# Reddit WallStreetBets Sentiment Analysis

## Artificial Intelligence – COMP.4200-201

Eddie Tran – Eddie_Tran@student.uml.edu – 01731595

## I. ABSTRACT

Some people rely on artificial intelligence techniques to make sense of stock market data. I attempted to apply some of these artificial intelligence techniques to data that is collected from the Reddit WallStreetBets community. I applied data analysis, prediction, and more to make use of the collected data. In this report, I discuss my steps taken to reach the results that I have achieved as well as the results themselves. The results may be beneficial for those that wish to make it big within the stock market.

## II. INTRODUCTION

Over the course of a semester, we were exposed to many different artificial intelligence techniques used for either classification, regression, or both. In this report, I test my knowledge and applied techniques that will help us identity the proper model to best fit the dataset.

## III. CLASSIFICATION DATASET

### *A. Dataset A*

The classification dataset, Reddit WallStreetBets Posts, from Kaggle is a dataset with the dimension of [53187, 8]. This dataset provides a mixture of titles, scores, IDs, URLs, number of comments, when it was created, the body and timestamps. As the dataset is processed, I have come to conclude that the value of titles reduces to 1,111.

### *B. Dataset B*

The dataset that I used to process dataset A, NYSE Tickers, from an online stock ticker database is a dataset with the dimension of [3497, 7]. This dataset provides a mixture of stock symbols, dates, open, high, low, close, and volume. As the dataset is processed, I have come to conclude that the value of symbols will reduce to [3497,1]

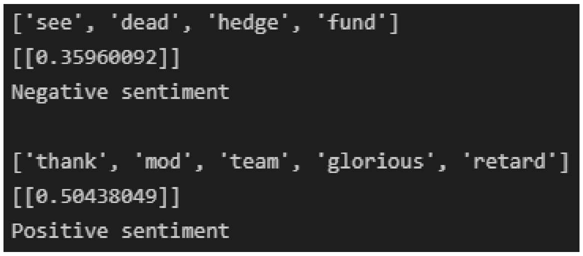
### *C. Data Processing*

The Reddit dataset does not determine whether each Reddit title is a positive, neutral, or negative sentiment so I must find these values myself. First, I must clean the text by removing any symbols, emojis, lowering the text, etc. to get the most appropriate results. I also cleaned up the NYSE dataset as well for later use. Next, I applied varderSentiment library to find the sentiment values and determined whether the titles were negative, neutral, or positive. After, I removed all neutral cases which is undesirable in our research study. I applied another type of cleaning, removing any stop-words and lemmatization. Now this is where the second dataset comes into play. I use the NYSE dataset to keep the Reddit titles that only have stock tickers for better results when applying the artificial intelligence techniques. Finally, we convert the titles back to strings and convert negatives to 0s and positives to 1s. With this, we can continue onto our classification techniques.

### *D. Logistics Regression*

My first algorithm is Logistic Regression which predicts a binary outcome based on a set of independent variables. With our processed dataset, I applied train_test_split to split our dataset into testing and training with a 20:80 ratio, respectively. Since, we needed to create our classification techniques by scratch, I found the frequencies of words found within my training sets first. This will be used to train the testing sets. Then, I defined our sigmoid function, gradient descent function, as well as extracting the features out of our titles. With all the current functions, I trained the model and resulted in the cost of training and the weight vectors. These weights will become the probability that will be used in prediction. Finally, I tested our model and found that the accuracy of the model is 91.93%, essentially 92%.

The accuracy of the Logistics Regression model is great as it is not too high, which means that overfitting has not occurred. In Figure 1, display the results of the model after being applied to a couple of Reddit titles. I was unfortunately unable to create a confusion matrix for Logistics Regression. However, we can use the accuracy to determine whether the model best fits our dataset, Reddit WallStreetBets, at the end.



```
['see', 'dead', 'hedge', 'fund']
[[0.35960092]]
Negative sentiment

['thank', 'mod', 'team', 'glorious', 'retard']
[[0.50438049]]
Positive sentiment
```

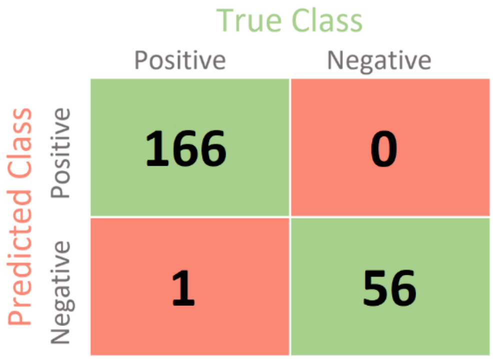
*Fig. 1. Logistic Regression on Reddit Titles Results*

### *E. Naïve Bayes Gaussian*

With Naïve Bayes Gaussian method, I used the code that our professor provided. I noticed that the sentiment scores were used to determine the resulting sentiment values. I used those scores to be applied in the prior and likelihood functions.

My results could not be converted to a confusion matrix via code, so I calculated it by hand by comparing the true values, y_test, and the predicted values, y_pred. I was able to conclude that the accuracy of the Naïve Bayes Gaussian method is 99.55%, basically a 100%.

The results are bad as having a perfect accuracy score means that we may have overfitting our dataset to the model, or that we have an issue in the code that was created from scratch.



| | | True Class | |
|---|---|---|---|
| | | Positive | Negative |
| Predicted Class | Positive | 166 | 0 |
| | Negative | 1 | 56 |

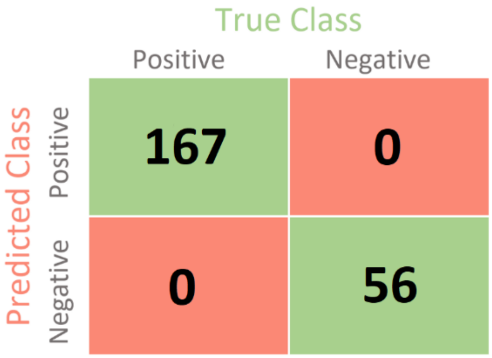
*Fig. 2. Naïve Bayes Gaussian Confusion Matrix*

### *F. Naïve Bayes Categorical*

With Naïve Bayes Categorical method, I used the code that our professor provided. I performed the same techniques such as converting the data frame of the sentiment scores to be categories. With that, I applied them to the prior and likelihood functions. Next, I trained the model and predicted the values.

Again, my results could not be converted to a confusion matrix via code, so I calculated it by hand by comparing the true values, y_test, and the predicted values, y_pred. I was able to conclude that the accuracy of the Naïve Bayes Gaussian method is a 100%.

Just like Naïve Bayes Gaussian method, the results are bad as having a perfect accuracy score means that we may have overfitting our dataset to the model, or that we have an issue in the code that was created from scratch.



| | | True Class | |
|---|---|---|---|
| | | Positive | Negative |
| Predicted Class | Positive | 167 | 0 |
| | Negative | 0 | 56 |

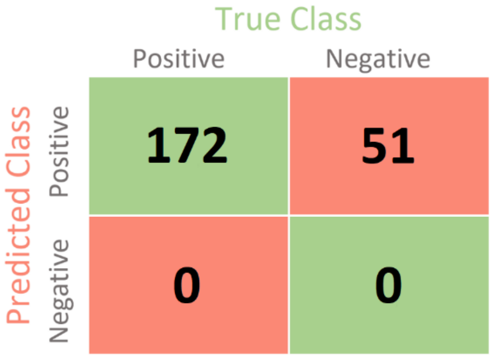
*Fig. 3. Naïve Bayes Categorical Confusion Matrix*

### *G. Neural Network*

In neural network, I used the code that was provided by the professor. I was able to get the code to compile and output a result, however, it seems that the results are incorrect. I am not 100% positive, however, I believe that there should not be a fixed probability when testing each Reddit title. I do not have faith in my results for neural network, but I calculated my accuracy of the model.

Again, I was unable to create a confusion matrix via code, so I calculated it by hand by comparing the true values, y_test, and the predicted values, y_pred, generated by the Neural Network model. I was able to conclude with an accuracy of 77.13%.

The resulting accuracy is not horrible, however, it is not credible as the previous output does not seem correct. With the fixed probability that I am getting per prediction of each Reddit title, I can say that the results of the Neural Network is not valid.

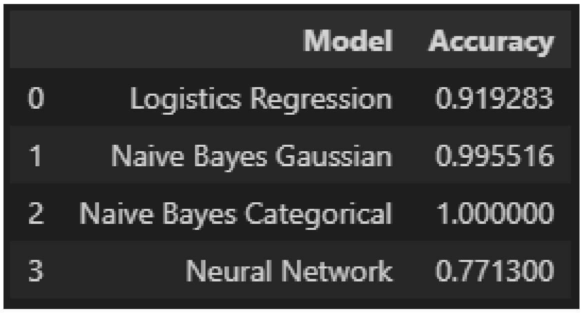


| | | True Class | |
|---|---|---|---|
| | | Positive | Negative |
| Predicted Class | Positive | 172 | 51 |
| | Negative | 0 | 0 |

*Fig. 4. Neural Network Confusion Matrix*

### *H. Conclusion*

Based on the data that was calculated, I can conclude that Logistics Regression came out the have the best accuracy score of 92% for this uncleaned dataset.



| | Model | Accuracy |
|---|---|---|
| 0 | Logistics Regression | 0.919283 |
| 1 | Naive Bayes Gaussian | 0.995516 |
| 2 | Naive Bayes Categorical | 1.000000 |
| 3 | Neural Network | 0.771300 |

*Table 1: Reddit WallStreetBets Classification Report*

## IV. CONCLUSION

I would like to say that there is a lot of techniques for all kinds of different problems. These techniques can be used in different way to improve our scores. There are endless choices in machine learning!