LOGISTIC REGREESION IMPLEMENTATION – REPORT

Language Used: Python

Software Requirement: Google Colab Notebook

Steps in implementation:

1. Import all the necessary libraries (pandas, NumPy, nltk, sklearn) and modules.
2. Load the training and test datasets.
    - Since the files are of '.dat' format use "read_csv( )" and "read_fwf( )" functions to read the data.
    - The training data has 2 colums separated by a tab (\t) which is given as the separator. The two columns are named "sentiment" and "review".

    Note: The training data is renamed as "Training_data.dat" and test data is renamed as "Test_data.dat" for convenience.
3. The input data may contain rows/columns with null values. These values are sentiments and cannot be filled. Hence, these are dropped from the training data set using the dropna( ) function.
4. Data Pre-processing:
    - **Stop-words:** There are a set of predefined words that are considered neutral i.e., neither positive nor negative. First, we get those words into "stop" remove the punctuations and then compare the training dataset with these stop words to eliminate them.
    - Also, convert the reviews to lower case since the stop words are all in lower case.
    - **Stemming:** The words might have affixes. We apply stemming to convert these words to the root form. For this purpose we use the PorterStemmer( ) function. We go through each word using the for loop and stem them.
5. **Term frequency – inverse document frequency:** Here the data is transformed from text to numeric vectors. This is used for feature extraction from the training data. This will be useful to train the logistic regression model.
6. **Training the logical regression model:** The training data file has been split into test and training sets to train the model and test the accuracy. The LogisticRegressionCV function is used to create the model. After training the accuracy of the model is checked with both training and test set.
7. **Predicting the sentiment for unknown data:** The training data is used to predict the output for the test data. This prediction vector is stored as .csv file.

Miner Username: ds18

Rank: 143

Accuracy score: 0.87

Methodology:

Null values, punctuations, stop-words and affixes are filtered to remove the unwanted data as part of pre-processing. Tf-idf is preferred over countvectorizer because this calculates the normalized count which is word count divided by no. of documents it appeared. If a specific word appears in one document repeatedly and less in other documents normalized frequency produces accurate results rather than countVecorizer.