# COMP30027 Report

## 1 Introduction

With the rapid growth of Twitter, millions of tweets are made everyday (Prusa et al., 2015). Therefore, the ability to mine, process and analyse this data is vital.

Sentiment analysis is among the most essential tasks. Sentiment analysis looks at emotions expressed via body of text to determine whether it holds a positive, negative, or neutral standpoint (Rosenthal et al., 2017).

For this study, we will look at the performance of each classifier to determine which is the most suitable for text classification problems. To achieve that, we use ZeroR as our baseline, and Multinomial Naive Bayes, Logistic Regression and Support Vector Machine.

Two datasets are provided: one training dataset, and one testing unlabeled. Each instance of the training dataset has a tweet's ID, its content and sentiment, while for the testing dataset, only a tweet's ID and its content is provided.

## 2 Method

### 2.1 Feature engineering

Tweets can be syntactically incorrect. Not rarely do we see abbreviation of common phrases, word elongation, or emojis and emoticons, which are not what Natural Language Processing is concerned with. To reduce feature space's size and computational expensiveness, performing noise reduction is important.

#### 2.1.1 Removing usernames/hyperlinks

Usernames and hyperlinks are unique. Although they can be included in multiple tweets, we do not have access to the actual content of an article, or previous tweets of a user. Hence, keeping them would increase our feature space, but not contribute meaningfully to our analysis.

#### 2.1.2 Removing numbers, non-English and non-alphanumeric characters

Non-English characters are similar to hyperlinks and usernames as they do not contribute meaningfully. Punctuation marks are also removed.

#### 2.1.3 Removing repeating characters

Elongated words conveyed heightened emotions. However, as it is difficult to find roots of words that are spelt incorrectly, we remove the repeated characters.

#### 2.1.4 Substitution of common phrases

algorithmMany common abbreviations are used in tweets, for example: 'idk' for "I don't know". We convert some of the most used abbreviations into their full-length meaning.

#### 2.1.5 Removing stop words, handling negation and contrast

Stopwords are commonly used words but do not considerably change the meaning of a sentence if omitted.

One special case is 'not'. 'Not' is important as it conveys negation, but without modification, it is removed in the process. Our approach is to replace 'not' or any abbreviated verbs with 'not' ("don't") with the token ***'negation'***. Doing this also reduces the feature space and helps convey the actual attitude of a tweet.

Contrast words such as 'yet' or 'but' are handled similarly to negation: we replace them with a ***'contrast'*** token.

### 2.2 Vectorisation

We use TF-IDF vectorisation. With sentences whose words are unique, BoW vectorisation method only returns each word's frequency, while TF-IDF considers how regularly each word occurs across the dataset and generates a corresponding weight (Imamah and Rachman, 2020). Therefore, TF-IDF allows us to determine which word is potentially deterministic of a tweet's sentiment.

### 2.3 Training/testing split

Our training dataset has 21802 instances. We have the following table:

| Sentiment | Count |
|---|---|
| Positive | 5428 |
| Neutral | 12659 |
| Negative | 3715 |
| Total | 21802 |

Table 1: Label distribution of dataset

From Table 1, our dataset is heavily biased towards 'Neutral' tweets, accounting for 58% of the number of instances. Given this information, we perform normal k-fold cross validation and stratified k-fold cross validation to determine whether imbalances in class labels affect the performance of our models. Our choice of fold for both situations is 10, as this ensures balance between sufficient training and testing data, and is time efficient. Results will be discussed in the *Results* section.

### 2.4 Models

#### 2.4.1 Baseline

For our baseline, we use ZeroR classification. ZeroR is the simplest classification method, which gives predictions based on the majority class. With our dataset, 'Neutral' is the majority class, so our baseline will produce a prediction where every test instance is labeled 'Neutral'.

ZeroR provides no predictive power, but it sets a performance threshold other models must surpass.

#### 2.4.2 Other machine learning models

We use Logistic Regression, Support Vector Machine and Multinomial Naive Bayes, and perform hyperparameter tuning on the first 2 models.

Logistic Regression 'extracts real-valued features from the input, multiplies each by a weight, sums them, and passes the sum through a sigmoid function to generate a probability' (Imamah and Rachman, 2020). For this dataset, we use multinomial logistic regression.

For Logistic Regression, choices of C are exponents of 10 from 0.01 to 100. A higher value of C, the fewer misclassfications there are, but it risks overfitting, while a lower value of C allows for better generalisation though there is potentially more misclassifications. For solver, 'newton-cg', 'lbfgs', 'liblinear' are considered. 'Liblinear' is a good candidate as we are working with a linear classification problem (Xue et al., 2019). For 'max_iter', we set it to 10000, so our model does not run out of iterations. After tuning, our model has C = 1.0, kernel = 'newton-cg'.

Support Vector Machine can take large margin classification tasks. It does this by creating a separating hyperplane using support vectors, so that the margin - the distance from one support vector to its closest data point - is maximised (Braun et al., 2011). Since more than one support vector can be chosen, Support Vector Machine is also a multi-class classification method.

For Support Vector Machine, choices of kernel are between 'rbf' and 'linear'. 'rbf' is default, while 'linear' is a good option, since we are working with linear vectors of text. 'Linear' kernel is also robust with large feature spaces (Zainuddin and Selamat, 2014). Similarly to Logistic Regression, a higher the value of C produces a smaller margin, but risks overfitting, while a smaller C value gives better generalisation by calculating hyperplanes with larger margins. Therefore, we let C be exponents of 10, from 0.01 to 100. Our 'gamma' choices are also exponents of 10, from 0.0001 to 1. Lower values of gamma means higher influence of one training instance towards surrounding instances, and vice versa. After tuning, our model has C = 1.0, gamma = 1.0, kernel = 'linear'.

### 2.5 Resampling

Given the imbalance, we perform oversampling, where instances of minority classes are repeated until classes are similarly distributed. We then train and compare our models' performance to our original dataset.

## 3 Results

Using the entire training dataset, ZeroR has an accuracy of 58%, representing the proportion of class 'Neutral'. As it provides no predictive power, cross validation is unnecessary.

### 3.1 Original dataset

Results are as follows:

With stratified k-fold cross validation, we have the following results:

### 3.2 Oversampled dataset

Results are as follows:

| | Precision | Recall | F1-score |
|---|---|---|---|
| Negative | 85.76% | 40.96% | 55.42% |
| Neutral | 72.43% | 93.58% | 81.66% |
| Positive | 81.24% | 55.00% | 65.58% |

Table 2: Multinomial Naive Bayes, accuracy 75.00%, 10-fold

| | Precision | Recall | F1-score |
|---|---|---|---|
| Negative | 58.27% | 33.64% | 42.17% |
| Neutral | 67.15% | 83.91% | 74.35% |
| Positive | 64.54% | 46.04% | 53.24% |

Table 3: Logistic Regression, accuracy 65.63%, 10-fold

| | Precision | Recall | F1-score |
|---|---|---|---|
| Negative | 51.22% | 41.24% | 45.17% |
| Neutral | 68.00% | 77.05% | 72.00% |
| Positive | 60.46% | 50.08% | 54.35% |

Table 4: Support Vector Machine, accuracy 63.97% 10-fold

| | Precision | Recall | F1-score |
|---|---|---|---|
| Negative | 85.76% | 40.96% | 55.42% |
| Neutral | 72.43% | 93.58% | 81.66% |
| Positive | 81.24% | 55.00% | 65.58% |

Table 5: Multinomial Naive Bayes, accuracy 75.00%stratified 10-fold

| | Precision | Recall | F1-score |
|---|---|---|---|
| Negative | 58.47% | 33.68% | 42.70% |
| Neutral | 67.17% | 83.58% | 74.48% |
| Positive | 63.86% | 46.03% | 53.47% |

Table 6: Logistic Regression, accuracy 65.73%, stratified 10-fold

| | Precision | Recall | F1-score |
|---|---|---|---|
| Negative | 70.69% | 11.49% | 19.71% |
| Neutral | 63.55% | 94.06% | 75.84% |
| Positive | 73.78% | 33.42% | 45.94% |

Table 7: Support Vector Machine, accuracy 64.94%, stratified 10-fold

With stratified k-fold cross validation, results are as follows:

| | Precision | Recall | F1-score |
|---|---|---|---|
| Negative | 94.23% | 40.93% | 57.07% |
| Neutral | 51.60% | 93.57% | 66.52% |
| Positive | 86.98% | 55.02% | 67.39% |

Table 8: Multinomial Naive Bayes, accuracy 64.57%, 10-fold

| | Precision | Recall | F1-score |
|---|---|---|---|
| Negative | 83.81% | 93.92% | 88.57% |
| Neutral | 81.61% | 68.71% | 74.60% |
| Positive | 80.02% | 84.89% | 82.38% |

Table 9: Logistic Regression, accuracy 81.88%, 10-fold

| | Precision | Recall | F1-score |
|---|---|---|---|
| Negative | 83.20% | 95.88% | 89.08% |
| Neutral | 84.64% | 66.89% | 74.72% |
| Positive | 80.33% | 87.43% | 83.72% |

Table 10: Support Vector Machine, accuracy 82.63%, 10-fold

| | Precision | Recall | F1-score |
|---|---|---|---|
| Negative | 94.19% | 40.94% | 57.06% |
| Neutral | 51.61% | 93.57% | 66.53% |
| Positive | 87.00% | 55.01% | 67.39% |

Table 11: Multinomial Naive Bayes, accuracy 64.57%, stratified 10-fold

| | Precision | Recall | F1-score |
|---|---|---|---|
| Negative | 83.91% | 93.82% | 88.59% |
| Neutral | 81.26% | 68.74% | 74.47% |
| Positive | 80.00% | 84.62% | 82.24% |

Table 12: Logistic Regression, accuracy 81.78%, stratified 10-fold

| | Precision | Recall | F1-score |
|---|---|---|---|
| Negative | 83.31% | 95.92% | 89.17% |
| Neutral | 84.28% | 66.31% | 74.22% |
| Positive | 79.68% | 87.07% | 83.20% |

Table 13: Support Vector Machine, accuracy 82.33% stratified 10-fold

## 4 Discussion

### 4.1 Challenges

One great challenge is contronyms. For example, while 'You look terrific' is a compliment,

'There is a terrific accident' indicates terror and destruction, which are opposite in meaning. TF-IDF does not consider this.

Another prominent challenge is our dataset's bias towards one label. The minority classes' proportions are significantly smaller compared to that of 'Neutral'. Given a small dataset, the numbers of instances of 'Positive' and 'Negative' are insufficient for training, which can lead to poor generalisation.

### 4.2 Performance

Performances are similar under both k-fold and stratified k-fold validation. Therefore, for this dataset, either cross validation method is accepted. In the sections below, we only use results collected from doing normal 10-fold cross validation.

Our hypothesis is that models will perform better on oversampled data, since equal proportions of class labels lead to unbiased predictions.

#### 4.2.1 Multinomial Naive Bayes

Our model performs well with the original dataset with 75% accuracy (Table 2) , but this drops by 13.9% to 64.57% (Table 8). From Table 2, 'Negative' and 'Positive' have high precision but low recall, suggesting the model does not predict many instances of the two labels, but most of their predicted labels is correct. 'Neutral' has high precision and high recall, which can be explained by itself being the majority class. With the oversampled dataset, the trend continues for 'Negative' and 'Positive' (Table 8). However, for 'Neutral', precision drops 28.7% to 51.60%, due to similar proportions between class labels.

#### 4.2.2 Logistic Regression

On the original dataset, our model achieves 65.63% accuracy (Table 3) and it increases by 24.7% to 81.88% on the oversample dataset (Table 9). Precision and recall for 'Positive' and 'Negative' are low, indicating that there are instances of those classes that are classified 'Neutral', and among the predicted instances, few of them are true positives. Precision and recall for 'Neutral' remain high, which is discussed above.

On the oversampled dataset, precision and recall for 'Negative' and 'Positive' increases considerably. However, due to our oversampling, it does not introduce new instances for training and testing, which bears the risk of overfitting, as multiple identical tweets are not realistic. Recall for 'Negative' drops by 22.1%, while precision increases by 21.5%, which keeps the f1-score unchanged (Table 9), suggesting that our model's performance on 'Neutral' is similar in both cases, which can be attributed to 'Neutral' not being re-sampled.

#### 4.2.3 Support Vector Machine

On the original dataset, our model achieves 63.97% accuracy (Table 4) and it increases by 24.7% to 88.47% on the oversample dataset (Table 10). From Table 3 and Table 4, performance of our Support Vector Machine and Logistic Regression model are fairly similar.

Same level of performance is observed again in the oversampled dataset. As mentioned above, this method of oversampling bears the risk of overfitting.

## 5 Conclusions

In this study, we perform tweet sentiment classification, using Logistic Regression, Support Vector Machine and Multinomial Naive Bayes. Overall, 3 models register better performance than our baseline. Multinomial Naive Bayes is the best classifier on the original data, while Support Vector Machine is performs best on oversampled data. Because of its ability to handle large feature space, Support Vector Machine is chosen as our classification model.

Future work could be done by implementing Stacking, where multiple models are used to generate the final prediction. Also, more tweets could be collected for a more generalised training process.

## References


Andreas Ch. Braun, Uwe Weidner, and Stefan Hinz. 2011. Support vector machines, import vector machines and relevance vector machines for hyperspectral classification — a comparison. In *2011 3rd Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS)*, pages 1–4.

Imamah and Fika Hastarita Rachman. 2020. Twitter sentiment analysis of covid-19 using term weighting tf-idf and logistic regresion. In *2020 6th Information Technology International Seminar (ITIS)*, pages 238–242.

Joseph Prusa, Taghi M. Khoshgoftaar, and Naeem Seliya. 2015. The effect of dataset size on training tweet sentiment classifiers. In *IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*, pages 96–102.

Yingbin Xue, Xiaoye Wang, and Zan Gao. 2019. Multi-classfication sentiment analysis based on the fused model. In *2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 1771–1775.

Nurulhuda Zainuddin and Ali Selamat. 2014. Sentiment analysis using support vector machine. In *2014 International Conference on Computer, Communications, and Control Technology (I4CT)*, pages 333–337.