The University of Melbourne

COMP90049 Knowledge Technologies

Project 2

1. Introduction

This report is written for the second project of the subject knowledge technology. The purpose of this project is to analyse the sentiment of the tweets using different data mining approaches and acquire knowledge from the results. After the feature engineering, four machine learning approaches including Naïve Bayes, Decision Tree, Random Forest, and KNN are applied in this project. Then, a comparison of the effectiveness of each method will be present. Finally, the acquired knowledge from the analysis will be shown.

2. Tools and Dataset

2.1 Dataset

The tweet collections used in this project is a sub-sample gathered from Twitter and own by a research group (Rosenthal, 2017). There are around 23000 data points in the train.txt which is the training data for the models, and around 5000 data points in the dev.txt which are used to evaluate the models.

2.2 Weka

Weka is used in the project, which is a commonly used tool providing a number of machine learning algorithms.

2.3 Classifiers

In this project, four classifiers are chosen to compare with each other. The four classifiers are Naïve Bayes(NB), Decision Tree(DT), Random Forest(RF), and KNN. The classifiers are chosen from the ones introduced in the subject due to their interpretability.

3. Feature Engineering

3.1 Twitter Specific Features

To construct more effective features for the tweets, it is critical to notice that a single tweet is not just a plain text message, but also contains features other than plain words which may contribute to its sentiment.

3.1.1 Negation Words

The present of the negation word will change the opinion of the tweet entirely. For example, not good means bad. But the representation of negations words could vary, such as not, didn't, doesn't, which means that for each of the negation words, the frequency could be relatively low, and to capture all of them may increase the dimension of the model significantly.
Another property of the negations words is that the existence of double negations. For example, "I couldn't not help her" actually means "I felt I should help her".
To solve the problems mentioned above, a feature named "NEGATIONWORD" is constructed. First, the system was given a collection of hardcoded negation words. Then, instead of using the number of appearance of the negation words in the tweet, the modulo operation was applied to handle the double negation. The attribute NEGATIONWORD of tweet *t* is calculated out of the following formula.

$$NEGATIONWORD(t) = c \bmod 2$$

c indicating the number of negative words in the tweet.

3.1.2 Emoticons

Emoji and other symbol expressions such as ":)", ":(", are widely used now to represent emotions. A good news for sentiment analysis is that the emoticons tend to indicate obvious positive or negative emotions and the negative words will not be applied on the emoji and facial expressions, which means that these emoticons usually express strong and clear opinions than normal words.
In this project, two features are constructed to have a better use of the emoticons. They are called "POSIEMOJI" and "NEGEMOJI". The value of each attribute is the number of appearances of the respective emoticons in the tweet.

3.2 Preprocessing

Although the tweets used in the project has been preprocessed to remove less informative contents such as author, time stamp, etc., further preprocessing is still needed for the project.
In the system, the URLs, tags, and mentions will be removed. Also, most of the stop words such as 'the', 'a' will be removed due to limited information they provide. Another critical step is stemming. Stemming is an approach to assemble the strongly related tokens to the same type of token (Shirbhate and Deshmukh, 2016). For example, the present tense and the past tense of the same word usually represents the similar opinion.

3.3 Feature Selection

To select features that are more related to the sentiment of the tweets, a score imitating the degree of purity of the classification have been given to each of the features. The purity score of feature *f* towards class *c* can be calculated using the following formula.

$$Score(f,c) = p(c|t)$$

The system ranks all the features based on this score for three classes. And select the top 200 features from the result of class positive and class negative.

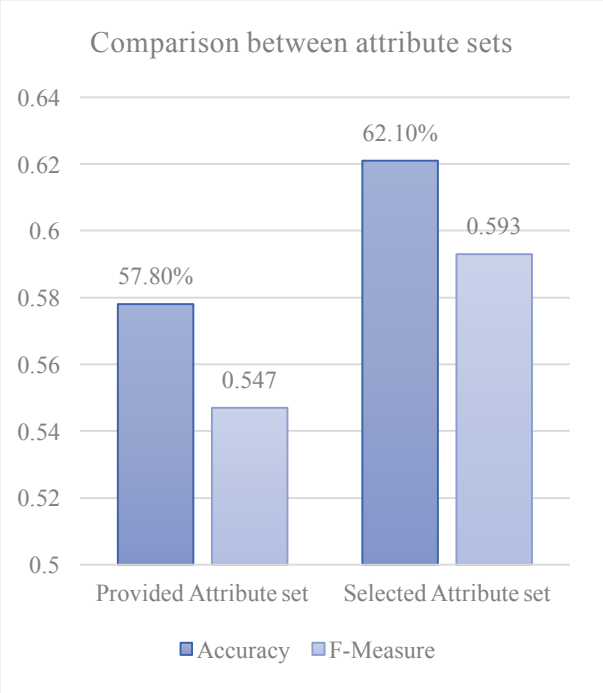

Figure 1: Comparison between attribute sets.

A comparison between the provided attribute set and the selected attribute set on the performance of the DT is shown in the Figure 1. It is easy to see that both the accuracy and the F-Measurement increased hugely in the selected attribute set. One of the reason is that in the provided attribute set, there are features representing the words like "at", "are", which may have high occurrence but barely contribute to the sentiment of the tweet. Another possible reason is that, the score function is basically assessing the purity of the classes, which would be beneficial for the DT to have a better performance.

4. Results and analysis

4.1 Results of classifiers

As mentioned above, the project chose four machine learning algorithms, including NB, DT, RF, and KNN. Specifically, for the KNN, the project use three nearest neighbours.
The main statistic results of the data are shown in table 1.

Table 1: Weka results of different classifiers

	NB	DT	RF	KNN
Testing Time(s)	2.02	0.24	3.15	217.72
Kappa	0.209	0.336	0.336	0.336
Avg. Precision	0.545	0.647	0.651	0.639
Avg. Recall	0.553	0.621	0.626	0.625
Avg. Accuracy	0.5534	0.6208	0.6265	0.6251
Avg. F-Measure	0.504	0.593	0.604	0.608
Avg. ROC area	0.679	0.679	0.73	0.73

The effectiveness measurements that will be used to evaluate the classifiers in the paper are Avg. ROC Area and Avg. F-Measure. The main reason is the imbalance of the dataset. Accuracy could be impacted significantly by the class distribution (Jeni, 2013). For example, a classifier who make wrong decisions to all instances of class A, and when class A is in the minority of the test set, the classifier could still get a high accuracy. On the contrary, the Avg. F-Measure and Avg. ROC Area can handle the unbalanced classes much better (Jeni, 2013).

A comparison of these measurements along with Accuracy between the classes are show as Figure 2.

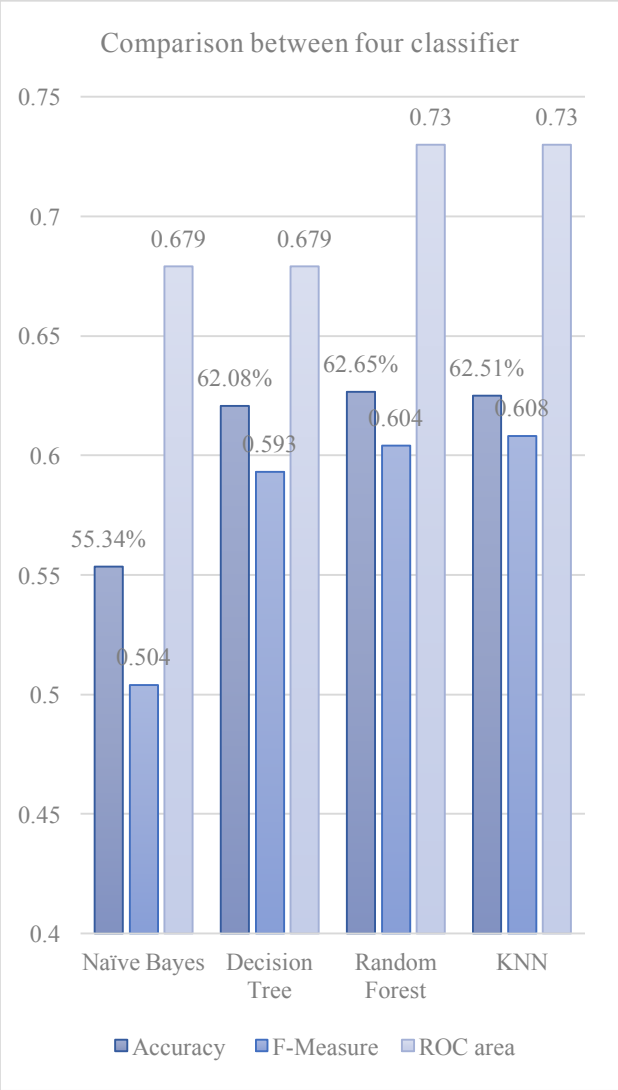

Figure 2: Comparison of main measurements between classifiers.

As can be seen from the Figure 2, NB behave badly in all three measurements. RF perform slightly better in F-Measure and Accuracy also have notably higher ROC area compare to DT. RF and KNN behave similarly with hardly noticeable differences.

4.2 Performance Analysis

NB is based on the assumption that the attributes are independent of each other, which is not true in this case. Another problem caused by the assumption is the zero-frequency problem. If a certain feature value has not occurred within a certain classification, which is quite common in sentiment analysis, the related conditional probability would be zero which will influence the performance of the classifier hugely.

DT performs better than NB mainly because as a rule-based classifier, it is able to handle the interacted features. Furthermore, redundant feature will not affect the accuracy of DT (Tan, 2006). On the other hand, main problem of DT is the over-fitting.

Due to the minor difference between the DT, RF and KNN, a detailed comparison between those classifiers have been show in the Figure 3. It compares on not only the Avg. F-Measure, but also on the F-Measure of each classes in the result.

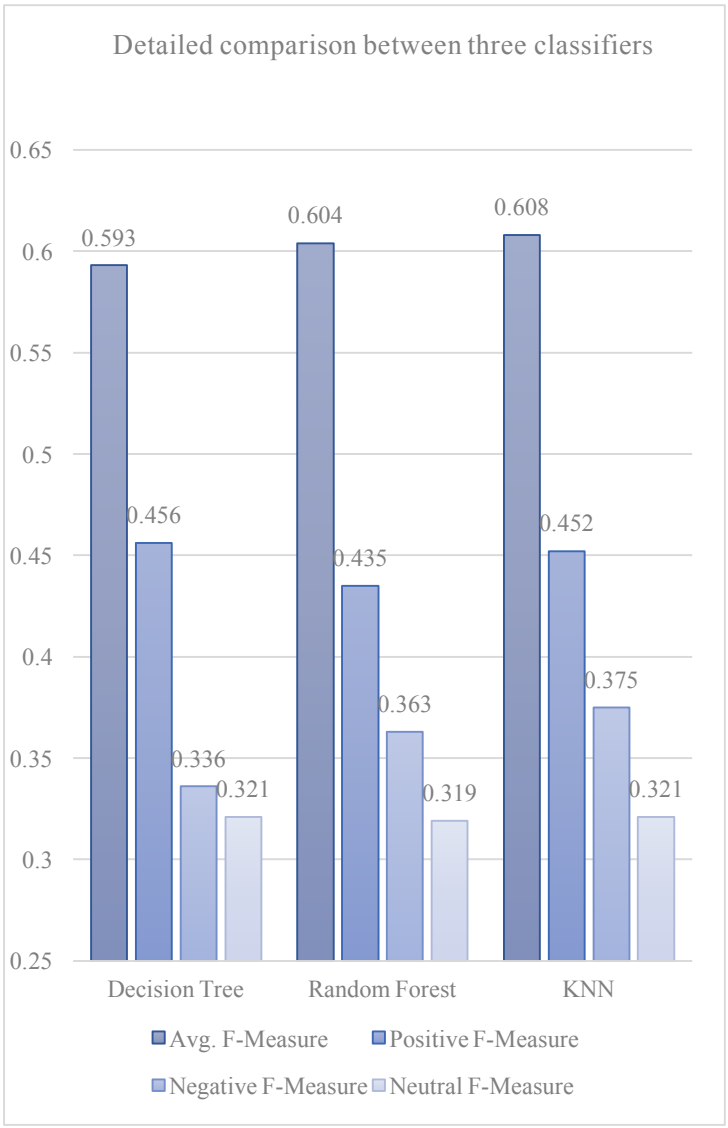

Figure 3: Detailed comparison between three classifiers.

RF using bagging to form multiple decision trees to overcome the over-fitting issue. It is obviously to see that in the Figure 3, the Negative F-Measure increased, although it is not shown in the Avg. F-Measure. Moreover, the Positive F-Measure of RF decrease notably compare to the DT. Because the stability of the decision tree decides the performance of the RF (Tan, 2006), it is fair to say that the decision tree is stable when identifying instances in class Positive. In other words, identifying Positive tweets generally needs more features than Negative tweets. That is the reason why the performance of classifying class Positive

dropped after bagging.

KNN perform slightly better than RF on most of the measurements. The reason is that as an instance-based classifier, KNN is able to generate more complicated decision boundaries than the rule-based classifiers. (Tan, 2016) As a lazy leaner, KNN will build the model only when the classification is required, which is not satisfied for the real time applications. In contrast, eager learners react rapidly once the training process is done.

5. Conclusions

To sum up, NB were found to behave worse than others after the comparison of their performances. RF and KNN are generally more effective than DT and NB, while the difference is that RF could be used in real time application due to rapid classification after training. Also, a feature selection process based on the purity of classification has been presented in the paper to derive a more sensible attribute set from the data.

References

Fawcett, T. (2006). *An introduction to ROC analysis. Pattern Recognition Letters*, 27(8), pp.861-874.

Jeni, L. A., Cohn, J. F., & Torre, F. D. L. (2013). *Facing Imbalanced Data--Recommendations for the Use of Performance Metrics*. 2013 Humaine Association Conference on Affective Computing and Intelligent Interaction.

Rosenthal, Sara, Noura Farra, and Preslav Nakov (2017). SemEval-2017 *Task 4: Senti- ment Analysis in Twitter.* In Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval '17). Vancouver, Canada.

Shirbhate, A. and Deshmukh, S. (2016). *Feature Extraction for Sentiment Classification on Twitter Data*. *International Journal of Science and Research (IJSR)*, 5(2), pp.2183-2189.

Tan, P., K. V, & S, M (2006), *Introduction to data mining*, 1st ed, Pearson Addison Wesley, Boston