CMPE 255-02-Spring 2018
Program1

Submitted to:
Prof. Gheorghi Guzun

Submitted by:
Vinayak Nigam
(011822213)

Goal:

To develop a predictive classification model that can determine, given a medical abstract, which of the 5 classes it falls in.

Dataset:

The training dataset consists of 14442 records and the test dataset consists of 14438 records. The data are provided as text in train.dat and test.dat, which should be processed appropriately.

- train.dat: Training set (class label, followed by a tab separating character and the text of the medical abstract).
- test.dat: Testing set (text of medical abstracts in lines, no class label provided).
- format.dat: A sample submission with 14438 entries randomly chosen to be 1 to 5.

Approach:

1. Data Preprocessing steps included cleansing the dataset using regular expression to remove all the special character from the dataset.
2. Filtering out words whose length is below 5 characters. Such words can be neglected as they fail to retain any meaningful inference.
3. Now carry out the above two steps for both train and test dataset and merger both the lists and then create sparse matrix.
4. Creating the CSR matrix followed by splitting it into two halve so that we get two sparse matrices with same dimensions.
5. We calculate the Cosine Similarity for both test and train data set we will create a cosine matrix which has similarity scores

6. For each row in the cosine matrix we will find the k similar value and store that index in a list using the KNN model.
7. Now for each index we check if the train index values belong to the one of the five classifiers (1,2,3,4,5) and then count them as well.
8. Then we take the max value for the count and write the file format with.

Methodology:

The methodology followed for this program was to clean the train and test data.
Use of regular expression was done to remove all the words with length less than 5.
Next step was to combine the training and test dataset and then generating a CSR matrix so that we get two matrices with same dimensions. Calculating similarity using Cosine Similarity function since its more accurate and fast to compute than Euclidean distance similarity function. After cosine similarity it will have 28k x 28k matrix.
Now using numpy arg partition we will sort each row of matrix to get the top K-Nearest Neighbors we will find from the similarity matrix each disease type classifier count and then based on the max value obtained we will write it in our prediction file.

Accuracy Achieved:

71.34