# Live Sentiment Analysis on /r/Wallstreetbets Comments

**Chad Comiter**
**Jordan Harrop**
Khoury College of Computer Sciences
Northeastern University
Boston, MA 02115

**Abstract**

We trained several classifiers and aggregated their output in a "classifier of classifiers" which was deployed to conduct sentiment analysis on comments from a popular investing forum in real-time. This model was limited by several technical limitations regarding the size of the training data, but regardless was able to classify comments in real-time as either positive or negative. Our experiments suggest that this model is successful although it does bias towards positivity. This bias is likely due to the limitations regarding computational power when training the models on our dataset and would likely be rectified by providing a larger training dataset.

## 1. Introduction

Sentiment analysis is a NLP technique by which target data is evaluated as positive, negative, or neutral. In both traditional and algorithmic investing, sentiment of news articles and discussion amongst the retail sector is considered an asset. While traditionally not used alone, sentiment analysis is frequently incorporated into more complicated overall strategies, and many firms charge large sums of money for robust sentiment analysis.

Reddit.com's WallStreetBets is a popular investing forum that has over 12 million active subscribers. It has gained notoriety in recent years for popularizing large trends in the retail sector that have made national news on several occasions. Undeniably, WallStreetBets represents a "pulse" amongst retail investors. Thus, it presents itself as a strong candidate for sentiment analysis and for the extraction of valuable information.

We aim to incorporate several classifiers into a "classifier of classifers" that can provide context as to the overall sentiment of WallStreetBets users at any given time. We aim to incorporate a number of naïve bayes, linear/logistic regression, and source vector machine classifiers in order to accomplish this. WallStreetBets presents several challenges when it comes to training models. Primarily, one must account for the unique vocabulary that separates WallStreetBets users from other investors – the site has its own language and slang that may cause difficulties when working with models. To combat this, we use training data unique to WallStreetBets to train our classifier of classifiers. This "master classifier" is then fed comments in real-time via PRAW, a python wrapper for Reddit's API.

## 2. Methods

### 2.1 Training Data

Training data was obtained from an open-source aggregation of over 300,000 sentiment-scored wallstreetbets comments. Only approximately 10% of the comments were used for training data due to limitations surrounding a lack of memory on our local machines. The body and score of each comment were extracted and a pandas dataframe was created with two columns - “Text” and “Sentiment”. Comments were pruned for stopwords, cleaned via a regex to remove special characters, standardized to lowercase, and finally tokenized to create an array of words that comprised each comment. Sentiment above 0.3 was deemed “positive”, and sentiment below -0.3 was deemed “negative”, and sentiment scores in the training data were adjusted accordingly to prepare for the creation of a featureset. The data was upsampled so that the minority sample data (in this case, negative sentiment), matched the majority sample data in frequency. A dictionary of words mapped to frequency was created for use in the various naïve bayes classifiers used in our model. Finally, a feature set of the training data was created, which is a dictionary of words mapped to sentiment. This was used in every model listed below. The various models explained below were imported from sklearn, a python package. We generally did not choose classifiers aside from the family of naïve bayes classifiers due to any advantage. Rather, this was an exercise in building a “classifier of classifiers” or an “algorithm of algorithms”.

### 2.2 Naïve Bayes Classifiers

We represent each comment as a set of features (X1,…..Xn) where X represents an individual token of the comment. This is similar to the way other Naïve Bayes experiments are structured. [1] Naïve Bayes classifiers are a family of classifiers which assume that the value of an independent variable/feature is independent, and not reliant on the value of any other independent variable/feature which comprises the same set of features. Bayes theorem gives us a formula by which we can determine the probability of an event given the probability of a set of events which have occurred.

$$p(\mathrm{A/B}) = p(\mathrm{B/A}) * p(\mathrm{A})/p(\mathrm{B})$$

In the above, P(A) represents the prior probability of A before some evidence B is provided. P(A/B) represents the probability of an event A after evidence B is seen. [2]
To calculate the probability p of observing features $f1$ through $fn$, given a class c, we can use the following formula:

$$p(f_1,\ldots,f_n|c) = \prod_{i=1}^{n} p(f_i|c)$$

While the assumption that each variable is truly independent of each other variable is rarely true (especially in text analysis), this classifier regardless typically provides a strong foundational starting point by which we can begin to analyze text and historically performs

surprisingly well. For our model, we used a Gaussian Naïve Bayes Classifier, a Multinomial Naïve Bayes Classifier, and a Bernoulli Naïve Bayes Classifier.

A multinomial naïve bayes classifier differs from traditional gaussian classifiers in that it recognizes that $p(fi|c)$ is a multinomial distribution. [1] It is known to work well on data which is translated to "counts" of features, which makes it a strong candidate to include in a text-based sentiment analysis. This is because words will occur in positive and negative sentiment in varying frequencies. The Bernoulli naïve bayes classifier assumes that our features are binary, working especially well in cases where words either do or do not occur in a document.

For our "classifier of classifiers", we incorporated sklearn's Naïve Bayes, Multinomial Naïve Bayes, and Gaussian Naïve Bayes classifiers. All were trained on our training data set and returned accuracy of over 70%.

### 2.3 Logistic Regression Classifiers

Logistic regression is a classification method used to predict the probability of occurrence for categorical dependent variables. These variables are typically mapped to boolean outcomes, such as true/false, 1 or 0, or the more relevant positive and negative. Linear regressions models generally predict P(A) as a function of B. Logistic regression takes the linear regression equastion and adds a sigmoid function. Whereas linear regression provides a continuous output, logistic regression is preferred for discrete classification. We use two logistic regression classifiers provided by sklearn – Logistic Regression and Stochastic Gradient Descent. Stochastic gradient descent is not a distinct classifier, but rather an optimization technique that provides efficiency and simple implementation to a traditional logistic regression model.

### 2.4 Support Vector Classifiers

Support vector machines are supervised machine learning methods used for classification, regression, and outlier detection. Our classifier of classifiers uses a standard SVM, a Linear SVM, and a Nu-SVM. Given a set of example training data which includes variables belonging to one of two categories, SVM algorithms build models that assign new variables to one of the two categories. This complements sentiment analysis incredibly well. SVM models represent examples as points in 2D space, and new variables are mapped into the same space and are assigned a category based upon which side of a division in the space they fall. [3]

### 2.5 Pickle

The python package pickle was used to save and load classifiers to massively reduce the runtime of the notebooks.

### 2.6 PRAW

Comments were streamed into our classification model in real-time via PRAW, a python wrapper for the Reddit API.

## 3. Results

Below are examples of the text output from our model with their sentiment score being represented by the following number. It has been divided into 3 groupings, successful outputs, or outputs where the model correctly identifies the sentiment when compared to a human judge (Table 1). Neutral outputs, where there wasn't enough information within the text to accurately define the sentiment (Table 2), and wrong output, where the sentiment output conflicts with a human judge (Table 3).

| Comment | Sentiment Score |
|---|---|
| Didn't think NFLX would drop over 25% being that not much should have changed from their last quarter release, boy was I wrong ! PE now in the teens, I think it will bounce back, not stressing. | 1 |
| No, still up, still green. | 1 |
| My post yesterday about my position almost everybody said 'kiss that money goodbye' | 1 |
| I bought more, thanks. | 1 |
| >RUSSIA EXPANDS U.S. SANCTIONS LIST, INCLUDES VICE-PRESIDENT HARRIS - RIA CITES FOREIGN MINISTRY 11:00:53 EDT-0400 | -1 |
| Yea sub has been trash for a while now. | -1 |
| Doubt | -1 |

**Table 1:** Comments and their sentiment analysis scores that agreed with the outcomes as seen by a human judge.

| Comment | Sentiment Score |
|---|---|
| I've been good. And you? | 1 |
| Yes | 1 |
| I should! | 1 |

| | |
|---|---|
| Anyone have a link to the Powell speech? | -1 |
| Youtube. Hasn't started yet | -1 |

**Table 2:** Comments and their sentiment analysis scores that agreed with the outcomes as seen by a human judge.

| Comment | Sentiment Score |
|---|---|
| but apparently the stock market hasn't crashed yet because the dow is only down 5% /s | 1 |
| I'm buying. Always buy in red days. | -1 |
| The only thing worse than losing money in the market is losing money when everything is green ![img](emote\|t5_2th52\|8880) | 1 |
| No. You just pissed away your money for nothing | 1 |
| No lol, losing subscribers for the first time in 10 years definitely hurt | 1 |

**Table 3:** Comments and their sentiment analysis scores that disagree with the outcomes as seen by a human judge.

As reddit is a message board there are a significant number of replies to comments that are short, and without the context of the parent comment do not make sense when read individually. This was probably the most common result after looking at the data and it was captured by the Neutral comments section of Table 2.

The other two tables are better judges of the model. Table 1 shows where the model was successful in judging sentiment. The model was more often correct when judging comments that had a positive sentiment analysis. There could be many reasons for this but likely it is a combination of the training data we had was limited by the strength of our computers and wallstreetbets as a subreddit tends to be overly positive in their comments. These factors lead to a model that is biased towards positive comments.

Table 3 shows comments that were not analyzed correctly by the model. There are different reasons for this but most commonly it seems to be sarcasm and lack of context from parent comments. Sarcasm is typically difficult to capture in analysis like this and can be marked and noticed by a human observer from the tone of the comment or the "/s" tag as seen in the first comment. Neither of these were accounted for in the model so it makes since it went unnoticed.

Included are 3 graphs of live streamed reddit comments taken at 1100-1300 EST which is an active time for the board. They chart the number of comments on the x-axis vs the cumulative sentiment score on the y of the comments. The coefficient for positive comments differs in the 3 charts to account for the positive bias of the model and the subreddit. Figure 1 has a bias coefficient of .3, Figure 2 has a coefficient of .2 and figure 3 has a coefficient of .1. All runs

were conducted until the model analyzed roughly 400 comments. The runs averaged roughly 6-8 minutes long.

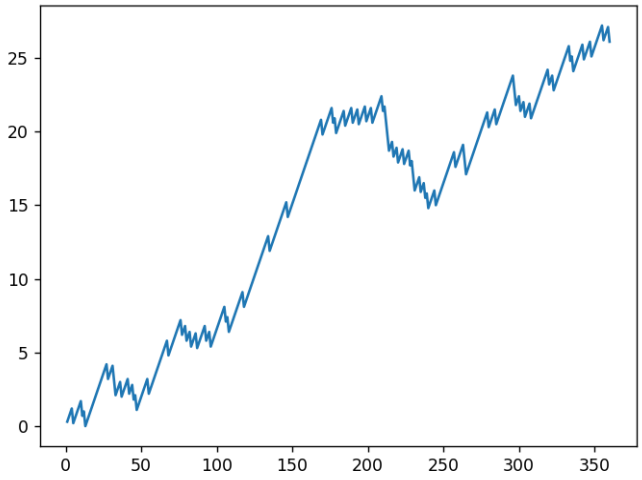


**Figure 1:** Comments on the x-axis vs Cumulative sentiment score on the y-axis for 375 comments. Positive bias coefficient of 0.3.

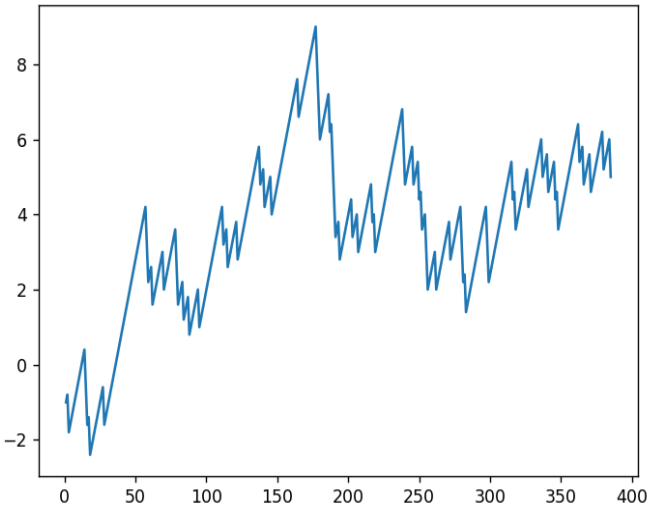


**Figure 2:** Comments on the x-axis vs Cumulative sentiment score on the y-axis for 375 comments. Positive bias coefficient of 0.2.

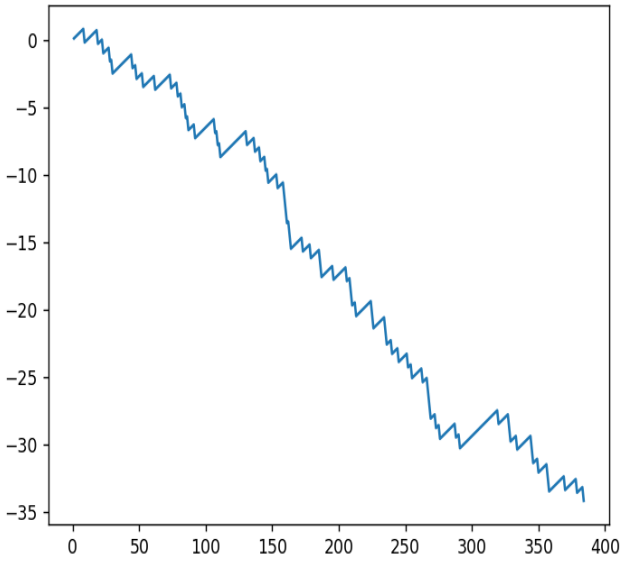


**Figure 3:** Comments on the x-axis vs Cumulative sentiment score on the y-axis for 375 comments. Positive bias coefficient of 0.1.

As demonstrated in the above figures, the model attributes positive sentiment to a comment approximately five times the amount it attributes negative sentiment to a comment.

## 4. Discussion

The goal of this project was to implement a model built upon a classifier of classifiers. Additionally, this model was to be deployed to categorize WallStreetBets comments in real-time by attributing to them positive or negative sentiment. Considering the above, we believe that we have achieved limited success. To some degree, there are considerations that must be accommodated when working with Reddit comments specifically that exceed the original scope of work planned in this project. Notably, the model cannot accommodate sarcasm or consider the "nested-ness" of some comments, whereby the comment only makes sense when considering the parent comment of the tree of comments.

There are other limitations that directly impacted the accuracy of the model. These are hardware issues and ultimately can be attributed to the fact that the two contributors to the project lack access to machines or virtual machines that could accommodate high degrees of CPU usage. This directly led to a smaller training dataset – as we ran into issues with executing the model training on data larger than 30,000-50,000 rows at a time. A smaller and less robust feature set is a direct consequence of this fact.

The model works well and is accurate when running it against testing data from the same source as our training data set – reaching accuracy levels of over 70%. However, it is ultimately inconsistent when categorizing live comments. This could be rectified by providing user feedback to the model, as it is live-streamed. For instance, if a user could confirm as to whether the model was correct, and if that feedback could be provided to the

model – our model would likely get significantly more accurate over time. This is an extremely common integration that is provided to other models of this type and should be considered a requirement if this model were to be deployed for use in the real world.

## 5. Conclusion

We implemented a classifier built upon the votes and aggregation of a number of classifiers including Naïve Bayes, Multinomial Naïve Bayes, Bernoulli Naïve Bayes, Logistic Regression, Stochastic Gradient Descent, Support Vector Classification, Nu-Support Vector Classification, and Linear Support Vector Classification. These models were trained and combined as a VoteClassifier class. While we were able to deploy this so that it classified comments as they were posted, we ultimately ended up with a model that was heavily biased towards positivity. In the future, we would like to explore a larger, more robust training dataset that would likely remove large degrees of the model's bias.

## 6. Acknowledgement

This project was completed for CS6120, taught by Professor Uzaid Ahmad.

We would like to thank each other for a collaborative and productive working environment, and for the successful completion of the term project.

## 7. Citations

[1] Metsis, Vangelis; Androutsopoulos, Ion; Paliouras, Georgios (2006). *Spam filtering with Naïve Bayes – which Naïve Bayes?***.** Third conference on email and anti-spam (CEAS). Vol. 17.

[2] Dhadhuk, H. *Performing Sentiment Analysis With Naïve Bayes Classifiers!* Analytics Vidhya. https://www.analyticsvidhya.com/blog/2021/07/performing-sentiment-analysis-with-naive-bayes-classifier/

[3] *1.4 Support Vector Machines.* (n.d.). Scitkit-Learn. https://scikit-learn.org/stable/modules/svm.html