Algorithmic Trading

Evan Goldstein, Joshua Hurd, Trent Jacoby, Kuzey Bektas, and Ethan Kawamara

April 2023

1 Abstract

Algorithmic trading is an automated system for buying and selling stocks based on predefined rules. In this paper, we describe our algorithmic trading system that utilizes deep neural networks to predict the direction of stock price movement. We trained our model on historical stock data from Yahoo Finance from the years 2010 to 2019, focusing on the period after the recession and before the COVID-19 pandemic. Our system uses technical indicators to identify patterns in the stock data and makes predictions based on these patterns. We discuss the challenges we encountered in choosing hyper-parameters and selecting the right technical indicators. We also describe the techniques we implemented, including recurrent neural networks (RNNs), long-short term memory networks (LSTMs) and a logistic regression model. Our system achieved promising results in predicting stock price movement and could potentially be used for trading in real-world scenarios.

2 Problem Space

Our trading system was designed to predict the direction of the stock price movement using historical stock data. We encountered several challenges in designing our system, including choosing hyper-parameters, selecting the right technical indicators, and obtaining enough data

3 Constraints, Rules and Objectives

The market is very unpredictable by nature and therefore recessions are too. Typically, in times of a recession the market is very volatile as evidenced by large gyrations in March of 2020. There is no model that could have predicted price movements during this time because in it itself recessions are for the most part unpredictable and dictated

largely by the unpredictable nature of human behavior. Each period we chose was unique as it had its own set of unique headwinds and tailwinds.

After the great financial crisis interest rates were essentially slashed to zero, leading the way to a zero-interest rate environment that would inevitably lead to economic growth but large-scale asset inflation. Although this is not normal as interest rates were the lowest, they have been for hundreds of years, it was for the most part without any major financial crisis. We viewed this as normal market operations, free of outside variables that are unpredictable such as the one-off financial stresses like the dotcom bubble burst of 2000. However, we risked the integrity of our model as we were only training our model on 'normal' behavior rather than what typically happens in the market. Additionally, we also choose this period as we view that at least in the United States is going into a recession and we wanted to train our model on post-recession price movement to prepare for our predicted recession. However, we at the end of the day do not know how the market will move if a recession occurs because every recession is unique.

Furthermore, another constraint we ran into is how much data we could obtain. Typically, in proprietary trading models, they are sourced to trade at high volume, or in other words on a second-by-second basis. The lowest time length we could obtain unfortunately was one day. Essentially, we were missing out on around 86,399 additional data points per day. Our training data only consisted of 2000 training points and if this were under a high-volume environment 172 million data points would have been used. To add on, with additional data points there would been less days used as 172 million data points would not be needed. In algorithmic trading there exists a curve called the efficient trading frontier that models the relationship between risk and cost. As high-volume trading is very expensive, but less risky, while low volume trading is cheap but very risky. The algorithms that individuals create try to obtain the efficient point on this relationship. As a result of how small our time length could get, we were constrained in finding the most optimal solutions.

4 Methodology

The first way we tried to implement our project was through a branch of a recurrent neural network called a long short term neural network. The way that this neural network worked was very interesting. The long short term (LSTM) neural network was very beneficial for our problem because we were working with time series data. Time series

data is essentially sequential data because it is all in order. The LSTM is very helpful when dealing with this sort of data because a LSTM neural network is able to pretty much retain information that it sees as important when going through iterations, and forget information that it does not see as helpful. This is especially helpful when trying to predict the stock market, because for our problem, there are a lot of trends that are within the training data, and when the LSTM neural network is able to analyze trends that it sees as helpful for making predictions, and trends that it sees as useless for making predictions, it is easier for us because we do not have to do as much work. We are able to just leave the work up to the machine. On top of being able to notice trends, the LSTM neural network was also able to look back at X amount of days in the training data so that it could analyze the trends on those sequences of days. In our code, we noted these as "backcandles", the name is a little odd, but it is what they were called every where else we looked. Being able to look back at an X amount of days was helpful because with everything in mind, it could analyze the trends more thoroughly, thus, getting a more accurate prediction because it could get rid of some outliers.

We required a separate model intended for comparative use: A logistic regression model. The logistic regression model was created to achieve better accuracy and reliability when predicting signals in a trading strategy. We utilized a training set in training the logistic regression model. The performance of the model was evaluated using a range of metrics: a confusion matrix, classification report, and accuracy scores. Using multiple metrics, we aimed at ensuring our model's accuracy was reliable and consistent. Scaling issues were addressed to obtain reliable results, and a 10-fold cross-validation model was employed to improve the model's ability to generalize to new data points making it more reliable in real-world scenarios. This technique helped to ensure that the model is trained and validated on different subsets of the data, minimizing the risk of overfitting. We then applied the logistic regression model to a trading strategy that used predictive signals. When the model predicted a positive signal, we purchased stocks, and when it predicted a negative signal, we sold stocks.

5 Challenges and Limitations

The proverbial challenge we faced was picking the hyper-parameters. The hyper-parameters we were particularly focused on were the features we were going to choose, the error function, the activation function, the lookback period, the number of epochs, how large or training data and test data were, and the number of hidden layers we should

use. The experiments were quite tedious to say the least as running the same code twice never yielded the same results. In terms of the features we chose, it was difficult to find what technical indicators worked the best together as to send the same buy and sell signals at the same time.

After doing some extensive research we learned that there exist 4 different types of technical indicators, and when used together they yield the best results. However, even with the new information it was difficult to choose the right technical indicators to use. Moreover, choosing the right error function was extremely difficult as we needed to find a function that maintains differences in data for a long period of time. Furthermore, we encountered this same problem with the activation function as it was difficult to determine which function worked best. Finally, when looking at the structure of neural networks it was difficult to assess what was the optimal structure as we risked over or underfitting which is something we were trying to avoid.

Data patterns and real-world events can conflict with each other, causing the model to garner conflicting representations and predictions from just the data. Because the models cannot precisely consider real-world events, the model will always be limited in its ability to predict future price movements. Along with misrepresentations of the real-world, the patterns of the data change over time. As the patterns change as time progresses, the model becomes more and more inaccurate the further it attempts to predict the future movement

6 Thinking Like a Human

The models that we provided are able to have some human though processes in some very interesting ways. To begin, with the long short term memory model that we implemented, it is able to think like a human in quite a odd way. As we stated previously in the model implementation, the long short term memory model is designed to model primarily sequential data. Basically, what this means is that it is able to capture information about the context of each input in relation to the ones that came before it. Which in some ways is similar to how humans are able to process language and other types of sequential information. Another way that it is similar to the way humans think is that it is able to remember certain trends and information in the data over a long period of time. This is done because the LSTM is able to store information for long periods in places called ”memory cells”, which is very on par with how human brains work as well. One last way this type of neural network is able

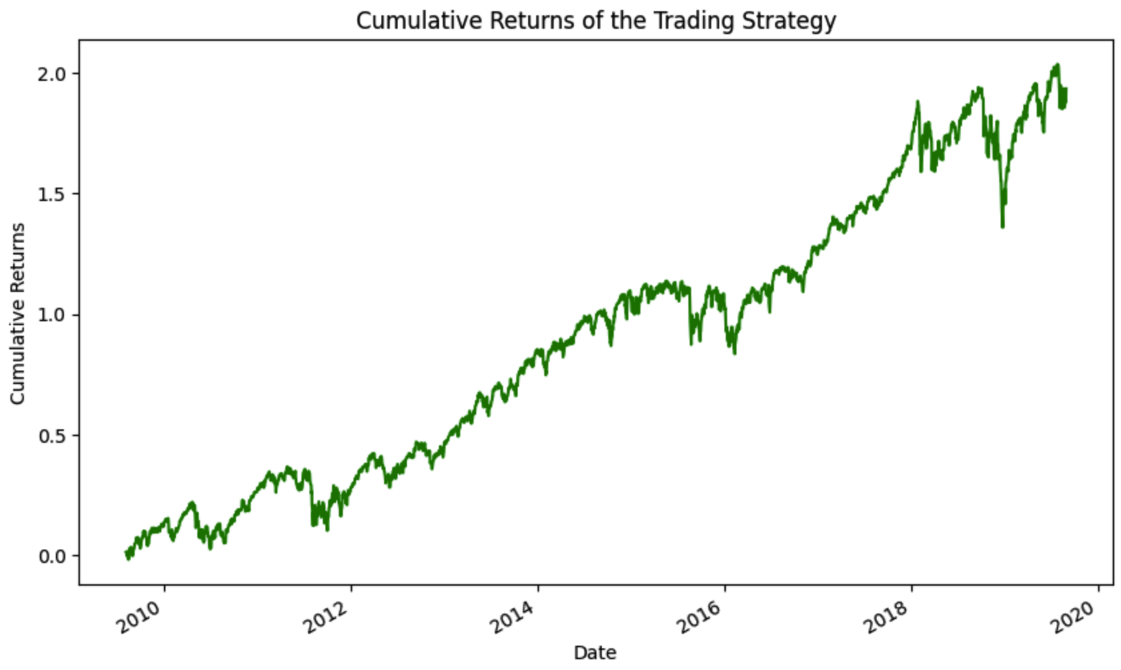
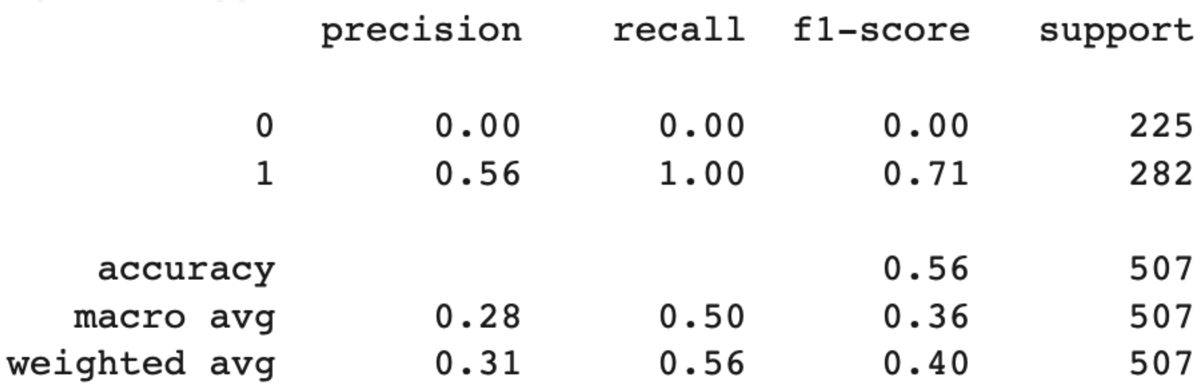


Figure 1: Logistic Regression Model Graph

to have a thought process like a human is because it is able to make predictions based on certain context. LSTM's are able to predict what the next input is likely to be, just like a human predicts what people are going to do or what they are going to say based off of their previous actions.

7 Results and Comparisons

Where we currently are right now, we are able to provide an accuracy of anywhere from 0.48-0.58, which shows us that a little more than half the time we are predicting it correctly. The final number that we approached as our accuracy was 0.56. Although this is merely 0.06 better than guessing if the market price will rise or fall using a coin flip, being consistently 0.06 more favoured to guess it right could mean that we can potentially make a profit over long-term investment. Furthermore, considering the fact that most stock market predictors created by professionals at the highest level are around 0.70 accuracy, we believe that our project is promising given our resources and time. At the end of the day, the stock market is directly tied to the real world, it's almost impossible to predict it correctly



```
              precision    recall  f1-score   support

           0       0.00      0.00      0.00       225
           1       0.56      1.00      0.71       282

    accuracy                           0.56       507
   macro avg       0.28      0.50      0.36       507
weighted avg       0.31      0.56      0.40       507
```

Figure 2: Accuracy Report (Logistic Regression Model)

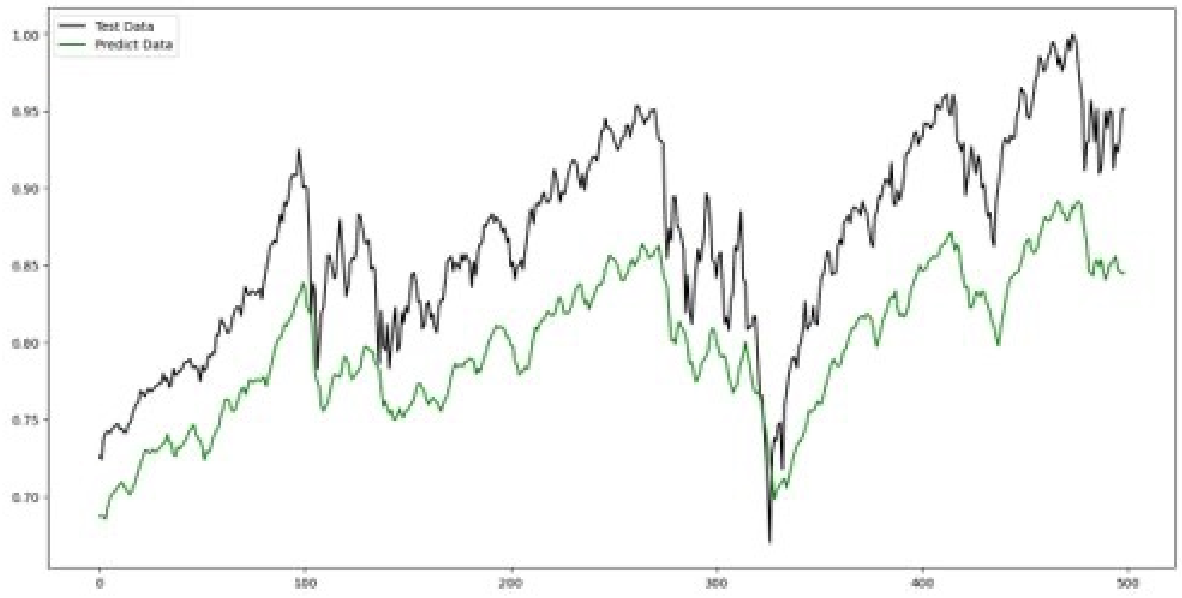


Figure 3: Regression Model Compared to the Testing Data

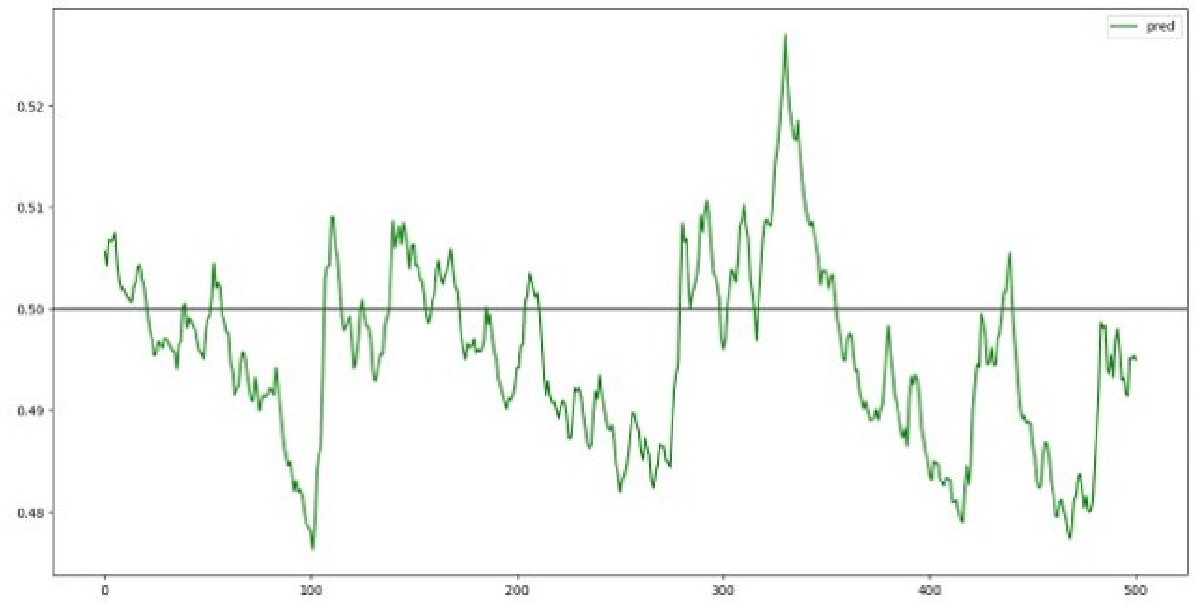


Figure 4: 0.5 Classification Line Graph

using only trading data. The following are some images showing the results we obtained:

8 Analysis

From our LSTM model, we determine that any times above the horizontal line at .5 indicate buy, while any times below indicate sell. Because of our challenges optimizing our technical indicators, number of nodes, and activation function, our results came out skewed, indicating to sell more than it should. From our logistic regression model, we were able to attain an accuracy rate between 46-58. Applying the model to a real-world scenario demonstrated its potential profitability, highlighting the model's practical applications. While we experienced imperfections in developing an optimal LSTM model, we accomplished our goal of producing a profitable model through our logistic regression model. Our logistic models proved to match trends and norms of attempts other by researchers and engineers.

Due to the variety of the challenges in optimizing our LSTM model, we recognized the limitations in predicting the stock market strictly based on data. Ultimately, the stock market is based on real-world events, constraining the perfection of the model. These results also

uncovered the complexity of the hyper parameters. As a result of our LSTM model's slight shortcomings, we recognize the troubles in picking the perfect technical indicators, number of nodes, and activation function. Lastly, because the accuracy of our logistic regression model is just slightly over 50%, we can firmly conclude that this is a difficult model to predict.

9 Potential Improvements

Seeing as though this project was only serving as a baseline so that we can further understand projects through the use of neural networks, there are a lot of ways that this project could be proved. I am not saying that in any means this project is bad or poorly made, I am really saying that working with the stock market is a challenge in itself, so getting higher accuracies requires a lot more data, as well as countless other types of models. For example, when a stock goes up or down, another attribute that is it dependent on is what people say on the news or in the media. What people say plays a very vital role in the fate of a stock either rising or falling because people are easily manipulated to either buy or sell their shares. This is able to be put in as an attribute by having a column that focuses on higher ups in the media saying positive or negative things that eventually caused the stock to rise or fall. This could be implemented with a convolutional neural network, which is something we did not learn about in this class, but is important to note if we were ever going to take this project any further.

10 Resources

https://www.youtube.com/watch?v=hpfQE0bTeA4 - Recurrent Neural Networks — LSTM Price Movement Predictions For Trading Algorithms – Code Trading

https://blog.quantinsti.com/machine-learning-logistic-regression-python/- Machine Learning Logistic Regression In Python: From Theory To Trading