ML Project Activity Log

IMT2020548 Tejas Sharma
IMT2020549 Dewanshi Dewan

Week 1 and 2: 5th October - 24th October

Tasks performed -

Basic understanding of the problem statement given. Looked at youtube videos and read resources online to understand natural language processing. Opened the dataset on google colab to perform initial analysis.
The text data from the train.csv file is extracted and passed into the summarizer. The summarizer performs the main pre processing tasks mentioned above. The text that is returned from the preprocessed text is now ready to be fed into the model, to train the model.

- Tokenizing comment. (A sentence is broken down into words and each word is made a token)
- Removing punctuation marks. Each token is checked. If the token is a punctuation mark then it is removed.
- Removing stopwords. (Stopwords from the English language have been removed from the token. The stopword “you” has been removed from the set of stopwords for the sake of the given task)
- Lemmatizing the tokens: (Each token is reduced to its lemma by first tagging each word to the parts of speech (POS Tags) and then lemmatizing it.)
- We also removed words of length less than 2.
- We converted all words into lower cases.
- We removed the url tags from the comments.

A train-test split is performed (80-20) on the train data for testing.

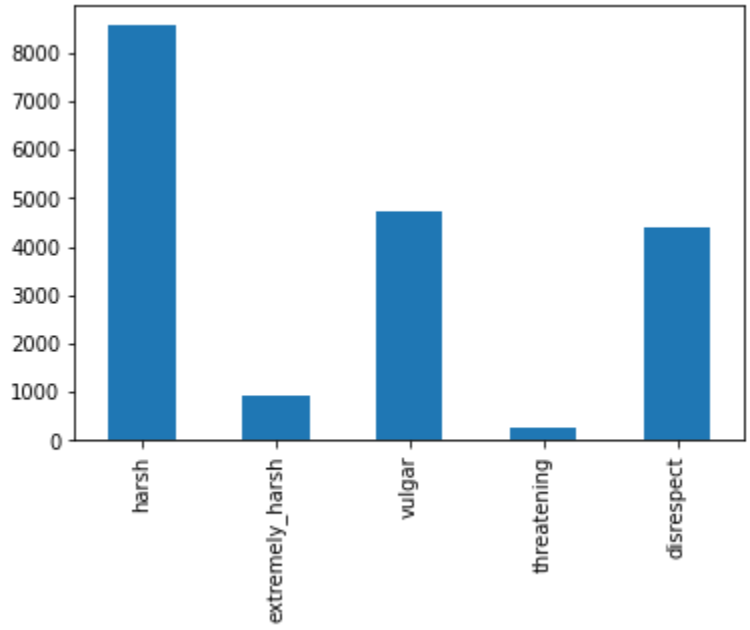
For training the model we use the following approach:

1. We create a vector representing the occurrence of each word in the corpus. It measures the number of times a particular word appears in a document.
2. Just the count of the occurrences of the words is not enough as words which are common to all comments will be given a higher score even when they might not contribute as much to the “meaning” of the text. Thus, we use a TF-IDF (Term Frequency - Inverse Document Frequency) method.
   Now we have a matrix representation of each word and its score for each document.

3. We use the OneVsRest Classifier which is used for multi class classification and modify it for multilabel classification. For our dataset, the OVR classifier makes a multiple classifiers like [harsh, rest], [extremely_harsh, rest], [vulgar, rest].. For each column. Then for each classifier, we have applied logistic regression and multinomial naive bayes to predict [harsh, rest] , [extremely_harsh, rest]..
4. We have also used an ensemble for the OVR with LR and OVR with multinomial naive bayes.
5. After training the model, we fit the testing data and evaluate using the roc_auc score. Looking at the values, we decided that its best to stick to OVR with logistic regression as it gave a better roc_auc_score.

Initial data analysis:

From the initial analysis of the dataset, we observe that there are 89359 rows. Each row contains a unique ID, a comment and the categories of ‘harsh’, ‘extremely harsh’, ‘vulgar’, ‘threatening’, ‘disrespect’, ‘targeted_hate’. For each category in a row a 1 marks that the comment has a flag for that particular category whereas a 0 means it doesn't.
Further analysis says that there are no null values in the dataset. Neither are there duplicate rows.
The number of comments which are flagged for each category is given by a histogram plot which is as follows:



Clearly, the number of harsh comments are about 10% with threatening comments being the least.

Results and Conclusion:

The output scores for each column of 'output' indicates that our implementation of the OVR classifier with multinomial naive bayes does not give us the best result. The ROC_AUC score is the best for accuracy; being 0.52 which is suggests that the model does not do a good job in discriminating.
The same OVR classifier with Logistic regression gave us a score of 0.68 which meant that it was a better discriminator.
The ensemble also did not give a better result for each column. Thus we decided to use the OVR with LR for the Kaggle submission.

References used:

Youtube video to understand NLP:
https://www.youtube.com/watch?v=5ctbvkAMQO4
Basic pre processing tasks that are performed in NLP:
https://www.kdnuggets.com/2019/04/text-preprocessing-nlp-machine-learning.html
To understand the use case of a wordnet:
https://www.youtube.com/watch?v=5wpm4Yvbop8
To convert the parts of speech tag to wordnet tag:
https://stackoverflow.com/questions/35458896/python-map-nltk-stanford-pos-tags-to-wordnet-pos-tags
Syntax to implement preprocessing using the natural language toolkit (NLTK) was found by googling for the task.
https://www.youtube.com/watch?v=ZvaELFv5IpM
To understand OneVsRest classifier.
https://en.wikipedia.org/wiki/Multi-label_classification
To understand the different approaches to solve multi label classification. (We used problem transformation)

Tasks planned for the coming weeks -

Work on the model to improve the scores. Try to implement the same task using neural networks.