Knowledge Technology Assignment2 Semester1

1 Introduction

Twitter has been widely used for a long time, using Twitter, people can post real-time reactions to 'everything' in characteristic short and simple manner of expression.

In this project, the assignment is to assess the sentiment of tweets using the given datasets. The problem is that there is no dictionary can show exactly sentiment of different words since the word might contain different sentiment in different words. Thus, in order to accomplish this goal, we should train the machine and let it 'memorize' that different sentiment classifications have different featuers.

This report will discuss two different classification models, Naive Bayes Models and Decision Tree Model, using engineered features.

2 Data-sets

All resource data-sets (train-tweets.txt, train-labels.txt, dev-tweets.txt, dev-labels.txt, test-tweets.txt, train.arff, dev.arff, test.arff) used in this report are given by Rosenthal and Nakov (2017).

File train-tweets.txt, dev-tweets.txt and test-tweets.txt contain the raw text of the tweets. File train-labels.txt and dev-labels.txt contain the manually assigned sentiment labels. File train.arff, dev.arff and test.arff combine the information contained within the previous two files.

3 Feature Engineering

In this project, the given feature does not give an ideal prediction precision. Therefore, a new set of features should be build. Based on the observation of all txt datasets, I find that there are some words which are very common (for example, "the", "a", "is" in English), which means these words has little useful information about the actual content of the tweets. Therefore, if I use those data directly to select features, those frequently occurring words might cover the words that has less frequency but more meaningful. Thus, before selecting the feature, the first thing to do is deleting those kinds of words. Since computer does not care about the integrity of the sentence, I can use this purer tweets to select feature. In this project, I care about both term frequency and inverse document frequency instead of only words frequency, so using TF-IDF statistical approach can help to select feature.

For programming, I choose TfidfVectorizer() function to get the feature matrix. To get a better result, some relevant parameter should be changed. The first one is 'max_df', I set it as 0.99, which means ignoring terms that appear in more than 99% of the documents, and set 'min_df' as 10, which means ignoring terms that appear in less than 10 documents. Another important parameter is 'max_features', which control the numbers of output features. Additionally, 'tokenizer' is also important, and it contains the rule of preprocessing the documents, and I write my own tokenizer in the program.

4 Classification Models

In this section, the report will discuss different Classification Models using engineered features.

4.1 Naive Bayes

Nave Bayes Model assumes that the features are independent of each other, but in real life, this assumption is basically not valid. Therefore, if there are a strong correlation between each feature and non-linear classification problems, Nave Bayes Model will have a poor classification effect.

I select different size of feature list to test the performance of Nave Bayes Models, the performance is as follow:(see Table 1).

Table1 shows the performance of using Naive Bayes Model, the precision is changed with the number of features increasing. But when the size of feature reaches the particular number,

Features size	Percision
N = 10	24%
N = 100	59%
N = 1000	62%
N = 5000	63%
N = 10000	63%
N = 20000	63%

Table 1: Performance of Naive Bayes

Features size	Percision
N = 10	42%
N = 100	50%
N = 1000	54%
N = 5000	57%
N = 10000	57%
N = 20000	57%

Table 3: Performance of Decision Tree

the precision will not change. I think the reason is that, in this project, 5000 features already have the ability to representative the characteristics of given tweets. Additionally, when the size of features is small, the precision increases sharply with the increasing of the size of features, which means, if there is not much data, this model will get better performance than many complex models, because the complex model with too many assumptions is more difficult to fit.

Negative	Neutral	Positive	Classified as
459	525	54	Negative
255	1876	269	Neutral
30	720	738	Positive

Table 2: Confusion Matrix when feature size is 5000

As we can see in the table2, the precision of predicting 'Neutral' correctly is higher than predicting others. Based on the observation of train-labels.txt, I find that the number of 'neutral' terms are much more than the number of the other two terms, which might make the machine learn more features about 'neutral' terms and less features about other terms. Thus, the probability of predicting neutral terms correctly is higher than predicting others.

4.2 Decision Tree

Decision tree is more in line with human thinking mode, which accounting to determine yes or no to decide to enter other judgments.

To test Decision Tree Model, I select different size of feature list to test the performance of Decision Tree Model, the performance is as follow:(see Table 3).

Table3 also shows that the precision of correct prediction will reach a highest point when the size of features reach a particular number. But we can see that, when the size of features is quite small, the precision of correct prediction in Decision Tree is much higher than that precision in Naive Bayes. The reason is that Naive Bayes Model predicts sentiment based on probability. Based on Bayes theorem, the probability will change a lot when the sample size increases from a small number. Therefore, when the feature size is small, the precision is quite low by using Nave Bayes Model. However, when the size of feature is quite large, Decision Tree will make probelm, since Decision Tree builds the tree in memory, so when the size of feature is large, it will take many time to build tree.

Negative	Neutral	Positive	Classified as
482	444	112	Negative
428	1507	465	Neutral
115	556	817	Positive

Table 4: Confusion Matrix when feature size is 5000

Table4 shows the approximately same result of Table2, which is that the precision of predicting 'Neutral' terms is higher than predicting others, but that precision in Decision Tree is lower than the precision in Nave Bayes Model. I think the reason is that Decision Tree Model is instability, which means, if the value in the tree changed a little, the whole tree will change, which will totally change the prediction.

5 Conclusions

In conclusion, I have using two different Classification Models to accomplish the assignment. In both model I find that the size of feature affect the performance of the models, but when the size of feature reach a particular number, the precision of those models will not change, and I believe that if the size of feature continues increasing, the precision might decrease. The reason is that machine in the learning process is unable to distinguish between local features

and global features, so when machine accomplish learning, except learning the global features of the data, it also learns part of the local features. Therefore, when new sample which only contains global features comes, the probability that machine with many local features can identify this sample correctly will get lower. I also find that when number of features of sample is quite samll but sample size is large, Naive Bayes Model does not work well.

References

Noura Farra Rosenthal, Sara and Preslav Nakov. 2017. Semeval-2017 task 4: Sentiment analysis in twitter. *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval 17)*.