584 HW1

Name : Sreeram Bangaru
Name on Miner : sreerambangaru
Accuracy : 79%
Rank : 172

Given two text data sets i.e, train dataset and test dataset, our aim is to create a machine learning model using logistic regression which classifies the sentiment of text data as either '+1' or '-1' with a good level of accuracy. For this I have used the programming language of **Python** and the open source IDE **Jupyter Notebook**.

Observing the data, it is identified that the data can be purified into a simpler form by using multiple pre-processing steps so that it is easy for the model to classify properly and reduce the runtime. For pre-processing of text the steps followed are:

- First I have removed all the punctuation marks from the text data as these do not help us in any way to predict a sentiment. The '**string**' package of python is used for this process.
- Next the text has been converted to lower case. This is done because when the machine learning model classifies the data then it might classify the same sentence wrongly if it is in upper case for one record and lower case for another record. This decreases the accuracy of the model.
- Next I have broken all the text records into tokens because tokenization helps us in splitting the paragraphs into smaller pieces which makes it easy to add further meaning to the records. For this the python package of '**re**' is used.
- To further purify the text data stop words are removed from the datasets. Stop words are a collection of articles, pronouns, prepositions, etc, which are removed from the text data before doing natural language processing because they do not add up to much information during classification. The '**nltk**' package of python is used to filter out the english stopwords.
- As the final step of pre-processing of text data, the technique of **lemmatization** is used. Lemmatization basically groups words of text without removing the meaning from it. Lemmatization is considered instead of **stemming** because stemming also groups the text but without a meaningful intent. The '**WordNetLemmatizer**' is used which is a part of nltk to help us break down the text into smaller groups with a meaning.

Pre-processing of text data is done and text has been purified, our next objective is to train a model on this data. The '**ngrams**' model is used to capture a pattern of words and then transform it into vectors and use these vectors in our logistic regression model. Also to convert the data into vectors we have used '**CounterVectorizer**' on ngrams where I have considered '**N=1 and N=2**' for a pattern of single words and two adjacent words.Out of the two basic approaches to classify text I have considered ngrams on '**bag of words**' because the latter classifies every word as a separate feature rather than giving us the flexibility to classify single words or a combination of words.

Now to check the accuracy of the train set I have used '**train test split**' where I have considered 80% train data and 20% test data. As the resultant accuracy is 86% it is a good fit. Now we have used the whole train data as input against the whole test data file in the logistic regression model. When tested this model against the test data of '**miner**' the accuracy was 72%. So to try a different approach, in the ngrams model I have considered only '**N=2**'. After this change there is a rapid increase in the accuracy to 79%.

To further increase the accuracy, the train and test data this time is used without any stemming or lemmatization. But the trained model on this data did not increase the accuracy of the model.

Referenced List :

https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html

https://www.analyticsvidhya.com/blog/2021/08/a-friendly-guide-to-nlp-bag-of-words-with-python-example/

https://practicaldatascience.co.uk/machine-learning/how-to-use-count-vectorization-for-n-gram-analysis