Stock Price Prediction Report

By: Mohsin Syed

July 21/2024

Table of Contents

Table of Contents **2**
Problem Statement **3**
Context 3
Purpose and Objectives 3
Data **4**
Data Collection 4
Data Wrangling 4
Exploratory Data Analysis (EDA) **5**
Visualizing Historical Stock Prices 5
Feature Engineering 6
Moving Averages (MA) 6
Relative Strength Index (RSI): 7
Moving Average Convergence and Divergence (MACD): 10
Daily % change: 12
Earnings Impact: 12
Correlation Matrix 13
Stock matrix Correlation 13
Feature Correlation 15
Modeling: **17**
Baseline Model - Dummy Regressor 17
Traditional ML Model 18
Deep learning Model 21
Recommendations for Improvements 26
Conclusion **27**

Problem Statement

In the dynamic and often unpredictable world of stock markets, accurately predicting stock prices remains a significant challenge due to the myriad of factors influencing market behavior, including economic indicators, market trends, and company-specific events. This capstone project aims to develop a predictive model using machine learning (ML) and deep learning (DL) algorithms to forecast the stock prices of Tesla, Nvidia, and AMD. The primary objective is to create a robust model that can provide valuable insights to investors and financial analysts, ultimately enhancing decision-making in the stock market.

Context

Investors and financial analysts continually seek advanced methods to gain insights and predict stock price movements in the highly volatile stock market. Companies like Tesla, Nvidia, and AMD are highly influential in the tech sector, with their stock prices reflecting broader market trends and investor sentiment. Leveraging ML and DL techniques holds the promise of improving the accuracy of stock price predictions compared to traditional statistical methods. However, the inherent complexity and volatility of the stock market pose substantial challenges to the development of a reliable predictive model.

Purpose and Objectives

This technical report provides an in-depth analysis of the stock prediction models developed, including their architecture, algorithms employed, evaluation metrics, and performance results. Despite the efforts to leverage ML and DL techniques, the models developed in this project did not achieve the desired level of robustness and accuracy. This outcome highlights the complexities involved in predicting stock prices and the limitations of current methodologies.

The report also explores the lessons learned during the development and evaluation phases, shedding light on the various factors that impacted the model's performance. Furthermore, it offers recommendations for potential improvements that could enhance the predictive accuracy of future models.

Data

Data Collection

For this project, I utilized the Yahoo Finance API to gather historical stock price data, trading volumes, and other relevant financial indicators for Tesla, Nvidia, and AMD. The data collected includes:

- **Daily stock prices**: open, high, low, and close prices
- **Trading volumes:** the number of shares traded each day

The dataset spans the last ten years, providing a comprehensive view of the stock performance and market activity over an extended period. This extensive historical data forms the foundation for developing and evaluating the predictive models.

Data Wrangling

One of the key aspects of data wrangling is ensuring that the dataset is clean and properly formatted for analysis. In this project, the following observations were made during the data wrangling process:

1. **No Null Values:** All rows in the dataset were complete, with no missing or null values. This eliminated the need for imputation or data cleaning related to missing data.
2. **Time Series Index:** The date column was imported as the index, converting the DataFrame into a time series format. This is crucial for time series analysis and modeling, as it allows for proper chronological ordering and manipulation of the data.

Given these conditions, minimal data wrangling was required. The dataset was ready for exploratory data analysis (EDA) and subsequent modeling steps without significant preprocessing. This streamlined the workflow and allowed for a more efficient focus on model development and evaluation.

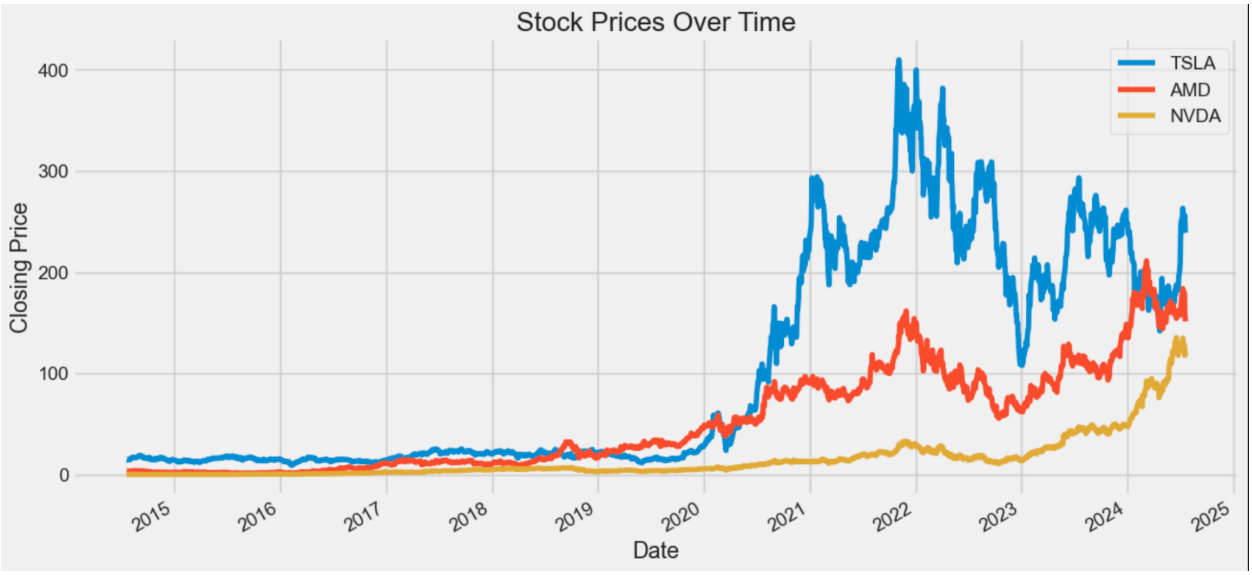
Exploratory Data Analysis (EDA)

In the EDA phase, the focus is on visualizing the historical stock prices and examining the dataset to uncover patterns, trends, and relationships within the data. This section covers the following key activities:

- **Visualizing Historical Stock Prices:** Creating visual representations of the daily stock prices (open, high, low, close) and trading volumes over the last ten years to identify trends and patterns.
- **Feature Engineering:** Developing new features that could potentially enhance the predictive power of the model. This involves creating additional variables derived from the existing data, such as moving averages, volatility measures, and momentum indicators.
- **Correlation Analysis:** Constructing a correlation matrix to analyze the relationships between different features and their impact on the close price. This helps to identify which features have the strongest correlation with the target variable, the close price.

Through these steps, the EDA aims to provide a comprehensive understanding of the dataset, highlight important features, and set the stage for effective model development and evaluation.

Visualizing Historical Stock Prices



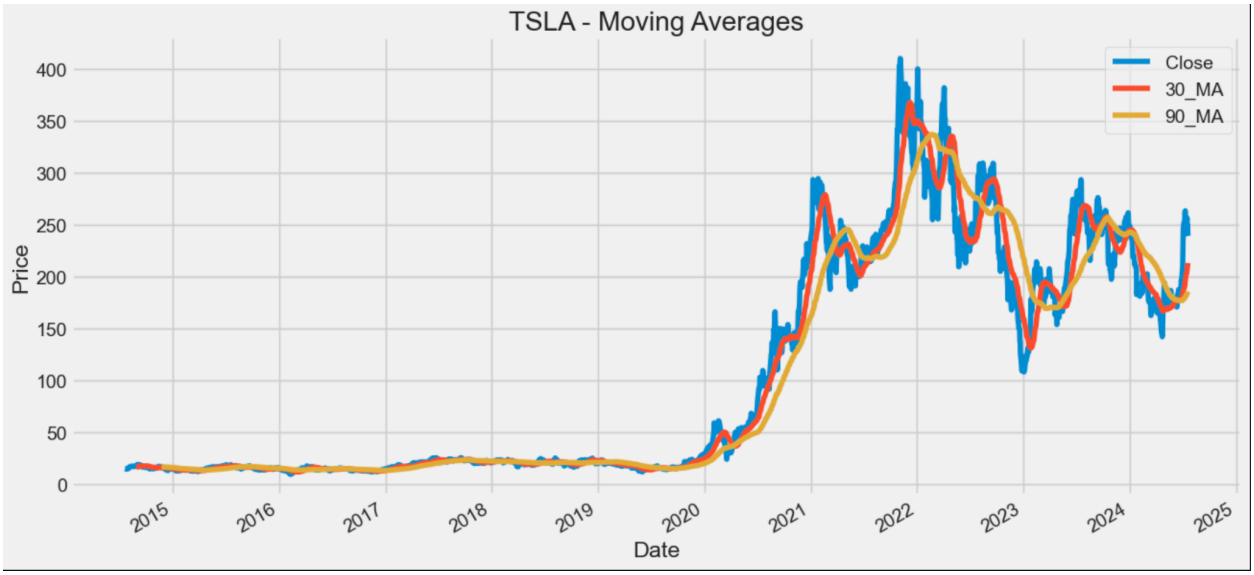
Above is the Historical Stock price trend for TSLA, AMD, and NVDA.

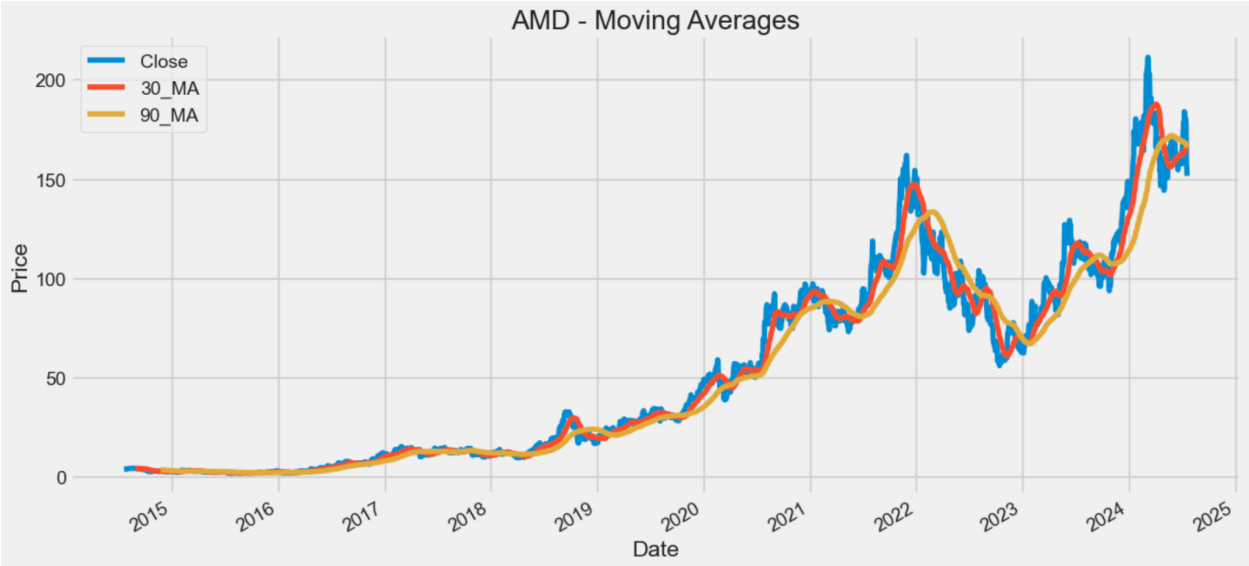
Feature Engineering

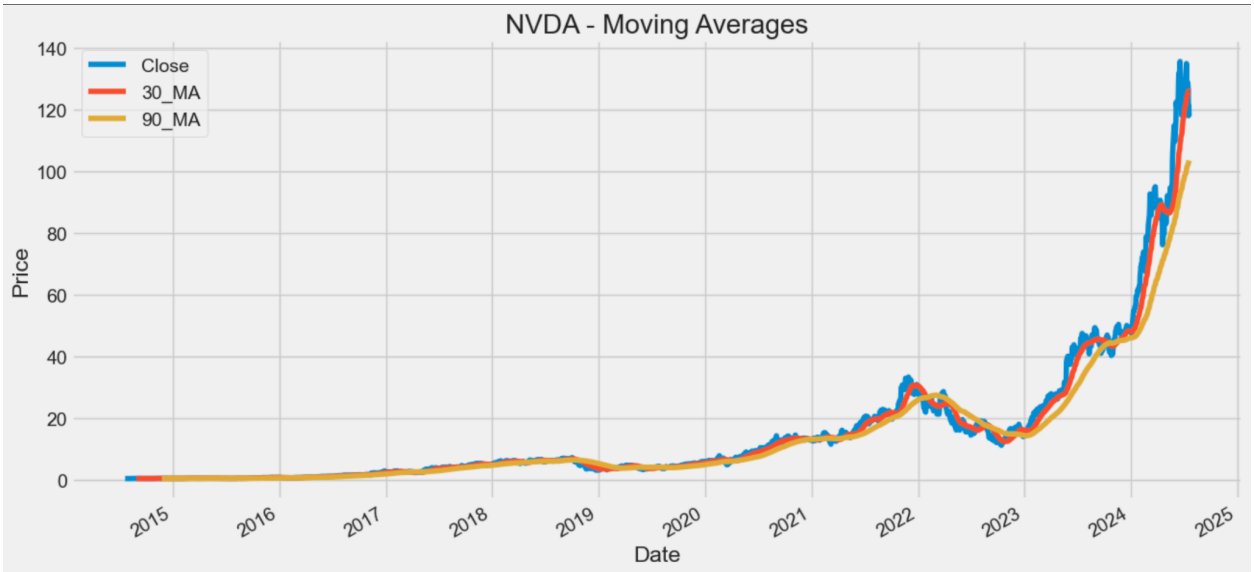
New features are engineered to potentially enhance the model's predictive capabilities. These mostly include technical indicators such as Moving Averages (MA), Relative Strength Index (RSI), Moving Average Convergence Divergence (MACD) and few other which will be highlighted below.

Moving Averages (MA)

Below we have 30 days and 90 days MA for each stock. When the stock price crosses above the moving average, it could signal a potential uptrend (buy signal). Conversely, when the stock price crosses below the moving average, it might indicate a downtrend (sell signal). Comparing the 30-day and 90-day moving averages can also help identify trend changes; for instance, a 30-day moving average crossing above the 90-day moving average could suggest a bullish trend.







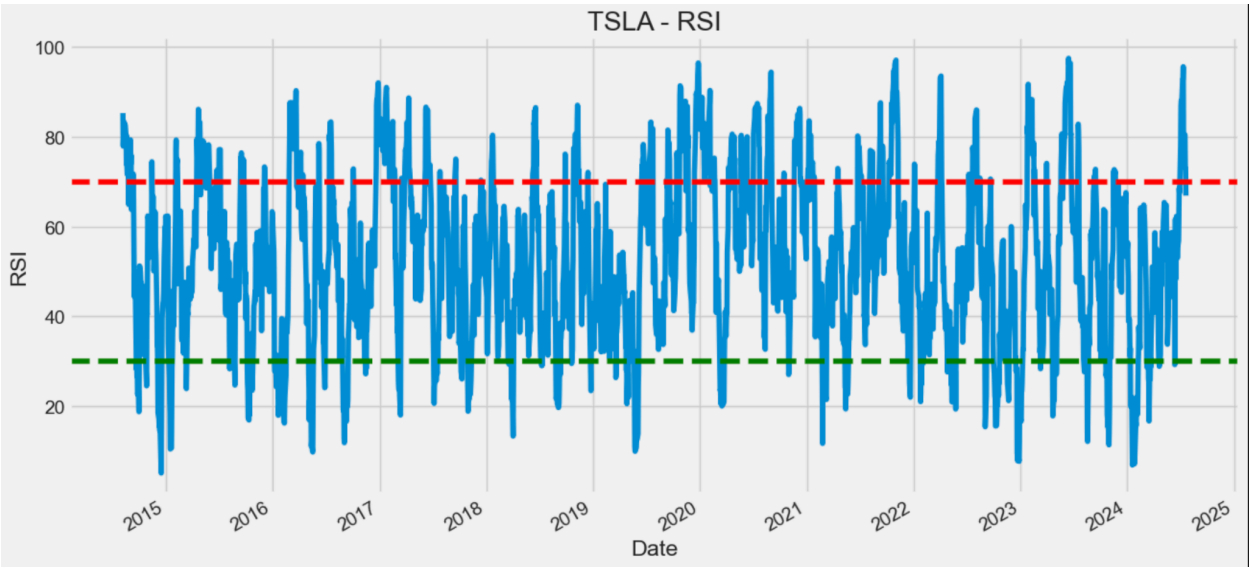
Relative Strength Index (RSI):

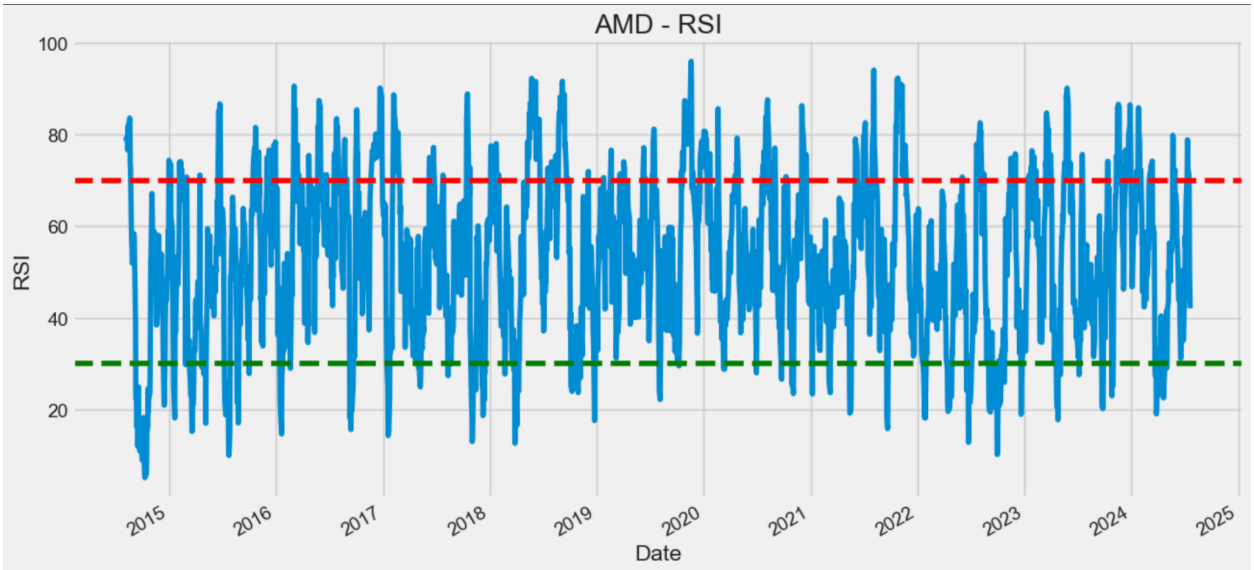
The Relative Strength Index (RSI) is a momentum oscillator that measures the speed and change of price movements on a scale from 0 to 100. It helps identify overbought or oversold conditions.

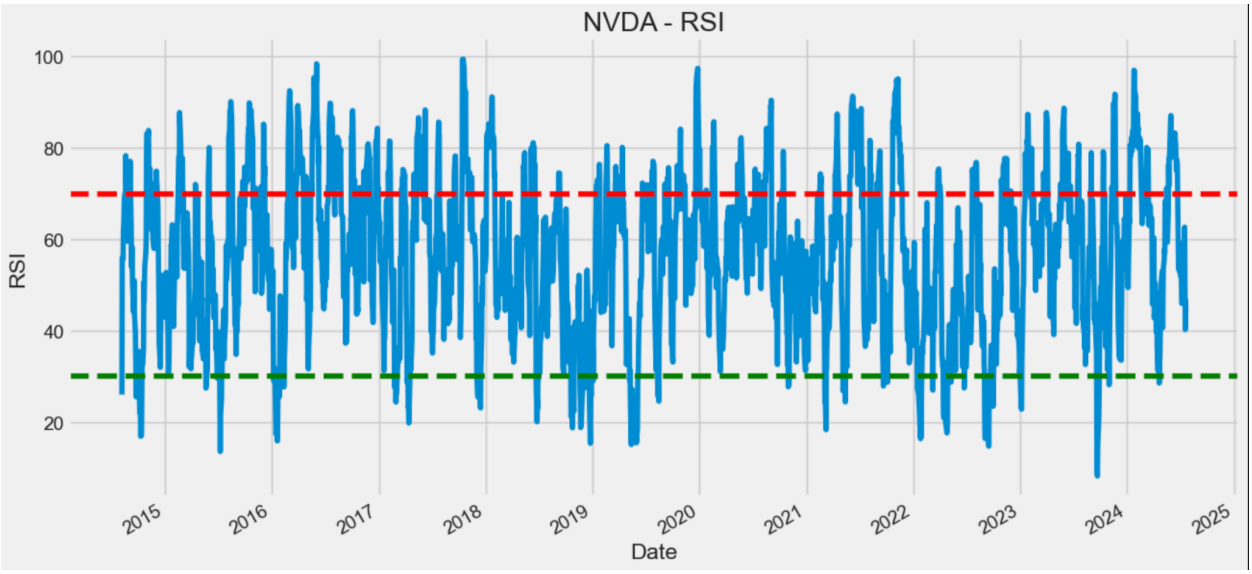
- **RSI > 70**: Indicates the stock might be overbought and could be due for a price correction.
- **RSI < 30**: Suggests the stock might be oversold and could be poised for a price increase.

Look for overbought or oversold signals. An RSI above 70 might indicate a potential selling opportunity, while an RSI below 30 might suggest a buying opportunity. Additionally, divergences

between RSI and price (e.g., price making new highs while RSI does not) can signal potential reversals.






NVDA - RSI
RSI
100
80
60
40
20
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
Date

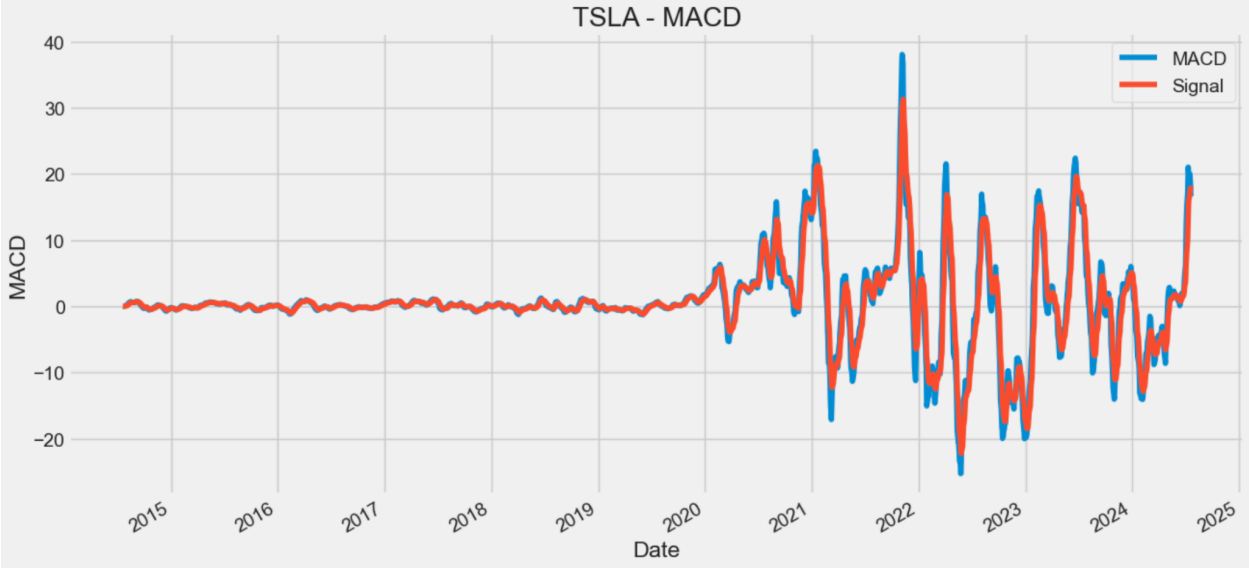
Moving Average Convergence and Divergence (MACD):

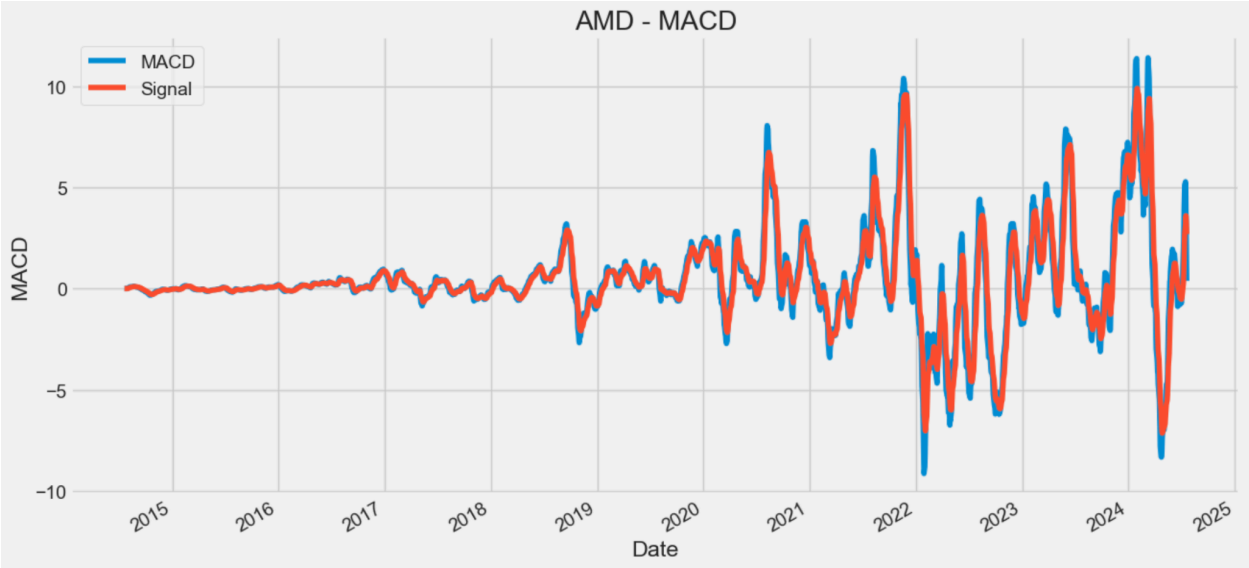
The Moving Average Convergence Divergence (MACD) is a trend-following momentum indicator that shows the relationship between two moving averages of a stock's price, typically the 12-day and 26-day exponential moving averages (EMAs). It consists of:

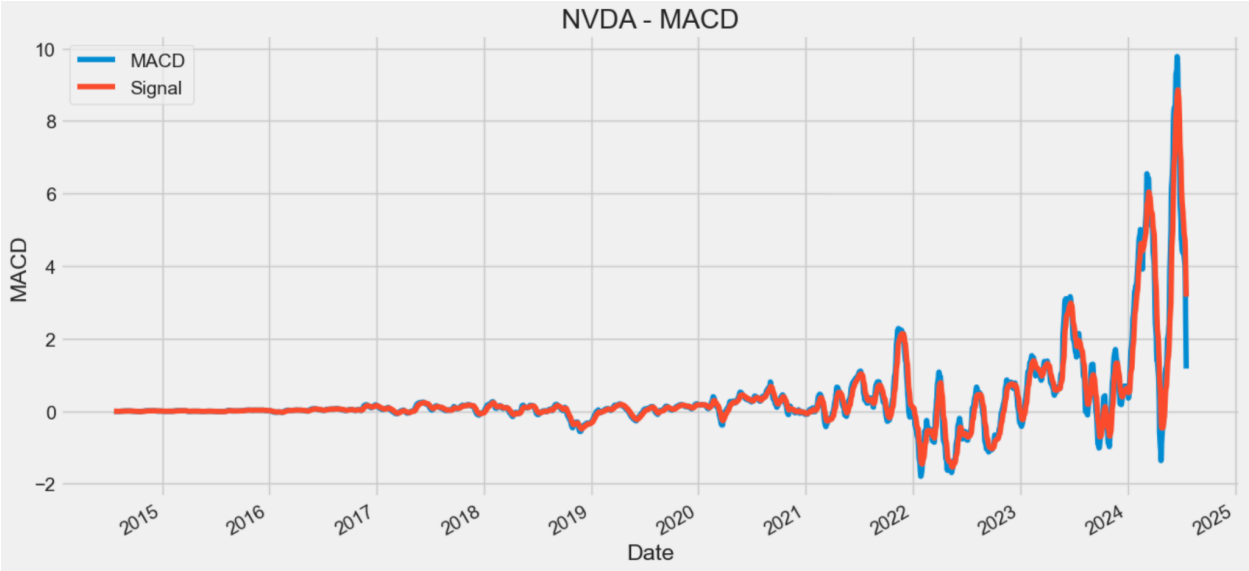
- **MACD Line**: The difference between the 12-day and 26-day EMAs.
- **Signal Line**: A 9-day EMA of the MACD line.

MACD Line Crosses Above Signal Line: Indicates a potential buy signal.

MACD Line Crosses Below Signal Line: Indicates a potential sell signal.

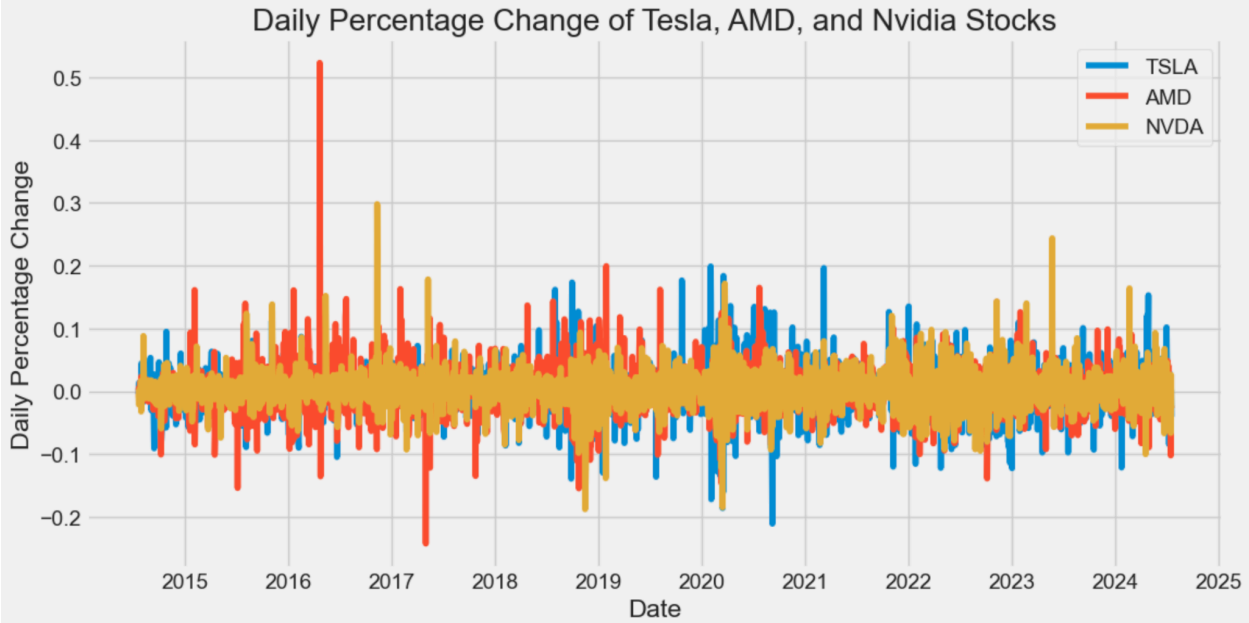



AMD - MACD
MACD
Signal
10
5
0
−5
−10
MACD
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
Date


NVDA - MACD
MACD
Signal
10
8
6
4
2
0
−2
MACD
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
Date

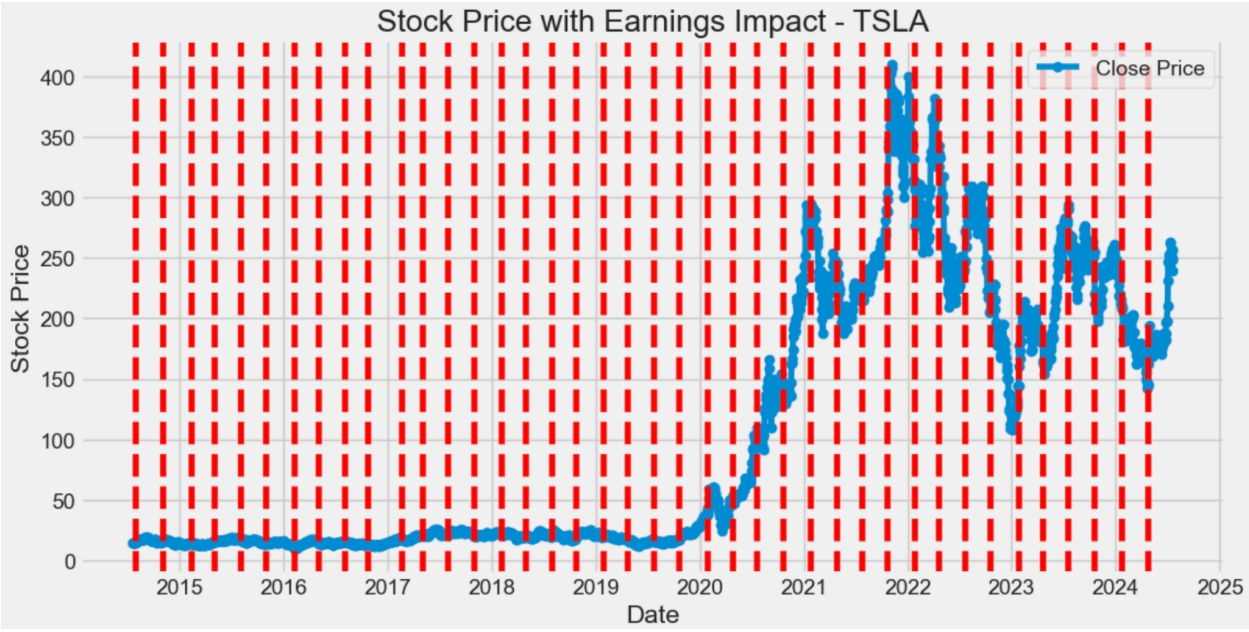
Daily % change:

What is the average % close price change trend?



Earnings Impact:

This plot here showcases earnings impact on stock prices, As you can see its quite difficult to make out any positive or negative impact from earnings in this visualization.



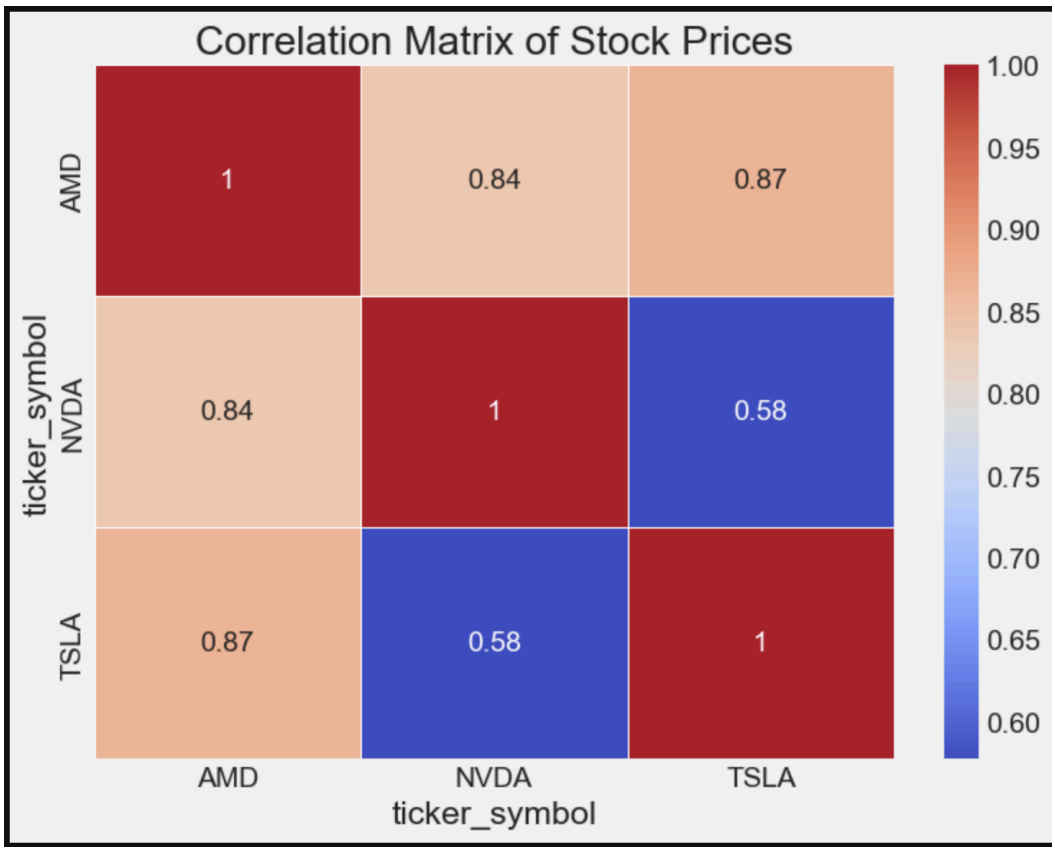
Correlation Matrix

Stock matrix Correlation

When comparing the stock prices of Tesla (TSLA), Nvidia (NVDA), and AMD (AMD), we observe the following correlations:

- **AMD and TSLA**: These two companies exhibit a high correlation. This can be attributed to Tesla's use of AMD chips in their vehicles, creating a direct business relationship that links their stock performance.
- **AMD and NVDA**: There is a moderate correlation between AMD and Nvidia. Both companies are major players in the GPU manufacturing market, so their stock prices are influenced by similar industry trends, competitive dynamics, and market demands.
- **NVDA and TSLA**: Nvidia and Tesla show a lower correlation. Although both are significant technology companies, they operate in different sectors without direct partnerships or dependencies, leading to less interconnected stock performance.

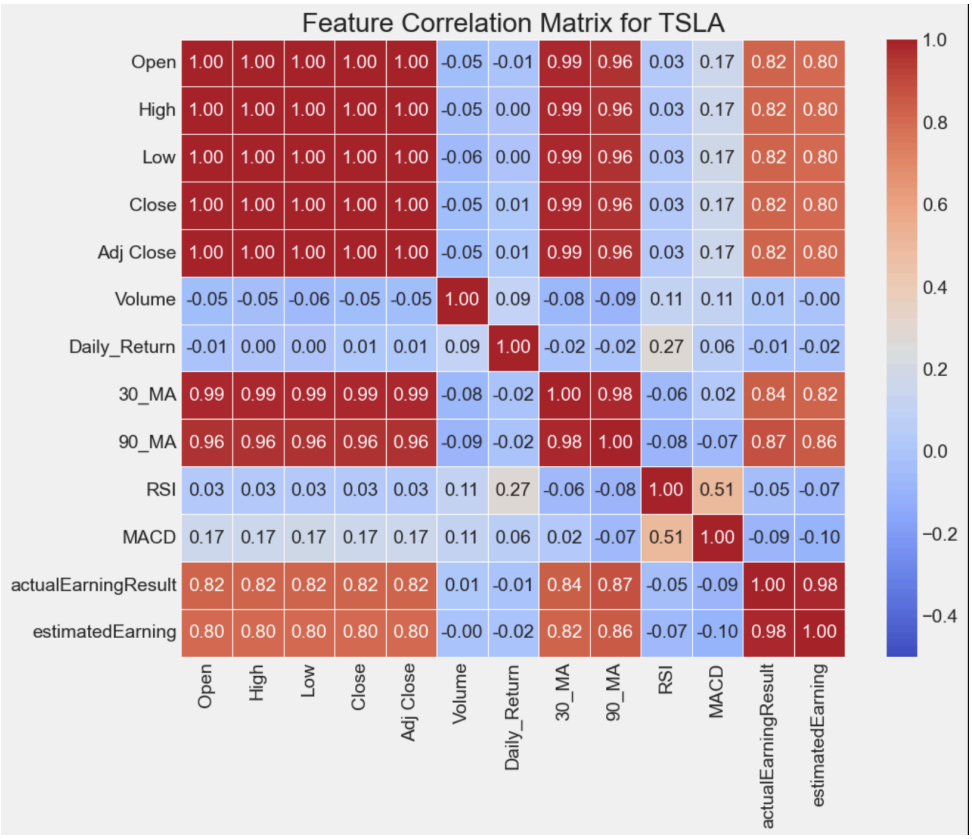
The correlations can be explained by the nature of their business relationships and market influences. Stocks with direct business ties or shared market segments tend to show higher correlation due to overlapping impacts from industry trends, market conditions, and competitive actions. Conversely, companies with no direct interaction or differing market focuses exhibit lower correlation.

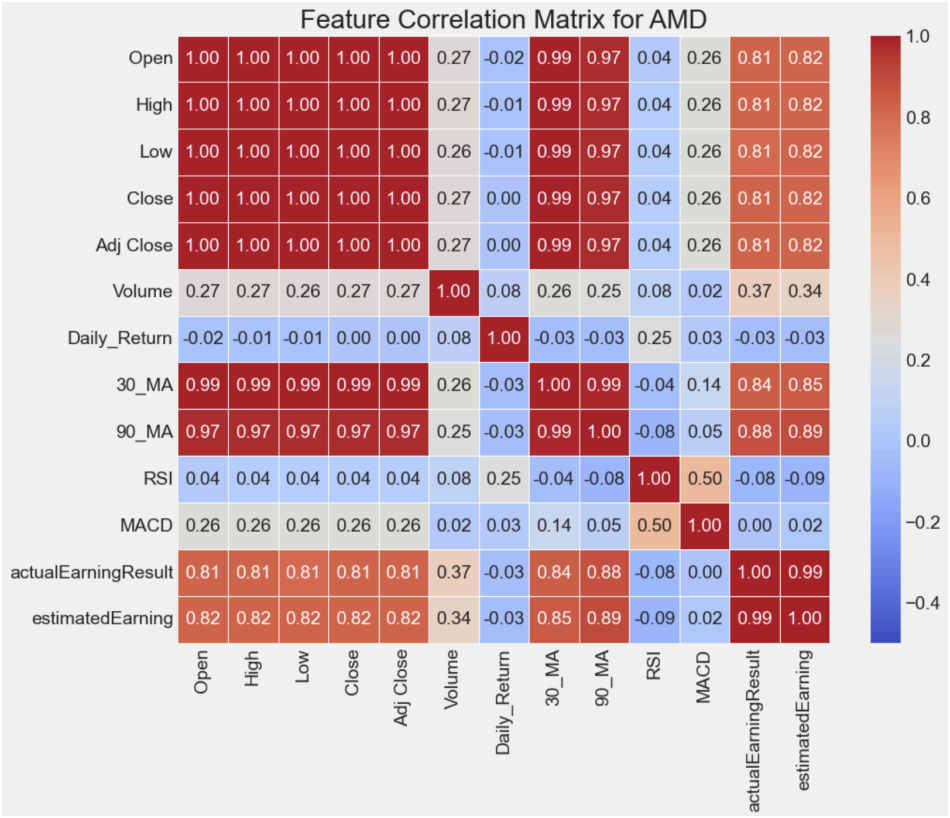


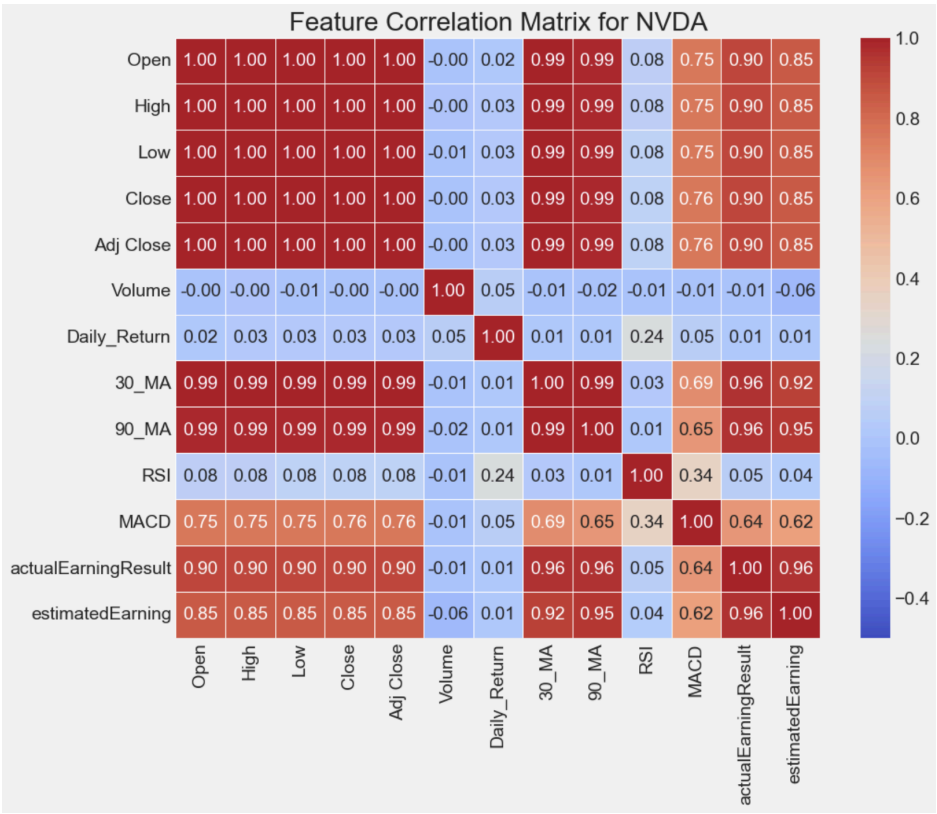
Feature Correlation

In this section, we examine the individual stock dataframes and their respective features to identify which variables have the largest positive or negative correlations with the 'Close' price. By analyzing these correlations, we aim to uncover the most influential factors affecting stock price movements and understand how they might impact the predictive accuracy of our model.

Upon closer examination, we observe that the 'Close' price is significantly influenced by several features, including 'Open', 'High', 'Low', '30_MA' (30-day moving average), '90_MA' (90-day moving average), 'ActualEPS' (actual earnings per share), and 'EstimatedEPS' (estimated earnings per share). Notably, the majority of these correlations are positive, indicating that as these variables increase, the 'Close' price tends to increase as well.




Feature Correlation Matrix for AMD
Open 1.00 1.00 1.00 1.00 1.00 0.27 -0.02 0.99 0.97 0.04 0.26 0.81 0.82
High 1.00 1.00 1.00 1.00 1.00 0.27 -0.01 0.99 0.97 0.04 0.26 0.81 0.82
Low 1.00 1.00 1.00 1.00 1.00 0.26 -0.01 0.99 0.97 0.04 0.26 0.81 0.82
Close 1.00 1.00 1.00 1.00 1.00 0.27 0.00 0.99 0.97 0.04 0.26 0.81 0.82
Adj Close 1.00 1.00 1.00 1.00 1.00 0.27 0.00 0.99 0.97 0.04 0.26 0.81 0.82
Volume 0.27 0.27 0.26 0.27 0.27 1.00 0.08 0.26 0.25 0.08 0.02 0.37 0.34
Daily_Return -0.02 -0.01 -0.01 0.00 0.00 0.08 1.00 -0.03 -0.03 0.25 0.03 -0.03 -0.03
30_MA 0.99 0.99 0.99 0.99 0.99 0.26 -0.03 1.00 0.99 -0.04 0.14 0.84 0.85
90_MA 0.97 0.97 0.97 0.97 0.97 0.25 -0.03 0.99 1.00 -0.08 0.05 0.88 0.89
RSI 0.04 0.04 0.04 0.04 0.04 0.08 0.25 -0.04 -0.08 1.00 0.50 -0.08 -0.09
MACD 0.26 0.26 0.26 0.26 0.26 0.02 0.03 0.14 0.05 0.50 1.00 0.00 0.02
actualEarningResult 0.81 0.81 0.81 0.81 0.81 0.37 -0.03 0.84 0.88 -0.08 0.00 1.00 0.99
estimatedEarning 0.82 0.82 0.82 0.82 0.82 0.34 -0.03 0.85 0.89 -0.09 0.02 0.99 1.00
Open
High
Low
Close
Adj Close
Volume
Daily_Return
30_MA
90_MA
RSI
MACD
actualEarningResult
estimatedEarning
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4


Feature Correlation Matrix for NVDA
Open 1.00 1.00 1.00 1.00 1.00 -0.00 0.02 0.99 0.99 0.08 0.75 0.90 0.85
High 1.00 1.00 1.00 1.00 1.00 -0.00 0.03 0.99 0.99 0.08 0.75 0.90 0.85
Low 1.00 1.00 1.00 1.00 1.00 -0.01 0.03 0.99 0.99 0.08 0.75 0.90 0.85
Close 1.00 1.00 1.00 1.00 1.00 -0.00 0.03 0.99 0.99 0.08 0.76 0.90 0.85
Adj Close 1.00 1.00 1.00 1.00 1.00 -0.00 0.03 0.99 0.99 0.08 0.76 0.90 0.85
Volume -0.00 -0.00 -0.01 -0.00 -0.00 1.00 0.05 -0.01 -0.02 -0.01 -0.01 -0.01 -0.06
Daily_Return 0.02 0.03 0.03 0.03 0.03 0.05 1.00 0.01 0.01 0.24 0.05 0.01 0.01
30_MA 0.99 0.99 0.99 0.99 0.99 -0.01 0.01 1.00 0.99 0.03 0.69 0.96 0.92
90_MA 0.99 0.99 0.99 0.99 0.99 -0.02 0.01 0.99 1.00 0.01 0.65 0.96 0.95
RSI 0.08 0.08 0.08 0.08 0.08 -0.01 0.24 0.03 0.01 1.00 0.34 0.05 0.04
MACD 0.75 0.75 0.75 0.76 0.76 -0.01 0.05 0.69 0.65 0.34 1.00 0.64 0.62
actualEarningResult 0.90 0.90 0.90 0.90 0.90 -0.01 0.01 0.96 0.96 0.05 0.64 1.00 0.96
estimatedEarning 0.85 0.85 0.85 0.85 0.85 -0.06 0.01 0.92 0.95 0.04 0.62 0.96 1.00
Open
High
Low
Close
Adj Close
Volume
Daily_Return
30_MA
90_MA
RSI
MACD
actualEarningResult
estimatedEarning
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4

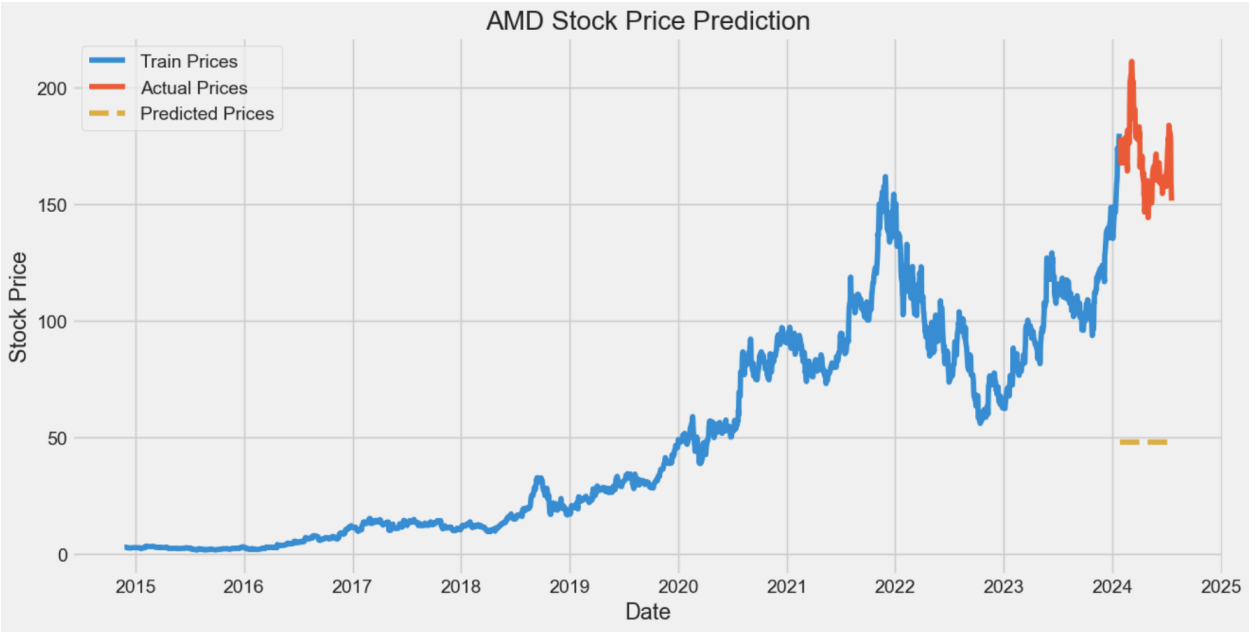
Modeling:

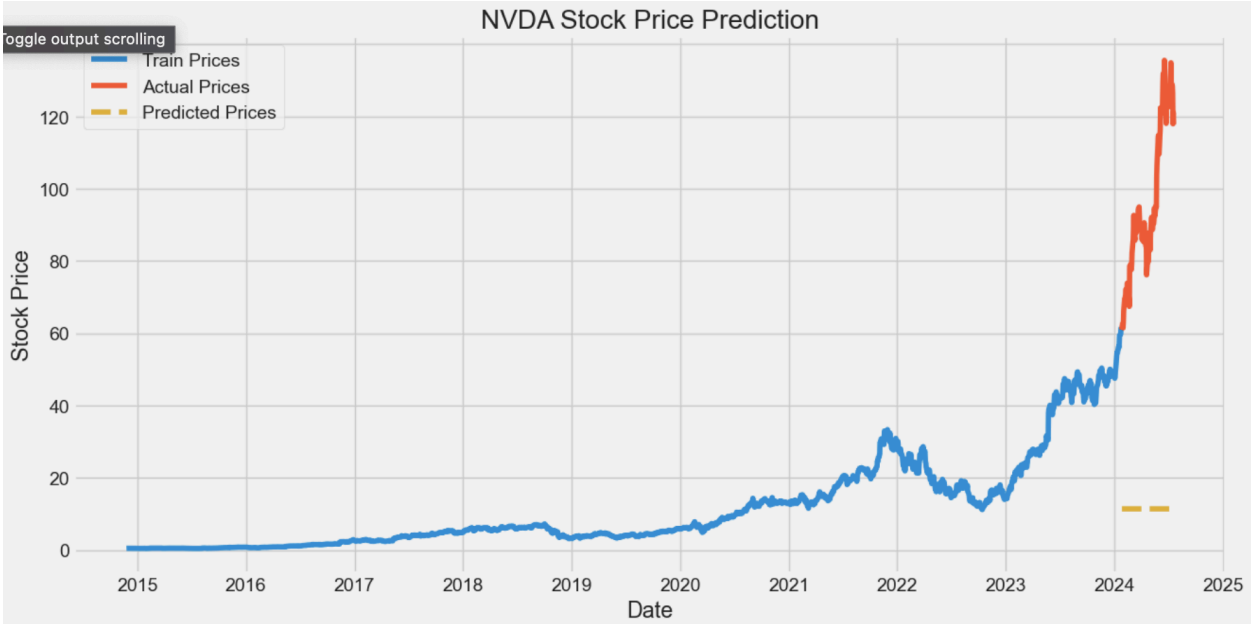
Now that we have completed the EDA and prepared our dataframe with the necessary features, we are ready to predict the 'Close' price on a given day. In this section, we will begin with a dummy regression model to establish a baseline score. Following that, we will explore our basic go-to machine learning algorithms and conclude with more advanced deep learning neural network models.

Baseline Model - Dummy Regressor

Let's start off with a basic dummy regression model to get a baseline score.

```
Evaluation results for TSLA:
MAE = 83.92599192178875, RMSE = 87.52686296465784
Evaluation results for AMD:
MAE = 121.98609533969403, RMSE = 122.81887787806762
Evaluation results for NVDA:
MAE = 84.69970528071703, RMSE = 87.12997743444586
```





Pretty baseline results with RMSE > $87 and just mean value as prediction for each of these stocks.

Traditional ML Model

In this section, we explore the use of Linear Regression, Decision Tree Regressor, and Random Forest Regressor for predicting the stock prices of Tesla, Nvidia, and AMD. The steps involved in this process are detailed below:

Train and Test Set Creation

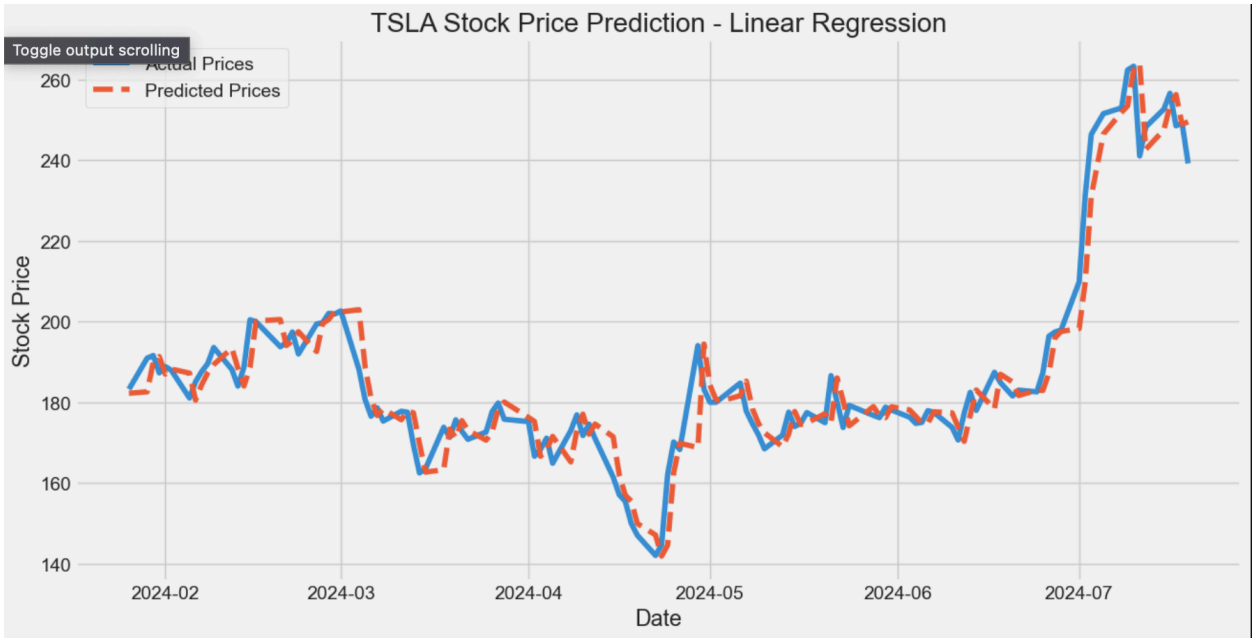
Creating a train and test set is a crucial step in the modeling process. This involves splitting the dataset into two parts: one for training the model and the other for testing its performance. For stock price prediction, it's essential to offset the input features to predict the 'Close' price for the following day. In this project, a 1-day offset is used, meaning all data up to today is used to predict tomorrow's price.

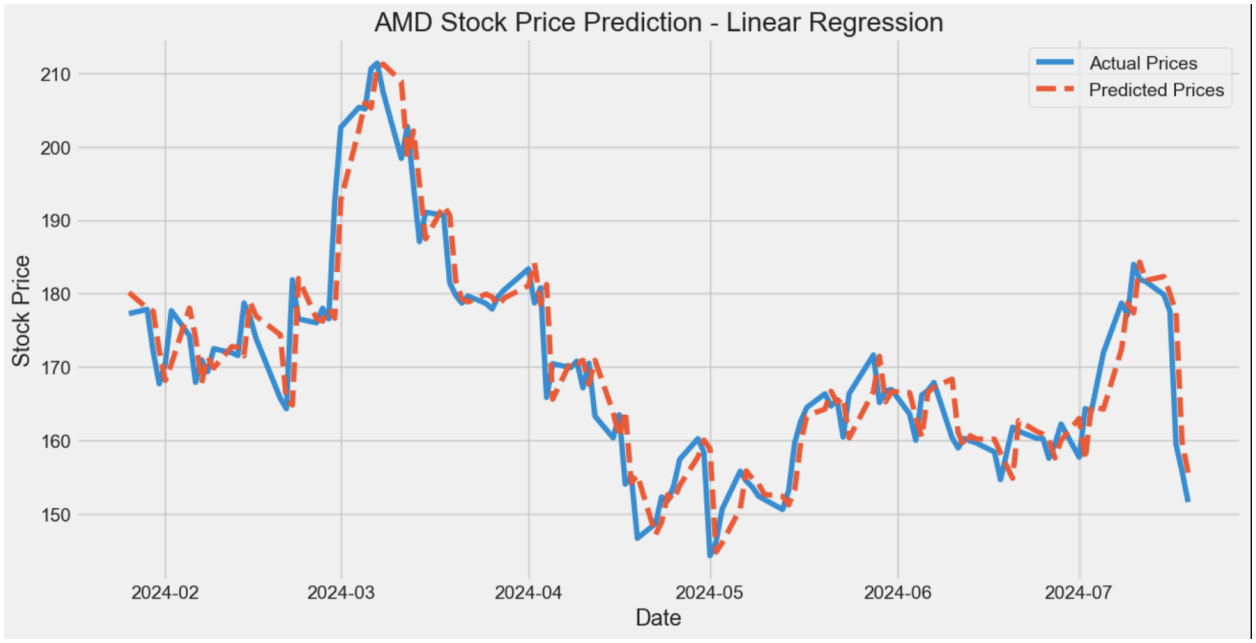
Model Training and Evaluation

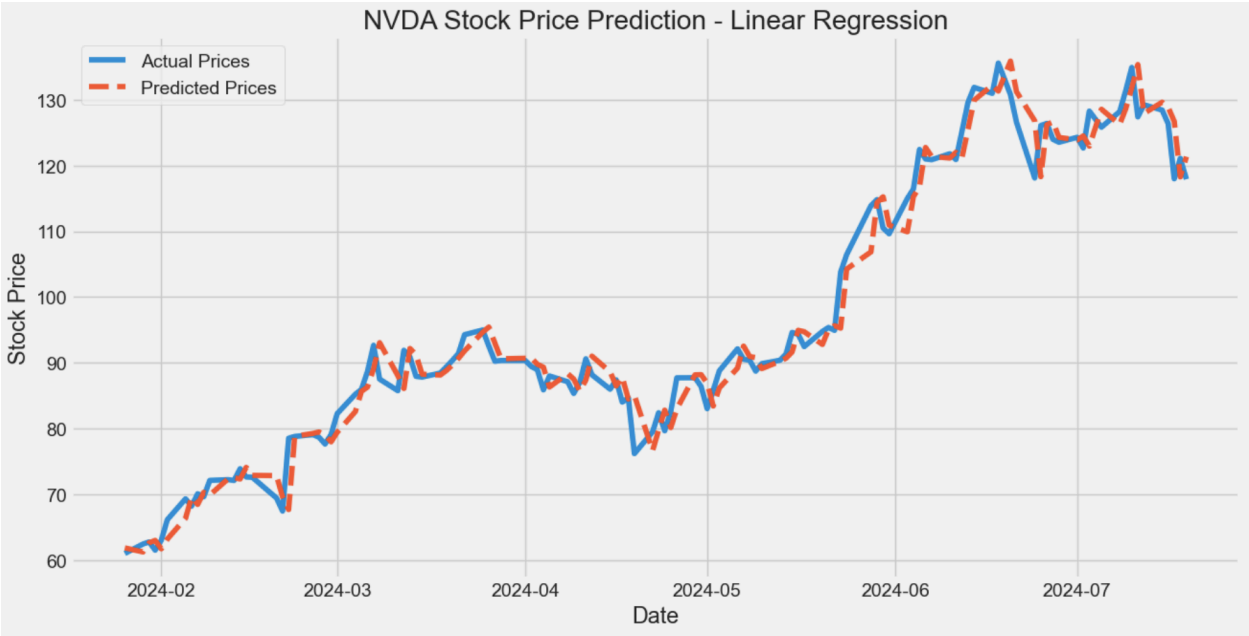
We implement three traditional machine learning models: Linear Regression, Decision Tree Regressor, and Random Forest Regressor. To determine which model performs best, we evaluate them using two key metrics: Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE).

```
Evaluation results for TSLA:
Linear Regression: MAE = 4.761281658580042, RMSE = 6.569041060293892
Decision Tree: MAE = 8.795233356065985, RMSE = 11.056552610346348
Random Forest: MAE = 5.117666692812578, RMSE = 7.1659821955096525
Evaluation results for AMD:
Linear Regression: MAE = 3.7955258079826337, RMSE = 5.225885299599488
Decision Tree: MAE = 9.825703392344073, RMSE = 13.465099004964687
Random Forest: MAE = 7.238814009359062, RMSE = 10.82822204466515
Evaluation results for NVDA:
Linear Regression: MAE = 2.420699357476408, RMSE = 3.2430916311819407
Decision Tree: MAE = 34.997586526161385, RMSE = 40.44316384673505
Random Forest: MAE = 35.609578680873895, RMSE = 41.023065318460425
```

For all stocks the best model was the Linear Regression model, with RMSE score ranging from $3 - $6.5. This is a suprisingly a really good score. Lets visualize the Linear regression model by predicting on our test dataframe.


TSLA Stock Price Prediction - Linear Regression
Toggle output scrolling
Actual Prices
Predicted Prices
260
240
220
200
180
160
140
Stock Price
2024-02
2024-03
2024-04
2024-05
2024-06
2024-07
Date


AMD Stock Price Prediction - Linear Regression
Actual Prices
Predicted Prices
210
200
190
180
170
160
150
Stock Price
2024-02
2024-03
2024-04
2024-05
2024-06
2024-07
Date



This looks great. Our Prediction model is trailing the actual prices. BUT WAITTTT, Upon closer examination, we notice an interesting pattern: the predictions for each day are essentially mirroring the previous day's actual prices, with a slight adjustment.

This indicates that our model is effectively capturing the trends, but it is lagging by one day. In other words, the predicted price for any given day closely resembles the actual price of the previous day, with a small incremental change. This trailing behavior suggests that while our model is responsive to recent trends, it may need further refinement to improve its predictive accuracy and better align with real-time price movements.

Deep learning Model

In this section, we will primarily focus on using the Long Short-Term Memory (LSTM) model. The goal is to determine whether we can improve upon our linear regression model by capturing more real-time price movements. LSTM models are particularly well-suited for time series data due to their ability to capture long-term dependencies and patterns.

Data Preparation for LSTM

Unlike traditional machine learning models that work with 2D dataframes, LSTM models require the input data to be in a 3D sequence format. This involves reshaping our existing 2D dataframe into a 3D structure suitable for the LSTM model. Steps taken were as follows:

- **Normalization**: MinMax Normalize the data to scale the features, ensuring that all values are within a similar range. This helps the LSTM model train more effectively.
- **Creating Sequences**: Transform the dataframe into sequences of 30 days. Each sequence should include the feature values for 30 days and the corresponding target value (close price) for the next time step (next day). For example, if using a sequence length of 30 days, each input sequence will contain the feature values for the past 30 days, and the target will be the close price for the 31st day.
- **Reshaping Data**: Reshape the data into a 3D array with dimensions [number of sequences, sequence length, number of features]. This format is required by the LSTM model to process the time series data.
- **Train-Test Split**: Split the sequences into training and testing sets of 95% toward training set, ensuring that the split maintains the temporal order of the data to avoid data leakage and maintain the integrity of the time series.

Hypertuning:

After splitting the data, the next step is hyperparameter tuning. I use the Keras Tuner module to find the optimal parameters for my multilayered LSTM model. Through this tuning process, I aim to identify the best configuration that minimizes validation loss. Upon completion, the tuning process resulted in a validation loss of 0.00028, which is the best score achieved. Therefore, I will proceed with this configuration for further model training and evaluation.

```
Trial 5 Complete [00h 20m 41s]
val_loss: 0.0003068597870878875

Best val_loss So Far: 0.00028865167405456305
Total elapsed time: 00h 57m 46s
Optimal hyperparameters for NVDA:
units1: 160
dropout1: 0.0
units2: 256
dropout2: 0.4
```

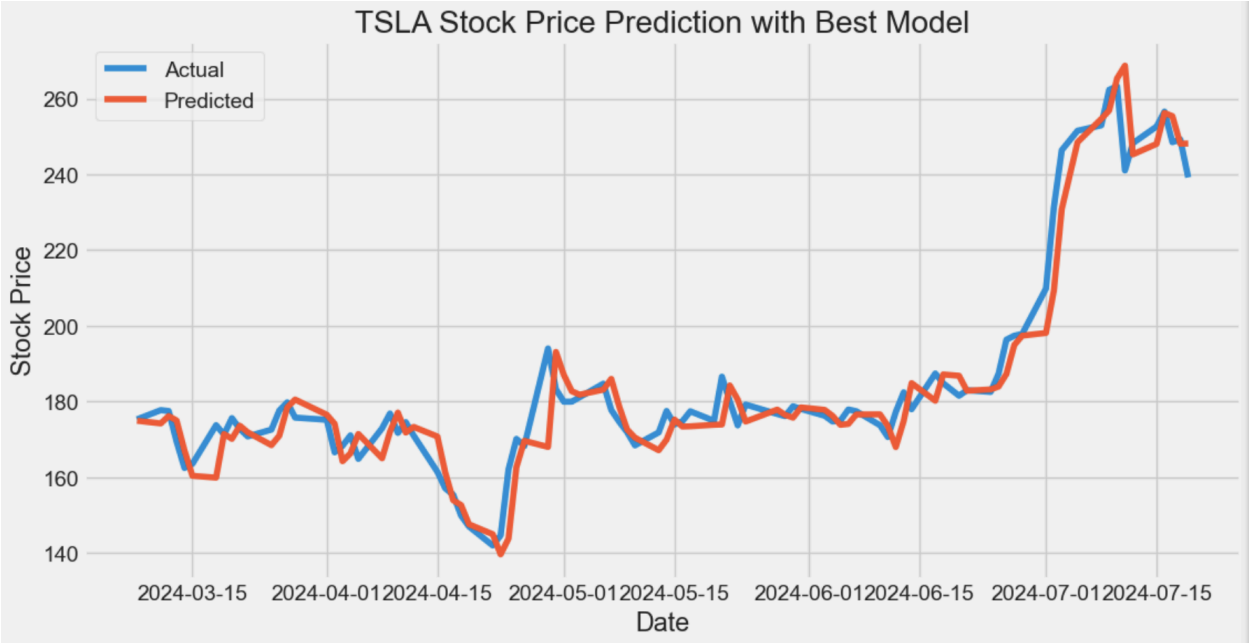
Model Training and Evaluation:

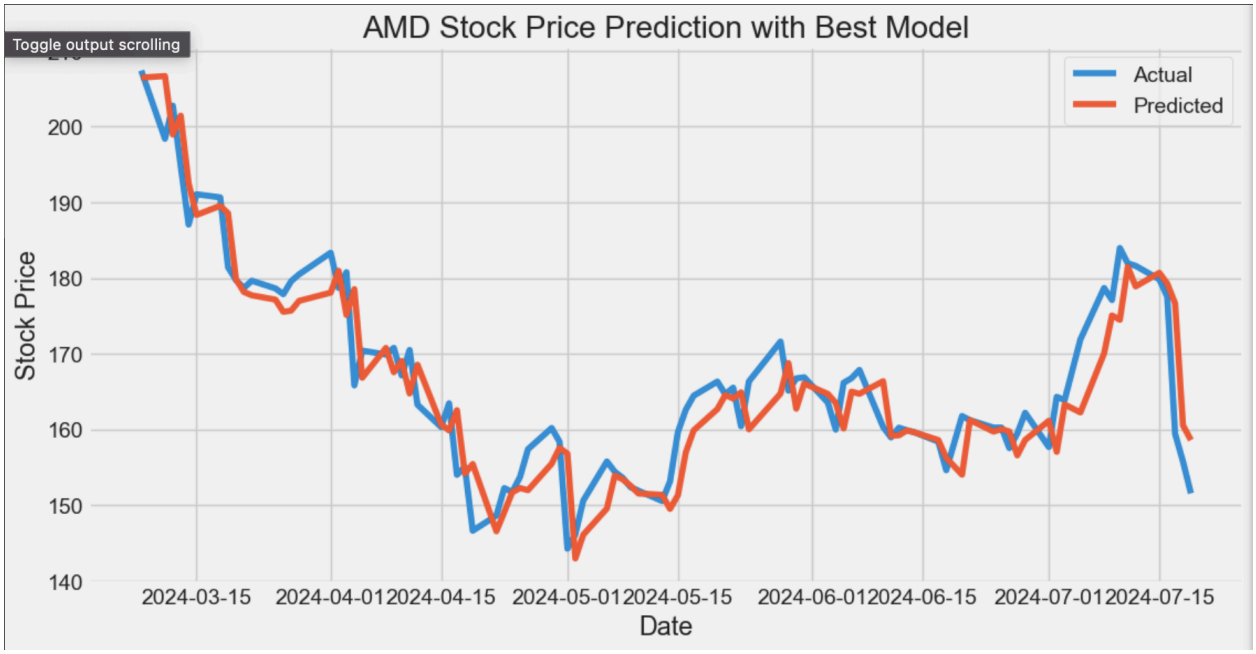
```
3/3 [==============================] - 0s 31ms/step
Hypertuned Evaluation results for TSLA:
MAE = 4.963618891839849, RMSE = 7.159550316857211
```

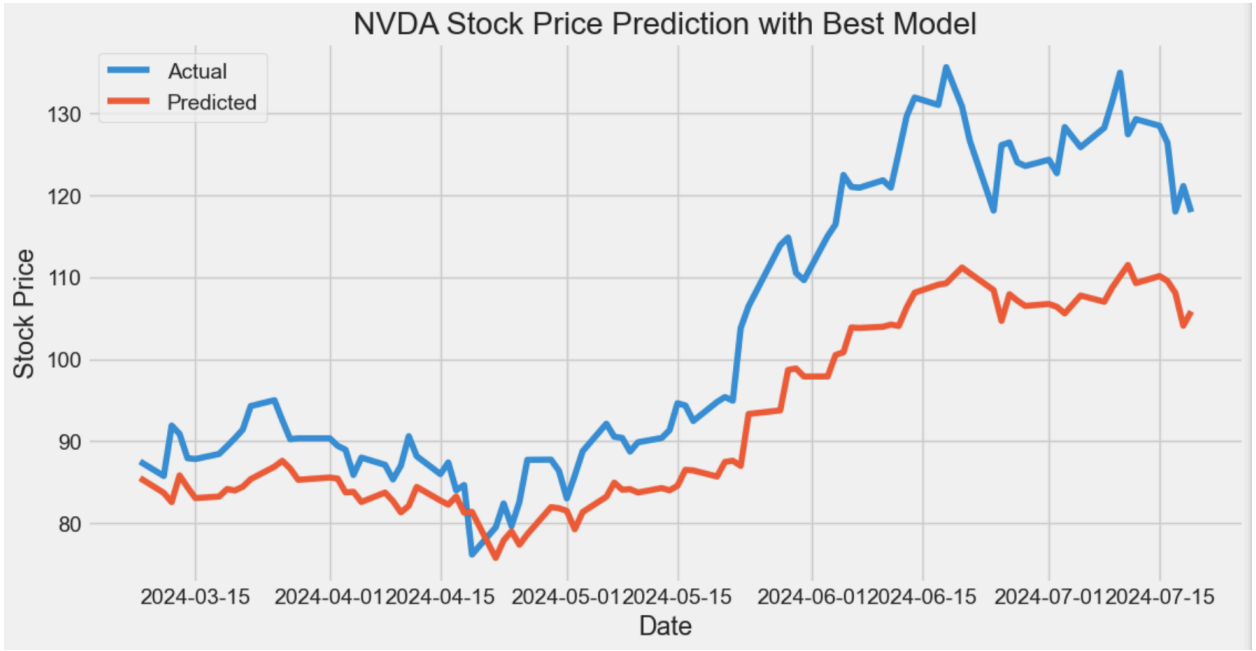
```
3/3 [==============================] - 0s 21ms/step
Hypertuned Evaluation results for AMD:
MAE = 3.65240046665591, RMSE = 4.907449024019578
```

```
3/3 [==============================] - 0s 20ms/step
Hypertuned Evaluation results for NVDA:
MAE = 10.742141554010868, RMSE = 12.787243059631917
```

Above are the results for the best LSTM model. Surprisingly lower than our Linear regression model. Lets plot the predictions against the actual to see how well it did.

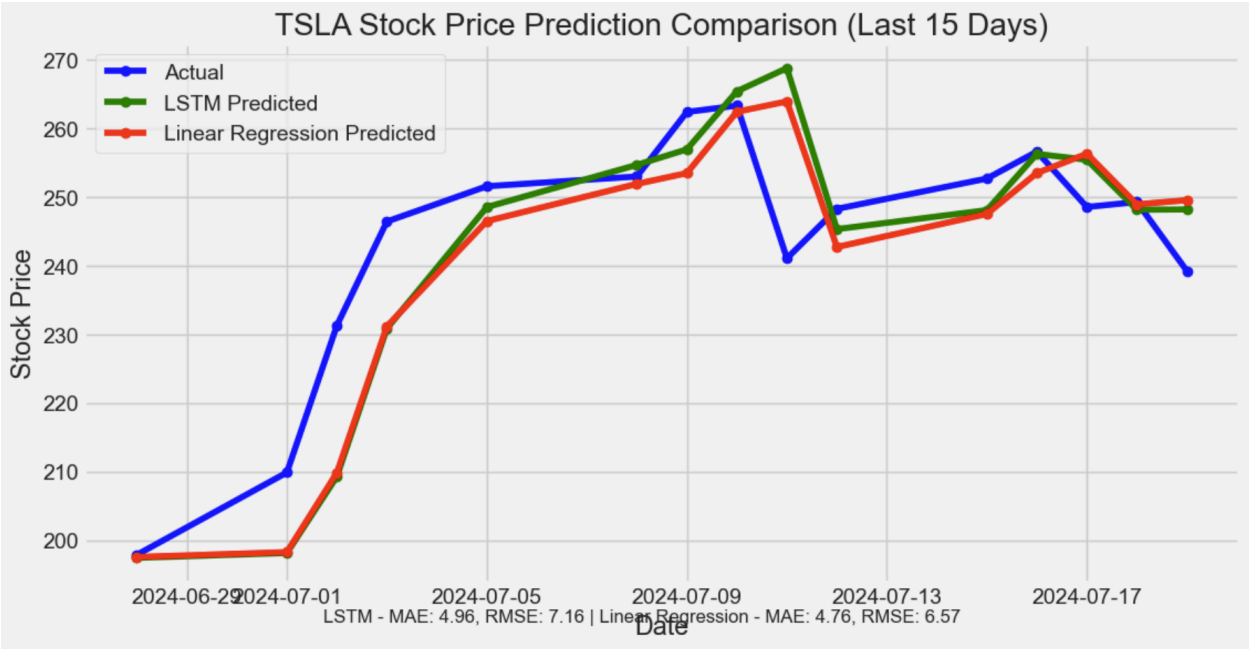


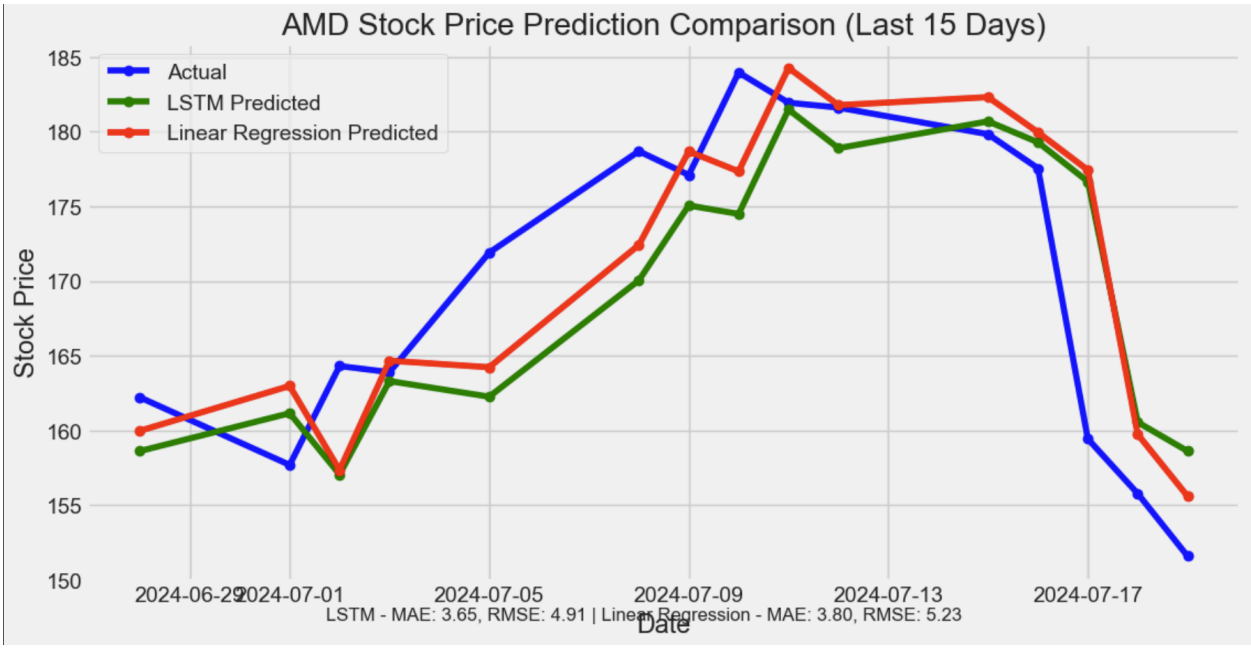


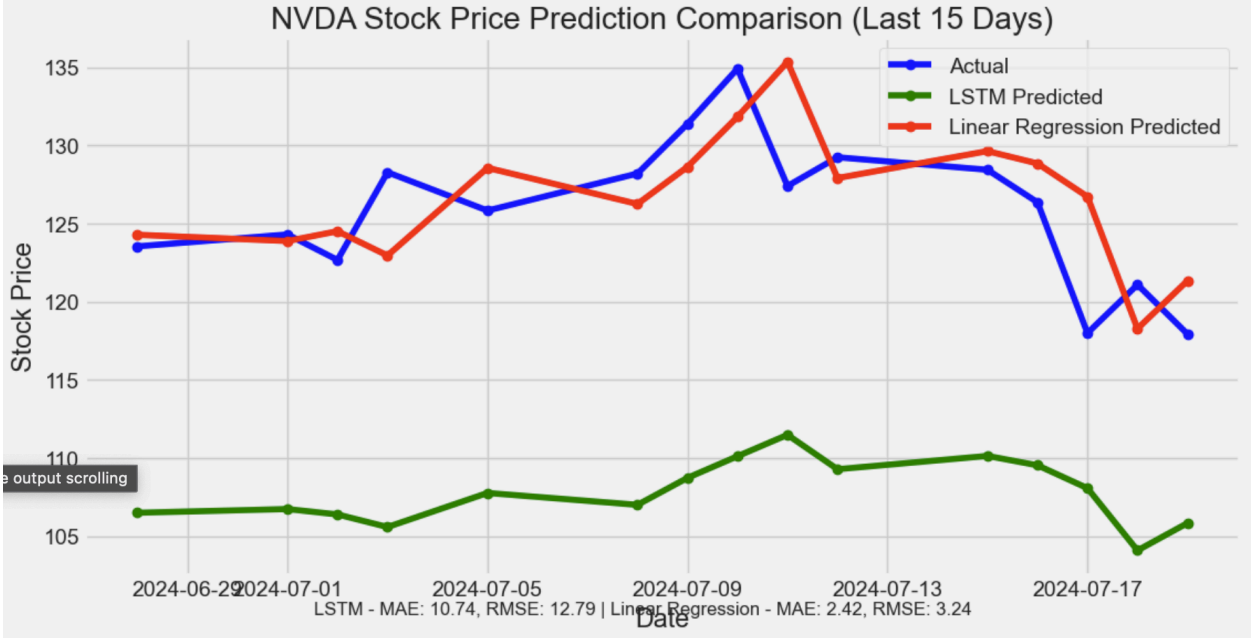


We encountered a similar issue with the LSTM model as with previous models: the predictions are trailing the actual prices by about a day. Despite leveraging the advanced capabilities of LSTM, we did not see the anticipated improvement in predictive accuracy or alignment with real-time price movements. In fact, the LSTM model performed worse than our linear regression model. This discrepancy is particularly evident with Nvidia (NVDA) stock, where the LSTM model failed to

capture the steep price climbs observed in the training data, resulting in predictions that are significantly off from the actual values.







Recommendations for Improvements

To address the issues with model accuracy and the inability to provide real-time predictions, the following improvements are recommended:

1. **Feature Engineering**:
    - Introduce additional features that may impact stock prices, such as macroeconomic indicators (e.g., US dollar index, unemployment rate, interest rate), sentiment analysis from the consumer sentiment index, news articles, and social media trends.
2. **Model Complexity**:
    - Experiment with more complex LSTM architectures, such as adding more layers, using bidirectional LSTMs, and other advanced configurations to capture better long-term dependencies and trends.

Conclusion

This report explored developing a stock price prediction model for Tesla, Nvidia, and AMD using traditional machine learning and deep learning techniques. Despite leveraging historical stock data and employing various models, we faced challenges with trailing predictions and day-off offsets, limiting real-time accuracy.

Our initial models included Linear Regression, Decision Tree Regressor, and Random Forest Regressor. We then utilized LSTM networks for their suitability with time series data but observed similar issues. Hyperparameter tuning did not yield significant improvements.

To enhance model performance, we recommend incorporating additional macroeconomic indicators and sentiment analysis and experimenting with more complex LSTM architectures.

In summary, while our models captured some stock price trends, robust and accurate predictions remain challenging due to market volatility. Further exploration of advanced techniques and comprehensive feature sets is essential for future improvements. This project provides a foundation for ongoing efforts to refine stock price prediction models.