Gamestop Corporation Share Price Movement Prediction Methods

Sean Yeo, Sean Sui, Yufei Chen
University of Victoria Department of Computer Science

Abstract

This report offers a comparison between decision trees and LSTM neural networks for predicting Gamestop Corporation daily closing share price movements year-to-date. Additionally, sentiment scores of daily online forum text from Reddit is used in an attempt to increase the prediction accuracy of the aforementioned methods. Among the two chosen methods, it has been found that LSTMs provide more accurate results in price movement prediction than decision trees, and the addition of sentiment data does not improve prediction accuracy.

1 Introduction

Within the last year, people have flocked to online trading platforms and forums to discuss investment strategies and opinions. In particular, one company this year has caught the attention of mainstream media, large hedge funds, the SEC, and average citizens alike: Gamestop. With how volatile the stock has become, it would be paramount to have a playbook of more advanced strategies other than buying and holding in hopes of a payout.

This report offers a comparison of machine learning methods to predict price movements using decision trees in Sklearn and LSTM neural networks in Keras using daily open, high, low, close, and volume data from Yahoo Finance, as well as four technical indicators calculated from those values: RSI (Relative Strength Index), ADX (Average Directional Movement Index), SMA (Small Moving Average), LMA (Long Moving Average). Additionally, daily sentiment scores of post titles from two subreddits (r/GME, r/Superstonk), as well as top comments from across Reddit with 'GME' as a substring are used to help in price movement prediction.

2 Method Choice Background

2.1 Decision Tree

Decision Trees offer the ability to derive classification rules from featureless, irregular data sets from various angles while retaining simplicity. They integrate well with large databases, and can handle multiple data types such as continuous, discrete, and Boolean values. We believe that the decision tree algorithm can find important features, accurately classify given data, all while preserving the accessibility of results.

2.2 LSTM Neural Network

Because LSTMs offer the ability to retain memory of previously encountered patterns in data for longer sequences, we thought that this method would be well suited for this problem. Although standard RNNs are also capable of working with time series data, due to susceptibility to the vanishing gradient problem, their usage was ultimately scrapped for this project.

3 Data Usage and Background

3.1 Data Processing

Daily open, high, low, and close price data was gathered from Yahoo Finance due to its availability and accuracy. One key problem with trying to predict Gamestop share price movements year-to-date was its volatility, as well as recent changes in its pattern at the start of 2021 due to subreddits such as r/wallstreetbets. In order to compensate for this, we decided to linearly interpolate the year-to-date open, high, low, and close data points to increase the size of the data set before doing any additional processing. In particular, before creating additional features, we wanted the data set to have around ten thousand rows. This required each day to have an additional seventy-one data points from linear interpolation.

If data augmentation methods are not used, the final data set size after all processing, would come out to be around one hundred and forty rows. However, with linear interpolation, the data set ends up being around seven-thousand eight-hundred rows after processing.

Although it is possible to train both decision trees and LSTMs using smaller data sets, using one would likely decrease the accuracy of both methods. This is due to the general higher variance of smaller data sets, which is further magnified by the volatility of this particular company's share prices year-to-date.

3.2 Sentiment Data

In order to acquire post titles and comments from Reddit, we decided to use the Pushshift API. This API allows scraping posts and comments from specific Subreddits. In particular the post titles of daily top posts based on the number of upvotes from two different Subreddits (r/GME and r/superstonk) were taken from January 1, 2021 to July 25, 2021. In addition to this, top daily comments from across Reddit with the substring 'GME' were also scraped and used for sentiment analysis.

After scraping, each day has a combination of around 30 post titles and comments. To analyze the sentiment of posts, we used SentimentIntensityAnalyzer from the Natural Language Toolkit (NLTK) Python library. The sentiment score of individual tokens in the library are sourced from ten independent human raters [1]. We calculate the average sentiment score using all post titles and comments within a day, and add this as a new column to the augmented data set using forward filling in Pandas. The sentiment scores range from extremely negative (-1) to extremely positive (1). Moreover, using the daily sentiment score, we classify each day as positive, negative, or neutral, and add this as another column to our dataframe in addition to the sentiment score column.

3.3 Technical Indicators

Technical indicators are values that can be calculated from open, high, low, close, and volume data. These values are used in practice to help predict future price movements. In particular, we chose three technical indicators: RSI, simple moving averages: one for fourteen days (SMA) and one for thirty days (LMA), as well as ADX.

ADX was chosen in combination with the RSI technical indicator. This is because ADX values help in detecting the severity of price trends; the higher its value, the stronger the trend. However, it does not know whether the price is going up or down. the RSI value compensates for this by detecting direction [2]. An RSI value of 30 or below indicates an undervalued condition, while a value over 70 indicates an overvalued stock [3].

Lastly, two moving average indicators are used, denoted by SMA (Small Moving Average) and LMA (Long Moving Average). These are both simple moving averages (commonly denoted as SMA as well). Another common technical trading strategy is the Golden/Death Cross trading strategy. This uses two moving averages, typically the fifty day and two-hundred day moving average values. When the shorter moving average is below the longer moving average, this means that the stock price is in a downward trend. However, at some point, the price will begin to go up. For this to occur, the shorter length moving average, must go higher than longer moving average. This is known as the Golden Cross and would indicate a "buy" entry point. On the other hand, when the shorter moving average crosses below the longer moving average, this would indicate an exit (sell) point, which is known as the Death Cross [4].

Although fifty and two-hundred day moving averages are the most commonly used, there is no reason other values for the number of days in each moving average cannot be used. It seems that as long as the shorter moving average is at most half of the longer and the smaller of the two is at least 15, the strategy can still be used [4]. Due to the actual number of days available for data, we decided to opt for the smallest possible value for each, to maximize the processed data set size.

4 Comparison of Methods without Sentiment Data

4.1 Decision Tree

4.1.1 Setup

First, a split is done on the augmented time series data set, such that the first 70% of data is used for training and the last 30% is used for testing. We then use GridSearchCV to tune the parameters for setting up the decision tree.

4.1.2 Parameter Testing

In order to find the best parameters to use, we do testing for maximum depth, minimum sample split and minimum sample leaves for the decision tree. The AUC (Area Under Curve) graph shows how much AUC we get for train and test while changing the parameters, and the graphs help with the parameter setting [5]. From this, we discover that the AUC scores in test data sets are much lower than the AUC scores in training data sets. This indicates that the decision tree perform better in training, which may indicate overfitting.

4.1.3 Cross Validation

Because we are using time series data, k-fold cross validation is not suited for this problem. Instead, we choose to use time series splitting with our best parameters [6]. We apply six incrementally sized intervals for the training data.

4.1.4 Results

Accuracy of the decision tree model is measured by seeing how many correct predictions the model can perform. As classification models will always predict values within actual labels, 1 represents up and 0 represents down.

Despite choosing different splits and parameter tuning, the decision tree gives low testing accuracy and AUC scores. We calculate the accuracy by dividing the number of correct predictions by the total number of predictions. Without sentiment data the results are as follows: value for AUC for training data is 0.67 and training accuracy is 68.60%. For testing, the value for the AUC is 0.50 and testing accuracy is 49.52%. From the value of AUC we think the decision tree model has no predictive value.

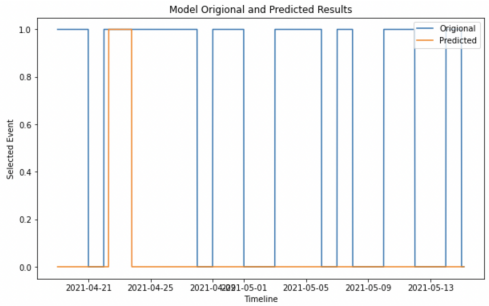

Figure 1: Line Plot for binary classification

We use a line plot to draw the binary classification for actual and predicted results. The graphs indicate that the decision tree generally predicts 'down'. The main reason for this situation may be the unbalanced data set. The data set has more rows that are labelled as 'down' than 'up', and for that reason will have more difficulty predicting positive closing price movements.

4.2 LSTM

4.2.1 Setup

The data is first normalized using a StandardScaler instance from Sklearn and split such that the first 70% of data is used for training, and the final 30% is used for testing. LSTMs also require that the input is a sequence of elements with an output. In order to create the proper sequence of data to feed into the LSTM, a TimeSeriesGenerator class instance is used, where the length parameter is initially set to 2 for testing.

```
<keras.preprocessing.sequence.TimeseriesGenerator
[[1 2]] => [3]
[[2 3]] => [4]
[[3 4]] => [5]
[[4 5]] => [6]
[[5 6]] => [7]
[[6 7]] => [8]
[[7 8]] => [9]
[[8 9]] => [10]
```

Figure 2: TimeSeriesGenerator example with length=2 for univariate data

The network consists of two LSTM layers followed by a single Dropout layer before the output. Each LSTM layer uses sigmoid as its activation function. SGD (Stochastic Gradient Descent) is used as the optimizer function, and MSE (mean squared error) for the loss function.

Since the LSTM we are working with uses multivariate data, the sequence of elements for input will be a chosen number of rows with all the features created are used to predict the label of the following row.

4.2.2 Hyperparameter Testing

In order to find the best parameters to use, some basic testing is done for different parameters. In particular, the length parameter of the TimeSeriesGenerator object used in the LSTM, the activation function in each hidden layer, the number of nodes, as well as the dropout value of the output is varied. (show photo)

Upon doing tests, it is found that the sigmoid activation function works best for both hidden layers. Additionally, a dropout value of 0.2, along with 64 units and 96 units in the first and second LSTM layers respectively give good accuracy scores.

The length parameter of the TimeSeriesGenerator input into the neural network is also varied. As the length parameter increases, there is a corresponding decrease in the accuracy of the model.

Furthermore, from testing different activation functions between ReLU, sigmoid, and softmax, it appears that using ReLU provides the most accurate results.

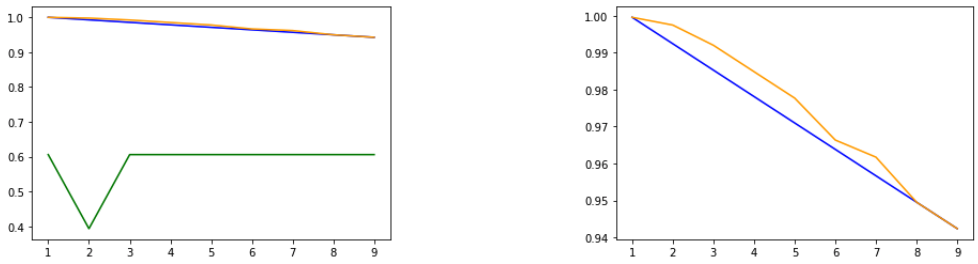

Figure 3: (Left) Accuracy as timesteps increase for TimeSeriesGenerator input for softmax=green, sigmoid=blue, and ReLU=yellow. (Right) A close up of sigmoid vs ReLU accuracy

4.2.3 Results

Accuracy of the model is measured by seeing how many ups (1) and downs (0) the model is able to predict. After the neural network outputs results, the values are inverse transformed from their normalized values, so that they may be better interpreted as binary results. If the inversely transformed

value is greater or equal to 0.5, then the value is seen as up (1). Otherwise, if the predicted value is less than 0.5, the value is seen as down (0).

From the graph, it appears that the best length for a TimeSeriesGenerator instance is 1, although this defeats the purpose of using LSTMs. The model is able to achieve an accuracy of 99.95% on the testing data set after training.

5 Comparison of Methods with Sentiment Data

5.1 Decision Tree

We add sentiment data to the data set and repeat the steps to set up the decision tree. We add a new sentiment score column for our data set. While applying with sentiment data the results are as follows: AUC for training is 0.67, training accuracy is 68.60%, AUC for testing is 0.50 and, and testing accuracy is 49.45%. There are no measurable difference in the accuracy with or without sentiment data.

5.2 LSTM

The setup and hyperparameter testing are almost exactly the same as without sentiment data, except for the new sentiment score column that is used in training and testing. Looking at the results in accuracy, there does not appear to be a measurable difference in the accuracy with or without the sentiment data. Although there appears to be, on average, a 9% decrease in loss, this value doesn't necessarily affect the accuracy level of the model. With sentiment data, the model is able to predict with 99.2% accuracy on the test data set.

6 Further Improvements

6.1 Using Actual Price Values

The reasoning behind using up and down labels as opposed to actual price values started because of a particular online example using a decision tree. The thought was that decision trees would have a lot of trouble handling non-discrete data, especially with the lack of data at the time. So to compensate for this, we decided to add more simplicity to our modelling by using a binary label for the change in closing prices.

Ideally, in the future it would be interesting to see more advanced methods for decision trees, such as using XGBoost versus LSTMs, and using RMSE (root mean-square error) for actual price values as opposed to binary up/down labels.

6.2 Adding Noise to Linearly Interpolated Data

Although using linear interpolation greatly increases the size of the data set, it loses some qualities that are typically for stock prices. Namely, the random fluctuations of values aren't present in our linearly interpolated values. Because of this, one future improvement would be to see how both models respond to the addition of noise to linearly interpolated data.

6.3 Bias in Sentiment Data

When scraping post titles and comments from Reddit, due to some recently removed functionality from the pushift API (aggregation), we were unable to retrieve comments from specific subreddits. Because of this, our scraper took top comments by number of upvotes across reddit, which always came from subreddits such as r/wallstreetbets or r/GME. The members of these subreddits have a strong incentive to stay positive since they may have invested money into Gamestop, and will likely not be shorting their positions. For that reason, our data may be missing the thoughts of those who have chosen not to invest into Gamestop, but may still have a valuable opinion regarding its trajectory.

Although the data will be sparse, it would be wise to gather data from all investing subreddits, not just the ones that are focused on one particular stock. This way the opinions of those who may not have

an incentive to say positive things about the company will offset those who do, and the sentiment score will better reflect the opinions of a more diversified group of people.

6.4 High LSTM Accuracy Issues

This coincides with the point about adding noise to the linearly interpolated points. After checking the code, there does not appear to be any data leaks. However, due to how the data is linearly interpolated, there is a severe amount of repetition of labels, and trends within the price values may be indirectly causing these labels to be inferred correctly. For example, due to the nature of the linear interpolation, on any given day, there will be 72 rows. Although these rows may have slightly different values, all their labels will all be identical.

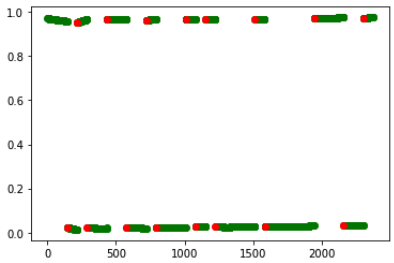

Figure 4: LSTM testing accuracy scatter plot with >99% accuracy (green=correctly labelled)

The above figure shows the LSTM exploiting this fact. It has a lot of trouble detecting changes in direction. The network only readjusts its prediction to match the new direction once it occurs. This shows that the network will fail given noisy data.

7 Conclusion

Based on the results of both methods, LSTMs provide more accurate results for price movement prediction on this particular data set. Although decision trees are more interpretable, they ultimately do not perform well for this problem. However, due to how the data set has been augmented, LSTMs are giving accuracy scores that do not truly reflect its abilities in predicting price movements. Additionally, the gathered sentiment data makes no measurable difference in the prediction accuracy of the chosen methods and approaches.

References

[1] C. J. Hutto and E. E. Gilbert, "vaderSentiment," VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text, Jun-2014. [Online]. Available: https://github.com/cjhutto/vaderSentiment. [Accessed: 12-Aug-2021].

[2] C. Stubbs, "Best adx strategy built by professional traders," Trading Strategy Guides, 07-Jun-2021. [Online]. Available: https://tradingstrategyguides.com/best-adx-strategy/. [Accessed: 12-Aug-2021].

[3] J. Fernando, "Relative strength index (rsi)," Investopedia, 02-Aug-2021. [Online]. Available: https://www.investopedia.com/terms/r/rsi.asp. [Accessed: 12-Aug-2021].

[4] A. Hayes, "Golden cross," Investopedia, 19-May-2021. [Online]. Available: https://www.investopedia.com/terms/g/goldencross.asp. [Accessed: 12-Aug-2021].

[5] S. Shrivastava, "Cross validation in time series," Medium, 17-Jan-2020. [Online]. Available: https://medium.com/@soumyachess1496/cross-validation-in-time-series-566ae4981ce4. [Accessed: 11-Aug-2021].

[6] B. Horton, "Calculating AUC: The area under a ROC CURVE," Revolutions, 22-Nov-2016. [Online]. Available: https://blog.revolutionanalytics.com/2016/11/calculating-auc.html. [Accessed: 12-Aug-2021].