Logistic Regression Text Classifier (CE314 assignment 2 report)

Student ID: 1801355

Reading data set:

Reading the data set using pandas library, also shuffling data set so it is less bias.

Pre-processing:

After reading the dataset with pandas, I get the data frame['review'] series and convert all the words into lower case, then removing all the English stop words, removing any non-alpha characters, and finally removing html tags since I noticed there are some of them as well. At least on my machine, removing English stop words and lowercase it takes around 20 seconds, only removing the html tags takes approximately 60 seconds, while removing all non-alpha characters takes most time, sometimes more than 3 minutes. All of this helps the classifier avoid any unnecessary computation or dealing with useless words and characters, it also splits train and test sets faster.

Feature extraction:

For feature extraction, I import "train_test_split" module from sklearn model selection library which takes the data frame["review"] and data frame["sentiment"] and splits them into 80% training feature set e.g. the first 4000 rows and 20% testing set e.g. last 1000 rows. After this I take the review testing and training sets and calculate TF-IDF using sklearn's TfidfVectorizer module.

Classifier method / model:

Classifier is using Logistic Regression model from sklearn library, I chose it because it is fast and gives high accuracy in a simple dataset, also it is easy to implement and it does the job required.

We are fitting the term-frequency review train data and sentiment train data into the classifier, afterwards the classifier makes a sentiment prediction based on term-frequency review test data that we created during feature extraction.

Once the sentiment prediction is generated we are using classification report from sklearn metrics module, which compares sentiment test data with the sentiment prediction.

From the report we get:

Accuracy around **90%**
Precision and **recall** between **89 – 91%** for either negative or positive.
F1 score around **90%** for either negative or positive.
Macro average around **90%**
Weighted average around **90%**

Improve Logistic Regression Classifier:

1. Most obvious that comes to mind is to analyse your errors, maybe the model works better on some parts and worse on others.
2. Adding Hyper parameter tuning - Grid Search and add cross validations.
3. Make sure there is a class balance, especially when we are dealing with reject / admit data.
4. Simply add more data.
5. Experimenting with your model/features/variables/etc. is probably the most correct answer since this is what machine learning is all about.