INTRODUCTION

Twitter is a platform which allows users to 'tweet' (or, publicly post) information, thoughts or stories. Boasting more than 300 million users, it is estimated that, on average, there are 200 billion tweets per day (Sayce.D, 2020). The potential information that can be retrieved from these tweets is enormous. This information can be valuable to companies in many ways such as keeping track of reviews on their services or products by filtering out negative and positive reviews on tweets. This report focuses on predicting the sentiment (positive, neutral or negative) of a sample of such tweets through feature engineering and various machine learning algorithms. The machine learning algorithms utilised will also be compared and critically analysed in terms of their performance and accuracy. The training data set contains 3 columns: tweet ID, text/tweet and the sentiment, with thousands of rows of tweets. The testing data set only contains two columns: tweet ID and text/tweet, and the algorithms are used to determine the sentiment of these tweets, after being trained on the training data.

METHOD

Influenced by the study done by Prusa.J and team (see Bibliography), the methodical approach to our problem is as follows. In terms of feature engineering, numerous pre-processing steps were undertaken to select the most reasonable features for determining a tweets' sentiment. Initially, the data was imported into separate training and testing variables. The tweets were then converted into lowercase, and all "stopwords", non-alphabetic characters, punctuation and words of length less than 2 were removed as they would provide no information on the sentiment. Now, all that was left to determine was if URL's, hashtags and tags were useful in predicting the sentiment. Through the use of a 'get_occurrence' function, all three aspects were placed into separate dictionaries and any repeated use of these aspects was observed. In terms of URL, it was determined that the highest occurrence was 6 – meaning 6 tweets out of thousands had used the same URL – a very insignificant number, hence URL's were removed from the feature selection process. Additionally, it was also found that tags and hashtags weren't significant for feature selection (highest occurrence was 79 for tags and hashtags for Neutral sentiment), so we removed tags and hashtags. For further simplicity in analysis, words in tweets were converted to their root form via the porterstemmer() method. This reduces all words to their word stem (e.g. "likes", "liked", "likely" will be "like"). The most frequent words used per sentiment were then visualised using wordcloud() (see Results), and common words between sentiments were removed (as they could skew results). These would then be the basis for our features.

In addition, we did feature selection on the features that were preprocessed by the above. We use filtering to evaluate the "goodness" of each attribute and select 'k' best features. Regarding the choice for numbers of k, we went with 10, 100 and 200 based on preliminary research done by Joseph D. Prusa. He and his team "recommended using Chi-Squared or Mutual Information with 100 to 200 features for subset size to achieve good performance". (Joseph D. Prusa et al., 2015). The filtering methods we tested were Chi-Squared and Mutual Information to compare with the entire feature set (without filtering). By using these methods, we want to find good features that are well correlated (or reverse) with an interesting class. By testing out different filtering methods and selecting different subsets, we would compare and select the most optimal filtering method and size of subset to select.

Before making a decision on our evaluation metrics, we first have to check for imbalances. Our training dataset is unbalanced with approximately 13,000 neutral sentiments and 5,000 positive and negative sentiments. In an attempt to tackle this issue, we will use Stratified K-fold Cross-Validation which preserves the proportions of sample for each class in every iteration. This will help control model bias, variance and tackle the unbalanced dataset bias by ensuring the proportion of each class "found in the original distribution is respected in all the folds" (He.H et al., 2013).

In terms of training models using Machine Learning methods, the classifiers models chosen for this study are: One-R, Multinomial Naïve Bayes (MNB), Random Forest (RF), Logistic Regression (LR) and Ensemble (Stacking). After thorough research conducted on previous studies concerning machine learning and tweet sentiments, these classifiers were chosen because they have, experimentally, the highest accuracy. For example, in a study done by Alsaeedi.A et al (see Bibliography), they stated "the random forest classifier outperformed other classifiers". Additionally, in a different study conducted by Gupta.B et al (see Bibliography), the classifiers with the highest accuracies were the Ensemble, Random Forest and Bayesian Logistic Regression, with values of 90%, 87.5% and 74.84% respectively. For stacking, meta classifiers that we will test will be Decision Tree and Logistic Regression. We will be looking at many different hyperparameters in order to select the best model as well which would then be incorporated into the stacking model which uses MNB, RF and LR as base classifiers.

Due to the unbalanced dataset, we have chosen to utilise the weighted F1 score. The F1 score combines precision and recall, and it is best used for unbalanced data (Korstanje.J, 2021). Additionally, we have used accuracy to observe the general performance of each individual model.

RESULTS

After feature engineering, the features selected for each sentiment were as follows:

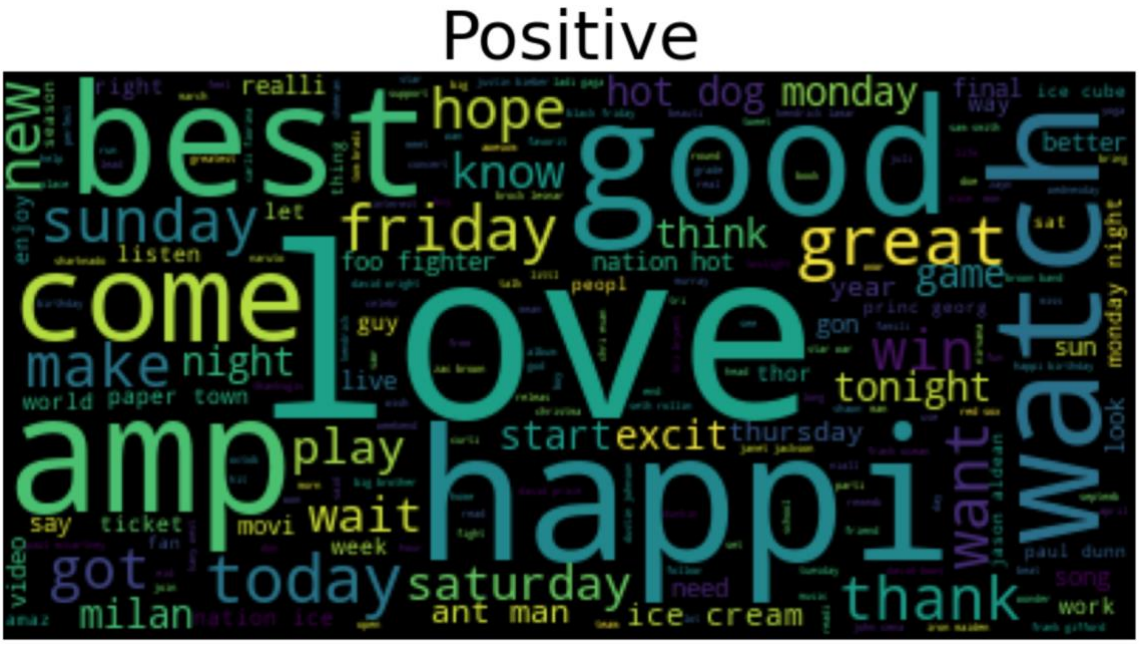

Figure 1: Tweets with Positive Sentiment

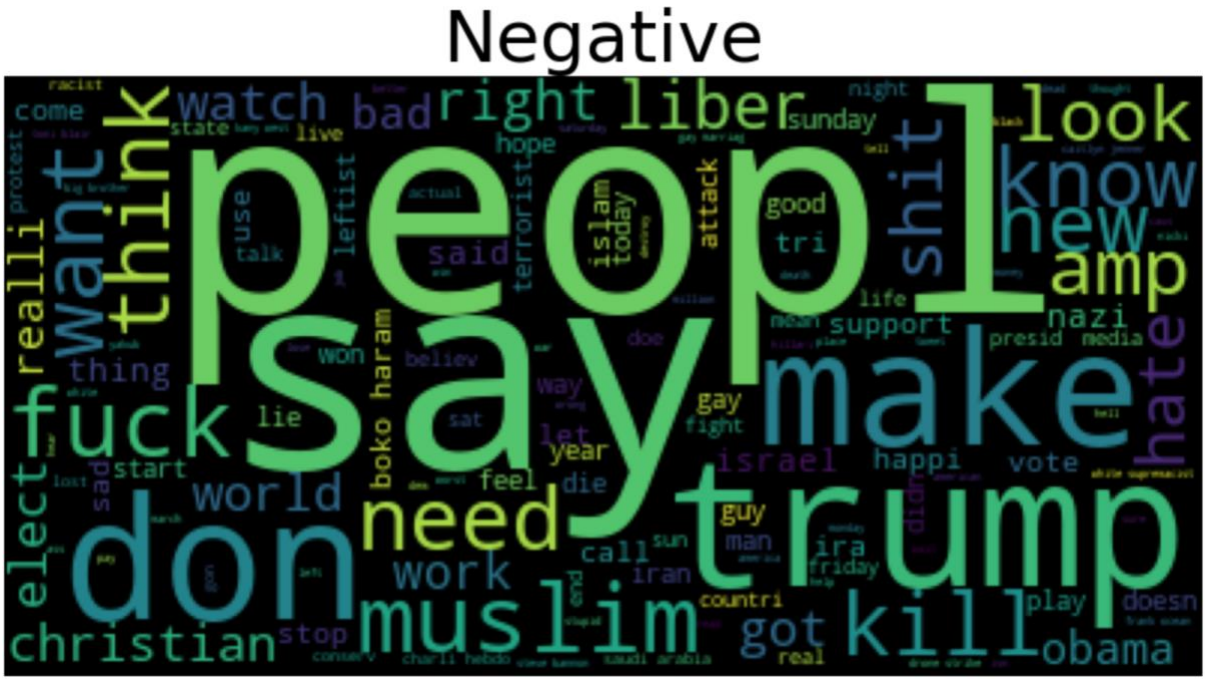

Figure 2: Tweets with Negative Sentiment

Figure 3: Tweets with Negative Sentiment

The following tables summarise the model evaluations and metrics:

ACCURACY

		k=10	k=100	k=200
MNB (Alpha = 1)	**All ***	0.6393 0.6450 0.6477 0.6294 **0.6404**		
	x2	0.5933 0.5971 0.5989 0.5910 **0.5951**	0.6436 0.6494 0.6475 0.6462 **0.6467**	0.6492 0.6507 0.6585 0.6503 **0.6528**
	MI	0.5938 0.5981 0.5998 0.5892 **0.5952**	0.6469 0.6487 0.6459 0.6492 **0.6477**	0.6511 0.6520 0.6578 0.6470 **0.6520**
One-R	**All ***	0.5900 0.5944 0.5877 0.5908 **0.5907**		
	x2	0.5900 0.5944 0.5877 0.5908 **0.5907**	0.5900 0.5944 0.5877 0.5908 **0.5907**	0.5900 0.5944 0.5877 0.5908 **0.5907**
	MI	0.5900 0.5944 0.5877 0.5908 **0.5907**	0.5900 0.5944 0.5877 0.5908 **0.5907**	0.5900 0.5944 0.5877 0.5908 **0.5907**
Random Forest (Max Depth = 100)	**All ***	0.6329 0.6481 0.6393 0.6398 **0.6400**		
	x2	0.6157	0.6415	0.6479

		0.6177 0.6178 0.6127 **0.6160**	0.6492 0.6411 0.6400 **0.6430**	0.6496 0.6475 0.6444 **0.6474**
	MI	0.6164 0.6160 0.6167 0.6154 **0.6161**	0.6432 0.6472 0.6422 0.6349 **0.6419**	0.6454 0.6454 0.6468 0.6424 **0.6450**
Logistic Regression (Max Iter = 1000)	**All ***	0.6458 0.6547 0.6479 0.6431 **0.6479**		
	x2	0.6159 0.6188 0.6187 0.6127 **0.6165**	0.6535 0.6584 0.6556 0.6512 **0.6547**	0.6610 0.6580 0.6626 0.6626 **0.6611**
	MI	0.6164 0.6170 0.6171 0.6154 **0.6165**	0.6553 0.6568 0.6552 0.6527 **0.6550**	0.6599 0.6568 0.6556 0.6594 **0.6579**

Table 1: Accuracy of Models (over 4-Fold Stratified Cross Validation) (Bold values are averages)

F1 SCORE

		k=10	**k=100**	**k=200**
MNB (Alpha = 1)	**All ***	0.6325 0.6389 0.6406 0.6205 **0.6331**		
	x2	0.4638 0.4687 0.4785 0.4564 **0.4669**	0.5978 0.6017 0.6033 0.6009 **0.6009**	0.6247 0.6286 0.6336 0.6301 **0.6293**
	MI	0.4671 0.4708	0.6074 0.6119	0.6310 0.6342

		0.4866 0.4516 **0.4690**	0.6046 0.6110 **0.6087**	0.6383 0.6285 **0.6230**
One-R	**All ***	0.4570 0.4652 0.4537 0.4583 **0.4586**		
	x2	0.4570 0.4652 0.4537 0.4583 **0.4586**	0.4570 0.4652 0.4537 0.4583 **0.4586**	0.4570 0.4652 0.4537 0.4583 **0.4586**
	MI	0.4570 0.4652 0.4537 0.4583 **0.4586**	0.4570 0.4652 0.4537 0.4583 **0.4586**	0.4570 0.4652 0.4537 0.4583 **0.4586**
Random Forest (Max Depth = 100, # of trees = 100)	**All ***	0.5593 0.5820 0.5720 0.5703 **0.5709**		
	x2	0.4570 0.5431 0.5378 0.5377 **0.5189**	0.6078 0.6164 0.6110 0.6068 **0.6105**	0.6137 0.6164 0.6123 0.6115 **0.6135**
	MI	0.5373 0.5386 0.5388 0.5360 **0.5377**	0.6084 0.6139 0.6073 0.5990 **0.6072**	0.6136 0.6153 0.6171 0.6121 **0.6145**
Logistic Regression (Max Iter = 1000)	**All ***	0.6338 0.6427 0.6360 0.6310 **0.6359**		
	x2	0.5319 0.5450	0.6144 0.6215	0.6314 0.6291

		0.5396 0.5390 **0.5389**	0.6169 0.6151 **0.6170**	0.6300 0.6325 **0.6308**
	MI	0.5390 0.5400 0.5398 0.5360 **0.5387**	0.6166 0.6191 0.6146 0.6148 **0.6163**	0.6290 0.6261 0.6221 0.6285 **0.6264**

Table 2: F1 Score of Models (over 4-Fold Stratified Cross Validation) (Bold values are averages)

Note *: k values do not apply to 'All', as 'All' includes all features in the dataset

In Tables 1 and 2 above, the feature selection methods utilised are Chi-Squared (x2) and Mutual Information (MI). The 'All' refers to the Accuracy and F1 Score values of the respective models using all the features in the dataset.

HYPERPARAMETER TUNING

Model	Hyperparameter	Avg. Accuracy	Avg. F1 Score
MNB	Alpha = 0	0.6524	0.6304
Random Forest	Max Depth = 10	0.6272	0.4988
	Max Depth = 200	0.6427	0.6158
Logistic Regression	Max Iter = 5,000	0.6611	0.6358
	Max Iter = 10,000	0.6611	0.6308

Table 3: Hyperparameter Tuning for Models (over 4-Fold Stratified Cross Validation)

ENSEMBLE STACKING

Model	Hyperparameter	Avg. Accuracy	Avg. F1 Score
Logistic Regression	Max Iter = 1,000	0.6315	0.6102
	Max Iter = 5,000	0.6294	0.6078
	Max Iter = 10,000	0.6304	0.6088
Decision Tree	Max Depth = 10	0.6388	0.6133
	Max Depth = 100	0.6379	0.6122
	Max Depth = 200	0.6387	0.6131

Table 4: Ensemble Stacking for Models (over 4-Fold Stratified Cross Validation)

DISCUSSION

Tables 1 and 2 showcase the different models and their relative performances. In regard to accuracy, we can see that for every model, the Chi-Squared and Mutual Information feature selection methods generally perform better than the All. For example, in Table 1 for Logistic Regression, the x2 and MI for k=100 is 0.6547 and 0.6550 respectively, whereas for All it's 0.6479. Although the difference is relatively statistically insignificant, it can still be observed that x2 and MI would be preferred over All. Additionally, for all models, the accuracy in x2 and MI increases as the number of features (k) increases. Overall, the x2 for each model outperforms the MI in accuracy at k=200, and the best model is Logistic Regression, which has an accuracy of 0.6611. However, x2 and MI for k=10 features is always lower than the All, due to an insignificant amount of features to classify data on. In regard to the F1 Scores in Table 2, it can be observed that as the number of features is increased from k=10 to k=200, the F1 Scores for x2 and MI for each model is also increased. For example, x2 in Logistic Regression is increased from 0.5389 to 0.6308 (k=10 to k=200). Again, we observe that x2 has higher outputs than MI and that Logistic Regression is the best model.

Another, more interesting trend that can be observed from the tables is that as the number of features increases, the Accuracy and F1 Score seem to be capped at approximately 60%. This was proven through some additional background tests with 1,000 features, which, after running, outputted almost similar accuracy and f1 scores. This might be due to the imbalance in the training data, which contains approximately 13,000 neutral sentiments and significantly smaller amounts of positive and negative sentiments. All classifiers might be better equipped to (and sometimes, wrongfully so) predict neutral statements, hence indicating some bias. There might be significant amounts of noise within the neutral tweets which prohibit the classifiers from gaining better accuracies.

Table 3 showcases the Accuracy and F1 Scores if we tune the hyperparameters. We can see that for each model, the hyperparameter tuning generally led to better accuracies and f1 scores for each model - except for the Random Forest F1 Score for the max depth of 10, which may be due to an insufficient sample size. Logistic Regression is still the optimal model after hyperparameter tuning. The overall results from Table 3 were used to facilitate the Ensemble Stacking. Selecting the attributes with the highest Accuracies and F1 Score, the following hyperparameters were chosen for ES: Alpha = 0, Max Depth = 200 and Max Iter = 5,000. On average, it can be seen in this table that the Decision Tree meta-classifier is better (although, there is not a significant improvement). A Decision Tree iteratively grows, putting significance on the number of units within a node, whereas Logistic Regression (LR) tries to fit all observations into some line of theoretical distribution. Due to uncertainty in fitting to a distribution, the Decision Tree tends to outperform LR (Bock.T, N.A).

In this study, the Random Forest classifier is the worst performing in terms of Accuracy (0.6427) and F1 Score (0.6158). This is contradictory to Gupta.B et al and Alsaeedi.A et al, both of which state Random Forest is one of the best classifiers, and Naive Bayes is one of the worst. This could be due to the fact that the number of trees generated for random forest is insufficient. However, this would require much more computational power and time to train, and due to time constraints, this would not be possible. On the other hand, the Logistic Regression (Max Iter = 10,000) classifier is the best performing. In contrast to the studies we have seen where ensembling is usually the top performing model, our LR classifier turned out

to be the best (Gupta.B et al., 2017). Stacking would help smooth errors over all the base classifiers with different biases, which would explain why stacking would be one of the top performing models, however, the difference of the scores between LR and Stacking (DT) are not that significant, therefore still showing similarities with the referenced research papers.

CONCLUSION AND FUTURE IMPROVEMENTS

In conclusion, after rigorously going through each pre-processing and feature engineering procedures, using stratified 4-Fold cross validation to evaluate a set of models with different feature selection methods and hyperparameter tuning, we chose Logistic Regression as our top performing model (other models were not too far off) based on the F1 score and Accuracy. Our worst performing (excluding One-R) was Random Forests, which is contradictory to other studies which are mentioned throughout the report. This was likely due to the limited number of trees that we used, which should be further studied. Additionally, Ensembling, which is considered to be one of the best performing models, was only slightly better in this case, and this could have been due to the lack of classifiers that we used or bias from the unbalanced dataset that was not fully tackled.

In order to improve the Accuracy and F1 Scores for future studies, it is recommended to utilise a computer with high computing power, as the Random Tree process would be more efficient and faster, and a final deduction can be made whether it is in fact a better classifier - as stated in the numerous referenced research papers. Additionally, further pre-processing/feature engineering steps can be applied to account for unbalanced data, such as other re-sampling techniques: oversample minority classes, or undersample majority classes. This would help reduce bias in the classifiers, resulting in higher accuracies and f1 scores. Finally, in the Ensembling stage, it should be studied whether adding more classifiers can improve the scores. Overall, the machine learning models and evaluations advocate a satisfactory class prediction process, which has the potential to work approximately 60% of the time. By incorporating these improvements, this accuracy can easily be increased to a higher level, hence predicting the sentiments of tweets well.

BIBLIOGRAPHY

Alsaeedi.A, Khan.M. (2019). *A Study on Sentiment Analysis Techniques of Twitter Data.* Medina: (IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 10, No. 2.

Bock.T. (N.A). *Decision Trees Are Usually Better Than Logistic Regression*. Retrieved from DisplayR Blog: https://www.displayr.com/decision-trees-are-usually-better-than-logistic-regression/#:~:text=Decision%20trees%20simplify%20such%20relationships,in%20terms%20of%20sample%20size

Gupta.B, Negi.M. (2017). *Study of Twitter Sentiment Analysis using Machine Learning Algorithms on Python.* Uttarkhand: International Journal of Computer Applications (0975 – 8887) Volume 165 – No.9.

He.H, Ma.Y. (2013). *Imbalanced Learning: Foundations, Algorithms, and Applications.* London: Wiley-IEEE Press.

Korstanje.J. (2021). *The F1 score*. Retrieved from Toward Data Science: https://towardsdatascience.com/the-f1-score-bec2bbc38aa6

Prusa.J, Khoshgoftaar.T & Dittman.D. (2015). *Impact of Feature Selection Techniques for Tweet Sentiment Classification.* Florida: Twenty-Eighth International Florida Artificial Intelligence Research Society Conference.

Rosenthal, S. N. (2017). *SemEval-2017 Task4: Sentiment Analysis in Twitter. Proceedings of the 11th* International Workshop on Semantic Evaluation . Vancouver, Canada: SemEval '17.

Sayce.D. (2020). *The Number of tweets per day in 2020*. Retrieved from David Sayce - Digital Consultant: https://www.dsayce.com/social-media/tweets-day/