Twitter Sentiment Analysis
COMP30027

Anonymous

1. Introduction

Any message from social media can be analysed sentimentally. The sentiment could be obtained by several characteristics, such as special words, patterns, and context. This Twitter text sentiment analysis project focuses on predicting labels by analysing the words. The goal is to correctly predict the sentiment given a Twitter text and then make predictions for new text. Therefore, the goal could be generalised to find the most suitable mapping function from feature to label. Several supervised machine learning models are implemented for data modelling to achieve the goal. And the performance of models is compared to conduct the most well-performed model.

This project follows three general steps. The first process is pre-processing, including data cleaning and transformation of the raw dataset to obtain a much cleaner and machine-readable data and features engineering. The second process is training various models and using the models to predict the labels for test data. The last process is evaluating and analysing every model.

2. Data

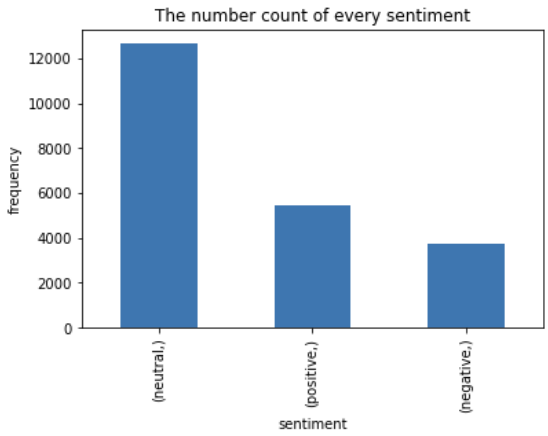

Figure 1- the distribution of sentiment labels

The dataset provided by Rosenthal (2017) comprises 21802 instances in train data and 6099 instances in test data. The raw training set consists of two attributes, id and text, and one label, sentiment, And the testing set only have two attributes. Observing the features, the ID feature contains the user's numerical ID. Another feature is the text posted by users. The text feature contains a single string text which contains both useful information and noise that need to be clean.

For the label, sentiment is given for every training instance. The distribution of the value counts, shown in Figure 1 above, suggests that the data distribution is highly imbalanced. There are 12659 neutral instances, 5428 positive instances, and 3715 negative instances. The number of neutral instances is much larger than the number of positive and negative instances combined. The imbalanced distribution of data could potentially hinder the data modelling.

3. Method

In this section, we describe three general steps we used to produce the mapping models, pre-processing, modelling, and hyperparameter tuning

3.1 Pre-processing

To begin with, the pre-processing of the raw data gets performed. There are three subprocesses get implemented during the pre-processing. They are data cleaning, transformation, and feature engineering.

3.1.1 Data cleaning

Data cleaning aims to only include the relevant information by reducing the noisy data. There are two features in the raw data set corresponding with one label. However, the ID feature is not chosen to conduct the supervised machine learning models because the numeric account ID does not contain sentimental information. In contrast, the text feature gets

included for supervised learning because the information posted by users can be analysed sentimentally. To further clean the text attribute, three kinds of texts get removed or transformed. They are words that are irrelevant to sentiment, the same words with different forms, and the different words with the same lexeme.

Firstly, the irrelevant text is removed, including special words such as @username, URL., punctuations, and numbers. For example, special words like @youtube should contain no emotional information because only the tweets posted by YouTube contain sentiment, but the username itself does not. Secondly, the same words with different forms are transformed into one base form by expanding the contraction, replacing the accented characters, lowering all cases, and converting the number into the word form. For instance, the abbreviation "I'm" gets expanded into two separate words. "I" and "am". The reason for transforming into a base form is that those words are identical both semantically and grammatically. So, they should be classified as the same words instead of separate words. Thirdly, the different words with the same lexeme get replaced by the basic lexeme form. By lemmatization, words like "is", and "are" get reduced to the lexeme, "be", so that the words with the same lexeme could be classified as one word.

3.1.2 Transformation

In addition to the pre-processing, the text features need to transform into a machine-friendly form. Term frequency-inverse document frequency (TF-IDF) is implemented instead of the bag of words because there are an enormous number of common words like stop words in every text. By using TF-IDF, the noise caused by common words gets reduced. And the two-gram model is used for the vectorizer to identify the difference between one word and the negation. For example, 'like' and not "like" have opposite sentiments, but the one-gram model could not identify them correctly. In contrast, the 2-gram model receives a more precise message that could potentially benefit sentiment identification.

3.1.3 Feature engineering

After the previous two pre-processing processes, the feature engineering gets implemented by the SelectKBest algorithm. Only 8000 most relevant features are included by the chi-square algorithm. By engineering the features, the dimensionality and noise are reduced.

3.2 Modelling

There are three types of models used to model the data and compare the performance.

3.2.1 Baseline Model

First of all, the baseline model 0R gets built. 0R model constantly predicts the most frequent labels. The role of the base model is to compare and measure the performance of other models

3.2.2 Basic Models

Secondly, several basic models are used to compare the performance and built for the following ensemble stacking model. They are decision tree, logistic regression model, and support vector machine. The decision tree is a simple machine learning model for categorical features. The advantages of the decision tree are simple implementation, high interpretability, and no distribution are assumed. However, the major drawback is overfitting. To be more specific, the decision tree has high variance and low bias, which lead to poor performance for new data input. Logistic regression uses the probability of success and failure of an event to build the classification model (Amiya 2020). The major benefits of implementing a logistic model are the same as the decision tree, but the disadvantage is that the model can only detect the linear relationship between features and labels without transforming the data. Support vector machine (SVM) benefits from effectiveness in high dimensional spaces and memory efficiency (3Dhiraj K). Nevertheless, the advantages of implementing SVM are highly sensitive to noise.

3.2.3 Ensemble Learning Models

Thirdly, more complex models using ensemble learning methods get built. Both bagging and stacking models combine several weak learners to obtain a stronger and more sophisticated model. Specifically, the data get modelled by the random forest model, a typical bagging model, and stacking models with logistic regression meta-classifiers. Random forest combines results from several decision trees to produce the prediction. The advantage of a random forest algorithm compared with a decision tree are that the chance of overfitting gets reduced. For the stacking model, basic

models from the second step are integrated by logistic regression meta classifier. The advantage of using stacking is that the ensembled model combines several well-performing models and usually has better practical results (Jason B., 2020). However, both time and space complexity for the ensemble algorithm become enormous, resulting from training multiple base learners and then using a meta classifier to train the final model (Naresh, 2019). In addition to the drawback of random forest and stacking models, models are hard to interpret and impossible to fully understand. However, the bagging models and stacking models perform reasonably well in this task, and the goal of supervised machine learning is to get correct predictions rather than interpret the logic behind the relations.

3.3 Hyperparameter tuning

To tune the parameter for every base model, the result from the default setting and the best result from the Search Cross-Validation (SearchCV) algorithm are compared to obtain the combination of hyperparameters with a better score. There are two SearchCV algorithms, RandomizedSearchCV and GridSearchCV. Considering the feasibility of this project, the randomized search method is better suited for this situation than the grid search because the dataset is considerably large. Although the grid search guarantees the parameters with the best score, the exhaustive grid search can consume considerable computation power and time, making this project infeasible. Therefore, the randomized search method with 10% samples of the search space of the hyperparameter is used to approximate the best parameter. By doing the 10% random sampling, the computation power and time are saved, and the potential overfitting problem is avoided by having a sub-optimal solution.

In addition to scoring each parameter combination for the RandomizedSearchCV function, "f1_weighted" is used to measure the performance. There are two advantages of scoring by "f1_weighted" for this imbalanced dataset. One advantage is that the weighted score considers the total number of each label. The more frequent label will take more weight, and the impact of the majority labels get reduced. Another advantage is that the F1 score encourages the model to balance the precision and recall. F1 score measures both correctness of predicting positive cases and the ability to detect the true positive.

$$F_1 = \frac{2(P*R)}{P+R} \quad P = \frac{TP}{TP+FP} \quad R = \frac{TP}{TP+FN}$$

As the formula shows above, the f1 score combines precision and recall. The precision gives the ratio of true positive out of all predicted positive, and recall gives the ratio of true positive out of all correctly predicted labels. By combing precision and recall, the F1 score measures both correctness of predicting positive cases and the ability to detect the true positive. In contrast, other scoring methods like accuracy can be strongly biased by the majority label, which can mislead the result and keep predicting one class.

4. Result

	STACKING	SVM	DECISION TREE	RANDOM FORESET	LOGISTIC REGRESSION
VALIDATION ACCURACY	**0.721**	0.715	0.579	0.607	0.645
WEIGHTED F1	**0.739**	0.749	0.579	0.611	0.721
PRECISION	**0.757**	0.745	0.599	0.627	0.721
RECALL	**0.752**	0.745	0.583	0.579	0.706

Table 1- a summary of selected metrics for all models.

Table 1 summarizes the selected metrics of 5 models, and it suggests that the stacking model has the best performance in terms of validation accuracy and weighted f1 score. The stacking model with logistic meta classifier has 0.71 validation accuracy and 0.725 weighted F1 score (0.724 precision and 0.73 recall). The training and validation accuracies express that an average of 71% of predictions are correct by five-fold cross-validation, and 72.5% of labels are correctly predicted for the validation set. Moreover, the weighted F1 score of 0.725 gives a preferable precision and recall balance.

5. Critical Analysis

5.1 Impact of the imbalance dataset

Observing the raw training dataset, the imbalanced distribution of the data could be found. Three sentiment groups have a huge difference in the total number, neutral labels

with 12659 cases, positive and negative labels have 5428 and 3715 cases, respectively. The neutral sentiment instances are more than all other two groups combined. This uneven number of instances for different groups could lead to low recalls and precision for the minority groups. Because of the imbalanced data structure, models are encouraged to predict the neutral labels. By predicting the majorities and ignoring minorities (0R model), the model accuracy could reach reasonable accuracy of 0.5. Because models will predict fewer minorities, the recall or precision decreases significantly. For example, the 0R model, which is the extreme case of predicting only the majority label, has low precision, recall and f1, 0.346, 0.588 and 0.435 respectively. Our group's solution is implementing a balanced class weight parameter for modelling and using weighted f1 as scoring.

		validation accuracy	weighted f1	precision	recall
before balanced	stacking	0.721	0.739	0.757	0.752
	svm	0.715	0.749	0.745	0.745
	decision tree	0.579	0.579	0.599	0.583
	random foreset	0.607	0.611	0.627	0.579
	logistic regression	0.645	0.721	0.721	0.706
after balanced	stacking	0.701	0.734	0.738	0.734
	svm	0.69	0.727	0.726	0.729
	decision tree	0.553	0.567	0.563	0.576
	random foreset	0.6	0.556	0.59	0.612
	logistic regression	0.62	0.705	0.705	0.711

Table 2 - a summary of selected metrics for all models before and after implement the “balanced” class weight

Table 2 shows the comparison before using “balanced” and after using “balanced”. The metrics above suggest that “balanced” weight class does not improve the model. However, this does not mean changing class weight could not improve the model because there are other weighting strategies that could be further investigated.

5.2 High FN and FP problem

One problem suggested by the confusion matrix (Figure 2) is that the stacking model has both high false negative and false positive for neutral labels. Models can identify the difference between negative and positive classes. However, models perform poorly in identifying the difference between neutral and positive classes and between neutral and negative classes.

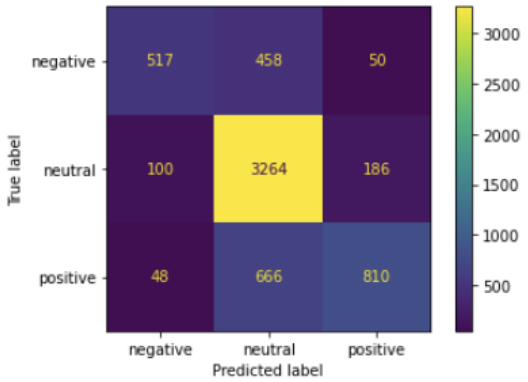

Figure2 - the confusion matrix of stacking model

5.2.1 Hypothesis

There are two potential hypotheses, one hypothesis relates to the text input, and another relates to the parameter tuning.

5.2.1.1 Mutual Words

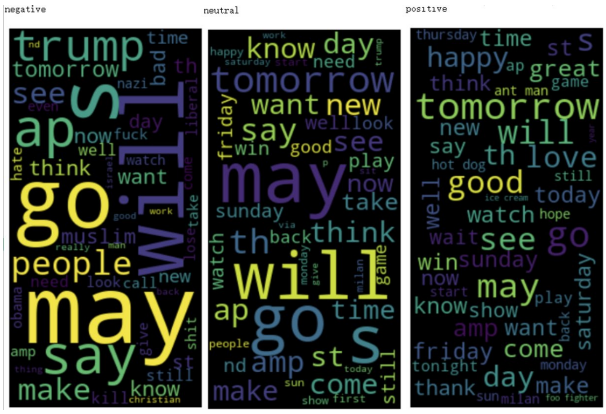

Figure3 - visualisation of word count for each sentimental groups. The group, from left to right, are negative, neutral and positive. The larger the front size is, the frequent the word is.

The first hypothesis of having both high FN and FP rate is that the neutral class shares many identical words with positive and negative classes. When the model tries to predict new labels, shared words confuse the model and potentially produce the wrong label. As Figure 3 indicates, many words cross over between different sentimental groups. For example, words like "tomorrow", “go”, and "may" are shared by all three groups of instances, "good", "happy", "st" are shared by neutral and positive, and "trump", "need" are shared by negative and neutral groups.

5.2.1.2 Insufficient Sample Space

The second hypothesis is that the sub-optimal approximation produced by the

RandomizedSearchCV is not close enough to the true optimal hyperparameter. To reduce the time complexity, only the most relevant portion of the hyperparameter gets randomly sampled. There is a chance that some other important hyperparameter not get considered. In addition to the second hypothesis, the randomised search algorithm only samples a small proportion of all hyperparameter sample space. This insufficient sample size problem could also make the incorrect approximation of the optimal hyperparameters.

5.2.2 Solutions

Some potential solutions are increasing the number of instances for two minority classes, applying resampling methods, and changing the class weight. One of the best solutions for solving high FN and FP is that increase the instances of positive and negative sentiment cases. By increasing the size of two minority classes, there is a higher chance that models could learn new deterministic features that could improve the prediction. Second, the resampling method could help to balance the dataset. Undersampling deletes the instances for major groups, and oversampling extends the minor groups. Both undersampling and oversampling secure the even distribution of data. However, the drawback of using the resampling methods is that models using undersampling could suffer from information loss since an extensive portion of data gets deleted, and the models using oversampling could be overfitting because the newly generated data is not observation but the mathematical implication of possible data or the repeated data of original data set. Thirdly, changes in class weight could also potentially benefit the prediction. For simplicity, most models use the "balanced" parameter, which takes the inverse proportion of the frequency, but other weighted strategies could also be investigated. Class weight parameters could affect the model enormously by adding more penalties when incorrectly predicting minorities.

6. Conclusion

In conclusion, we construct a stacking classifier based on logistic regression, decision tree and support vector machine for twitter sentiment analysis. The prediction produced by the model seems reasonable assuming training and testing datasets have the same class distribution. However, there are still room for improvement. For example, neural network might be a better choice to get higher accuracy, especially RNN or LSTM since texts are sequential information, which is our future research topic.

7. Bibliography

Amiya R.R. (2020) Advantages and Disadvantages of Logistic Regression. GeeksforGeeks. https://www.geeksforgeeks.org/advantages-and-disadvantages-of-logistic-regression/

Dhiraj K. (2019). Top 4 advantages and disadvantages of Support Vector Machine or SVM. Medium. https://dhirajkumarblog.medium.com/top-4-advantages-and-disadvantages-of-support-vector-machine-or-svm-a3c06a2b107

Jason B. (2020). Stacking Ensemble Machine Learning With Python. Machine Learning Mastery. https://machinelearningmastery.com/stacking-ensemble-machine-learning-with-python/#:~:text=The%20benefit%20of%20stacking%20is,single%20model%20in%20the%20ensemble.

Naresh K. (2019). Advantages and Disadvantages of Random Forest Algorithm in Machine Learning. The Professionals Point. http://theprofessionalspoint.blogspot.com/2019/02/advantages-and-disadvantages-of-random.html

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

Teemu, K. (2020) A Look at Precision, Recall, and F1-Score. Towards Data Science. https://towardsdatascience.com/a-look-at-precision-recall-and-f1-score-36b5fd0dd3ec#:~:text=Some%20advantages%20of%20F1%2Dscore,metric%20across%20positive%2Fnegative%20samples.