NVIDIA Next? Predicting Peaks and Spotting Sneaks in Stock Prices

Avinash Chaluvadi - CS5390

May 6, 2024

Contents

1 Abstract

What is Time Series Forecasting? A time series is a series of observations recorded over a period of time. These observations are dependent on the time component, which cannot be neglected. Thus, we must analyze this data while keeping the time component in mind. Time series forecasting is perhaps one of the most common types of machine learning techniques used in real-world scenarios. Time series forecasting refers to predicting future values from historical data by statistical analysis of trends and patterns from certain time series data. It falls under the unsupervised learning category but is often referred to as a self-supervised learning or supervised learning technique. Time series data can be complex to analyze, as it is challenging to find patterns due to the irregular components of the time series.

How is time series forecasting essential to AI? Time series forecasting is vital to AI-based predictive systems. These systems use algorithms to analyze historical data patterns, enabling them to forecast future events or trends. By utilizing time series forecasting, AI can provide accurate predictions that help in planning and decision-making processes. This capability is crucial for enhancing operational efficiency across various sectors such as finance, manufacturing, and retail. Therefore, time series forecasting significantly boosts the effectiveness of AI applications.

Motivation The motivation behind this project is to use deep learning for predicting the stock prices of Nvidia, which can be a complex and dynamic task. The goal is to develop supervised learning algorithms using Long Short-Term Memory networks (LSTMs) for time series forecasting. LSTM networks are particularly well-suited for financial time series data because they can capture long-term dependencies and patterns that are crucial for accurate predictions.
In conclusion, this paper presents a dataset of Nvidia's stock prices and demonstrates the effectiveness of LSTM models in predicting financial markets. The dataset and the models can serve as a benchmark for developing more sophisticated financial forecasting systems, which can potentially help investors and financial analysts in making informed decisions.

2 Dataset

The **"Nvidia Stock Price Dataset"** For the experimental study, I downloaded live dataset, namely Nvidia, from the Yahoo Finance website Yahoo Finance. The dataset includes historical stock price data for Nvidia over the past 10 years, enriched with several technical indicators to analyze market trends and stock performance. It details the date, as well as the opening, highest, lowest, and closing prices for each trading day, and also the volume of shares traded.

The Nvidia stock price dataset comprises 2518 samples. The data is first pre-processed to handle any missing values, normalize figures, and derive necessary technical indicators before being split into training and testing sets. Typically, 80% of the data is used for training the LSTM model, while the remaining 20% is reserved for validating/testing its performance. This split helps in evaluating the model's effectiveness in predicting stock movements under varied market conditions. The input features pertain to certain fields: **Open, High, Low, Close, Adj Close, Volume.**

2.1 Comprehensive Analysis of Nvidia Stock Performance (2014-2024)

The graph illustrates the daily closing stock prices of Nvidia over a decade, culminating in a record high of $950 in 2024.

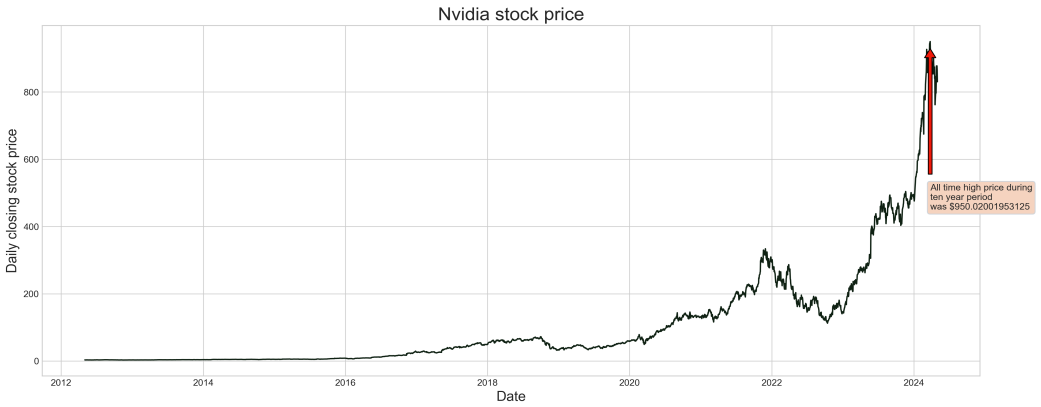

Figure 1: Nvidia Closing stock price and maximum price during 5 years

The graph clearly depicts a significant upward trend in Nvidia's daily stock prices during 2023 and 2024, illustrating a marked increase over these years.

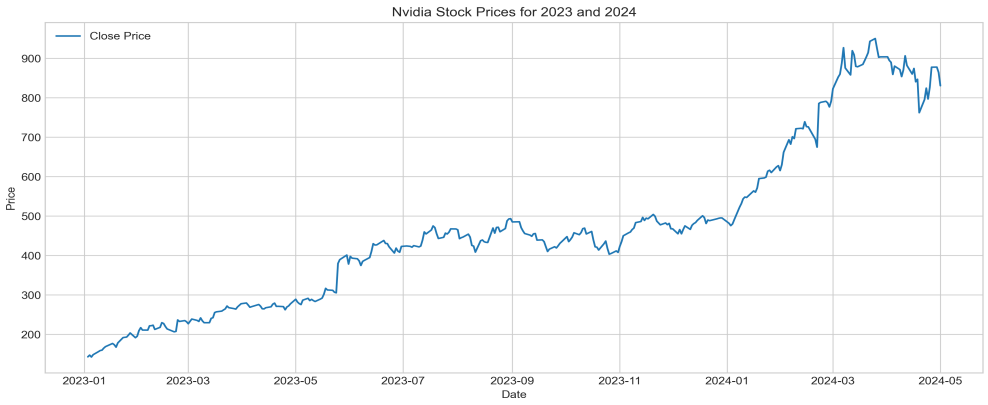

Figure 2: Nvidia Stock Prices During Peak Periods in 2023 and 2024

The graph represents Nvidia's daily trade volume from 2012 to 2024, highlighting a significant peak in 2018 reaching an all-time high of approximately 0.3 billion shares.

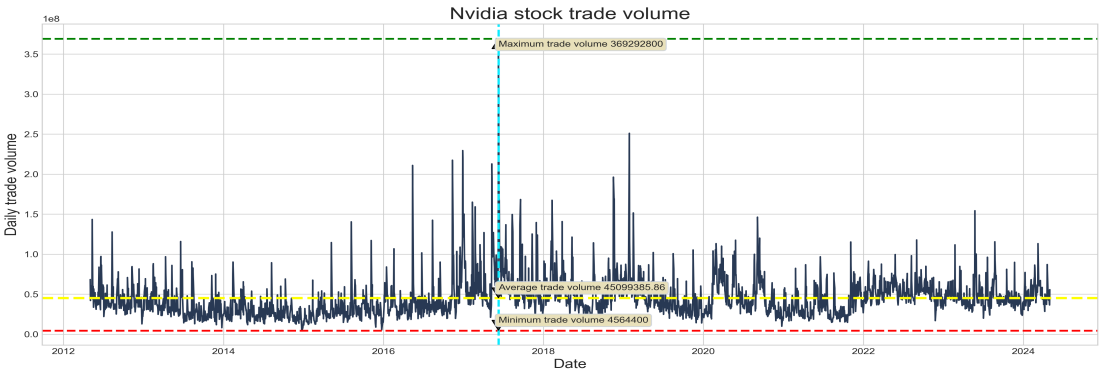

Figure 3: Trade volume of Nvidia stock over a period of 2014-2024

The plot illustrates the decomposed components of Nvidia's stock prices, including the trend, seasonal, and residual components. The trend subplot highlights a substantial upward movement post-2020, whereas the seasonal and residual subplots show relatively stable values across the observed period.

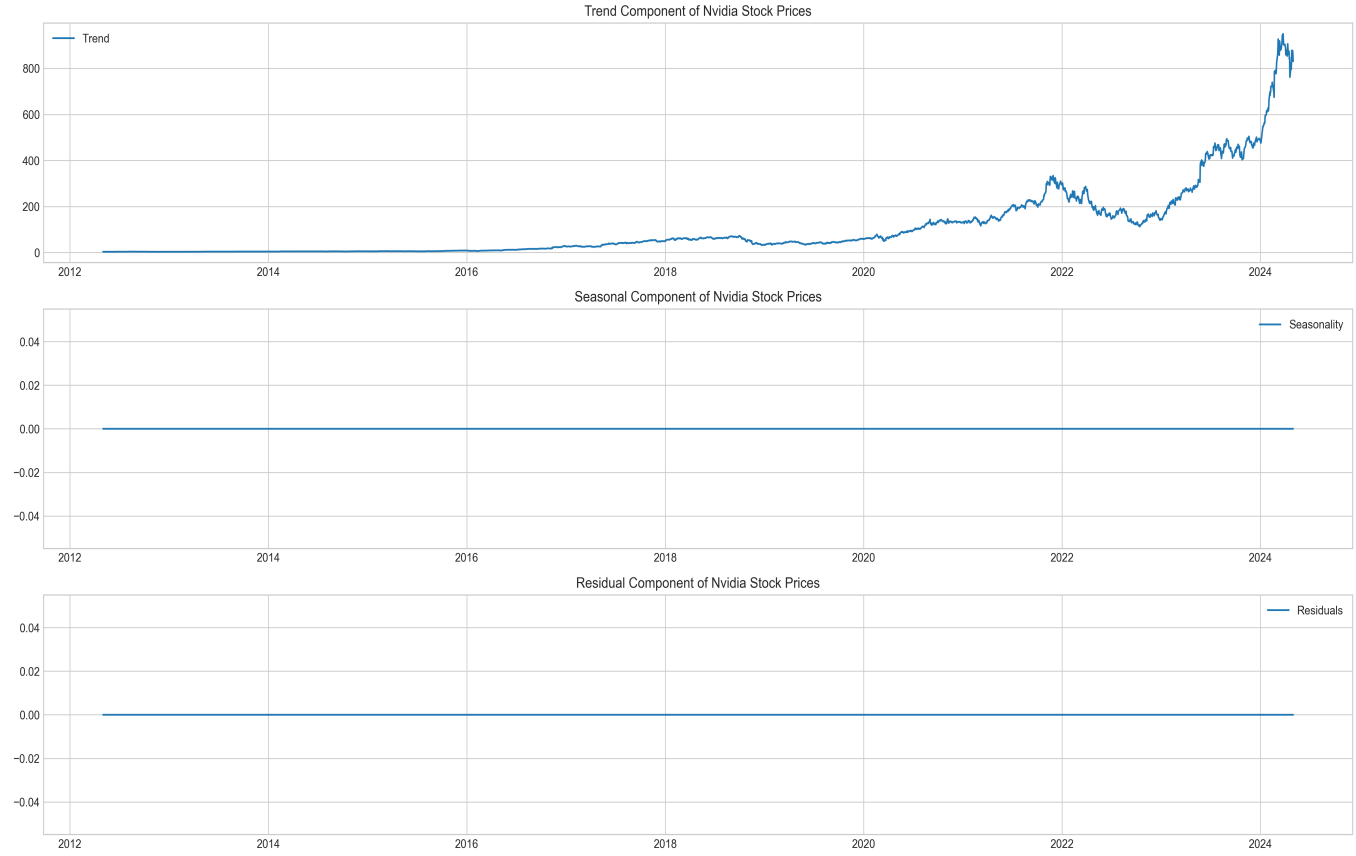

Figure 4: Decomposition Analysis of Nvidia Stock Prices

The plot illustrates the daily returns of NVIDIA stock from 2012 to 2024, highlighting notable volatility with several significant spikes, particularly around 2018 and 2020.

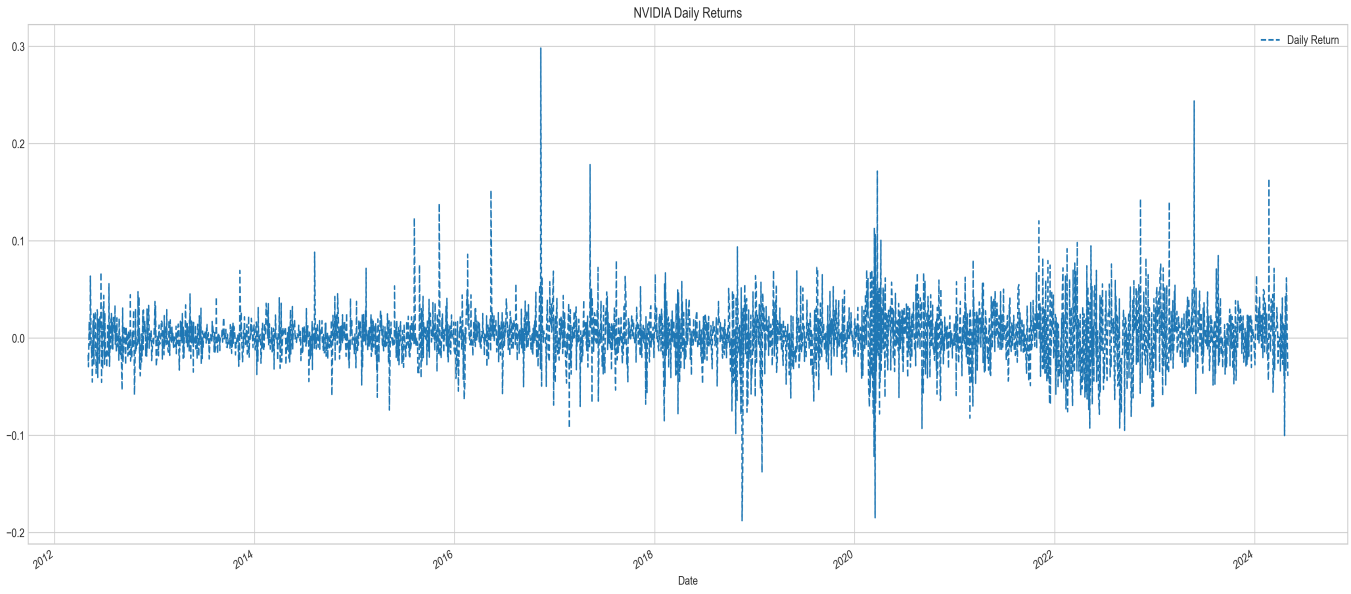

Figure 5: Daily return of the Nvidia stock on average

3 Data Processing

For the analysis, twelve years of Nvidia stock price historical data were utilized. The dataset was structured using two distinct windowing techniques to capture temporal dependencies in stock price movements, described below:

1. **90-Day Window Technique**
   The first technique involved using the closing prices of the previous 90 days as input features. This approach aims to predict the current day's closing price based on short-term historical data.

2. **180-Day Window Technique**
   The second method extended the window to include the closing prices of the previous 180 days. This broader window is designed to capture a more extensive range of historical price movements, providing a deeper context for the prediction of the current day's closing price.

3.1 Data Splitting

In the preparation of the dataset, the complete data was partitioned into training, validation, and testing segments. Specifically, 80% of the data was designated for training, while the remaining 20% was equally divided between validation and testing.

1. **For the first set of data (90-day window):** 2326 train samples, 212 validation samples, 211 test samples.
2. **For the second set of data (180-day window):** 2236 train samples, 122 validation samples, 121 test samples.

3.2 Data Normalization

Data pre-processing is crucial prior to data modeling, particularly to manage the non-uniform distribution of data. Normalization techniques are implemented to enhance the numerical stability of the optimization process and improve training. Normalization ensures that all values are scaled between 0 and 1, making outliers more apparent in the normalized dataset. For this project, the Min-Max Scaler was used.

Min-Max Normalization Formula

$$X_{\text{normalized}} = \frac{X - X_{\text{min}}}{X_{\text{max}} - X_{\text{min}}}$$

Overview of Data Post-Normalization

Date	Close	Day of Week
2012-05-02	0.000388	0.500000
2012-05-03	0.000330	0.750000
2012-05-04	0.000232	1.000000
2012-05-07	0.000288	0.000000
2012-05-08	0.000285	0.250000

4 Modelling

An LSTM-based neural network architecture was employed to build the model, aimed at Nvidia stock price prediction. Long Short-Term Memory (LSTM) architectures were fitted to the data. The LSTM network was developed using Keras, with several variations involving different numbers of LSTM and dense layers.

NOTE: The data was not shuffled due to its sequential nature, ensuring consistent results across multiple runs.

4.1 Overfitting

In the initial step, I aimed to identify the optimal architecture model that overfits the entire dataset without splitting. I experimented with different combinations of LSTM and dense layers, adjusting the number of neurons and layers to identify the best model configuration. The first model, a baseline, consists of a single LSTM layer and a dense layer. Subsequent models progressively increase the complexity up to a maximum of five LSTM layers.

4.1.1 Performance of the Overfitting Models

LSTM, Dense Layers(LSTM,Dense Units)	Loss	MAE	Windowed Data
1 LSTM(128)	6.3571e-05	0.0043	3 months
1 LSTM(128)	6.9634e-05	0.0045	6 months
2 LSTM(128-64), 1 Dense(8)	9.9356e-05	0.0059	3 months
2 LSTM(128-64), 1 Dense(8)	9.7770e-05	0.0057	6 months
3 LSTM(128-64-64), 2 Dense(8-1)	6.8383e-05	0.0044	3 months
3 LSTM(128-64-64), 2 Dense(8-1)	5.8841e-05	0.0043	6 months
4 LSTM(8-16-16-32), 3 Dense(32-16-16)	1.1268e-04	0.0067	3 months
4 LSTM(8-16-16-32), 3 Dense(32-16-16)	1.4419e-04	0.0068	6 months
5 LSTM(8-16-16-32-64), 2 Dense(8-8)	1.4442e-04	0.0077	3 months
5 LSTM(8-16-16-32-64), 2 Dense(8-8)	1.2227e-04	0.0071	6 months

Table 1: Model Performance for Different LSTM and Dense Layer Architectures - Overfitting

As shown in the table above, the basic architectures featuring 1, 2, or 3 LSTM layers, along with 1 or 2 dense layers, perform relatively similarly. However, architectures with 2 or 3 LSTM layers and 1 or 2 dense layers demonstrated slightly better performance during this iteration. Ultimately, the model configuration comprising 3 LSTM layers and 2 dense layers will be selected.

4.2 Learning Curve of Selected Neural Network Architecture

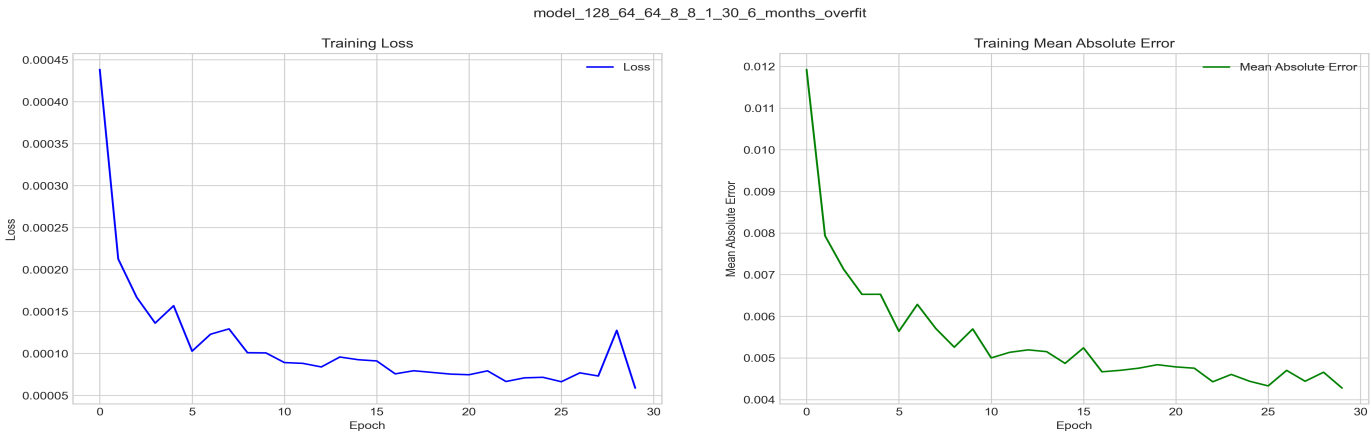

Figure 6: Curve showing change in over-fitting MSE and MAE vs epoch

5 Model Evaluation

In this step, I evaluated all the models established during the overfitting phase, exploring various combinations of LSTM and dense layers and adjusting the number of neurons and layers to determine the optimal model configuration. The batch-size parameter was also adjusted, and Mean Absolute Percentage Error (MAPE) was incorporated alongside Mean Absolute Error (MAE).

Model Configuration	Window Period	Validation MAE	Validation MAPE
1 LSTM(128)	3 months	6.152192	0.037394
2 LSTM(128-64), 1 Dense(8)	3 months	5.962914	0.035794
3 LSTM(128-64-64), 2 Dense(8-1)	3 months	6.445919	0.038820
4 LSTM(8-16-16-32), 3 Dense(32-16-16)	3 months	9.583087	0.059262
5 LSTM(8-16-16-32-64), 2 Dense(8-8)	3 months	11.107743	0.070483

Table 2: Model Performance of Architectures: Validation with 3-Month Windowed Data

Model Configuration	Window Period	Validation MAE	Validation MAPE
1 LSTM(128)	6 months	5.885041	0.039111
2 LSTM(128-64), 1 Dense(8)	6 months	5.801102	0.037494
3 LSTM(128-64-64), 2 Dense(8-1)	6 months	5.871108	0.038194
4 LSTM(8-16-16-32), 3 Dense(32-16-16)	6 months	10.011879	0.067544
5 LSTM(8-16-16-32-64), 2 Dense(8-8)	6 months	8.775115	0.059131

Table 3: Model Performance of Architectures: Validation with 6-Month Windowed Data

From the above tables, it's evident that the LSTM models trained using a 6-month window surpassed those trained with a 3-month window in forecasting future stock prices. Among the various LSTM configurations, the most effective model is either the one featuring 2 LSTM layers or the one with 3 LSTM layers. Both configurations performed comparably. Ultimately, the model configuration comprising 3 LSTM layers and 2 dense layers will be selected.

5.1 Learning Curve of Selected Neural Network Architecture

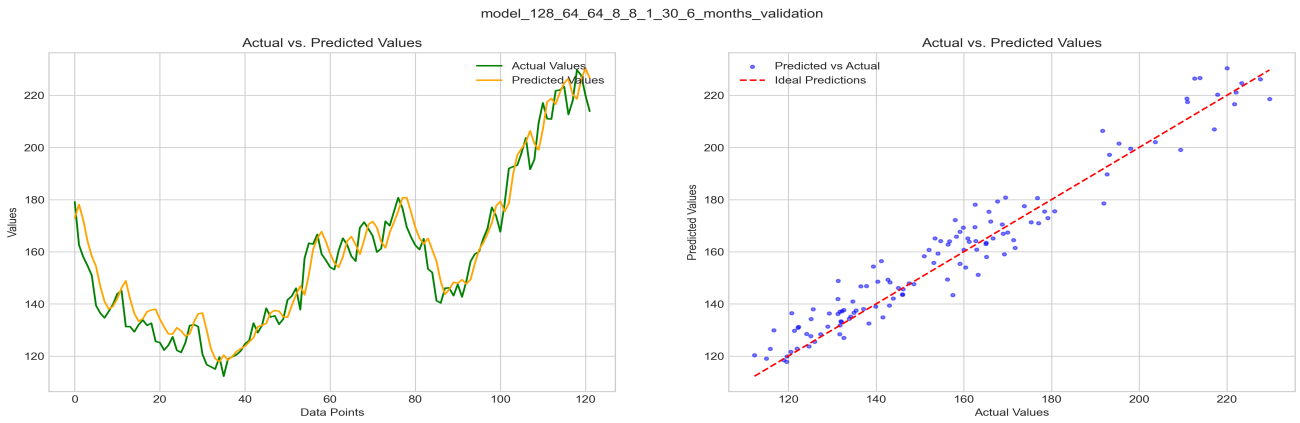

Figure 7: Comparison of Actual vs. Predicted Values and Scatter Plot of Predicted vs. Actual Values

6 Improving Validation Accuracy

6.1 Data Augmentation

In this step, I explored three different techniques to improve validation accuracy:

1. **Adding Jitter**: A small amount of noise (jitter) was added to the data to make the model more robust. I experimented with three noise levels: 0.005, 0.01, and 0.03.

2. **Scaling Technique**: Each time series value was scaled by a small factor. I tested three different scaling values: 0.01, 0.1, and 0.3.

3. **Magnitude Warping**: Each time series' magnitude was adjusted using a curve generated by 'cubic spline' interpolation with a set number of knots at random magnitudes: Sigma of 0.2 and knot of 1 and Sigma of 0.2 and knot of 4

Augmentation Technique	Period	Validation MAE	Validation MAPE	Test MAE	Test MAPE
Jitter 0.01	6 months	7.872104	0.050886	38.037239	0.052338
Jitter 0.03	6 months	12.814412	0.082905	92.741540	0.127254
Jitter 0.005	6 months	6.802560	0.044311	19.602779	0.029615
Scaling 0.1	6 months	6.249270	0.040435	99.480830	0.148395
Scaling 0.3	6 months	7.824560	0.049986	258.880185	0.359946
Scaling 0.01	6 months	6.142313	0.039434	18.831326	0.027169
Warping Sigma 0.2, Knot 1	6 months	10.648151	0.070893	203.769306	0.281648
Warping Sigma 0.2, Knot 4	6 months	10.371709	0.069998	204.302959	0.280349

Table 4: Model Configuration Results after Data Augmentation Techniques

The table above, displaying configurations with added noise and scaling magnitude augmentations, demonstrates a distinct noise or scaling constant for each row. These models were trained using a 6-month window with 3 LSTM layers and 3 dense layers to predict daily stock prices. From the results, it's clear that the LSTM model trained with a scaling constant of 0.01 and a 6-month window outperformed those trained with other noise and scaling constant values.

6.1.1 Learning Curve of Selected Neural Network Architecture

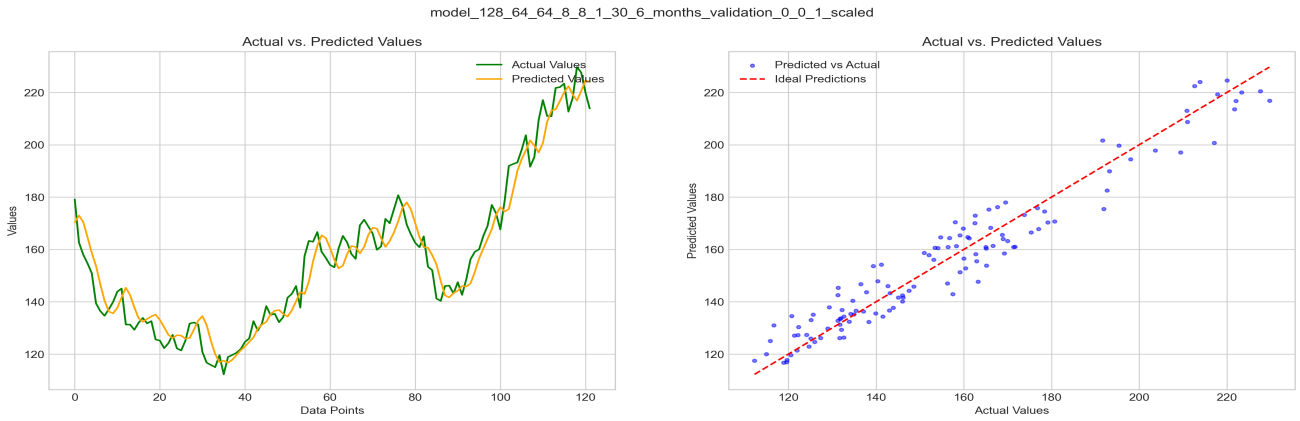

Figure 8: Comparison of Actual vs. Predicted Values and Scatter Plot of Predicted vs. Actual Values

6.2 Effects of Regularization

In this step, I explored two different ways to improve validation accuracy:

1. **Batch Normalization**: The first technique involved adding a batch normalization layer after the first LSTM layer and another two batch normalization layers after the first and last LSTM layers.

2. **Dropout Layers**: The second technique incorporated dropout layers by adding one after the first LSTM layer and two more after the first and last LSTM layers.

Layer Configuration	Period	Validation MAE	Validation MAPE	Test MAE	Test MAPE
1 Batch Normalization layer	6 months	6.799067	0.044139	247.337539	0.331576
2 Batch Normalization layers	6 months	8.567218	0.055651	168.914832	0.233823
1 Dropout layer	6 months	6.603170	0.042127	21.736594	0.030256
2 Dropout layers	6 months	7.053871	0.045771	19.266117	0.027392

Table 5: Model Configuration Results with Different Batch Normalization and Dropout Layer Configurations

The table above compares configurations featuring either one or two batch normalization layers or one or two dropout layers, with each row representing a unique setup. These models were trained over a 6-month window using three LSTM layers and three dense layers to predict daily stock prices. The results indicate that the LSTM model trained with a single dropout layer at a rate of 0.2 and a 6-month window outperformed those with one or two batch normalization layers or two dropout layers.

6.2.1 Learning Curve of Selected Neural Network Architecture

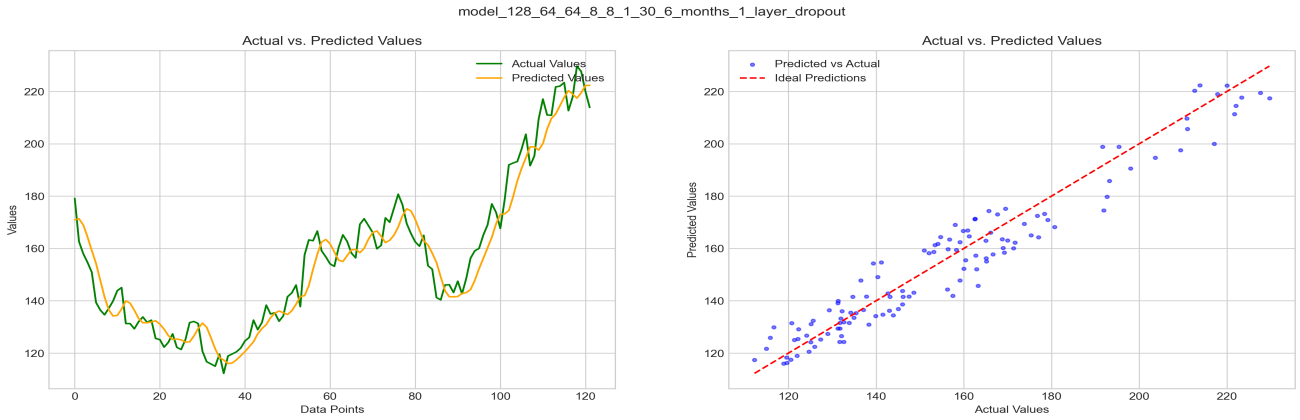

Figure 9: Comparison of Actual vs. Predicted Values and Scatter Plot of Predicted vs. Actual Values

6.3 Exploring Powerful Architectures and Pre Trained Models

In this step, I explored two different approaches. First, I designed a dense Bi-directional LSTM network configuration to assess if it could improve validation accuracy. Second, I utilized a pre-trained model like Prophet to evaluate its performance.

Model Configuration	Period	Validation MAE	Validation MAPE	Test MAE	Test MAPE
6 Bi-directional LSTM, 2 Dense	6 months	9.702311	0.064795	33.59239	0.046592
Prophet,	6 months			284.29	0.51141

Table 6: Model Configuration with dense network and Pre-trained model(Prophet)

From the table above, it's clear that neither the pre-trained model nor the dense LSTM architecture resulted in a reduction in loss or Mean Absolute Error (MAE).

7 Challenges Faced

1. I initially faced challenges in feature creation, as I used a 60-day window size, which did not reduce the loss due to the high volatility of stock prices. A 60-day lookback period was insufficient to capture meaningful patterns. Subsequently, I decided to increase the lookback to 90 and 180 days, which led to a significant reduction in loss.

2. I encountered difficulties in determining the appropriate model architecture and number of neurons to achieve over-fitting, despite trying numerous different architectures.

3. In the early phase of this project, I didn't use an early stopping technique which in consequence causing training error and time.

4. I encountered challenges in detecting anomalies, particularly in setting the threshold to identify a specific stock price as an anomaly. I decided to classify data points as anomalies based on their loss values; if the loss exceeded the threshold, the data point was considered anomalous. The threshold was determined from an overfitting model on the entire dataset.

8 Conclusion

In this report, I compared different deep learning models, including LSTM, using Nvidia data to predict future stock values. The findings reveal that the proposed methods effectively capture the data's interrelation. After evaluating various model architectures with different configurations of LSTM and dense layers, the highest validation accuracy was achieved with a model containing three LSTM layers (128-64-64) and three dense layers (8-8-1). Despite experimenting with data augmentation techniques such as jittering, scaling, and trend warping, no significant improvement in validation accuracy was observed. Furthermore, adding Batch Normalization and Dropout layers did not enhance performance, and more complex architectures, like a dense Bi-directional LSTM, led to overfitting. Pre-trained models, such as Facebook Prophet, also did not deliver better validation results.

Across the different phases, I have gained knowledge about several aspects, including data normalization, augmentation methods, the potential to overfit data by identifying features that are most important, effectively utilizing early stopping and model checkpointing techniques, analyzing learning curves to understand model behavior.

Further improvements, such as cross-validation and hybrid model architectures like LSTM and CNN, are needed to enhance the project's performance. This project has enabled the acquisition of new knowledge in developing LSTM-based neural networks and provided insights into understanding their behavior.

9 References

1. Data Augmentation techniques in time series domain: A survey and taxonomy
2. Correct way to Pre-Process Time Series
3. Time Series Prediction using LSTM Keras