 10 classes. In each class I stored number of occurrences of each word that passes in each class. After storing each word, I calculated each words probability in each class so that I can use this result in naïve bayes. I used log scaled probability to prevent underflow.

After this I preprocessed test set. Since I will be using document in testing, I stored them in the dictionary by their `news_id` as key. In each document I stored each word number of occurrences in that document to use that in both naïve bayes and knn. I removed stopwords, as in previous assignments. I removed punctuations using `string.punctuation`. I also made case folding.

After finishing naïve bayes preprocessing I started to preprocess knn training set. Test set for knn is the same with naïve bayes so I didn't do any preprocess again for test set. In training dataset, I removed stopwords and punctuations. And case folding also made. I stored each words occurrences in each document. These occurrences were used in tfidf calculations. I also stored the squared root of the sum of  $tf*idf$  squared values. So in knn algorithm I am not making this calculations again and again.

Provide information about what the top 10 classes are and how many documents each class has in the training and test sets. What is the total number of documents in the training set and the total number of documents in the test set. How many documents are labeled with more than one of the top 10 classes.

I found top ten classes in preprocessing part and I select my training and test set based on these classes. These classes are [('earn', 2861), ('acq', 1648), ('money-fx', 534), ('grain', 428), ('crude', 385), ('trade', 367), ('interest', 345), ('wheat', 211), ('ship', 191), ('corn', 181)]. They are in sorted order and you can see the numbers that they occur in training dataset. These numbers shows the occurrence of each class in training dataset.

In test dataset I found the results as the following. {'trade': 117, 'grain': 148, 'crude': 186, 'corn': 56, 'ship': 87, 'wheat': 71, 'acq': 718, 'money-fx': 179, 'interest': 131, 'earn': 1080}

As you can see “earn” class is very dominant in the data set.

Total number of documents in the Training set is 6454.

Total number of documents in the Test set is 2532.

In test data there are 212 docs that have multiple top 10 classes.

In training data there are 613 docs that have multiple top 10 classes.

Describe how you performed parameter tuning. Did you use cross-validation or did you use part of your training set as a development set. Please explain and report the best set of parameters that you determined (such as k in kNN) and justify your selection. Please describe what strategy you used to assign a document to more than one class and discuss how successful your strategy is.

I have used part of my training set as development as in each case. In naïve bayes I looked the logarithmic results of each test and determined a threshold in percentage to make this program multilabel result producer. But I found that it does not work on my algorithm, so I removed that part. So, in this code I am returning one label for each test document. In knn, I tried different k numbers to tune my algorithm. I found the best k at k = 3. Precisions and recalls are optimum in this value. In knn earn causes some troubles because frequency of earn is very

Muhammed Göktepe  
2017400162

high compared to others. Increasing k causes all test documents to be labelled as earn. So, some classes results are not satisfying.

Provide your evaluation results on the test set, and provide and discuss your randomization test results.

For naïve bayes:

microAVGPrecision: 0.9518167456556083

microAVGRecall: 0.8694083694083694

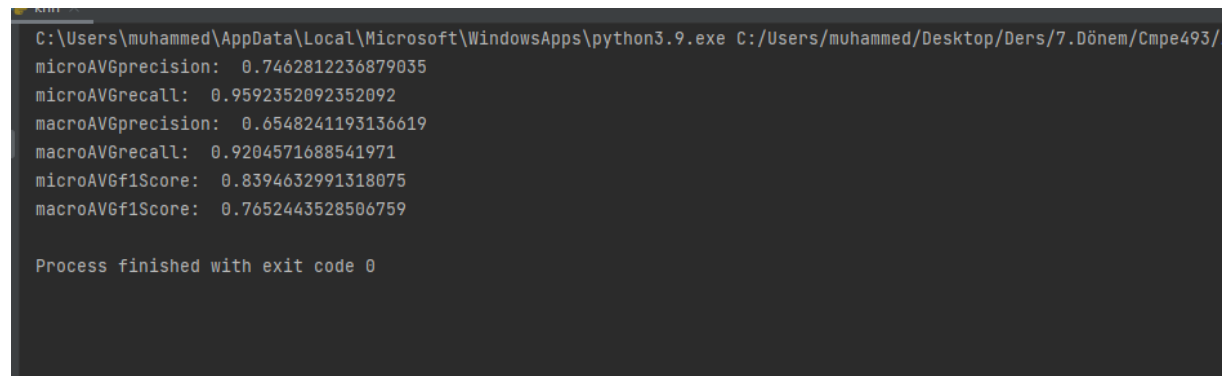
macroAVGPrecision: 0.8200006138147093

macroAVGRecall: 0.6515200797052743

microAVGF1Score: 0.9087481146304677

macroAVGF1Score: 0.726115327665527

For knn:



```
C:\Users\muhammed\AppData\Local\Microsoft\WindowsApps\python3.9.exe C:/Users/muhammed/Desktop/Ders/7.Dönem/Cmpe493/
microAVGPrecision: 0.7462812236879035
microAVGRecall: 0.9592352092352092
macroAVGPrecision: 0.6548241193136619
macroAVGRecall: 0.9204571688541971
microAVGF1Score: 0.8394632991318075
macroAVGF1Score: 0.7652443528506759

Process finished with exit code 0
```

Provide screenshots of running your algorithms.

Before running my program you need to install BeautifulSoup library it must be installed on working environment.

Muhammed Göktepe  
2017400162

```
PS C:\Users\muhammed\Desktop\Ders\7.Dönem\Cmpe493\Assignment1> pip install bs4
WARNING: Ignoring invalid distribution -ip (c:\python310\lib\site-packages)
WARNING: Ignoring invalid distribution - (c:\python310\lib\site-packages)
WARNING: Ignoring invalid distribution -ip (c:\python310\lib\site-packages)
WARNING: Ignoring invalid distribution - (c:\python310\lib\site-packages)
Requirement already satisfied: bs4 in c:\python310\lib\site-packages (0.0.1)
Requirement already satisfied: beautifulsoup4 in c:\python310\lib\site-packages (from bs4) (4.10.0)
Requirement already satisfied: soupsieve>1.2 in c:\python310\lib\site-packages (from beautifulsoup4->bs4) (2.3.1)
WARNING: Ignoring invalid distribution -ip (c:\python310\lib\site-packages)
WARNING: Ignoring invalid distribution - (c:\python310\lib\site-packages)
WARNING: Ignoring invalid distribution -ip (c:\python310\lib\site-packages)
WARNING: Ignoring invalid distribution - (c:\python310\lib\site-packages)
WARNING: Ignoring invalid distribution -ip (c:\python310\lib\site-packages)
WARNING: Ignoring invalid distribution - (c:\python310\lib\site-packages)
WARNING: Ignoring invalid distribution -ip (c:\python310\lib\site-packages)
WARNING: Ignoring invalid distribution - (c:\python310\lib\site-packages)
WARNING: You are using pip version 21.2.3; however, version 21.3.1 is available.
You should consider upgrading via the 'C:\Python310\python.exe -m pip install --upgrade pip' command.
PS C:\Users\muhammed\Desktop\Ders\7.Dönem\Cmpe493\Assignment1> python .\preprocessing.py
```

You can install it by typing “pip install bs4”. Then you can run my preprocessing code by typing “python preprocessing.py”.

```
PS C:\Users\muhammed\Desktop\Ders\7.Dönem\Cmpe493\Assignment3>
PS C:\Users\muhammed\Desktop\Ders\7.Dönem\Cmpe493\Assignment3>
PS C:\Users\muhammed\Desktop\Ders\7.Dönem\Cmpe493\Assignment3>
PS C:\Users\muhammed\Desktop\Ders\7.Dönem\Cmpe493\Assignment3> python3 .\createDict.py
```

With this command you will create necessary dictionaries for knn and naïve bayes. It will dump several json files in the current directory. These files will be used in algorithm runs.

```
PS C:\Users\muhammed\Desktop\Ders\7.Dönem\Cmpe493\Assignment3>
PS C:\Users\muhammed\Desktop\Ders\7.Dönem\Cmpe493\Assignment3>
PS C:\Users\muhammed\Desktop\Ders\7.Dönem\Cmpe493\Assignment3>
PS C:\Users\muhammed\Desktop\Ders\7.Dönem\Cmpe493\Assignment3> python3 .\naive.py
```

With this command you will get the results of the naïve bayes in console and as a json file.

```
PS C:\Users\muhammed\Desktop\Ders\7.Dönem\Cmpe493\Assignment3>
PS C:\Users\muhammed\Desktop\Ders\7.Dönem\Cmpe493\Assignment3>
PS C:\Users\muhammed\Desktop\Ders\7.Dönem\Cmpe493\Assignment3> python3 .\knn.py
```

With this command you will get the results for knn algorithm. This code is little slow. It ends in 5 minutes for all test set.