COMP30027 Sentiment Classification of Tweets Report

Anonymous

1. Introduction

With the normalisation of social media, more and more people share their thoughts and opinions on social media, and in this case Twitter. In this report, I utilised supervised machine learning models to learn from 21802 tweets to help predict the sentiment of 6099 tweets. These machine learning models will help classify the tweets into three categories: 'Positive', 'Negative' or 'Neutral'. More importantly, through sentiment classification of tweets I will also critically analyse the differences between models and evaluate the performances of the predictions, in order to find the best performing model for sentiment analysis. Aside from just comparing models, the choice in text pre-processing, the selection in features and how they are engineered will also influence the performance of the models. Lastly, using appropriate evaluation methods to critically assess the performance of each distinct Machine Learning Models.

The dataset used for Twitter sentiment analysis is split into two data sets: training data and testing data. Within the training data set there is 21802 instances and each row in the datafile contains a tweet ID, the text of the tweet and a sentiment.

Sample row from training data set:

[805582613687713000, doctors hit campaign trail as race to medical council elections heats up https://t.co/iifdwb9v0w #homeopathy, neutral]

A breakdown of the training dataset in terms of sentiment:

Train Data	21802
Positive	5428
Neutral	12659
Negative	3715

While the test data set has 6099 instances, and will contain only a tweet ID and the text of the tweet.

Sample row from testing data set:

[802217876644052000, @loogunda @poroshenko putin abducted ukrainian citizens in occupied territories & mobilized them /human shield against ukraine = war crime.]

2. Method

To summarise, the following steps were needed to pre-process the data to allow it to be tokenised while simultaneously reducing the noise within the dataset. Once the texts were tokenised it also need needed to be vectorized in order to generate a set of features that was receptive for the machine learning models. The set of features that were generated where numerical data types that returned the weights of each term in each text instance. Now having knowledge in the data type of the features, this influenced my choice in selecting Gaussian Naïve Bayes, Support Vector Machine with a Linear Kernel, and Logistic regression. Finally, utilising evaluation methods such as the accuracy value, confusion matrix and F1-score with weighted-averaging.

2.1 Data Pre-processing

Aside from English words, a tweet can contain punctuation to express emotions, username tags, hashtags, links, and colloquial language. Therefore, I conducted some text pre-processing to reduce noise and help increase the performance, also allowing to reduce close duplicates but at the same time maintain a balance between useful features and the meaning of the words. The text pre-processing methods I used to reduce duplication and to also help configure a more accurate count of words/feature appearances in a text, consists of making all text lower case, removing repeated characters, stemming and lemmatisation to only keep root-forms of a word. In addition,

the following text pre-processing techniques were also used to reduce words/features that had low information as thy had little contribution to the sentiment of a tweet: removing 'stop words', removing URLS and removing non-alphanumerical characters.

2.2 Feature Engineering

Machine Learning models are unable to take text as input, therefore I have chosen 'Term Frequency Inverse Document Frequency' a text vectorization method to aid feature selection.

$$W_{d,t} = f_{d,t} \times \log \frac{N}{f_t}$$

Figure 1- TFIDF Formula

The Term Frequency Inverse Document Frequency vectorizer (TFIDF) essentially measures the weight of a term through reducing the weight of frequent terms and increasing the weight of rare and indicative ones. In figure 1, '$f_{d,t}$' is the frequency of term t in document d, 'f_t' is the number of documents containing t, and N is the total number of documents. This allows the models to identify what terms had more influence on the final sentiment of a text. I decided to use TFIDF as I needed to take into account how informative a term is towards a sentiment, therefore, I needing to reduce the influence of those terms that appeared frequently in many tweets.

2.3 Training the data

Due to the test dataset not having labelled sentiments, therefore I adopted a holdout approach on the train dataset to evaluate the models built. I decided to have an 80-20 split for the train and test data, because I wanted to achieve a balance between overfitting the training data and having enough test data where the results will be a good representation of the data.

2.4 Classification

2.4.1 Baseline Model – Zero R

I used a baseline model as a benchmark for other models. I chose Zero R, where it will always predict only the majority class.

$$\hat{c} = \arg\max_{c_j \in C} P(c_j)$$

Figure 2 – Baseline Zero R Formula

2.4.2 Naïve Bayes

Naïve Bayes classifies instances into possible classes through calculating the prior and conditional probabilities of the attributes conditioned on classes. Naïve Bayes was a suitable model as it was a probabilistic learner and is receptable to continuous attributes. Since the attributes are continuous, I have chosen to use Gaussian Naïve Bayes, because according to the central limit theorem for a large enough sample size the data should approach normal distribution.

$$P(c_j|x) \approx P(c_j) \prod_k P(x_k|c_j)$$

Figure 3- Naïve Bayes Formula

2.4.3 Support Vector Machine

The Support Vector Machine (SVM) aims to find a linear hyperplane to separate the classes. This is done by finding a decision boundary and finding an optimal solution where there is a trade-off between maximising the margins between difficult points close to the decision boundary and minimising training error. I used a support vector machine as it was good for handling datasets with many features and was suitable for numerical features.

$$\min_w \frac{1}{2}||w||^2 + C\sum_{i=1}^{N} \xi_i$$
$$subject\ to\ y_i(w^T x_i + b) + \xi_i - 1 \geq 0,$$
$$\xi_i \geq 0, i \in \{1,2,\ldots,N\}$$

Figure 4 - SVM's Formula

The C in figure 4 is a regularisation parameter that penalises the errors in training.

2.4.4 Logistic Regression

The logistic regression is a probabilistic model that takes in continuous data to predict discrete labels. It completes this through determining the appropriate parameter vector $\beta^{(c_j)}$ for each class by determining the β that minimises the loss function. This process is done through many iterations of using gradient descent, where the β is updated to

reduce the error until it is at the minimum.

$$P(y = c_j|x, \boldsymbol{\beta}) = \frac{e^{\boldsymbol{\beta}^{(c_j)}x}}{\sum_{c=1}^{|C|} e^{\boldsymbol{\beta}^{(c_j)}x}}$$

Figure 5 – Logistic Regression's Formula

The logistic regression model in comparison to the Naïve Bayes only learns to distinguish classes.

2.5 Evaluation methods

I will be evaluating the performance of the 4 models with the following metrics:

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN}$$

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F1 = \frac{2PR}{R}$$

Figure 6 – Evaluation metrics Formulas

In terms of the F1-score, I decided to use weighted-averaging because the distribution of 'negative', 'neutral' and 'positive' are not equally distributed.

3. Results

The dataset used for this sentiment analysis was sourced from Rosenthal (2017). In the training-evaluation phase, these models were fitted onto 80% of the training data TFIDF vectorized features to learn and generalise. Therefore, the results and evaluation below is generated from the 20% of the training data that was held out.

3.1 Baseline Model Zero R

Accuracy: 0.6111
F1 – Score: 0.4636

		Predicted		
		Neg	Neu	Pos
Actual	Neg	0	658	0
	Neu	0	2665	0
	Pos	0	1038	0

Table 1- Zero R's Confusion Matrix

3.2 Naïve Bayes (Gaussian)

Accuracy: 0.4357
F1 – Score: 0.4474

		Predicted		
		Neg	Neu	Pos
Actual	Neg	376	190	92
	Neu	862	947	856
	Pos	228	233	577

Table 2- Naïve Bayes' Confusion Matrix

3.3 Support Vector Machine

For a Linear Kernel (C = 1.0):
Accuracy: 0.6705
F1 – Score: 0.6412

		Predicted		
		Neg	Neu	Pos
Actual	Neg	182	459	17
	Neu	144	2344	177
	Pos	8	632	398

Table 3- Support Vector Machine's Confusion Matrix

For a Linear Kernel (C = 10.0):
Accuracy: 0.6143

For a Radial Basis Function (RBF) Kernel (C = 1.0):
Accuracy: 0.6687

3.3 Logistic Regression

Accuracy: 0.6666
F1 – Score: 0.6362

		Predicted		
		Neg	Neu	Pos
Actual	Neg	172	465	21
	Neu	139	2341	185
	Pos	9	635	394

Table 4- Logistic Regression's Confusion Matrix

4. Discussion / Critical Analysis

The overall performance of the models has indicated a need for better selection in features due to the accuracy scores falling in a range between 0.6-0.7. In the following section, I will

be discussing the model interpretations, error analysis and how I tuned the hyperparameters.

4.1 Not Randomising Holdout

With this particular dataset, when the holdout approach was randomised, the performance of the Naïve Bayes model was weaker in comparison. The accuracy between a non-randomised holdout and a randomised holdout fell from 0.4357 to 0.4212. The discrepancy only suggests that the distribution of sentiments changed between the train and test data sets within the training evaluation phase. This result may also suggest that the data between train and test data is not proportionally distributed.

4.2 Comparing Models and Tuning Hyperparameters

In comparison, both the SVM (0.6705) and Logistic Regression models (0.6666) had better accuracy scores than the Zero R model (0.6111), however not by a lot. Naïve Bayes (0.4357) performed poorly compared to all the models, and by comparing the actual labels and predicted labels it has classified more 'positive' and 'negative' labels in comparison to the others. This may be subjected to the assumption that attributes are independent of the class or that the multinomial Naïve Bayes may have been better suited for the distribution of the attributes. Nevertheless, since all models did not perform well it may suggest that feature selection can be improved.

The SVM and Logistic Regression displayed similar performances. For SVM, when deciding for a suitable C, I ran the model with both a C = 1.0 and a C = 10.0. When C = 1.0 it performed better with an accuracy score of 0.6705, while for C = 10.0 the accuracy score fell to 0.6143. This suggests that it performed better when the margin was maximised and when the model was more lenient towards errors.

Initially, for the kernel choice of the Support Vector Machine I tested both a linear kernel and an RBF kernel. For when C = 1.0, the linear kernel outperformed RBF, where the accuracy scores were respectively 0.6705 and 0.6687. This suggests that the data was more linearly separable in comparison.

The superior performance by the linear kernel is partially aligned with the performance of the logistic regression as they both search for a linear decision boundary.

In comparing the classes, for both SVM and Logistic Regression there was very few cases where the model predicted a 'positive' for an actual 'negative' and vice versa. This indicates that most of the error occurred between predicting 'neutral' for cases when it was not 'neutral' or predicting 'negative' or 'positive' for when the actual was 'neutral'. Therefore, this could mean that some features could influence both 'neutral' and 'positive'/ 'negative' sentiments.

Across both SVM and Logistic Regression, it is also plausible that the trained models overfitted and had a bias for 'neutral' sentiments, and again this may be due to feature selection or the distribution of the training dataset.

Ultimately, this echoes the earlier suggestion that to improve the performances of the three models, feature selection methods like Pointwise Mutual Information (PMI) where it identifies attributes that are most correlated with classes. This can not only help improve performance but also the computation time as it will have less features.

5. Conclusions

For this dataset, I found that the SVM with a linear Kernel and a C = 1.0 performed the best, as the data was linearly separable. Even though, there is areas of improvement such as employing a feature selection method that helped identify the best attributes in predicting the correct class. This sentiment analysis helped identify which methods and strategies worked best in identifying the sentiments of tweets.

6. References

Ehinger, K. (2022). *COMP30027-04-NaïveBayes(Ni)*. https://canvas.lms.unimelb.edu.au/courses/124130/pages/week-2-probability-and-naive-bayes?module_item_id=3196365

Ehinger, K. (2022). *COMP30027-06-Evaluation(Ni)*. https://canvas.lms.unimelb.edu.au/courses/124130/pages/week-3-discrete-and-continuous-data-and-model-evaluation?module_item_id=3196366

Marius, H. (2020). *Multiclass Classification with Support Vector Machines (SVM), Dual Problem and Kernel Functions*. https://towardsdatascience.com/multiclass-classification-with-support-vector-machines-svm-kernel-trick-kernel-functions-f9d5377d6f02

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

Samadi, H. (2022). *COMP30027-09-SVM-Handout*. https://canvas.lms.unimelb.edu.au/courses/124130/pages/week-5-svm-and-interpretation?module_item_id=3196368

Samadi, H. (2022). *COMP30027-10-Interpretation-Hasti*. https://canvas.lms.unimelb.edu.au/courses/124130/pages/week-5-svm-and-interpretation?module_item_id=3196368

Samadi, H. (2022). *COMP30027-12-LogisticRegression-Hasti*. https://canvas.lms.unimelb.edu.au/courses/124130/pages/week-6-linear-regression-and-logistic-regression?module_item_id=3196369

Samadi, H. (2022). *COMP30027-14-FeatureSelection-Hasti*. https://canvas.lms.unimelb.edu.au/courses/124130/pages/week-7-classifier-combination-and-feature-selection?module_item_id=3196370