Project Proposal for Deep Learning

Project Report by - Anshita Singh Bais-1222426855
Hitaxi Harshadkumar Mistry- 1222126334

Task - Irony Detection in English Tweets

Research paper referred to -
Unified Benchmark and Comparative Evaluation for Tweet Classification by Cynthia Van Hee, Els Lefever and Ve´ronique Hoste LT3 Language and Translation Technology Team Ghent University Groot-Brittannie¨laan 45, 9000 Ghent

Link for the research paper - https://arxiv.org/pdf/2010.12421.pdf

Definition of the task - Given a tweet, predict whether the tweet includes irony intents or not. The dataset consists of 2 fields - the label and the text. The label is either 0 or 1 denoting not irony and irony respectively and the text consists of the tweets with emojis converted to UTF-8 encoding. The input of the dataset will be the tweet and the output will be either 1 or 0 based on whether the tweet has irony or not.

Dataset Description - We use the Subtask A dataset of the SemEval2018 Irony Detection challenge (Van Hee et al., 2018). Note that this dataset was artificially balanced to make the task more accessible. The Subtask A dataset consists of tweets which have no emoji. They replaced emoji by UTF8 descriptions and removed irony-related hashtags as these hashtags will not be present in the test set.

Example of the Dataset -

Label	**Text**
1	Planned on an early night last night. Oh yaa course that happened.. #gotinterupted
0	2014 can't end fast enough! Stupid dead battery.
1	Doesn't lucky and fortunate mean the same thing?

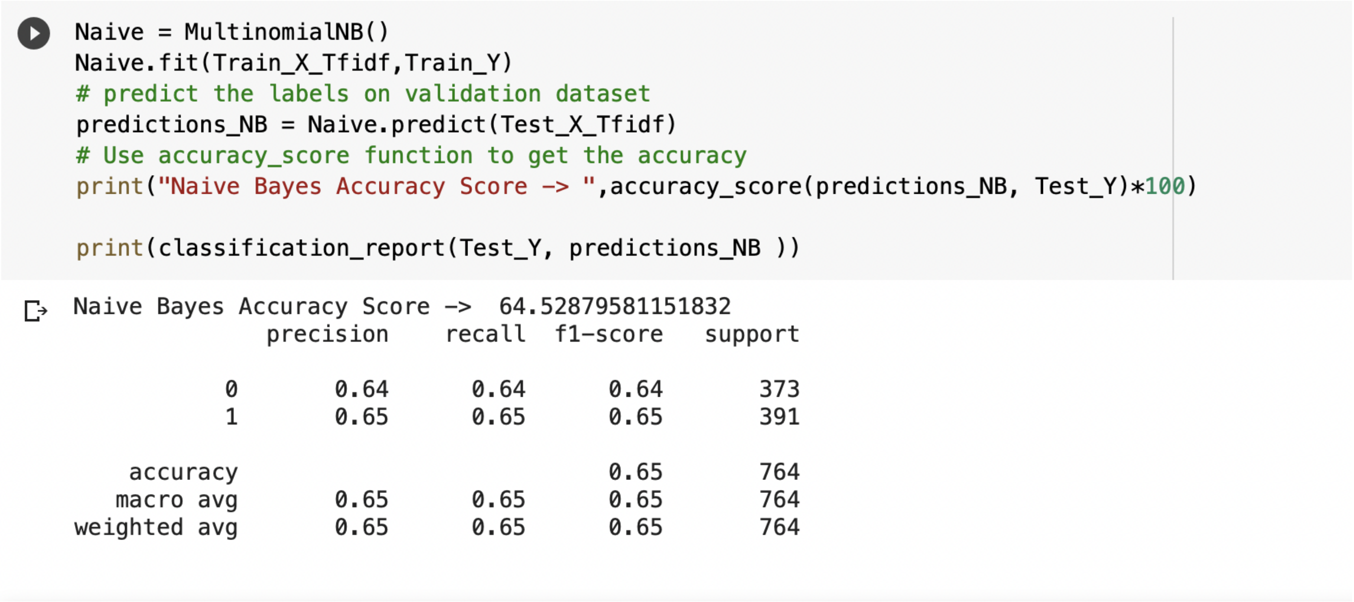
Dataset Link -
https://github.com/Cyvhee/SemEval2018-Task3/blob/master/datasets/train/SemEval2018-T3-train-taskB.txt

Baseline - The model was trained using Naive Bayes and the accuracy received in classifying the labels was 64.53%. The label 0 represents that the tweet wasn't ironic, while the value 1 represents otherwise.

Evaluation Matrix - The evaluation matrix that we are using is the classification report that gives an insight to the recall rate, f1-score and precision of the model. The value is around 64-65%, along with the accuracy score.

Link of the Colab Notebook -
https://colab.research.google.com/drive/1GouvwHOXIC4Wu7_5VGNAnrf2C5LNiL3i?usp=sharing

Baseline Screenshot -



```
Naive = MultinomialNB()
Naive.fit(Train_X_Tfidf,Train_Y)
# predict the labels on validation dataset
predictions_NB = Naive.predict(Test_X_Tfidf)
# Use accuracy_score function to get the accuracy
print("Naive Bayes Accuracy Score -> ",accuracy_score(predictions_NB, Test_Y)*100)

print(classification_report(Test_Y, predictions_NB ))
```

```
Naive Bayes Accuracy Score ->  64.52879581151832
              precision    recall  f1-score   support

           0       0.64      0.64      0.64       373
           1       0.65      0.65      0.65       391

    accuracy                           0.65       764
   macro avg       0.65      0.65      0.65       764
weighted avg       0.65      0.65      0.65       764
```

Future Goal - We will try different transformers and various other models to train on the given dataset and expect to improve the accuracy upto 70-75% or more for the model.