|  |  |
| --- | --- |
| Reddit's WallStreetBets plans to decentralize | Level Up Coding  1 Image via 1. Louis Petrik. Reddit’s WallStreetBets plans to decentralize | Level Up Coding (gitconnected.com) | Can Comments in the Wallstreetbets Subreddit be Predictive of Stock Price Movements? |

# Introduction

In late 2020 the public became aware of the Subreddit “Wallstreetbets” when followers drove the price of ailing Gamestop from around $4.00 per share to as much as $325.00 per share. There is no doubt that the convictions of these non-traditional investors moved the share price.

We can see correlations between comment volume and stock price movement, but is there any predictive value in the volume and/or the content of these comments?

Chart, line chart

Description automatically generated

In the following exercise I explored whether chatter in the group (comments) could be used to predict stock price movements. I gathered comments referencing Tesla, Inc. as a popular discussion topic, and chose a seven-month time-frame of relatively choppy movement rather than overarching trend to see if machine learning could be used to turn those comments into actionable daily predictions.

Initially I set out perform a two-step process using BERT sentiment analysis to analyze the content of the comments and then LTSM time series analysis to create a prediction. These choices were adapted as the nature of the data became evident.

# Data Collection

Reddit comments were collected using PushshiftAPI . This is a repository of comments that allows easy querying by date. The seven month’s-worth of comments was pulled in three batches to avoid limits and amounted to ~182,000 rows. Price movement data for the same period was pulled from Yahoo using the yfinance package.

# Data Pre-Processing

As the primary target was content of comments and the count of comments in the Reddit data, there was not a lot of scrubbing to do other than to confirm that the comments were English language using langdetect and to check for duplicates.

The Reddit data included a number of additional fields describing the Authors of the comments and various other details. These may be worth exploring in a future iteration but were not the target of this project. I scrolled through summaries of unique values for these columns looking for categorization opportunities but found none of use for this analysis.

Text

Description automatically generated with medium confidence

# Feature Engineering

Since the object was a two-step machine learning process, it was necessary to engineer the intermediate prediction value: “sentiment” - as I needed to know if comments were positive or negative to be able to train the BERT model. I set out to label 300 rows ‘positive’ or ‘negative.’

The process of rating comments proved nearly impossible given the snark, sarcasm, and profanity prevalent on Reddit. It was also difficult to know the context of the comment given that they might have been replying to another comment in the string, even if their comment was not labeled as a reply. There was the additional difficulty of knowing if someone was ‘positive’ or ‘negative’ regarding the stock, the market, other commenters, or their general mood.

|  |  |
| --- | --- |
|  | Knowing this, I looked into BERT a little further and found reviews stating that BERT did not do particularly well with text under one hundred words. Filtering the dataset for entries with more than one hundred words left only ~6,000 records out of the original 182,000. |

What I did notice however, was the prevalence of emoticons and specific phrases common to Wallstreetbets. Wallstreetbets is well know for particular phrases and memes. HODL is an example which may have originated with a typo of “HOLD” or may stand for “Hold On for Dear Life.” Given the presence of these features, I continued with a sentiment-naïve analysis of the predictive value of the prevalence of these in daily comments.

Using extract\_emoji I encoded the most popular emojis into the dataset, as well as the most common meme phrases.

|  |  |
| --- | --- |
| Text  Description automatically generated | Image result for wallstreetbets |
| Graphical user interface, text, application, email  Description automatically generated | A picture containing graphical user interface  Description automatically generated |

I then grouped the data by date and added the next-business-day stock price close value as a prediction value as well as an indicator of positive or negative movement from the previous day close.

# Modeling

## LTSM

The first model was an LTSM multivariate time-series prediction model adapted from an example by Florian Müller. This entailed scaling values, creating train and test datasets, and tuning the hyperparameters of the model. As there is currently no implementation of Gridsearch for LTSM, the model was run multiple times with a variety of values for Sequence, Batch Size, and Epoch values.

Using Mean Absolute Percentage Error as a guide, the following settings were chosen to result in a MAPE of 3.64%.

*Sequence Length 10 Epochs 10 Batch size 5 Median Absolute Error (MAE): 22.5 Mean Absolute Percentage Error (MAPE): 3.64 % Median Absolute Percentage Error (MDAPE): 3.59%*

Chart, line chart

Description automatically generated

The result is impressive but the errors are still large enough to make it unlikely to be implemented for a day-to-day trading algorithm.

Feature importance is also difficult to assess as this type of deep learning algorithm is somewhat of a “black box.”

## XGBoost

The next model is a XGBoost Classifier applied against the next\_day\_plus feature (stock price up). XGBoost does not require scaling but does require creation of a train and test dataset. I performed a VIF test of multicollinearity to remove correlated features.

I used GridsearchCV to optimize the hyper-parameters and the model returned the following performance:

Graphical user interface, text, application, email

Description automatically generated

|  |  |
| --- | --- |
| The Shapley TreeExplainer shows the most impactful features. |  |

The XGBoost model shows some potential to predict positive or negative moves in the stock price on the following business day. The accuracy and the ROC AUC Score are not high enough to justify implementation but they do indicate that there may be value in further investigation.

## Logistic Regression

Finally I ran logistic regression on the train and test sets prepared for XGBoost. This returned a similar result: Some prediction (Accuracy 53%), not enough yet to put to use. Precision would be important as well in that you would not want to act on a false positive – and that is under 50% in this case.

Chart

Description automatically generated

# Results

The XGBoost has the most promising performance and is easily interpretable. All of the models indicate there might be something there but aren’t ready to justify use in a live setting.

With a better accuracy and AUC ROC Score, this would be relatively easily moved to production. Comments could be pulled daily and the model run against them for each day’s trading. An option position could be opened (call option for up-prediction/put option for down prediction) at the opening of trading and optionally closed out at the end of the day.

# Next Work

This preliminary work I believe would justify additional exploration to see if the additional features and data could push the performance results to useful levels.

Some things to consider:

* Weighting prolific authors
* Splitting up dates for market hours/after market hours
* Running for individual days of the week
* Using a basket of similar stocks and a longer time frame to get a larger final dataset
* Normalizing for overall market moves



Attributions:

1. [Louis Petrik](https://louispetrik.medium.com/?source=post_page-----5d0763434f1e-----------------------------------). [Reddit’s WallStreetBets plans to decentralize | Level Up Coding (gitconnected.com)](https://levelup.gitconnected.com/decentralize-wallstreetbets-5d0763434f1e)
2. Florian Müller: https://www.relataly.com/stock-market-prediction-using-multivariate-time-series-in-python/1815/