Sentiment Analysis of Tweets via Binary Classification

Liam Heeger and Jungmin Park
Course: CSC 240 (Data Mining)

***Abstract*— This project report focuses on the data mining techniques to analyze the sentiment of Twitter text data. The classification models generated for this report have been analyzed on a dataset from Kaggle which contains 1.6 million tweets extracted using the Twitter API. Each entry in the dataset is one tweet each of which is annotated with a binary sentiment.**

I. Introduction

The work completed for this project addressed the problem of performing sentiment analysis on tweets. Sentiment analysis is an automated data mining process to analyze text data for the attitude of the written text. In this case due to the formulation of the dataset, our methods perform prediction for positive or negative attitude in a tweet. Sentiment analysis is widely used in business for gaining insights from social media comments to inform data-driven decisions from marketing campaigns to political campaigns.

The focus of this report will be on various classification models to identify sentiment in text data, in particular tweet data. The following section will outline, in detail, the problem which we addressed. The remainder of the report will discuss the dataset and its properties, the techniques needed to preprocess and clean the data, and descriptions of the models used for sentiment analysis and classification. The report concludes with the authors' notes for future expansion of this work and some closing remarks on the project.

II. Problem Statement

The problem addressed in this work was to classify a given tweet from the dataset as negative or positive with high accuracy. Figure 1 shows some examples tweets and the associated sentiment.

The approach to solving this problem was to generate a suite of classifiers. Each of these classifiers examines individual words in the tweets, producing a prediction based on the trained model.

I **love** data mining. It is a **fascinating**!
It is **raining** outside. Guess I am staying **inside.**

Fig. 1. Two tweets with positive and negative sentiment respectively. Words that are bolded strongly indicate the possible sentiment of the tweet.

Through analyzing the sentiment of words in a tweet, the classifier predicts the overall sentiment of the tweet. Several classification models were created, tested, and analyzed using different algorithms to accurately predict sentiment.

III. Dataset Description

This project utilized a dataset containing 1.6 million tweets extracted using the Twitter API and annotated with a given sentiment. The authors of the dataset derived the sentiment for each tweet by making the assumption that the presence of a positive (`:)`) or negative (`:(`) emoticon was a suitable indicator for the sentiment of the tweet.

The dataset has 6 columns: `target`, `id`, `date`, `flag`, `user`, and `text`. From these 6 columns, the focus of the analysis and classification was primarily on two of these columns: the `target` column and `text` column. The `target` column contains the sentiment of the tweet and have been annotated as 0 for negative and 4 for positive. The official description of this dataset claimed that a value of 2 indicated neutral sentiment but this was unused within the tweet data. The `text` column contains the content of the tweets that some user has written. The dataset creators have stripped the emoticons as this would defeat the purpose of pursuing this analysis.

IV. Data Preprocessing

The following section is a description of the data preprocessing performed on the dataset.

A. Sentiment Re-scaling

The sentiment value given in the dataset is indicated by 0 or 4 for negative or positive respectively. This is not desirable for classification. Therefore, this value is re-scaled so that negative has sentiment -1 and positive has sentiment 1.

B. Text Cleaning

The most involved preprocessing step was text cleaning. This is due to the fact that the original tweets are unfiltered, misspelled, streams of consciousness. The following steps outline the order and details of the various text cleaning procedures. An example of this result of this process at each step is shown in Figure 2.

1) **Force lowercase:** All tweets are made lowercase so that words that are the same but different case are the same.
2) **Remove hyperlinks:** A regular expression captures hyperlinks in the tweet and removes them. Hyperlinks are unnecessary in understanding the sentiment of the tweet.
3) **Remove HTML entities:** Another regular expression captures HTML entities, such as "`&and;`" to represent "&". These artifacts do not contribute meaningfully to the sentiment analysis.
4) **Remove repeated characters:** Repeated characters in a tweet or word are sometimes used to denote exaggeration, such as saying "farrrrr awaaaayyy" which is just "far away". The text cleaning process removes any excess repeated character which appears 3 times or more consecutively. It is assumed that once the word has 2 occurrences of the same letter, such as "farr awaayy", the spell checking step will automatically correct this to right word.
5) **Remove punctuation:** Punctuation was not considered when analyzing the sentiment, so it is removed entirely.
6) **Remove hashtags and handles:** Although analysis on the contribution of hashtags and handles to the sentiment of tweets was performed, it was excluded from the final results. A simple regular expression is used to extract words that start with "@" or "#" to clean out the handles and hashtags.
7) **Replace internet slang:** Internet slang like "lol" or "fomo" which abbreviate "laughing out loud" and "fear of missing out" respectively, contain words in their expansion which more clearly indicate the sentiment of that term. Therefore, a database of these internet slang terms were identified and used to remap these terms with their corresponding expansion.
8) **Correct spelling:** Words within the tweets in this dataset are frequently misspelled, enough so to justify correcting the spelling of all of the tweets. A library was used to perform the spelling correction.

Hennnlo & Check out haha.com lol #sofunny
1) hennnlo! & check out haha.com lol #sofunny
2) hennnlo! & check out lol #sofunny
3) hennnlo! check out lol #sofunny
4) hennlo! check out lol #sofunny
5) hennlo check out lol #sofunny
6) hennlo check out lol
7) hennlo check out laughing out loud
8) hello check out laughing out loud

Fig. 2. An example of the text cleaning process and the result of each step.

C. Tokenization

Tokenization is the process which converts sentences into lists of words in order. In this case, the tweets' words are split by the whitespace into a list of words, or tokens.

D. Stopword Removal

Stopwords are very common words in the English language which are useless for classification and increase dimensionality needlessly. Examples of stopwords may be: is, the, or, from, and but. Stopwords are removed after tokenization by finding and removing a fixed list of predefined stopwords.

E. Vectorization

Vectorization is the process which takes as input the lists of tokens and maps each unique word within the token lists to an integer, making the classification process much simpler.

F. TF-IDF Evaluation

TF-IDF evaluation is the process which scores each word in each tweet by their importance in that tweet. TF-IDF stands for term frequency-inverse document frequency and is a product of those two components. This product is defined informally below.

$$TF = \left(\frac{\text{\# appearances of a term in this document}}{\text{\# terms in this document}}\right)$$
$$IDF = \left(\frac{\text{\# documents with this term}}{\text{\# documents in total}}\right)^{-1}$$
$$\text{TF} - \text{IDF} = TF * IDF$$

This score is used in classification to differentiate important terms from less important terms which may not affect sentiment as much.

V. Classification Models

As part of this project's implementation, several standard algorithms were generated to classify the sentiment of the tweets.

A. Linear Classifier

Two variants of linear classifier were implemented for sentiment analysis. The first was a logistic regression classifier and the other was a stochastic gradient descent classifier.

B. Naive Bayes Classifier

The naive Bayes classifier which was implemented was conditioned on the multinomial distribution. This type of classifier is known to perform well when the system we are predicting from has discrete features, such as we have from TF-IDF vectors for each tweet.

C. Decision Tree Classifier

The decision tree classifier modeled in this project followed a standard implementation of this kind of classifier. The splitting criteria chosen was Gini index and no restriction on the depth of the tree was stipulated.

D. Random Forest Classifier

Similar to the decision tree classifier, the random forest classifier builds an decision trees which act as estimators. For the experiments performed, 1000 was chosen as the estimators to allocate for this classifier. Each of these estimators has the Gini index splitting criteria and no restriction on the depth of the tree was stipulated.

E. K-Nearest Neighbors Classifier

The instance based classifier that was implemented utilized the k-nearest neighbors algorithm. Determining the distance to neighbors for a sample was calculated using Minkowski distance.

F. Neural Network Classifier

A long-term short-memory based neural network architecture was implemented to perform classification. This model is outlined in detail in the following section (section VI).

VI. Neural Network Model

The most successful classifier model generated as a result of this project was the neural network approach. This model utilized a long-term short-memory (LSTM) layer as its centerpiece. These kinds of networks are very powerful for text analysis and time series prediction models, which inspired its use in this project. The architecture takes a vectorized tweet as input and passes it to an embedding layer, followed by dense layer and then the LSTM layer. Following the LSTM layer is another another dense layer and lastly a single neuron output giving the predicted sentiment. A diagram showing the model architecture can be found in section VI.

VII. Results

The primary metric collected during testing for each classifier was accuracy. Excluding the neural network model, which was tested using a different procedure, each classifier ran on 10 tests set sizes from 10 percent to 90 percent test fraction. Each test fraction for each classifier was ran 10 times to get a mean and standard deviation of the accuracy. Plots for accuracy of each classifier are shown in Figure 4. Receiver operator characteristic (ROC) curves, are also generated for each classifier in Figure 5.

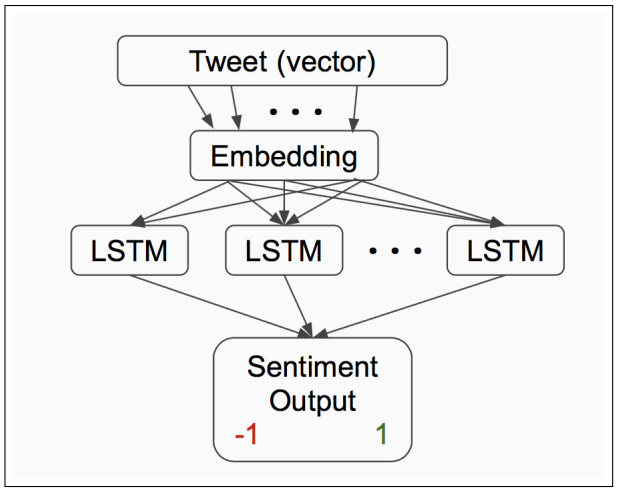

Fig. 3. A diagram displaying the architecture of the neural network classifier implemented.

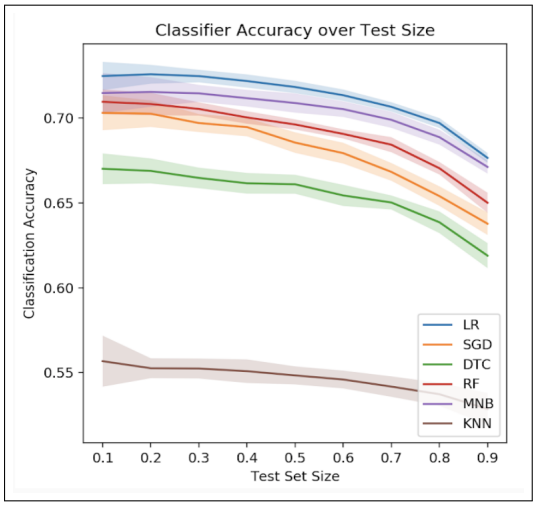

Fig. 4. Classification accuracy for a range of test fractions for each of the various classification methods.

The neural network classifier was tested at a 20 percent test fraction with a final accuracy of 83 percent. Performance of this classifier was the best, exceeding the other models by around 10 percent in accuracy. The neural network training procedure stops early prior to over-fitting. A training curve for this model is in Figure 6.

VIII. Future Work

For the future work, refining classifiers to get better accuracy can be considered since the classification accuracy achieved might not be optimal. Another future task that can be done for sentiment analysis is would be to examine the sentiment changes over time. The graph in Figure 7 is the time series plot of sentiment changes throughout days from Monday to Sunday.

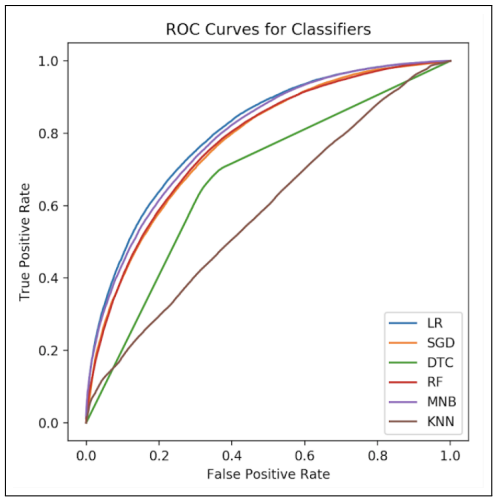

Fig. 5. Receiver operator characteristic (ROC) curves for the various classification methods.

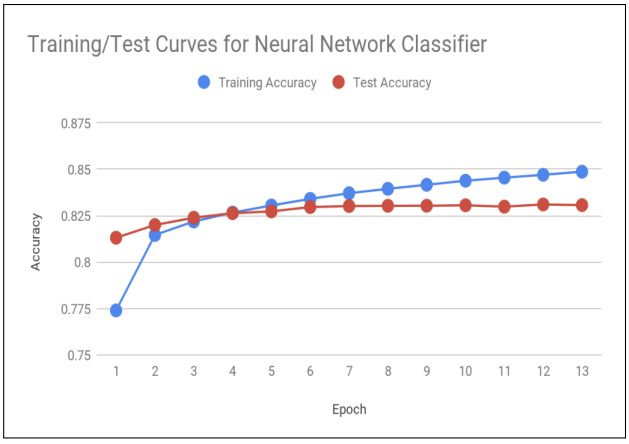

Fig. 6. The training and test accuracy for each epoch of training for the neural network model.

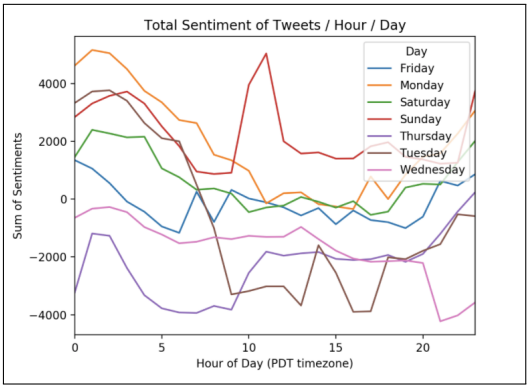

Fig. 7. A time series plot showing the total sentiment for all tweets for each hour. Each line corresponds to a day within a week.

Based on the results from the graph shown, Tuesday shows the most negative sentiment on average. Conversely, Sunday shows the most positive sentiment on average. This time series analysis can also be conditioned on each user to predict the user's future sentiment. Lastly, finding association rules which connect highly negative or positive words to less frequent terms can be helpful to further analyze sentiments that are unsure and harder to predict. This work can be used to to aid in more accurate overall sentiment prediction.

IX. Conclusion

This sentiment analysis project implemented various classification techniques using different classifier model algorithms that effectively predict the overall sentiment of the given tweets as positive or negative. The optimal classification model that has shown the best performance was neural network classifier. This exploration of sentiment analysis classification has brought a lot of insights of what kinds of models are good at predicting sentiment of tweets.

References

[1] Go, Alec, Richa Bhayani, and Lei Huang. "Twitter sentiment classification using distant supervision." CS224N Project Report, Stanford 1, no. 12 (2009): 2009.

[2] KazAnova. Sentiment140 Dataset with 1.6 Million Tweets. Kaggle, September 13, 2017. https://www.kaggle.com/kazanova/sentiment140.

[3] Smetanin, Sergey. Sentiment Analysis of Tweets Using Multinomial Naive Bayes. Medium. Towards Data Science, September 1, 2018. https://towardsdatascience.com/sentiment-analysis-of-tweets-using-multinomial-naive-bayes-1009ed24276b.

[4] Agrawal, Samarth. Sentiment Analysis Using LSTM Step-by-Step. Medium. Towards Data Science, February 21, 2019. https://towardsdatascience.com/sentiment-analysis-using-lstm-step-by-step-50d074f09948.

[5] Chablani, Manish. Sentiment Analysis Using RNNs(LSTM). Medium. Towards Data Science, July 22, 2017. https://towardsdatascience.com/sentiment-analysis-using-rnns-lstm-60871fa6aeba.

[6] Taiuti, Alberto, Alberto Taiuti, Matt, Matt, Franco, Franco, Vijay R Sathish, et al. Twitter Sentiment Analysis Using Combined LSTM-CNN Models. Bsides, February 20, 2018. http://konukoii.com/blog/2018/02/19/twitter-sentiment-analysis-using-combined-lstm-cnn-models/.

[7] Jones, Anna Bianca. Sentiment Analysis of Reviews: Text Pre-Processing. Medium. Medium, March 15, 2018. https://medium.com/@annabiancajones/sentiment-analysis-of-reviews-text-pre-processing-6359343784fb.