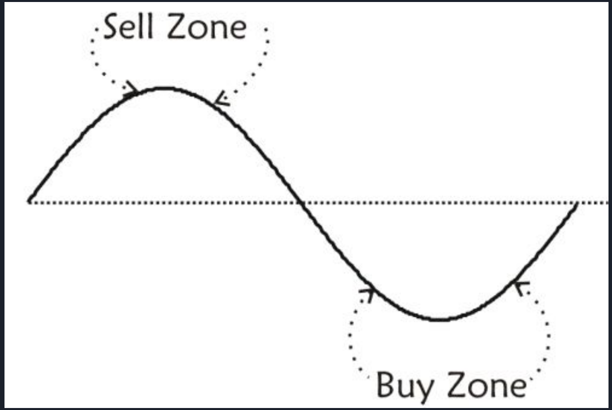
Stock Prediction Model

Clay Roberts, Pete Acevedo, Pedro Alvarado, Awaleh Houssein, Mateo Rivas

Introduction

- The stock market is a system that offers participants the ability to buy and sell portions of ownership in an organization.
  - In return a participant’s investments increases or decreases depending on the organizations success.
- Traders work to predict the market and optimize individual gains by buying and selling stock the moment before a stock swings positive or negative.

- Stock prediction models work to identify complex relationships that influence the behavior of a stock, allowing a market participant to invest in anticipation of a desired outcome.

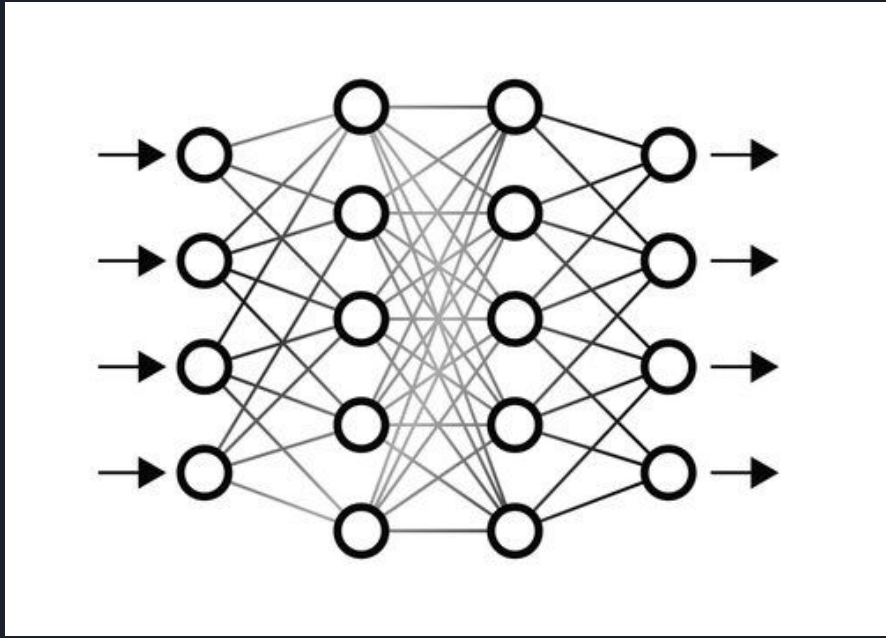


Research Problem

- Stocks are affected by a plethora of factors including illogical human reaction and emotion. Some of which make things like predicting the behavior of stocks inherently difficult.
- The research problem we address in this iteration of our project is the short term prediction of stock prices in spite of these factors, using a variety of machine learning methods.

Methods

- Long Short Term Memory (LSTM)
  - Recurrent Neural Network (RNN) that sequentially captures dependencies between data points.
- Moving Average Time Series
  - Model that forecasts performance using historical data.
- Self-Attention
  - A technique that eliminates the sequential nature of RNNs and allows for parallelization of relationship identification



ARIMA and Time Series

- Time series forecasting is the process of analyzing time series data using statistics and modeling to make predictions and inform strategic decision-making.
- Time series forecasting is useful for prediction of data that spans over a period of time. We used it for stock prediction in particular to try and predict how NVIDIA stock would move over time. Hence the time series aspect of it.
- Our particular model is called the ARIMA model and we achieved accuracy of up to 71% in some of our predictions.

- Autoregressive Integrated Moving Average (ARIMA) combines three components - autoregression, integration, and moving averages to model time series data.
- Autoregression (AR) uses a regression model that utilizes the dependent relationship between current values and lagged values of a time series.
- Integration (I) transforms non-stationary time series data into stationary data through differencing.
- Moving Average (MA) models the error term as a linear combination of error terms occurring contemporaneously and at various times in the past.

LSTM and The Transformer

- LSTM memory utilizes a RNN that maintains some “memory” of important context from information passed from node to node.
- Memory is updated at each step in the network, and past and new inputs are considered.
- Captures long term dependencies , which is useful for data with complex patterns.

- The transformer, and more specifically self-attention allows the model grasp dependencies between inputs that are “far” apart.
- A relationship map is created between input that highlights meaningful connections between their behaviors and values.
- Because computations aren’t performed sequentially, self-attention allows relationships to be attended to all at once instead of relying on the previous input.

Experimental Setup

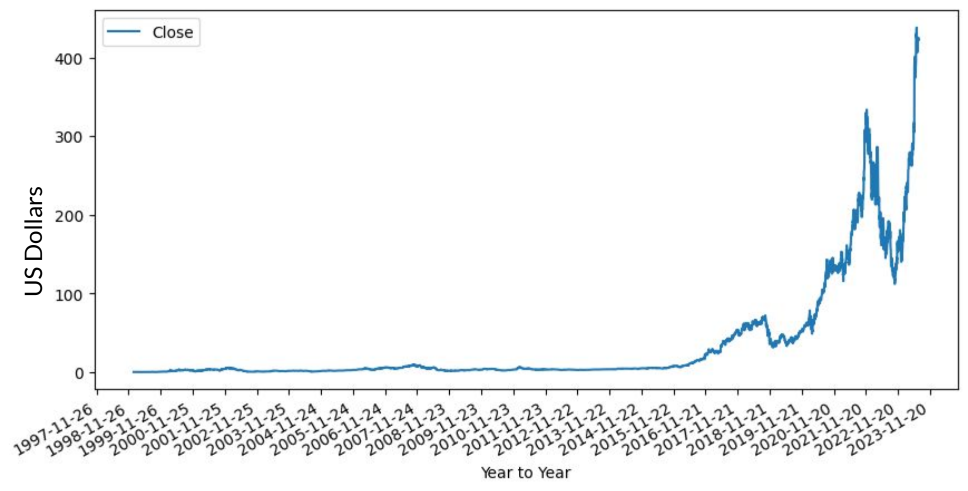
- The model will be evaluated using a two key metrics:
  - Mean Absolute Percentage Error (MAPE)
    - Average difference between predicted and actual price
  - Root Mean Squared Error (RMSE)
    - Adds focus to outlier events
- Predictions were based on predicting the closing stock price a single day in the future.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} (Predicted_i - Actual_i)^2}{N}}$$

$$\text{MAPE} = \frac{\Sigma \frac{|A - F|}{A} \times 100}{N}$$

Dataset

- LSTM, Moving Average, and Self-Attention were all trained and tested using NVIDIA Corporation stock performance from 1999 to 2023.
- The stock performance data was gathered from yahoo finance on July 11th, 2023 as a CSV file containing the Date, Open, High, Close, Adjusted Close, and Volume of shares traded each day.



Results

- Time Series:
  - MAPE: 0.0813
  - RMSE: 0.5396
- LSTM 32 Neuron, Single Dense Layer, 100 Epochs:
  - MAPE: 0.0563
  - RMSE: 0.2373
  -
- Two LSTM 50 Neuron, Single Dense Layer, 100 Epochs:
  - MAPE: 5.0222
  - RMSE: 1.4836
- LSTM 20 Neuron, Single Dense Layer, 20 Epochs:
  - MAPE: 0.0548
  - RMSE: 0.2495
- Self-Attention:
  - MAPE: 0.0695
  - RMSE: 0.3286

Deciphering Results

- Based on the Mean Absolute Percentage Error the 20 Neuron LSTM had the lowest error.
- Based on these Root Mean Square Error, the LSTM 32 Neuron, Single Dense Layer, 100 Epochs model had the highest accuracy with the 20 Neuron LSTM model as a close second.

- The LSTMs with 50 neurons had interesting behavior around when the stock really started growing in price. As the actual price began to trend upward, the predicted price followed a trend line that remained flat.
  - Most likely due to overfitting by the model.
  - The training data was fairly stable, whereas the testing data had significantly more movement.

Challenges Faced

- Training times on the Neural Networks were significant, even for the less complex models.
- When training a complex LSTM model, it was very for the model to overfit, which is the case that happened with the multilayer LSTM.
- Although fairly accurate predictions could be made on a short time scale, those predictions can’t be used very practically for much.
- Issues would arise when trying to find previous implementation to model out code after, where the libraries used had deprecated.
- The neural network output dimensionality was not always what was expect, and required analysis to extract the proper prediction value.

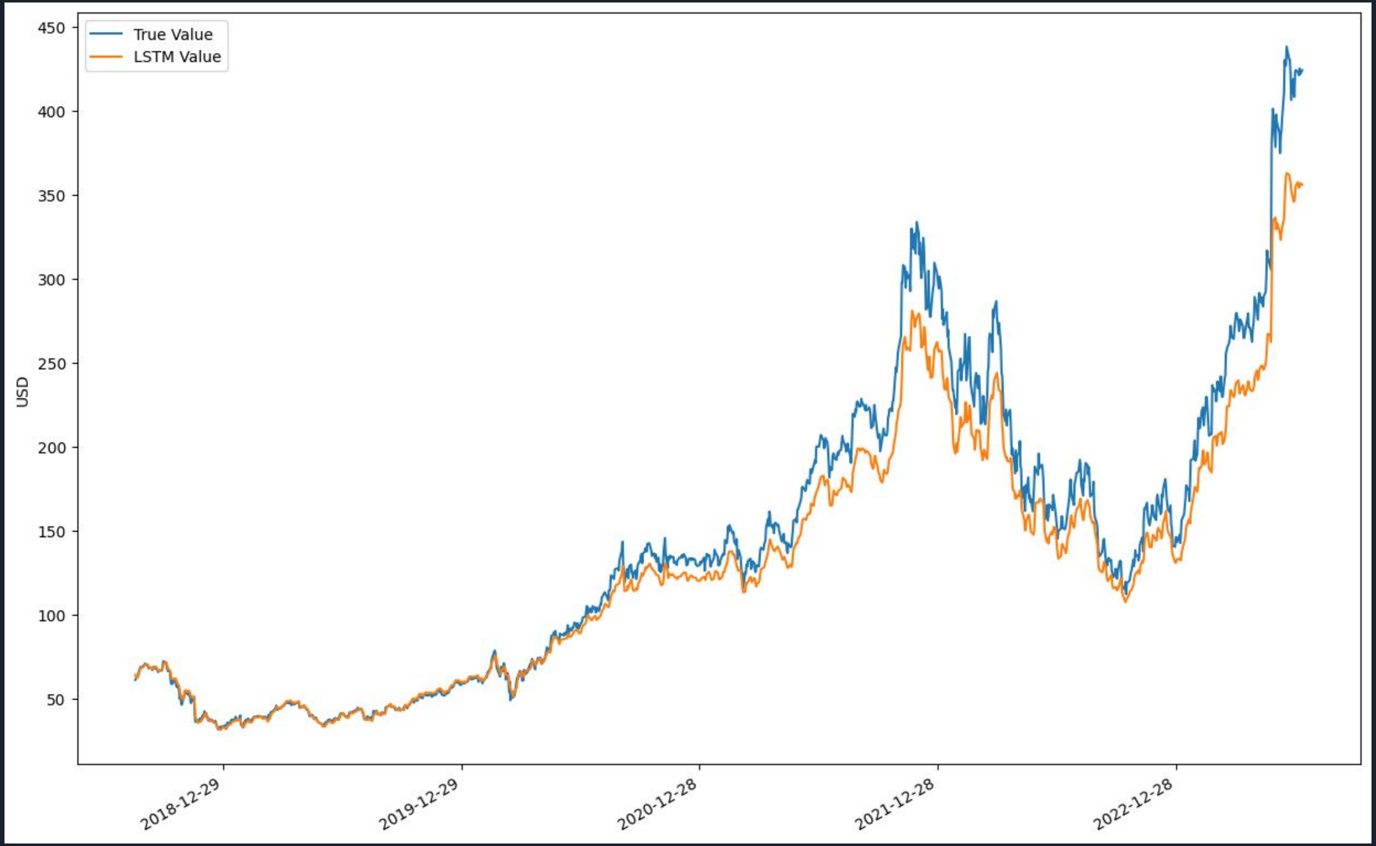


Figure 1: LSTM 32 Nodes, Single Dense Layer, 100 Epochs

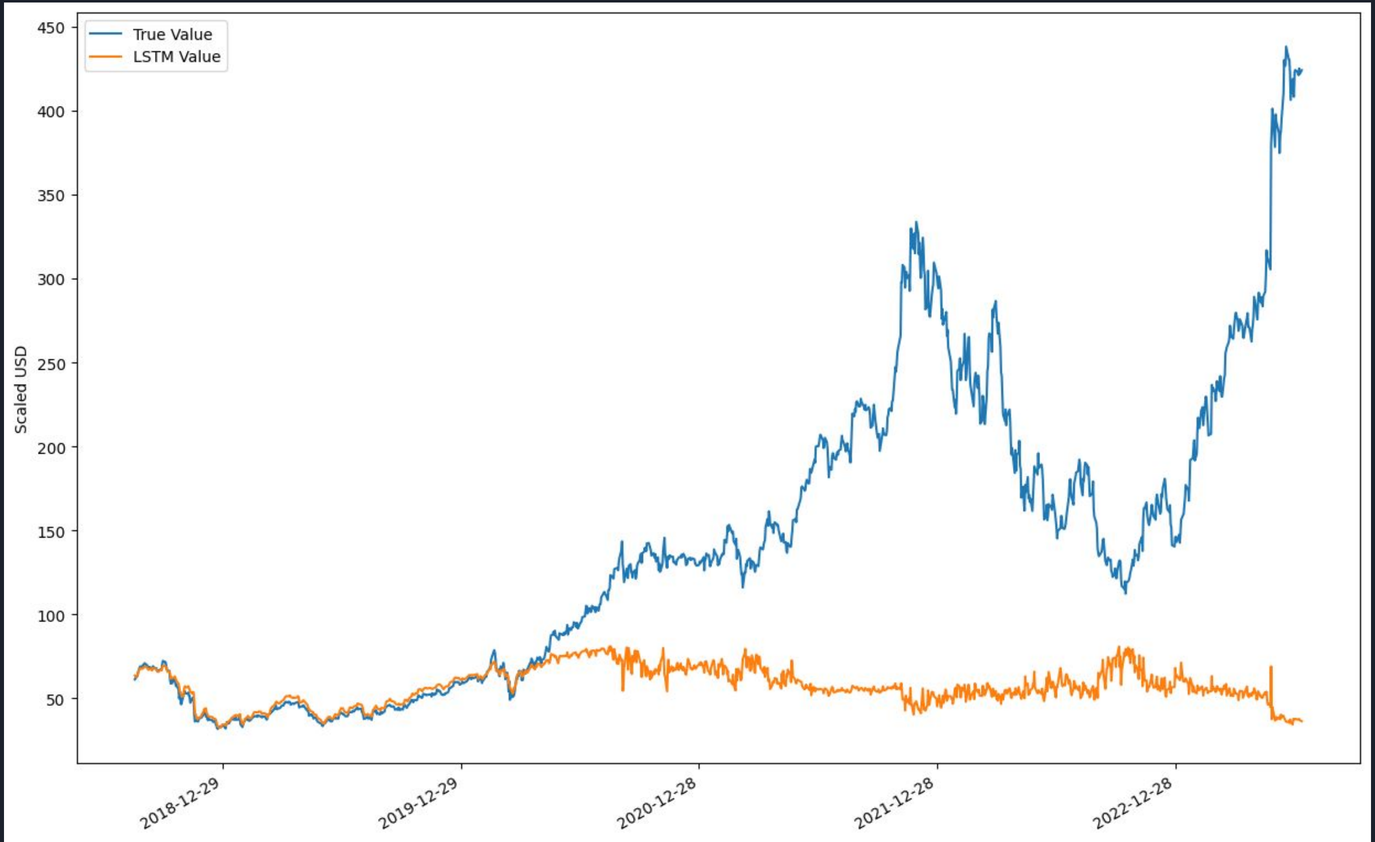


Figure 2: Two LSTM, 50 Nodes Each, Single Dense Layer, 100 Epochs

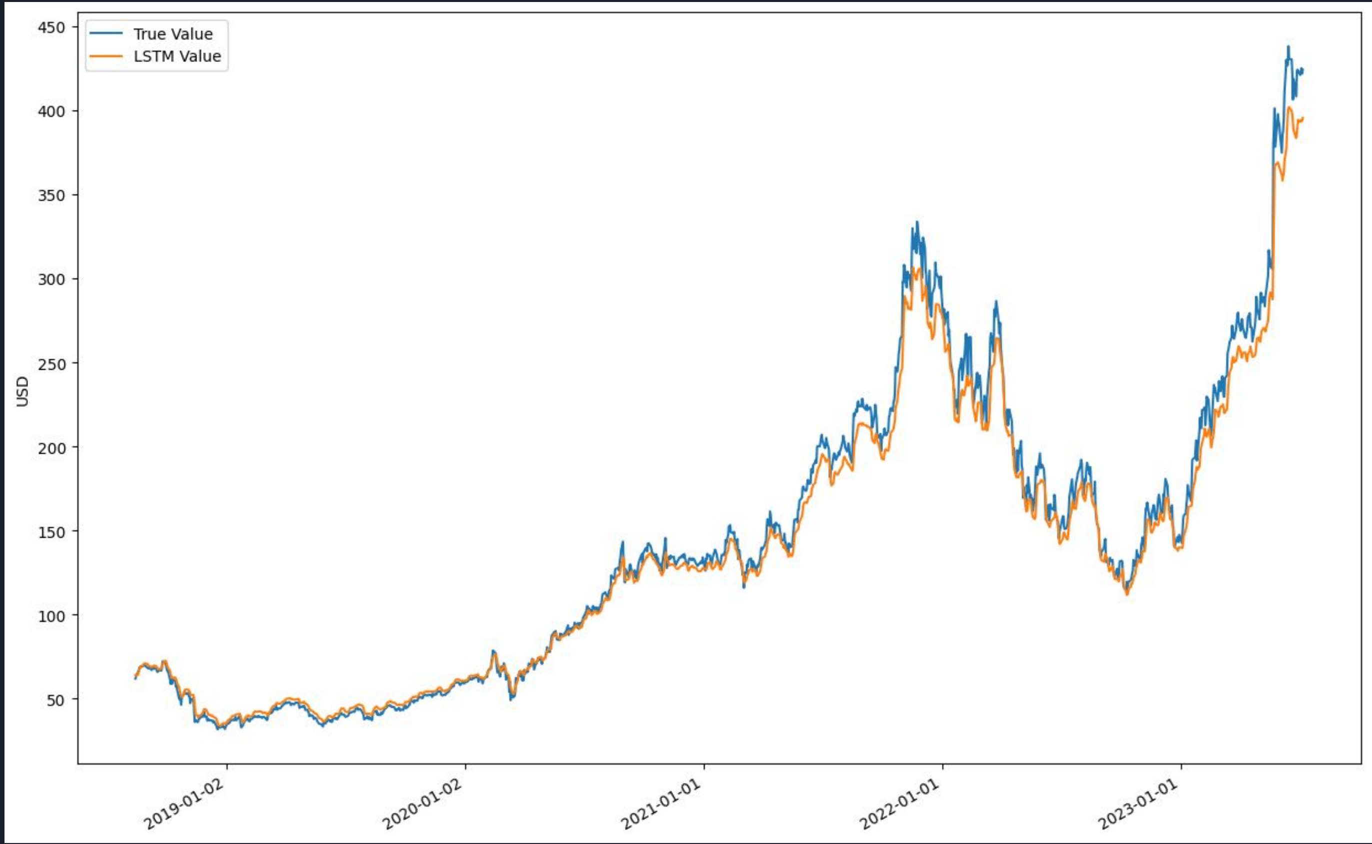


Figure 3: LSTM, 20 Nodes , Single Dense Layer, 20 Epochs

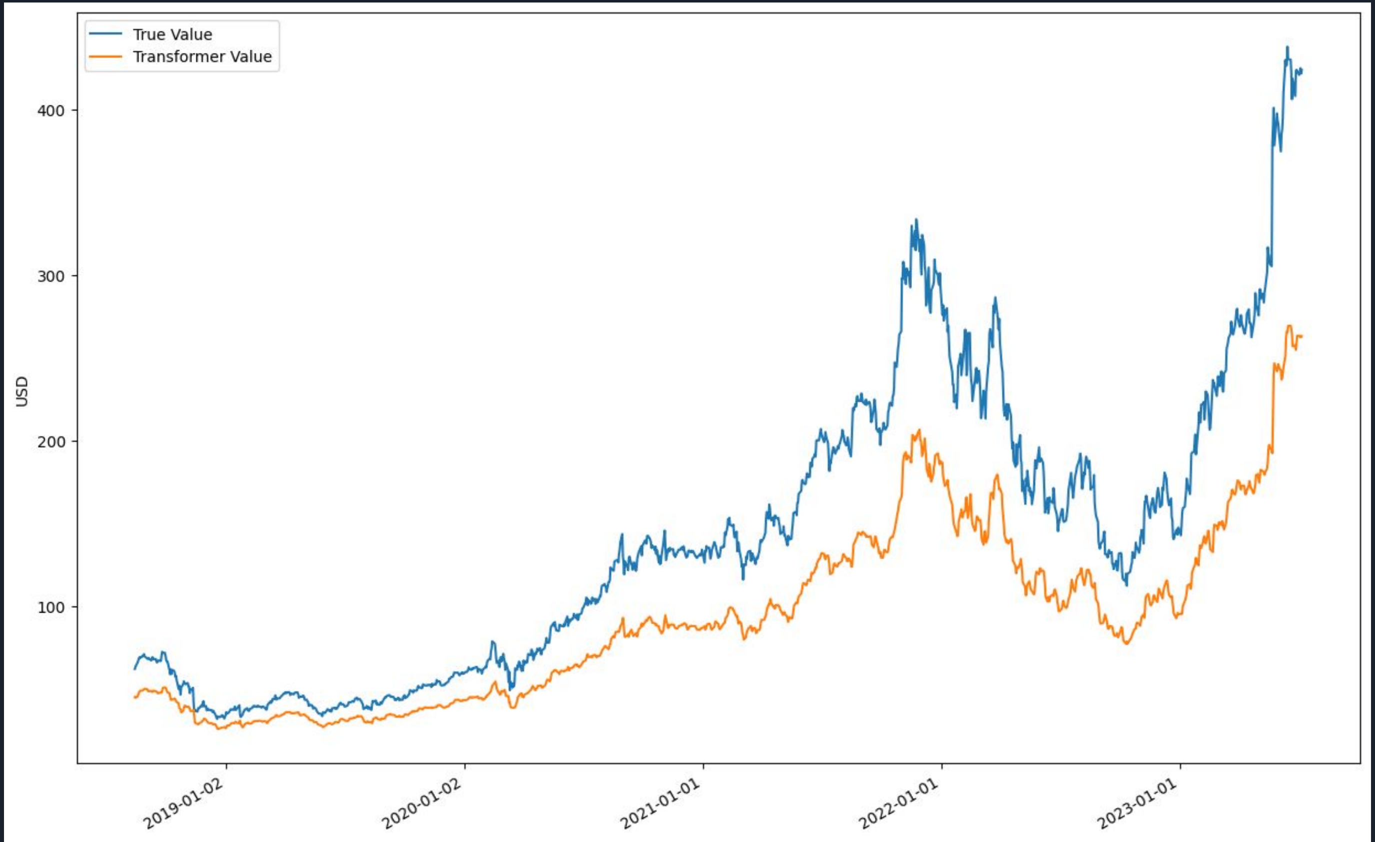


Figure 4: Self-Attention, 46 Heads