ASSIGNMENT-1 REPORT

Name: Hasmitha Bhutham
GNumber: G01205552
Username on Miner: acchickens
Rank :226, Score on Miner: 0.81

PROGRAM IMPLEMENTATION:

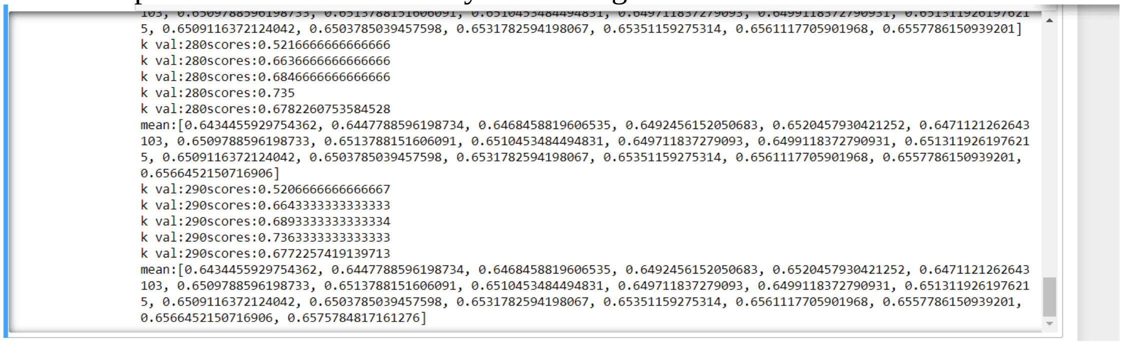
Main steps include:

1. Preprocessing the data
2. TFIDF Vectorizing
3. Cosine Similarities
4. Using KFold to determine better accuracy for a k value.
5. Prediction using KNN

1. **Preprocessing the data:**
   - The function used to preprocess the the cleaning_data().
   - Important packages used here are nltk, BeautifulSoup, sklearn.
   - Stopwords.words(‘english’) is used to remove stop words from the text.
   - WordNetLemmatizer is used to cut down the stems of the words.
2. **TFIDF Vectorizing:**
   - from sklearn.feature_extraction.text import TfidfTransformer package, the train and test reviews are vectorized using TFDIFVectorizer with l2 normalization. The train data and the test data are transformed and returned.
   - The purpose of this is to turn the text data into a vectorized or numeric data.
   - This is used to shrink the text by lowering the terms that have high frequency.
3. **Cosine Similarities:**
   - The cosine similarity between TRAIN_TFIDF and TEST_TFIDF is given by using np.dot() which gives the dot product of both the arrays.
   - The result is given as a NumPy array.

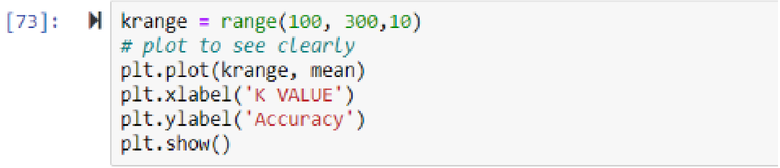
4. **K Fold Cross-Validation:**
   - K fold is used for determining better accuracy for a given K value.
   - sklearn.model_selection package is used for this.
   - The clean train data is split into 5 splits and a range(100,300) is given with an increment.(10)
   - Accuracy is determined by the accuracy() function which takes the TRAIN_TFIDF and TEST_TFIDF data and computes the cosine similarity between them.
   - Using the predictionCount() function, the positive and negative sentiments are appended by 1 and if positive > negative 1 is returned and -1 is returned otherwise.

- After splitting the clean data into X_train, X_test and y_train and y_test, the accuracy is tested by passing these values into accuracy() with k value.
- Mean is determined by sum of kvalaccuracy/no.of total kvalaccuracy which gives the accuracy for each fold.
- The below picture shows the accuracy and averages for each fold.

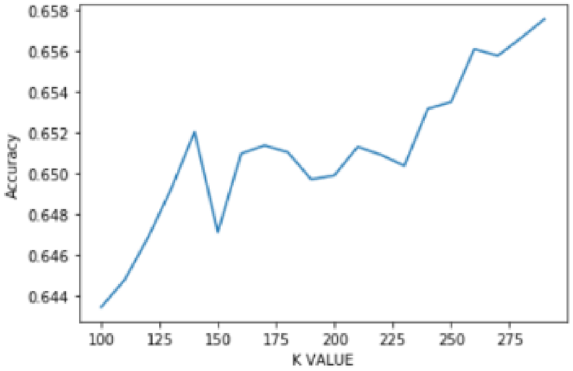


```
103, 0.6509788596198733, 0.6513788151606091, 0.6510453484494831, 0.649711837279093, 0.6499118372790931, 0.651311926197621
5, 0.6509116372124042, 0.6503785039457598, 0.6531782594198067, 0.65351159275314, 0.6561117705901968, 0.6557786150939201]
k val:280scores:0.5216666666666666
k val:280scores:0.6636666666666666
k val:280scores:0.6846666666666666
k val:280scores:0.735
k val:280scores:0.6782260753584528
mean:[0.6434455929754362, 0.6447788596198734, 0.6468458819606535, 0.6492456152050683, 0.6520457930421252, 0.6471121262643
103, 0.6509788596198733, 0.6513788151606091, 0.6510453484494831, 0.649711837279093, 0.6499118372790931, 0.651311926197621
5, 0.6509116372124042, 0.6503785039457598, 0.6531782594198067, 0.65351159275314, 0.6561117705901968, 0.6557786150939201,
0.6566452150716906]
k val:290scores:0.5206666666666667
k val:290scores:0.6643333333333333
k val:290scores:0.6893333333333334
k val:290scores:0.7363333333333333
k val:290scores:0.6772257419139713
mean:[0.6434455929754362, 0.6447788596198734, 0.6468458819606535, 0.6492456152050683, 0.6520457930421252, 0.6471121262643
103, 0.6509788596198733, 0.6513788151606091, 0.6510453484494831, 0.649711837279093, 0.6499118372790931, 0.651311926197621
5, 0.6509116372124042, 0.6503785039457598, 0.6531782594198067, 0.65351159275314, 0.6561117705901968, 0.6557786150939201,
0.6566452150716906, 0.6575784817161276]
```

- The below picture shows the graph plotted against Accuracy and K value.



```
[73]:  krange = range(100, 300,10)
       # plot to see clearly
       plt.plot(krange, mean)
       plt.xlabel('K VALUE')
       plt.ylabel('Accuracy')
       plt.show()
```



- High accuracy is for `k val:260scores:0.7386666666666667`

5. KNN Classification:

- K nearest neighbors is found using np.argsort() using a k value = 280
- With the cosine similarity, the k nearest neighbors can be determined for each similarity.
- Then the predicted sentiment is appended by ‘1’ and ‘-1’ for positive and negative sentiment respectively.
- A result file is written in the folder with the result sentiments.