Predicting market volatility and building short-term trading strategies using data from Reddit's WallStreetBets

CS229 Project: Final Report
Shashank Rammoorthy (sr22)
Kiara Nirghin (knirghin)
Abhishek Raghunathan (abhi2020)

Abstract

The aim of this project is to use data from posts made on the sub-reddit "WallStreet-Bets" to make a prediction on stock prices and market volatility. Various features were extracted from these posts and a learning model was trained to predict if specific stocks rose or fell in the given time frame.

1 Introduction

The GameStop "short squeeze" dominated financial, and even primetime news headlines in January 2021. Stories of individual investors becoming millionaires overnight became virtually dime a dozen. Analysts pontificated that the phenomenon signaled a shift in markets - a shift favoring individual investors for the first time in years.

The primary driver of the GameStop short squeeze saga was the social news website Reddit - specifically the "subreddit" r/WallStreetBets. WallStreetBets (WSB) is an online hub for discussions surrounding speculative stock picks, and really grew into its own during the short squeeze. A few users on WSB essentially fueled the crazed buying of the GME or GameStop stock. Groups like WallStreetBets have ushered in a new era of trading in which the general public has the power to cause a stock market disruption on its own.

Our project aims to carefully follow data scraped from groups like Reddit's WallStreetBets as a means to predict market volatility and build a robust short-term trading strategy.

In this paper, we describe our experiments with algorithms that aim to predict the future price of a security through analyzing posts on WSB. The long-term goal of the project would be to build a trading bot that intelligently executes short-term trades based on insights gleaned from somewhat unconventional sources like WSB - aggregating results obtained by the algorithms described in this paper in order to make a decision.

2 Related Work

Predicting market trends through data forms the basis for a behemoth of an industry. Machine learning methods have lately gained traction alongside more traditional statistical methods widely employed at investment banks and quantitative hedge funds over the years.

Jiao et al[2] looks at the S&P 500, and attempts to predict the future price of the index based on historical data using various learning algorithms. A key finding was that taking into account recent information - such as recently closed European and Asian indexes - lead to a vast increase in predictability.

The general consensus that is established in the literature is that stock prices are very dynamic and susceptible to quick changes because of the underlying nature of the financial domain and in part because of the mix of known parameters (previous closing price, P/E ratio etc.) and unknown factors (like election results, rumors etc.). Shah[3] considered various machine learning techniques for stock prediction, and implemented as simple linear regression model, alongside a Support Vector Machine model.

Another common approach was the use of a neural network to forecast stock market prices. Neural networks not only have the ability to discover patterns in nonlinear and chaotic systems but also offer the ability to predict market directions more accurately than most current techniques. Lawrence[4] considers common market analysis techniques such as technical analysis, fundamental analysis, and regression - and compares these with neural network performance.

In the sentiment analysis applied to markets realm, Awan et al take a big data approach to the problem of short-term forecasting, incorporating sentiments from mainstream news sources to inform their predictions. Kesavan et al consider time series data from social media platforms and incorporate the polarity of the sentiments observed to enhance the prediction accuracy. Unsurprisingly, considering sentiment as a part of the input to a stock price prediction algorithm seems to lead to favorable outcomes.

3 Dataset and Features

title	score	id	url	comms_num	created	body	timestamp
It's not about the money, it's about sending a message. 🚀💎🙌	55	l6ulcx	https://v.redd.it/6j75regs72e61	6	1611862661.0		2021-01-28 21:37:41

Figure 1: A sample feature

We used a dataset hosted on kaggle.com that consisted of scraped posts from the WSB subreddit.[1] The dataset included the following for each post: the title, id, url, the body of post, number of comments, total score of each post and a time stamp. An example from the data set is included above.

There were several steps of preprocessing needed to make the data usable for our purposes. A lot of these steps are conventional for a problem in this domain - we dropped NAN values and converted all strings to lowercase, among other things. For the sentiment analysis stage of the preprocessing, we removed handlers, URLs, special characters, single characters, and substituted multiple spaces with a single space. We analyzed the mentions of various in-the-news companies, as indicated by the histogram below, and decided to focus our analysis on GME (GameStop), AMC (AMC Theatres), and TSLA (Tesla).

Using both pandas and numpy, we were able to process the Kaggle dataset of Reddit - r/WallStreetBets. We were then able to filter these posts to ones that explicitly mention the stocks we had decided on above. Initially, we decided to map these posts on date keys where we were able to group posts from the same day and calculate values such as: total comments and average score for that day. However, based on short-term market volatility and the limits this set on data availability, we concluded with creating a seven element feature vector for every post that mentions the stock and created a sub-matrix for each stock category. The seven features include:

1. post_score
2. total_comms_num
3. len_of_post
4. positive_sentiment
5. neutral_sentiment
6. negative_sentiment
7. bias (column of 1's)

Our sentiment analysis on each post leveraged prior work done for this same dataset[9].

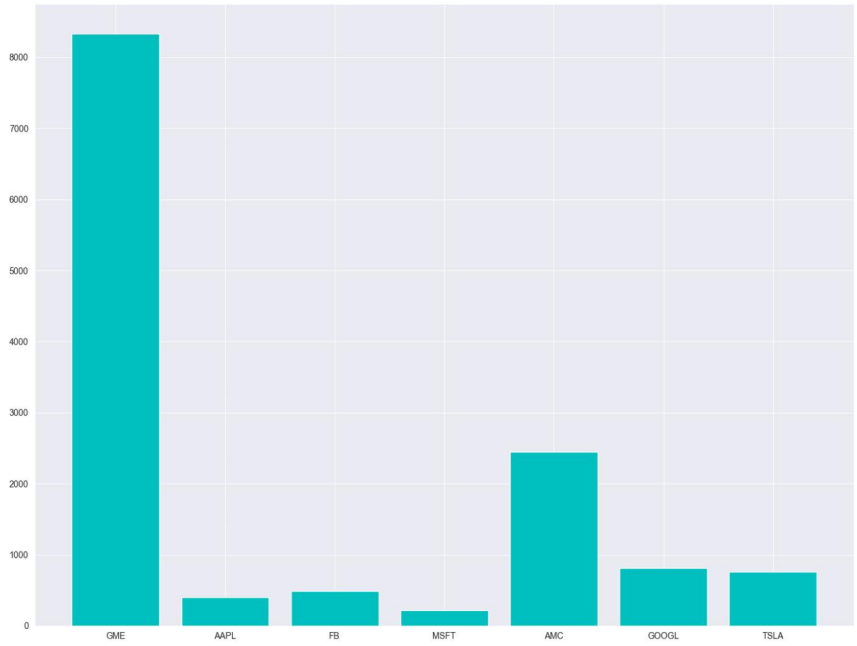

Figure 2: Mentions of various stocks in the dataset

4 Methods

We started out with a binary classifier as the baseline. Logistic regression was used; input features were created from the data-set, and passed into the logistic regression classifier. Feature scaling was applied on the input features to ensure a reasonable range of each weight. Different label ideas were experimented with, which are described below. What this classifier does is that it tries to minimize a loss function to maximize the probability of each label being correct. The result is the impact of each variable on the odds ratio of the observed event of interest, which is represented through a weight vector. Newton's method was used to solve for the optimal weights. This is a numerical algorithm that helps us solve an equation of the form $\frac{\partial F}{\partial \theta} = 0$.

As an alternative to the logistic regression employed above, and to fit a nonlinear function to this prediction task, we also constructed a 3-layer neural network to do binary classification on individual posts. The network consisted of 3 256-dimensional linear layers with ReLU nonlinearities, and a sigmoid at the head.

The resultant weights answer the question of whether the stock price went up at market open the next day - in other words, whether $P_k < P_{k+1}$ where P_i represents the price of a security at market open on day i.

The idea with the neural network was to create a weight vector for posts about a specific security, and then when trying to incorporate the weight vector into our bot, we take a majority-wins approach to deciding whether the price of a security will go up given all the posts seen in a day.

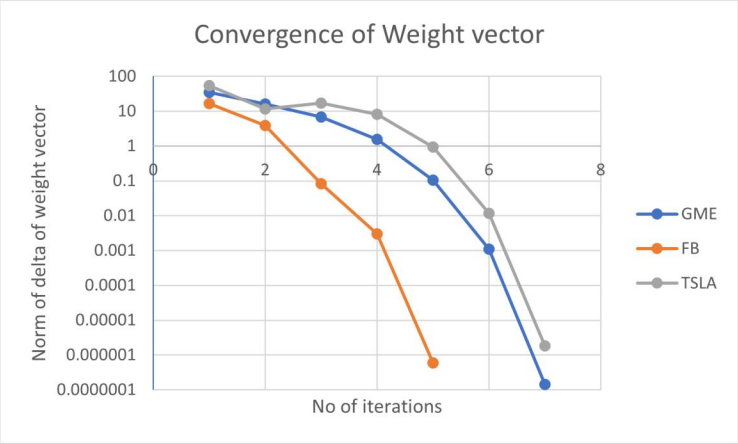

Figure 3: Convergence of weights for various stocks

5 Experiments, Results and Discussion

5.1 Logistic Regression

In our baseline model with 80 training examples and 20 elements in the test set, we obtained an accuracy of about 73%. This baseline mostly served to reinforce our belief that there is something to be learned here. Our labels were simple - a 1 indicates the price of the security at the opening of the market on the following day was higher than it was on the day of the post/comment. A 0, on the other hand, indicates that the opening price of the selected security was higher on the day of the post/comment in question, and went down the next day at market open. This was also highly contextualized to GME.

An attempt was made to improve on the baseline logistic regression. A few key changes were incorporated:

- The labels were changed to indicate whether there was a significant shift in stock value. The idea behind this change was that mention of the stock in the sub-reddit could be correlated to a rise or fall in the stock but not necessarily either.
- Each feature vector was remodelled to represent a single post, rather than an aggregate of posts made on the day. This allowed us to extract the fine details present in the data-set, while providing more examples for training and testing.

We also implemented a random baseline, which simply guesses if the stock price changes significantly or not. This served as a floor which helped us understand the performance of the other algorithms used.
The convergence of the algorithm for 3 different stocks(GameStop, Facebook, Tesla) is presented below. This is a semi-log plot showing the decreasing norm of the difference in weight vector between subsequent iterations. We found that while the machine made reasonable predictions for these 3, it failed to converge for AMC. Upon investigating, it was found that this algorithm failed to bring out any correlation that might have existed with the features. This prompted us to explore other more complex algorithms.

5.2 Neural Network

We tried various hyper-parameters as is standard in the literature, and settled on a batch size of 16, using Adam with a learning rate of 0.001, and running for 30 epochs. Where practical, we increased batch size (in our case, we trained TSLA and GME with a batch size of 64). We implemented an 80-10-10 train/validation/test split, and report the results on the held out test set.

As compared to the random baseline - a large improvement is clearly visible. Comparing the logistic regression and neural network test accuracy, we see that the logistic regression accuracy for TSLA is high. Logistic regression for AMC never converged with our criteria.

Stock	NN Test Accuracy	LogReg Test Accuracy	Baseline accuracy
TSLA	0.64	0.740	0.532
AMC	0.819	(N/A)	0.42
GME	0.844	0.719	0.527
FB	0.693	0.68	0.42

Table 1: Accuracy results of the methods employed

The neural net test accuracies were consistently high across the board, with more variability in the logistic regression predictions. The logistic regression also appeared to do better with predictions involving negative changes in general, one wonders if stock market context has a role to play as well.

6 Conclusion and Future Work

Our most important conclusion is that some amount of correlation exists, and that WallStreetsBets posts do have an impact on the actual stock prices. The real challenge lies in bringing out this correlation, and we made an attempt at overcoming it by building a neural network.

While these results may not hold much value by themselves, they declare the start of a new spectrum of ideas and possibilities for prediction of stock prices. The neural network was more reliable than a simple logistic regression, but only by a slight amount. That being said, there is a lot of scope for expansion and advancement with this idea.

We were dealing with a relatively small dataset in this project, all things considered - and had to tailor our methods to work in this low-data regime. Possible extensions include scraping more data, including those from conventional mainstream media sources, and instead of relying on a black-box sentiment analyzer, make the sentiments trainable weights that are tailored to the task of stock price prediction. We would also like to explore variable timings in periods when looking at the stock volatility, perhaps time periods such as 6 hours, 1 week etc. as opposed to 24 hours.

In terms of the feature vector we believe that our current feature vector is a simple representation of the data we were able to extract from Reddit - r/WallStreetBets, but can be further expanded. We would like to incorporate more features than the ones that exist right now, by considering additional sources such as: other finance focussed Reddit groups, finance related podcast transcripts and news articles. With this more robust feature vector, we would also like to explore some pre-trained models for sentiment analysis and general language modeling such as BERT. We think this could be an interesting addition as we focus more closely on the language used within these groups. We would also like to look at the semantic meaning of the posts in more detail, this would entail looking at word embedding and building a language model from a pre-trained BERT. The idea would again be to contextualize the language model to this specific task.

Since we incorporate post-level granularity with our predictions, there are applications for streaming algorithms in this domain. We envision a bot that makes buy/sell decisions 'on-the-fly'.

A Contributions from team members

Shashank Rammoorthy – Processed financial data, provided the delta API for the rise and fall classification of daily gamestop stock, implemented the neural network.
Kiara Nirghin – Processed Reddit - r/WallStreetBets group data including creating the seven element feature vector, performed sentiment analysis on the stock posts and graphed stock frequencies to determine the subset used for the project.
Abhishek Raghunathan – Implemented the logistic regression algorithm, trained it and predicted rise or fall for a test subset of the data, and compiled final results comparing to a random baseline. Also integrated different aspects of the learning algorithm.

B References

1. https://www.kaggle.com/gpreda/reddit-wallstreetsbets-posts
2. Y. Jiao and J. Jakubowicz, "Predicting stock movement direction with machine learning: An extensive study on S&P 500 stocks," 2017 IEEE International Conference on Big Data (Big Data), Boston, MA, USA, 2017.
3. Shah, Vatsal H. "Machine learning techniques for stock prediction." Foundations of Machine Learning| Spring 1.1 (2007): 6-12.
4. R. Lawrence, "Using Neural Networks to Forecast Stock Market Prices," University of Manitoba, Canada, 1997.
5. M. J. Awan, M. Shafry, H. Nobanee, A. Munawar, A. Yasin et al., "Social media and stock market prediction: a big data approach," Computers, Materials & Continua, vol. 67, no.2, pp. 2569–2583, 2021
6. M. Kesavan, J. Karthiraman, R. T. Ebenezer and S. Adhithyan, "Stock Market Prediction with Historical Time Series Data and Sentimental Analysis of Social Media Data," 2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS), 2020, pp. 477-482, doi: 10.1109/ICICCS48265.2020.9121121.
7. Ran Aroussi's yfinance python package (https://pypi.org/project/yfinance/)
8. Chollet, F. & others, 2015. Keras. Available at: https://github.com/fchollet/keras.
9. https://www.kaggle.com/thomaskonstantin/reddit-wallstreetbets-posts-sentiment-analysis