Implementation

Representation of text / Preprocessing:

I have started by using a multinomial naïve bayes classifier, this means the frequency of occurrence of each word will be recorded instead of a Boolean expression.

I have also decided to use all words instead of top 1000 words so that some informative words are not missed out.

For the multinomial version, I created 4 dictionaries based on the corresponding class. Each dictionary has words that occurred as key and frequency of occurrence as value.

For the complement version, I created one dictionary of dictionary where each word is a key and values are another dictionary that has 4 keys corresponding to 4 classes and the value is the count of this word occurring in document of each type.

It was found that removing common words had no significant effect on performance, therefore we did not remove the common words.

Using logarithm:

To handle the underflow problem (multiplying many small floating numbers and receive 0), I have decided to apply a simple log-transformation to the probabilities and the final probability will be the sum of all log-probabilities. The final predicted value will be the class with highest log-probability.The log transform is also necessary later on when I implement the complement version of naïve bayes classifier.

Laplacian smoothing:

There will be words that are not being recorded in the frequency dictionary (i.e. with a frequency of 0). Since we cannot log-transform 0 and having 0 multiplied by other probabilities will leave everything as 0. I decided to add 1 to all word frequencies so that we have a minimum frequency of 1 and there will be no 0 probabilities.

Method extension

TF-IDF Document frequency normalization:

It is obvious that if a word appears in every text, it is useless for prediction. Whereas high informative words only appear in small number of texts. A TF-IDF document frequency normalization was used such all word frequency has been multiplied by the IDF value of that word. Some rare words would have higher IDF value and common words have lower IDF value. This improvement is aimed to increase the weight of those rarely occurred words so that we do not miss any important information.

Complement Naïve bayes

The training dataset we received is heavily imbalanced. There are 2144 rows of class e, 1602 rows of class b and only around 100 rows for class a and class v. From "Tackling the Poor Assumptions of Naive Bayes Text Classifiers" We know that a complement naïve bayes would perform better than a multinomial naïve bayes on some imbalanced dataset. Therefore, we have attempted to implement a complement version of naïve bayes classifier.

Evaluation

I have split the training data into training and validation set. The size of training set is 70% of all data and 30% of data are used for testing.

For the multinomial version: We obtained an accuracy of 0.97775 when the entire dataset was used and 0.94 on the validation set. We obtained an accuracy of 0.9533 on Kaggle.

For TF-IDF improved version, we obtained an accuracy of 0.99025 when the entire dataset was used and 0.944 on the validation set. We obtained an accuracy of 0.94 on Kaggle.

For the complement version, We obtained an accuracy of 0.99225 when the entire dataset was used and 0.948 on the validation set. We obtained an accuracy of 0.96 on Kaggle, this is our best score.

Lastly, for the complement + TF-IDF version we obtained an accuracy as high as 0.9975 when the entire dataset was used and 0.954 on the validation set. However, we only obtained 0.9566 on Kaggle, which is a very minor improvement from the multinomial version.

Conclusion

It is clear that any implementation/extension of naïve bayes classifier preforms far better than the ZeroR model (predicting everything as the majority class) which has an accuracy of 51.3% on Kaggle.

All of the extensions had better performance than the standard naïve bayes classifier which had an accuracy of around 92%.

I found that the complement version had a significant improvement than the multinomial version, this is not a surprise because we did have very imbalanced class.

After also using TF-IDF (Document frequency normalization) on the complement version, the performance on the validation set was increased very slightly. The effect of Document frequency normalization was rather insignificant on the unseen dataset, but it does have a significant effect on the training dataset.

Overall, the complement + TF-IDF version did have a noticeable higher accuracy than standard naïve bayes model and a much higher accuracy than the majority model. However, the effect of TF-IDF was not obvious, it is possible that this method caused some overfitting.

Our best score was 0.96 on Kaggle, from the complement class version.