Future Computing Technologies Lab: Creative Inquiry

SEMESTER REPORT
LSTM STOCK FORECASTING

April 25, 2020

Matt Franchi
Clemson University
Department of Electrical and Computer Engineering
mwfranc@clemson.edu

1 Introduction

For the semester project, I chose to try and predict future stock prices using historical data from Yahoo finance. I mainly focused on ETF analysis, as these stocks usually represent one of the major stock indexes (i.e. QQQ mirrors NASDAQ). The task that I focused on was regression, utilizing a Long Short Term Memory (LSTM) neural network. The motivation behind this project was a desire to delve into the stock market, as I've always viewed it as a sort of "untameable beast".

2 Materials and Methods

First, I focused on creating a functional python class to represent a stock, holding a member variable for the stock name and a Pandas dataframe (form shown in Table 1) with the stock's historical data.

Table 1: Head of GDX Stock Dataframe

Date	Open	High	Low	Close	Adj Close	Volume
2006-05-22	36.520000	37.290001	35.869999	37.230000	34.038883	197100
2006-05-23	37.750000	39.220001	37.750000	37.959999	34.706322	620900
2006-05-24	37.130001	37.570000	35.869999	36.520000	33.389740	638600
2006-05-25	37.180000	38.320000	36.980000	38.320000	35.035458	367000
2006-05-26	38.740002	38.740002	37.770000	38.549999	35.245750	269400

Second, I focused on preparing the data for the LSTM model. This portion of the project was the most difficult, as the Keras LSTM layer requires data in a specific shape of at least 3 dimensions. This section of the project entailed making the data column the index of the stock's dataframe; splitting the data into testing and training sets based off of a user-entered proportion; and normalizing the data using SciKitLearn's built in MinMaxScaler functionality, which scaled all of the financial data into a (0,1) range, desirable to the LSTM layer and the sigmoid activation used in the Activation layer.
Third, I focused on creating the actual model. This portion was fairly straightforward, due to an abundance of online resources being available on Keras model creation. However, I did face the dilemma of what kind of model to use - TimeDistributed or Sequential. Eventually, I settled on a Keras Sequential Model, as it was more manageable and easier to tweak. Once I settled on the type of model, I added Dense, Dropout, and Activation layers. Lastly, I decided to create a concatenated model, in order to use additional "indicator" metrics (at this point, only a simple moving average) to further improve the model. A diagram of the layers of the model that I created can be found on the next page.
Lastly, I evaluated the model, using graphs of predicted results, the mean-squared error score (MSE) and the R-squared score. These evaluation metrics allowed me to tweak my model, changing the batch size, sequence size, test/train split, and number of epochs.

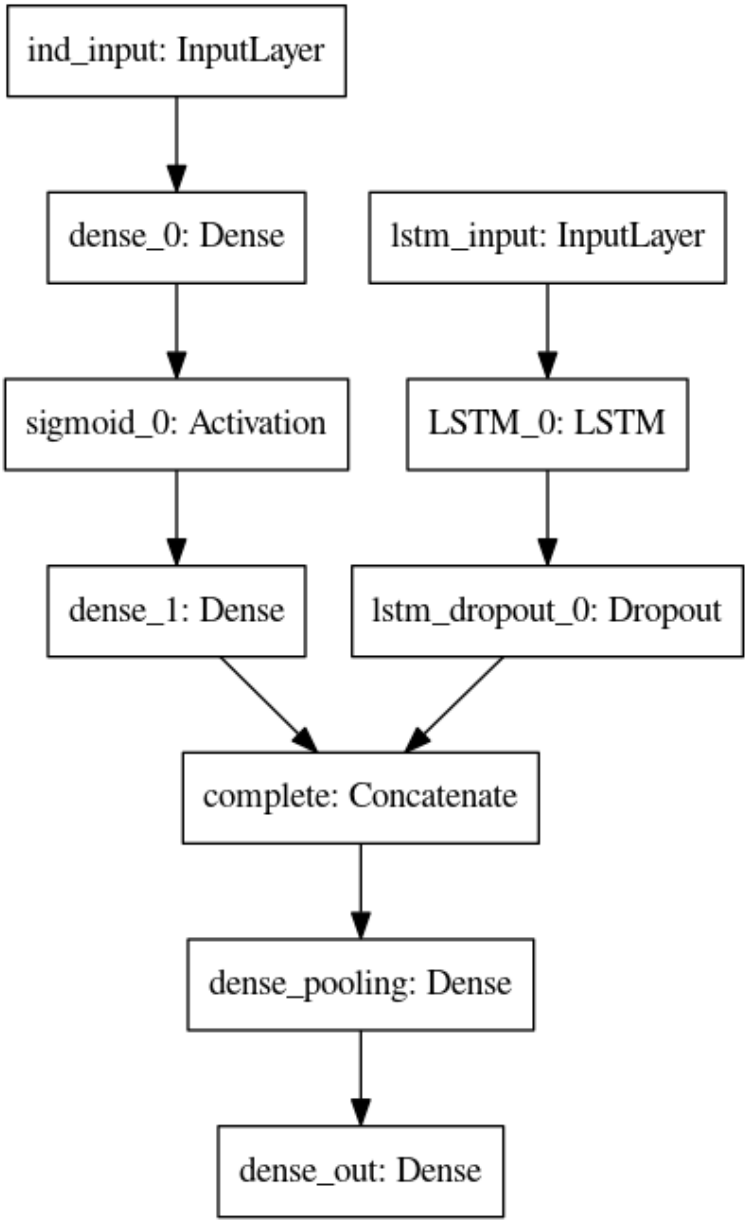

Figure 1: Model Layer Diagram

3 Results

Overall, my model performed fairly well, achieving an R-squared score of at least .95 after training for roughly 20 epochs. Table 2 illustrates this trend; however, performance seemed to level off after about 20 epochs, with the prediction accuracy actually decreasing from there. This is an indication of overfitting.

Table 2: Model Evaluation (GDX Stock) with Increasing Epochs

# Epochs	MSE	R^2
5	8.4917	.8879
10	1.9439	.9743
15	2.8393	.9625
20	1.7910	.9763
25	2.9908	.9605
50	3.0339	.9599

Below is a graph of the predicted vs. actual prices of the Apple, Inc. (AAPL) Stock, from its inception on December 12, 1980, to now. As expected, the model gets almost perfect accuracy on the training set, and then sees a small performance decrease when trying to predict the test set.

Figure 2: AAPL Forecast Plot

Figures 3 and 4 depict the change in model performance with an increase in epochs; the model performs much better when working with 20 epochs of training, relative to only 5 epochs.

Figure 3: GDX Forecast Plot - 5 Epochs

Figure 4: GDX Forecast Plot - 20 Epochs

Figure 5 depicts model performance on the QQQ ETF, which closely mirrors that of the NASDAQ index. I chose to test the model on the QQQ ETF because it represents an aggregation of several stocks. However, the model did not perform as well on this dataset, likely because it has a fairly erratic shape, with sharp increases and decreases occuring more frequently, relative to GDX or APPL. I predict that this shape is due to spikes in the QQQ volume around its start in 2006, in addition to global phenomena like the 2008 Recession. However, even with decreased performance, the model still accurately predicts whether the stock will increase or decrease the next day at a rate of >95%, which makes it useful for some sort of automated trading algorithm.

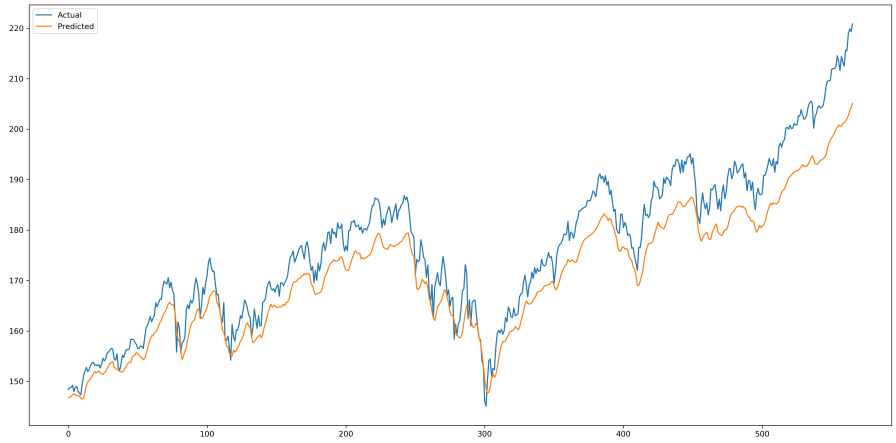

Figure 5: QQQ Forecast Plot

4 Experience

I enjoyed my experience in this Creative Inquiry (CI), as it focused on independent-learning and explored using existing tools, rather than creating them from the ground up. I've heard that the Computer Science Department's Applied Data Science course starts very low level, creating models and layers similar to those provided by Tensorflow and Keras, but from the ground up. For me, I'd rather work on innovating existing tools and putting them into action, rather than trying to re-invent the wheel.

I learned how to use matplotlib, creating effective data visualizations; SciKitLearn, taking advantage of built-in preprocessing tools and evaluation metrics; Tensorflow and Keras, designing a two-pronged neural network with several layers; and numpy and pandas, effectively organizing and manipulating complex datasets into series and dataframes.

The only challenge I had during the CI was carving out time to work through notebooks and engage my semester project; with independent learning comes the responsibility of taking the initiative to complete assignments. Aside from that, I had a few roadblocks in implementing my semester project, but nothing that reading through a library documentation or a search on StackOverflow couldn't solve.

While a data science class with Python is provided by the CPSC department, it ultimately is a 4000-level course, meaning that only upperclassman generally have the ability to take it. This CI provides a more accessible, streamlined way of learning these same concepts, while also giving us a chance to start a meaningful project that is easily expandable in the future.

5 Conclusions and Future Work

Overall, I am fairly satisfied with how my project turned out. I created an effective LSTM model that can predict future stock prices with about 95% accuracy, after being trained on 20 epochs. Ideally, I would have liked to actually create a "bot" that could make trading decisions based off of my model, and then sum up the total profit/loss of the bot's decisions. Aside from that, there are several other technical indicators that would likely bolster the performance of the model, including:

- Other simple moving averages (30 days, 90 days, 250 days)
- Exponential moving average
- Scalper alerts
- Stochastics

Additionally, the project could be expanded by creating a Portfolio class that simulates a real-time stock portfolio, thus allowing a "bot" to work with multiple stocks / models at once.