**The Analytics Edge Data Competition**

|  |  |
| --- | --- |
| Reuben Tan | 1003462 |
| Karan Nair | 1003603 |
| James Gan | 1003330 |

**Table of Content**

1. Methodology
2. Data Pre-Processing and Exploratory Data Analysis
3. Machine Learning Algorithm
4. Conclusion
5. **Methodology**

2 parts

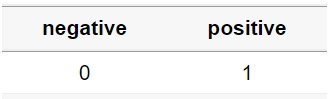
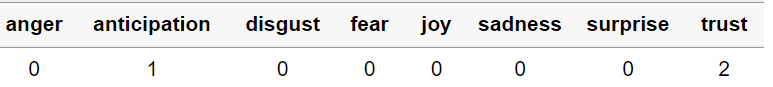
Various iterations of Pre-processing & Modelling

1. **Data Pre-Processing and Exploratory Data Analysis**

Firstly, for data pre-processing, we applied the preprocessing techniques learnt in class, mainly removing numbers, punctuations and making sure all the text was in lower case. Next, we proceeded to remove the top 50 most common words such as @mention and #WEATHER. The reason for this is that these words do not provide much information and may actually contribute to noise in the model, which would negatively affect performance.

However, we figured that emoticons would play a significant role in predicting sentiment because a smiley face implies a positive sentiment while a frowning face implies a negative sentiment. Thus, we decided to include several important emoticons in the document term matrix (DTM) in order to better train our model.

We then observed that our model performed better with emoticons present. To further train our model to understand the different emotions and sentiments, we used the get\_nrc\_sentiment from the library syuzhet. The NRC emotion lexicon contains a list of words and their 8 associated emotions (anger, fear, anticipation, trust, surprise, sadness, joy, and disgust) along with two sentiments (negative and positive). This was quite powerful because our model could train using these emotions instead of just random words in the DTM.



Due to the limitations of stemming, we also tried using lemmatization as we thought our model would train better using complete words as opposed to stemming which cuts off the word. However, our model did not have a significant improvement with lemmatization , thus we continued using stemming as the code ran much faster.

1. **Machine Learning Algorithm**

We tried several algorithms for the model building including algorithms from the course and other algorithm. The process of our model training is finding the model that has the highest accuracy score from the holdout(validation) set.

The models we tried are:

* Naïve Bayes
* K Nearest Neighbor
* CART
* Gradient Boosting Classifier
* Random Forest
* Neural Network
* Linear Discriminant Analysis

The model that was the most suitable for us was Linear Discriminant Analysis with relatively acceptable training time and accuracy.

We first started out using the algorithms that were taught in this course such as Naïve Bayes, K Nearest Neighbor, CART and Random Forest. Of the 4 models that we tried, Random Forest had the best result and compared to the other 3 models which did not have high accuracy scores.

After identifying random forest as our best option, we decided to tune the performance of the random forest model to optimize its performance. We ran a for loop through the measure of the number of trees against the OOB value (Figure 1).

Chart

Description automatically generated

Figure 1 - OOB against the number of trees

Through model tuning of the random forest model, we were able to achieve a satisfactory score of 0.84577. We decided to try out additional model beyond the scope of the course which were Gradient Boosting Classifier, Neural Network and Linear Discriminant Analysis.

The gradient boosting classifier library was not implemented successfully due to depreciated support for the SoftMax classification parameter where warning messages mention of possible inaccurate results. We ran the model anyways and achieve result similar to the random forest model but decided to try other models instead.

Then here we can talk about LDA and NN

1. **Conclusion**