KangHyuk (Chris) Lee

NLP: Text Categorization Report

System Overview

The program was written in Python 3.9 and developed on a macOS operating system. Prior to running the program, the necessary libraries can be installed by downloading them from the requirements.txt file, which is done by typing the following command in the terminal of the directory containing the file: **pip install -r requirements.txt** It should be noted that Python and pip package installer must be installed on the machine prior to downloading the requirements and running the program. In order to run the program, assuming a set of labeled test documents exists, the user can run the script, text_categorization.py, by typing the following command in the terminal of the directory containing the script and the necessary input files: **python3 text_categorization.py**

If there is no set of labeled test documents available, the user can run the script, split_data.py, to randomly split a set of labeled training documents into a smaller training set, labeled validation set, and unlabeled validation set using a chosen test size by typing the following command in the terminal of the directory containing the script and the labeled training documents file: **python3 split_data.py**

The source code and other necessary files for the project can be found on GitHub.

KNN Classifier

The machine learning method used for text categorization was a KNN classifier. The program tokenizes the training and test document texts by using the word_tokenize() function of the nltk library. To be more specific, the "punkt" tokenizer, which is a pre-trained model, is downloaded and used to tokenize the texts. The TF*IDF weighting scheme was used for the tokens. An implementation of the stop list that is provided with the nltk library was attempted in order to remove trivial words from the vocabulary, but the accuracy did not change much (decreased in some cases), so it was removed.

For calculating the similarity scores, both Euclidean distance and cosine similarity methods were attempted. It should be noted that utilizing the vectors as NumPy arrays decreased the computation time by a significant amount. The results are shown and discussed in the section below.

Results

In order to compare the methods of Euclidean distance and cosine similarity, an arbitrary value of K=5 was chosen and both methods were evaluated for each corpus. As shown below, the cos similarity resulted in significantly higher accuracies across the board, so it was chosen as the optimal method. Arbitrarily low and high K values of 1 and 20 were attempted with cos similarity for Corpus 1, but the resulting accuracies were similar, which suggested that K values are insignificant for accuracy compared to the significance of the computational methods.

<u>Corpus 1:</u>

K = 5, Method = Euclidean distance

```
Perl script output: Found 5 categories: Cri Str Oth Dis Pol

209 CORRECT, 234 INCORRECT, RATIO = 0.471783295711061.

CONTINGENCY TABLE:
        Cri     Str     Oth     Dis     Pol     PREC
Cri     50      45      18      60      70      0.21
Str     0       89      1       0       35      0.71
Oth     0       0       3       0       1       0.75
Dis     0       0       0       29      0       1.00
Pol     0       1       3       0       38      0.90
RECALL  1.00    0.66    0.12    0.33    0.26

F_1(Cri) = 0.341296928327645
F_1(Str) = 0.684615384615385
F_1(Oth) = 0.206896551724138
F_1(Dis) = 0.491525423728814
F_1(Pol) = 0.408602150537634
```

K = 5, Method = Cos similarity

```
Perl script output: Found 5 categories: Dis Cri Oth Pol Str

364 CORRECT, 79 INCORRECT, RATIO = 0.821670428893905.

CONTINGENCY TABLE:
        Dis     Cri     Oth     Pol     Str     PREC
Dis     85      2       5       2       6       0.85
Cri     0       38      0       1       0       0.97
Oth     0       0       11      2       0       0.85
Pol     1       4       6       110     9       0.85
Str     3       6       3       29      120     0.75
RECALL  0.96    0.76    0.44    0.76    0.89

F_1(Dis) = 0.899470899470899
F_1(Cri) = 0.853932584269663
F_1(Oth) = 0.578947368421053
F_1(Pol) = 0.802919708029197
F_1(Str) = 0.810810810810811
```

K = 1, Method = Cos similarity

```
Perl script output: Found 5 categories: Dis Str Oth Pol Cri

364 CORRECT, 79 INCORRECT, RATIO = 0.821670428893905.

CONTINGENCY TABLE:
        Dis     Str     Oth     Pol     Cri     PREC
Dis     86      6       6       1       7       0.81
Str     2       115     1       25      3       0.79
Oth     1       0       12      3       0       0.75
Pol     0       10      6       114     3       0.86
Cri     0       4       0       1       37      0.88
RECALL  0.97    0.85    0.48    0.79    0.74

F_1(Dis) = 0.882051282051282
F_1(Str) = 0.818505338078292
F_1(Oth) = 0.585365853658537
F_1(Pol) = 0.823104693140794
F_1(Cri) = 0.804347826086957
```

K = 20, Method = Cos similarity

```
Perl script output: Found 5 categories: Oth Dis Str Pol Cri

365 CORRECT, 78 INCORRECT, RATIO = 0.82392776523702.

CONTINGENCY TABLE:
        Oth     Dis     Str     Pol     Cri     PREC
Oth     13      0       0       2       1       0.81
Dis     4       85      1       0       4       0.90
Str     2       3       120     30      6       0.75
Pol     5       1       13      111     3       0.83
Cri     1       0       1       1       36      0.92
RECALL  0.52    0.96    0.89    0.77    0.72

F_1(Oth) = 0.634146341463415
F_1(Dis) = 0.92896174863388
F_1(Str) = 0.810810810810811
F_1(Pol) = 0.8014440433213
F_1(Cri) = 0.808988764044944
```

Corpus 2:

For corpus 2, a 0.2 test size was chosen and the following files were created for evaluation: corpus2_train_subset.labels, corpus2_validation.list, corpus2_validation.labels

K = 5, Method = Euclidean distance

```
Found 2 categories: O I
Processing prediction file...

141 CORRECT, 38 INCORRECT, RATIO = 0.787709497206704.

CONTINGENCY TABLE:
        O       I       PREC
O       118     25      0.83
I       13      23      0.64
RECALL  0.90    0.48

F_1(O) = 0.861313868613139
F_1(I) = 0.547619047619048
```

K = 5, Method = Cos similarity

```
Perl script output: Found 2 categories: O I

147 CORRECT, 32 INCORRECT, RATIO = 0.82122905027933.

CONTINGENCY TABLE:
        O       I       PREC
O       114     15      0.88
I       17      33      0.66
RECALL  0.87    0.69

F_1(O) = 0.876923076923077
F_1(I) = 0.673469387755102
```

Corpus 3:

For corpus 3, a 0.2 test size was chosen and the following files were created for evaluation: corpus3_train_subset.labels, corpus3_validation.list, corpus3_validation.labels

K = 5, Method = Euclidean distance

```
Perl script output: Found 6 categories: Fin USN Wor Spo Ent Sci

98 CORRECT, 93 INCORRECT, RATIO = 0.513089005235602.

CONTINGENCY TABLE:
        Fin     USN     Wor     Spo     Ent     Sci     PREC
Fin     6       1       0       0       0       0       0.86
USN     1       20      2       0       0       0       0.87
Wor     16      26      68      19      8       20      0.43
Spo     0       0       0       0       0       0       0.00
Ent     0       0       0       0       0       0       0.00
Sci     0       0       0       0       0       4       1.00
RECALL  0.26    0.43    0.97    0.00    0.00    0.17

F_1(Fin) = 0.4
F_1(USN) = 0.571428571428571
F_1(Wor) = 0.599118942731278
F_1(Spo) = 0
F_1(Ent) = 0
F_1(Sci) = 0.285714285714286
```

K = 5, Method = Cos similarity

```
Perl script output: Found 6 categories: Ent Spo USN Sci Wor Fin

169 CORRECT, 22 INCORRECT, RATIO = 0.884816753926702.

CONTINGENCY TABLE:
        Ent     Spo     USN     Sci     Wor     Fin     PREC
Ent     5       0       0       0       0       1       0.83
Spo     0       19      0       0       0       0       1.00
USN     0       0       43      2       3       5       0.81
Sci     2       0       1       19      1       0       0.83
Wor     1       0       2       3       66      0       0.92
Fin     0       0       1       0       0       17      0.94
RECALL  0.62    1.00    0.91    0.79    0.94    0.74

F_1(Ent) = 0.714285714285714
F_1(Spo) = 1
F_1(USN) = 0.86
F_1(Sci) = 0.808510638297872
F_1(Wor) = 0.929577464788732
F_1(Fin) = 0.829268292682927
```

Future work & Recommendations

To determine an optimal K value, either cross validation methods or a for loop of various K values can be evaluated and the resulting accuracy values can be compared.