Cmpe 493 Introduction to Information Retrieval, Fall 2021
Assignment 3 – Text Classification
Zuhal Didem Aytaç – 2018400045

1. Data Preprocessing

I created a file preprocess.py. This code handles the initial processing of data. It reads the reutx-xxx.sgm files and the stopwords.txt file. It extracts the necessary content from the sgm files. Then with the helper functions, it performs case-fold, punctuation removal and stop-word removal operations on the extracted texts. It also removes the numerical values with stop words. It keeps the normalized, tokenized versions of texts.

2. Creating the Train and Test Datasets

Preprocess.py groups articles into train and tests datasets, depending on the LEWISSPLIT value of the article. It extracts the topics for articles within the TOPICS tags. After the preprocessing is completed, it selects the top 10 classes with highest number of documents in total.

From the train and test dataset, it removes the articles that do not belong to any one of these 10 classes. It returns the train and test datasets {new_id:[tokens]}, selected classes {class:[new_id]}, train and test classes {new_id:[classes]}.

It prints the information like vocabulary size, number of train and test documents for each class, train and test dataset sizes, number of train and test documents in multiple classes.

```
Size of vocabulary after preprocessing = 54774
Size of vocabulary in selected documents = 32068

Train Dataset Size = 6494
Test Dataset Size = 2548

SELECTED CLASSES
      earn  train=2877  test=1088
       acq  train=1650  test=719
  money-fx  train=539   test=180
     grain  train=434   test=149
     crude  train=391   test=189
     trade  train=369   test=117
  interest  train=347   test=133
     wheat  train=212   test=71
      ship  train=198   test=89
      corn  train=182   test=56
```

3. Multinomial NB Classifier

I implemented the multinomial NB classifier in nb.py file. It calls the preprocess module to get the train and test datasets, selected classes and train and test labels. The algorithm works in 3 steps: learning, classification and resulting. These components are explained in detail in comments. The algorithm has 1 parameter, range, which decides the range of acceptance. It uses add one smoothing. The algorithm detail can be found in the code comments.

It prints the outputs to nb_result_targets.csv and nb_real_targets.csv files. It prints to each row a new id and 1,0 values, indicating whether that article belongs to that class or not.

4. kNN Classifier

I implemented the KNN classifier in knn.py file. It calls the preprocess module to get the train and test datasets, selected classes and train and test labels. The algorithm works in 3 steps: learning, classification and resulting. These components are explained in detail in comments. The algorithm has 2 parameters, range, which decides the range of acceptance and k, as expected. It uses tf-idf based cosine similarity. The algorithm detail can be found in the code comments.

It prints the outputs to knn_result_targets.csv and knn_real_targets.csv files. It prints to each row a new id and 1,0 values, indicating whether that article belongs to that class or not.

5. Parameter Tuning

I extracted a part of the training set as development set. The NB classifier has one parameter and the KNN classifier has two parameters, as explained in sections 3 and 4. The range parameter is related to the strategy to assign a document in more than one class. In NB, the classifier module returns for each new_id in test dataset, a probability score for each class. The article is assigned to the class with highest probability. It is also assigned to the classes within the predetermined range of the highest value. This parameter defines that value. The same logic also applies to kNN. The kNN classifier module returns for each new_id in test dataset, the count of classes its k nearest neighbors belong to. The article is assigned to the class with highest count. It is also assigned to the classes within the predetermined range of the highest value. This parameter defines that value.

Cross Validation Results for NB (F scores)

```
range=1 macro=0.7397630211127896, micro=0.7905635423309274
range=1.5 macro=0.7574580587299049, micro=0.798890911288509
range=2 macro=0.8108613562881477, micro=0.8247556837242452
range=2.5 macro=0.8138727231507922, micro=0.826840938432595
range=3 macro=0.8180402485306063, micro=0.8274611712636898
range=3.5 macro=0.7817978520791726, micro=0.7928140032494204
range=4 macro=0.7364699040605467, micro=0.7517558400841696
range=4.5 macro=0.7103285161691723, micro=0.7284724842576935
range=5 macro=0.6876418239506772, micro=0.7066518540227597
```

Cross Validation Results for KNN (F scores)

```
k=1 range=30 macro=0.8238761293997328, micro=0.8291026070992534
k=3 range=30 macro=0.8047228435715372, micro=0.825535480945816
k=5 range=30 macro=0.8422260502749495, micro=0.8504712688379231
k=7 range=30 macro=0.8354197591817505, micro=0.8432460020641431
k=9 range=30 macro=0.822101911036353, micro=0.8343913071955361
k=1 range=35 macro=0.8238761293997328, micro=0.8291026070992534
k=3 range=35 macro=0.8364336844245871, micro=0.8414030697903208
k=5 range=35 macro=0.8382934216339567, micro=0.8463588337051025
k=7 range=35 macro=0.8417959655653504, micro=0.849683526186858
k=9 range=35 macro=0.836626496571264, micro=0.8441086703045473
k=1 range=40 macro=0.8238761293997328, micro=0.8291026070992534
k=3 range=40 macro=0.8364336844245871, micro=0.8414030697903208
k=5 range=40 macro=0.8448253854804971, micro=0.8508581449042688
k=7 range=40 macro=0.850011314142202, micro=0.8560679260636995
k=9 range=40 macro=0.8457390542266555, micro=0.8514649597474037
```

For NB, I chose range=2.5, it's not the highest but values decrease fast after 3, so 2.5 seemed safer rather than 3.

For kNN, I chose range=40 and k=7 because it gives the highest values.

6. Evaluation

6.1 NB Classifier

```
CONFUSION MATRIX
earn                                    trade
1074  34                                110  71
14  1426                                7  2360

acq                                     interest
708  51                                 112  27
11  1778                                21  2388

money-fx                                wheat
179  86                                 35  17
1  2282                                 36  2460

grain                                   ship
139  29                                 70  12
10  2370                                19  2447

crude                                   corn
186  63                                 25  11
3  2296                                 31  2481


 PRECISION                              ACCURACY
      earn  0.9693140794223827               earn  0.9811616954474097
       acq  0.932806324110672                 acq  0.9756671899529042
  money-fx  0.6754716981132075          money-fx  0.9658555729984302
     grain  0.8273809523809523             grain  0.9846938775510204
     crude  0.7469879518072289             crude  0.9740973312401884
     trade  0.6077348066298343             trade  0.9693877551020408
  interest  0.8057553956834532          interest  0.9811616954474097
     wheat  0.6730769230769231             wheat  0.9791993720565149
      ship  0.8536585365853658              ship  0.9878335949764521
      corn  0.6944444444444444              corn  0.9835164835164835
 macro avg  0.7786631112254465         macro avg  0.9782574568288854
 micro avg  0.868048700230339          micro avg  0.9782574568288854


 RECALL                                 F1
      earn  0.9871323529411765               earn  0.9781420765027322
       acq  0.9847009735744089                acq  0.9580514208389715
  money-fx  0.9944444444444445          money-fx  0.8044943820224718
     grain  0.9328859060402684             grain  0.8769716088328074
     crude  0.9841269841269841             crude  0.8493150684931506
     trade  0.9401709401709402             trade  0.7382550335570469
  interest  0.8421052631578947          interest  0.8235294117647058
     wheat  0.49295774647887325            wheat  0.5691056910569106
      ship  0.7865168539325843              ship  0.8187134502923977
      corn  0.44642857142857145             corn  0.5434782608695653
 macro avg  0.8391470036296147         macro avg  0.796005640423076
 micro avg  0.9451809387316374         micro avg  0.8180464746659369
```

6.2 kNN Classifier

```
CONFUSION MATRIX
earn                                    trade
1064  263                               108  45
17  1191                                8  2374

acq                                     interest
636  37                                 105  35
82  1780                                28  2367

money-fx                                wheat
171  59                                 51  22
9  2296                                 20  2442

grain                                   ship
134  19                                 75  14
14  2368                                12  2434

crude                                   corn
174  43                                 33  19
12  2306                                23  2460


 PRECISION                              ACCURACY
      earn  0.80180859080633                 earn  0.8895463510848126
       acq  0.9450222882615156                acq  0.9530571992110454
  money-fx  0.7434782608695653           money-fx  0.9731755424063117
     grain  0.8758169934640523              grain  0.98698224852071
     crude  0.8018433179723502              crude  0.9783037475345168
     trade  0.7058823529411765              trade  0.9790927021696253
  interest  0.75                         interest  0.9751479289940829
     wheat  0.6986301369863014              wheat  0.9834319526627219
      ship  0.8426966292134831               ship  0.9897435897435898
      corn  0.6346153846153846               corn  0.9834319526627219
 macro avg  0.7799793955130159          macro avg  0.9691913214990138
 micro avg  0.8210492436433859          micro avg  0.9691913214990138


 RECALL                                 F1
      earn  0.9842738205365402               earn  0.883720930232558
       acq  0.8857938718662952                acq  0.9144500359453631
  money-fx  0.95                         money-fx  0.8341463414634146
     grain  0.9054054054054054              grain  0.8903654485049834
     crude  0.9354838709677419              crude  0.8635235732009925
     trade  0.9310344827586207              trade  0.8029739776951673
  interest  0.7894736842105263           interest  0.7692307692307692
     wheat  0.7183098591549296              wheat  0.7083333333333335
      ship  0.8620689655172413               ship  0.8522727272727273
      corn  0.5892857142857143               corn  0.611111111111111
 macro avg  0.8551129674703015          macro avg  0.813012824799042
 micro avg  0.9189481268011528          micro avg  0.8228588145008803
```

7. Statistical Significance

I implemented a randomization test in randomization_test.py file. It works as follows:

- It calculates the macro F1 scores for the NB and KNN result files.
- For 1000 iterations on the result files of NB and KNN classifiers:
    - It creates new randomized result files for NB and KNN.
    - It traverses the rows and classes in rows
    - For each row-class, it gets the original value from NB and KNN result files.
    - If the random value is greater than 0.5, it prints to the randomized file by swapping the values between them.
    - If not, it leaves them as they are.
    - After all rows are traversed, there are 2 generated randomized result files.
    - It calls the evaluation module.
    - It gets the macro F1 values for randomized NB and KNN files.
    - If the randomized macros have difference is greater than the original macros, it increments the counter.
- It prints the p=(conter+1)/(R+1) value

The null hypothesis suggests that, NB and KNN are not different. That is, the difference |F(A) – F(B)| has occurred by chance.

Suppose we have rejection level 0.05. The printed value is 0.046995300469953004. Then we can reject the null hypothesis that the systems are the same.

8. Screenshots

8.1. NB

```
didemaytac@Zuhal-MacBook-Pro src % python3 nb.py
Size of vocabulary after preprocessing = 54774
Size of vocabulary in selected documents = 32068

Train Dataset Size = 6494
Test Dataset Size = 2548

SELECTED CLASSES
      earn  train=2877  test=1088
       acq  train=1650  test=719
  money-fx  train=539   test=180
     grain  train=434   test=149
     crude  train=391   test=189
     trade  train=369   test=117
  interest  train=347   test=133
     wheat  train=212   test=71
      ship  train=198   test=89
      corn  train=182   test=56
Train Documents in Multiple Classes  = 622
Test Documents in Multiple Classes  = 214
didemaytac@Zuhal-MacBook-Pro src %
```

```
didemaytac@Zuhal-MacBook-Pro src % python3 evaluate_results.py nb
CONFUSION MATRIX

earn
1074  34
14  1426

acq
708  51
11  1778

money-fx
179  86
1  2282

grain
139  29
10  2370

crude
186  63
3  2296

trade
110  71
7  2360

interest
112  27
21  2388

wheat
35  17
36  2460

ship
70  12
19  2447

corn
25  11
31  2481
```

```
PRECISION
      earn  0.9693140794223827
       acq  0.932806324110672
  money-fx  0.6754716981132075
     grain  0.8273809523809523
     crude  0.7469879518072289
     trade  0.6077348066298343
  interest  0.8057553956834532
     wheat  0.6730769230769231
      ship  0.8536585365853658
      corn  0.6944444444444444
 macro avg  0.7786631112254465
 micro avg  0.868048700230339

ACCURACY
      earn  0.9811616954474097
       acq  0.9756671899529042
  money-fx  0.9658555729984302
     grain  0.9846938775510204
     crude  0.9740973312401884
     trade  0.9693877551020408
  interest  0.9811616954474097
     wheat  0.9791993720565149
      ship  0.9878335949764521
      corn  0.9835164835164835
 macro avg  0.9782574568288854
 micro avg  0.9782574568288854

RECALL
      earn  0.9871323529411765
       acq  0.9847009735744089
  money-fx  0.9944444444444445
     grain  0.9328859060402684
     crude  0.9841269841269841
     trade  0.9401709401709402
  interest  0.8421052631578947
     wheat  0.49295774647887325
      ship  0.7865168539325843
      corn  0.44642857142857145
 macro avg  0.8391470036296147
 micro avg  0.9451809387316374

F1
      earn  0.9781420765027322
       acq  0.9580514208389715
  money-fx  0.8044943820224718
     grain  0.8769716088328074
     crude  0.8493150684931506
     trade  0.7382550335570469
  interest  0.8235294117647058
     wheat  0.5691056910569106
      ship  0.8187134502923977
      corn  0.5434782608695653
 macro avg  0.796005640423076
 micro avg  0.8180464746659369
```

8.2. KNN

```
didemaytac@Zuhal-MacBook-Pro src % python3 knn.py
Size of vocabulary after preprocessing = 54774
Size of vocabulary in selected documents = 32068

Train Dataset Size = 6494
Test Dataset Size = 2548

SELECTED CLASSES
      earn  train=2877  test=1088
       acq  train=1650  test=719
  money-fx  train=539   test=180
     grain  train=434   test=149
     crude  train=391   test=189
     trade  train=369   test=117
  interest  train=347   test=133
     wheat  train=212   test=71
      ship  train=198   test=89
      corn  train=182   test=56
Train Documents in Multiple Classes  = 622
Test Documents in Multiple Classes  = 214
```

```
didemaytac@Zuhal-MacBook-Pro src % python3 evaluate_results.py knn
CONFUSION MATRIX

earn
1064  263
17  1191

acq
636  37
82  1780

money-fx
171  59
9  2296

grain
134  19
14  2368

crude
174  43
12  2306

trade
108  45
8  2374

interest
105  35
28  2367

wheat
51  22
20  2442

ship
75  14
12  2434

corn
33  19
23  2460
```

```
PRECISION
     earn  0.80180859080633
      acq  0.9450222882615156
 money-fx  0.7434782608695653
    grain  0.8758169934640523
    crude  0.8018433179723502
    trade  0.7058823529411765
 interest  0.75
    wheat  0.6986301369863014
     ship  0.8426966292134831
     corn  0.6346153846153846
macro avg  0.7799793955130159
micro avg  0.8210492436433859

ACCURACY
     earn  0.8895463510848126
      acq  0.9530571992110454
 money-fx  0.9731755424063117
    grain  0.98698224852071
    crude  0.9783037475345168
    trade  0.9790927021696253
 interest  0.9751479289940829
    wheat  0.9834319526627219
     ship  0.9897435897435898
     corn  0.9834319526627219
macro avg  0.9691913214990138
micro avg  0.9691913214990138

RECALL
     earn  0.9842738205365402
      acq  0.8857938718662952
 money-fx  0.95
    grain  0.9054054054054054
    crude  0.9354838709677419
    trade  0.9310344827586207
 interest  0.7894736842105263
    wheat  0.7183098591549296
     ship  0.8620689655172413
     corn  0.5892857142857143
macro avg  0.8551129674703015
micro avg  0.9189481268011528

F1
     earn  0.883720930232558
      acq  0.9144500359453631
 money-fx  0.8341463414634146
    grain  0.8903654485049834
    crude  0.8635235732009925
    trade  0.8029739776951673
 interest  0.7692307692307692
    wheat  0.7083333333333335
     ship  0.8522727272727273
     corn  0.611111111111111
macro avg  0.813012824799042
micro avg  0.8228588145008803
```