The Neutral Problem: Twitter Sentiment Analysis Using Machine Learning

Nathan Neeley
Department of Computer Science
Kennesaw State University
Kennesaw, GA
nathanneeley94@gmail.com

Nathan Neeley
Department of Computer Science
Kennesaw State University
Kennesaw, GA
nathanneeley94@gmail.com

Nathan Neeley
Department of Computer Science
Kennesaw State University
Kennesaw, GA
nathanneeley94@gmail.com

ABSTRACT

Social media platforms have become a huge medium for companies, consumers, and individuals to express their opinions about a particular product, person, or company. It has come to influence our politics and consumer interests. Because of the increased reliability on social platforms, it is important to create techniques to analyze the sentiment of peoples' opinion with a relatively accurate degree of reliability. In my project, I will be conducting a sentiment analysis of tweets from Twitter. The goal here is to use machine learning to analyze the sentiment of tweets and discuss the problem with classifying neutral sentiment.

CCS CONCEPTS

• Computing methodologies • Applied computing • Software and its engineering

KEYWORDS

Machine learning, Sentiment analysis, Regular expression, Naïve Bayes

ACM Reference format:

FirstName Surname, FirstName Surname and FirstName Surname. 2018. Insert Your Title Here: Insert Subtitle Here. In *Proceedings of ACM Woodstock conference (WOODSTOCK'18). ACM, New York, NY, USA, 2 pages.* https://doi.org/10.1145/1234567890

1 Research Statement and Conjecture

Social media platforms have become a huge medium for companies, consumers, and individuals to express their opinions about a particular product, person, or company. It tends to influence our politics and consumer interests. It has come to influence buying habits and voting habits so much that companies and politicians have chosen to gear their products and voting habits around social platforms. Because of the increased reliability on social platforms, it is important to create techniques to analyze the sentiment of peoples' opinion with a relatively accurate degree of reliability. In my project, I will be conducting a sentiment analysis of tweets from Twitter that can be geared around a particular topic, such as a politician (e.g. Donald Trump), a company (e.g. Target), or a consumer product (e.g. iPhone). The goal here is to use machine learning to analyze the sentiment of tweets and discuss the problem with classifying neutral sentiment.

Prior research mostly deals with classifying text into positive and negative sentiments, reaching accuracy of 85% to 90% and upward. A dataset of airline tweets was chosen, which contain 2928 tweets, labeled as positive, negative, and neutral. This research shows that positive and negative sentiment are easier to classify and neutral sentiment, such as an Ad, is more difficult for machine learning algorithms to classify. I tweaked the preprocessing and vectorization to improve the accuracy of the model from 70% to close to 80% and show here that the Naïve Bayes algorithm is a good model choice, and I will conclude with statistics from the model and discuss how to improve accuracy further, especially among neutral sentiment.

Many other algorithms were tried in the implementation but had downsides. *RandomForestRegressor*, *SVM*, and *Logistic Regression* were all implemented and are popular choices for sentiment analysis. Each one had their drawbacks. *Logistic Regression* was determined to have around the same accuracy as the Naïve Bayes model without being as fast performance wise, while *SVM* and *RandomForestRegressor* produced slightly better accuracy but at the cost of greatly decreased performance, enough to where the models were not practical. Naïve Bayes, on the other hand, offers good accuracy and performance. Clearly, we could extend this model to deep learning models and compare those as well, but this research only considers machine learning algorithms.

1.1 Related work

There has been a multitude of research done on the area of sentiment analysis, using deep learning and machine learning models to create usable systems in a growing world built around online opinion.

M.S. Neethu et. al [1] analyzes the sentiments of Tweets similarly to how I am in this paper. Their literature review focuses on machine learning techniques, processing of text, sentiment analysis, twitter research to present a representative corpus of prior research. They formed their Twitter datasets by extracting posts about electronic products. They explain that they perform a sentence level sentiment analysis by performing preprocessing first, creating feature vector second, and then finally using different classifier to group tweets in positive and negative classes. They experimented with multiple classifiers, including *Naïve Bayes Classifier*, *Maximum Entropy Classifier*, and *SVM Classifier*, to compare results. It determined that Naïve Bayes had comparable accuracy between the models. They had difficulty dealing with the preprocessing of tweets just as I did in my research, due to the misspellings, syntax, and slang words.

1.2 Methodology

After choosing my topic for the source of my research, I decided on a dataset. Many of the Twitter datasets I noticed only had positive and negative sentiments text. I thought this was odd because tweets are going to include a good number of ads and other indifferent sentiments on a topic, so to make the research more usable and to filter out the ads, neutral sentiment also had to be considered. That is when I found the airline sentiment dataset. It contains 2928 tweets all relating to airlines.

For this analysis, all I was concerned with was the text and the sentiment, and the algorithm would predict the label with the training and testing datasets without additional features. Additional features could have been added to improve accuracy, but that would increase performance time. Performance time was carefully considered in this analysis because for this analysis to be effective, it must be able to determine sentiment relatively quickly because there are so many tweets on a particular subject when applying this analysis to new tweets. If the model takes too long, the implementation is obsolete and cumbersome to apply to any real-world analysis.

Below are steps I went through when developing the implementation in Python and the code for the Naïve Bayes training and testing used in my implementation:

1. Read dataset into dictionary data type where columns are keys (Only text and sentiment are important)
2. Preprocess Text (Covered below)
3. Split dataset into training and testing parts by X and Y
4. Transform X's (training and testing) into matrix of token counts and removes stop words
5. Train Naïve Bayes model on dataset (training), compute accuracy of label recognition, print classification report, and display confusion matrix
6. Test Naïve Bayes model on new unseen data (testing), compute accuracy of label recognition, print classification report, and display confusion matrix

```
def NB_train(self):
    self.NB_model = ComplementNB()
    self.NB_model.fit(self.X_Tr, self.Y_Tr)
    self.NB_accuracy_Tr =
    self.NB_model.score(self.X_Tr, self.Y_Tr)
    plot_confusion_matrix(self.NB_model, self.X_Tr,
    self.Y_Tr)
    plt.show()

def NB_test(self):
    self.NB_accuracy_Ts =
    self.NB_model.score(self.X_Ts, self.Y_Ts)
    plot_confusion_matrix(self.NB_model, self.X_Ts,
    self.Y_Ts)
    plt.show()
```

When first developing the implementation, the testing accuracy was lower in the high 60s to low 70s percent range, but once I started adding multiple ways to preprocess the data, the accuracy increased to high 70%.

Processing text from Twitter is difficult because of the syntax and slang that is used. To reduce some of the complexity of Twitter's text from our algorithm, we use preprocessing to simplify or remove unnecessary text. Below are the steps I went through when preprocessing the data:

1. Lower case all text to make it uniform
2. Expand all contractions
3. Remove all urls (regular expression): r"((http://)[^]*|(https://)[^]*|(www\.)[^]*)"
4. Remove all usernames (starting with @) (regular expression): "@[^\s]+"
5. Remove all punctuations
6. Convert all words to stem words, reducing the dictionary size

This preprocessing section is paramount to producing the most accurate sentiment analysis.

1.2.1 Naïve Bayes Classifier. The naïve bayes classifier takes all the features in the vector and uses them as independent of one another, assuming that none of them is related. Assuming that all features are equally independent is the main axiom of the naïve bayes classifier. That is why it works well in our model, where only the text and sentiment are used in predicting class labels, finding correlations between the text and sentiment by applying the Bayes' theorem. The conditional probability equation for the Naïve Bayes classifier can be stated as such:

$$y = argmax_y P(y) \prod_{i=1}^{n} P(x_i|y)$$

'X' is the feature vector and x_i is each individual feature while y = is the class label. We could add additional features to the model that might be helpful, but minimal is typically better with the Naïve Bayes classifier given that each feature is considered independent of the other ones. Furthermore, each additional feature slows down the performance and may or may not improve the accuracy, depending on if the naïve bayes classifier finds the feature as important in calculating the conditional probabilities.

1.3 Experimental Design, Results, Analyses, and Comparison

After trying multiple classifiers to find the best suited one for this analysis, naïve bayes classifier's performance and accuracy outweighed the other algorithms. In Neethu et al. [1], they achieved an accuracy of 88.50% by using the naïve bayes classifier, which is higher than the other models. However, precision and recall were lower in the naïve bayes classifier, and they were not using classifying neutral sentiment at all, only positive and negative. As already discussed, neutral sentiment seems to produce a lower accuracy than positive and negative, at least based on my results.

Below are my results from the training and testing datasets from each sentiment as well as the confusion matrices for both datasets:

TABLE 1: Naïve Bayes Statistics of Training Dataset

	Precision	Recall	F1-Score	Tweets
Negative	91%	93%	92%	7289
Neutral	84%	72$	77%	2519
Positive	79%	88%	83%	1904
Accuracy			87%	11712
Macro Avg.	85%	84%	84%	11712
Weighted Avg.	87%	87%	87%	11712

FIGURE 1: Naïve Bayes Training Confusion Matrix

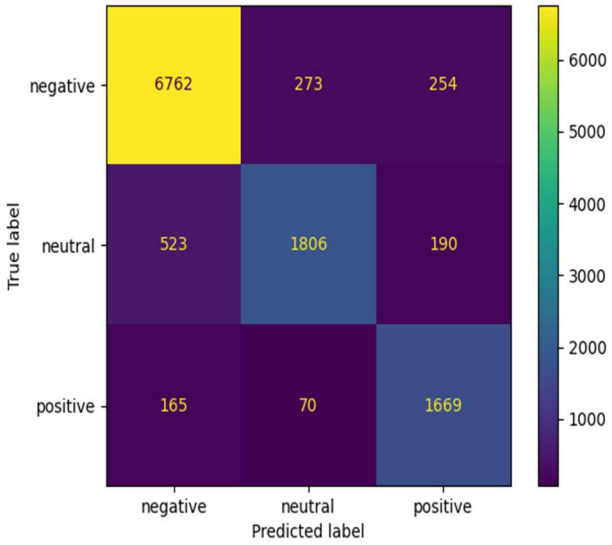
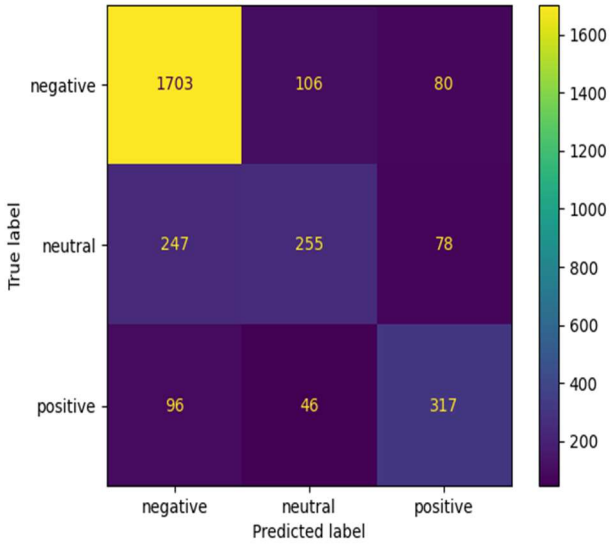


TABLE 2: Naïve Bayes Statistics of Testing Dataset

	Precision	Recall	F1-Score	Tweets
Negative	83%	90%	87%	1889
Neutral	63%	44%	52%	580
Positive	67%	69%	68%	459
Accuracy			78%	2928
Macro Avg.	71%	68%	69%	2928
Weighted Avg.	77%	78%	77%	2928

FIGURE 2: Naïve Bayes Testing Confusion Matrix



As you can see, accuracy is lower for neutral sentiment on both the training and testing datasets. It is negligible in the training dataset, but significantly lower in the testing dataset. At first glance, I thought it might be because of the disproportionate number of tweets of each sentiment, but that does not look to be the primary problem since positive has less tweets in both datasets and positive's accuracy is higher in both as well. Augmenting the dataset with additional tweets for positive and neutral should improve the accuracy based on the increased accuracy for the negative sentiment results. It produced a 92% accuracy on the training dataset and an 87% accuracy on the testing dataset, significantly higher than both positive and neutral sentiments in both cases.

After examining the model, it performs well on both training and testing dataset, producing similar results just a little higher across the board for the training dataset. The confusion matrices visualize the data clearer, illuminating the problem areas.

Let's consider the neutral sentiment a little closer. Since we determined augmenting the data does not seem like that would entirely fix the problem, we need to find the problem first to be able to fix it. When observing the charts above, precision for neutral is similar to positive, interestingly performing better for the training dataset, whereas recall performs significantly worst for both training and testing, calculated as 72% and 44%, respectively. This drop in recall means there is an abundance of false negatives. I believe this might be a result of there not being enough distinctive markers in neutral sentiment tweets that the model does not know how to label them. One fix for this is augmenting the data and see if neutral sentiment results improve. Another possible option would be to add more features to the model to see if that gives the classifier more distinctive markers to find.

1.3.1 Logistic Regression Classifier. Previously in this paper, I discussed how other models do not perform as well or are lacking in performance. To demonstrate this, I will show the results of the logistic regression classifier. Below are my logistic regression classifier results from the training and testing datasets from each sentiment as well as the confusion matrices for both datasets:

TABLE 3: Logistic Regression Statistics of Training Dataset

	Precision	Recall	F1-Score	Tweets
Negative	95%	98%	96%	7289
Neutral	91%	86%	88%	2519
Positive	93%	90%	91%	1904
Accuracy			94%	11712
Macro Avg.	85%	84%	92%	11712
Weighted Avg.	87%	87%	94%	11712

FIGURE 3: Logistic Regression Training Confusion Matrix

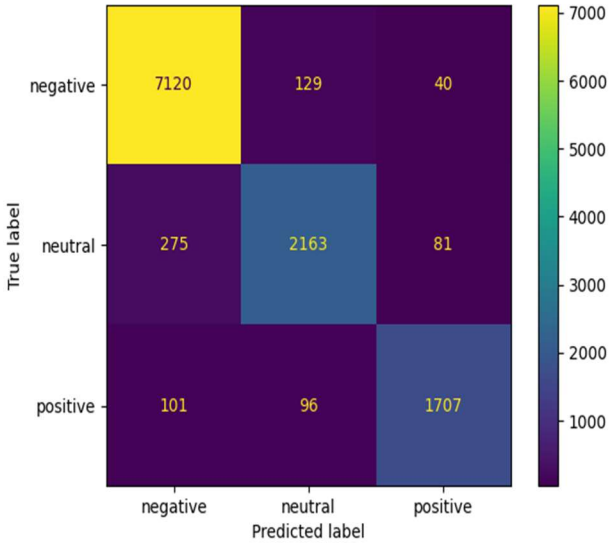
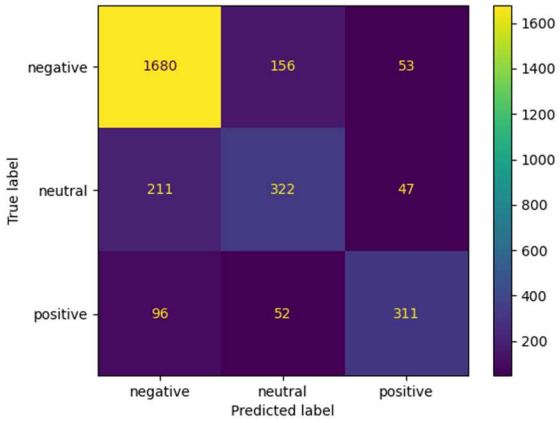


TABLE 4: Logistic Regression Statistics of Testing Dataset

	Precision	Recall	F1-Score	Tweets
Negative	85%	89%	87%	1889
Neutral	61%	56%	58%	580
Positive	76%	68%	71%	459
Accuracy			79%	2928
Macro Avg.	74%	71%	72%	2928
Weighted Avg.	78%	79%	79%	2928

FIGURE 4: Logistic Regression Testing Confusion Matrix



When first considering this data compared to the previous classifier's data, it appears to outperform the naïve bayes model, but there are two problem areas here. First, the testing data is comparable to the naïve bayes testing data, but the training data is significantly higher in almost every area, achieving over 90% most of the time. Then the model performs significantly worst on the testing data. This trend is a clear sign of overfitting. Second, the performance time for the logistic regression model is higher than the naïve bayes model. Both problems influenced my decision on which algorithm to choose. The other classifiers previously discussed improved accuracy but at the cost of a significant increase in performance time.

1.4 Conclusion

Sentiment analysis has become an important growing research area since sales, politics, and other areas have continued to be influenced by social media and online consumers' opinions. In this paper, I produced a usable, effective sentiment analysis that uses the naïve bayes classifier and was able to show case the problem when including neutral sentiments. I outlined my methodology and source code, so others can follow-up with my research and add their own ideas to it to make it fit their needs. Naïve bayes classifier also proved to be effective in performing sentiment analysis. For future work, some of the ideas proposed in the results section could be implemented to improve outcomes, such as augmenting the dataset and adding additional features to the model. This research built on the previous corpus by discussing the neutral sentiment problem regarding tweets. For many areas of sentiment analysis, such as consumers' reviews, there would not be any need to include neutral sentiment, but in places such as social media where there are ads, neutral sentiment is important to cipher through the junk data. By presenting a thorough walk through of my research and methodology, other researchers can take my data and methodology and make it their own for future investigations.

REFERENCES

[1] M. S. Neethu and R. Rajasree, "Sentiment analysis in twitter using machine learning techniques," *2013 Fourth International Conference on Computing, Communications and Networking Technologies (ICCCNT)*, Tiruchengode, India,