Stock Price Prediction using Machine Learning: A Case Study on Google Stock Data

Maitreya Sameer Ganu, IISER Thiruvananthapuram

5^{th} December 2024

Abstract

This project uses machine learning (ML) for predicting stock prices using historical market data of more than 20 years of Google stock obtained from Yahoo Finance. The study makes use of Long Short-Term Memory (LSTM) networks, to predict future stock prices. The models are evaluated using performance metrics like Root Mean Squared Error (RMSE). This project focuses on predicting the adjusted closing price using LSTM of the google stock prices. The concept of moving averages has been used extensively and the moving average for 100 days has been used.

1 Introduction:

The goal of this project is to apply advanced machine learning methods to predict Google stock prices based on historical data. The results of this study aim to demonstrate the potential of machine learning, specifically LSTM networks, in providing more accurate stock price predictions compared to conventional approaches. The model was very accurate because the root mean squared error for this data set was 9 . The model has been trained for 2 epochs but with the batch size of 1. This was done to ensure the ease of computational calculations. The project was done in Python and included the foloowing libraries :

- sklearn
- keras
- matplotlib
- numpy
- pandas
- yfinance
- datetime

2 Data Visualization:

The following diagram describes the dataset . It contains 5109 entries containing no null values and missing values. Thus , the dataset is already cleaned.

2.1 Overview of the dataset

Price	Adj Close	Close	High	Low	Open	Volume
Ticker	GOOG	GOOG	GOOG	GOOG	GOOG	GOOG
count	5109.000000	5109.000000	5109.000000	5109.000000	5109.000000	5.109000e+03
mean	47.285894	47.394604	47.872732	46.901723	47.375096	1.140482e+08
std	45.768176	45.860644	46.332335	45.384147	45.833861	1.490378e+08
min	2.484811	2.490913	2.534002	2.390042	2.470490	1.584340e+05
25%	13.079234	13.111355	13.222439	12.969637	13.113846	2.674390e+07
50%	27.445065	27.512465	27.730865	27.225252	27.537395	5.321472e+07
75%	62.127926	62.280499	62.777100	61.778000	62.211498	1.400269e+08
max	192.406723	192.660004	193.309998	190.619995	191.750000	1.650833e+09

Figure 1: General Description of the dataset

```
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   (Adj Close, GOOG)  5109 non-null   float64
 1   (Close, GOOG)      5109 non-null   float64
 2   (High, GOOG)       5109 non-null   float64
 3   (Low, GOOG)        5109 non-null   float64
 4   (Open, GOOG)       5109 non-null   float64
 5   (Volume, GOOG)     5109 non-null   int64
```

Figure 2: Description about null and non-null values

Price	Ticker	
Adj Close	GOOG	0
Close	GOOG	0
High	GOOG	0
Low	GOOG	0
Open	GOOG	0
Volume	GOOG	0

Figure 3: Description about missing values

2.2 Graphical visualization of features:

This section analyzes the behavior of the feature variables over time. The analysis focuses on Google stock data, beginning from 2004, and examines the variation in feature values from 2004 to 2024.

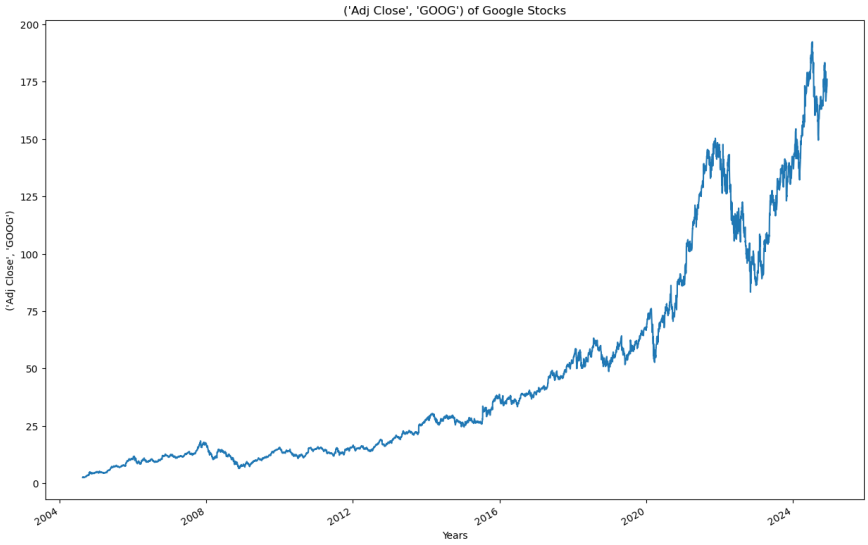

Figure 4: Variation of adjusted closing price with time

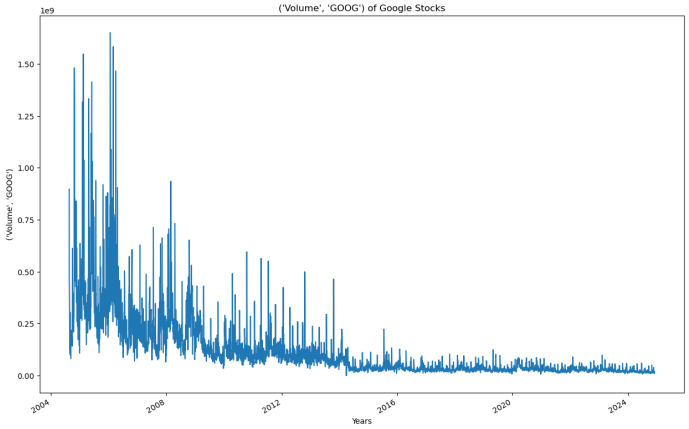

Figure 5: Variation of volume of stocks with time

Now , the 100-day moving average of the adjusted closing prices in the Google stock data is calculated. A rolling window of 100 days is applied to the adjusted closing prices, and the average of the values within each window is calculated.

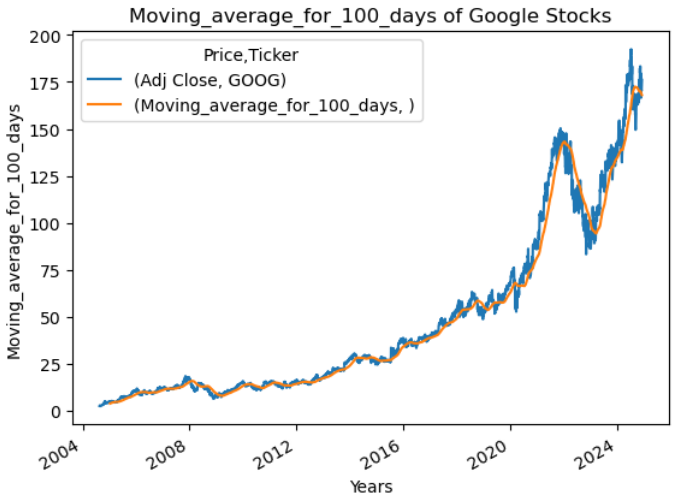

Figure 6: 100-Day Moving Average of Google Stock Prices (2004-2024)

The daily percentage change in the adjusted closing price of Google stock was calculated. It measures the relative change between consecutive adjusted closing prices, providing insight into the daily price fluctuations as a percentage.

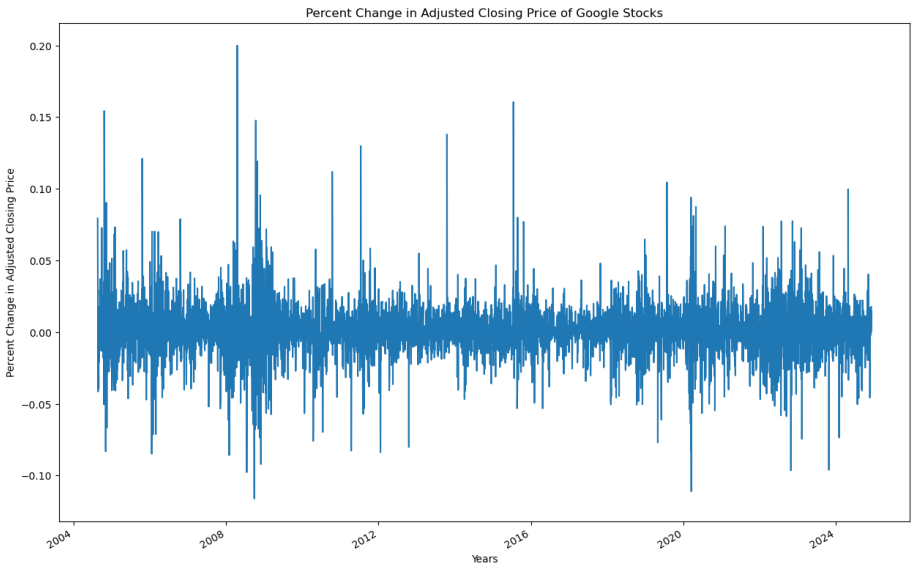

Figure 7: Daily percentage change in the adjusted closing price of Google stock (2004-2024)

3 Data Preprocessing

The MinMaxScaler is employed to normalize the dataset by transforming all data points into a range between 0 and 1. This scaling technique ensures that the values in the dataset are rescaled proportionally, preserving the relationships between the original data while standardizing the range for further analysis or modeling. 70% of the original dataset (3506 data points) is used to train our model and 30% (1402 data points , here the first 100 days of moving average are not considered) is used to test it.
The model is built using a sequential architecture, which allows for the stacking of layers in a linear order. The first layer is an LSTM layer with 128 units, designed to capture temporal dependencies in the input data. This layer returns sequences to enable the subsequent LSTM layer to process the outputs further. The second LSTM layer contains 64 units and does not return sequences, effectively reducing the output to a single vector. Following this, a dense layer with 25 units is added to refine the learned features. Finally, an output dense layer with a single neuron is included to generate the final prediction.
The model is compiled using the Adam optimizer, which is well-suited for adaptive learning, and the Mean Squared Error (MSE) loss function, commonly used for regression tasks. The model is trained with a batch size of 1 over 2 epochs, allowing it to iteratively learn patterns in the training data.

Model: "sequential_1"

Layer (type)	Output Shape	Param #
lstm_2 (LSTM)	(None, 100, 128)	66,560
lstm_3 (LSTM)	(None, 64)	49,408
dense_1 (Dense)	(None, 25)	1,625
dense_2 (Dense)	(None, 1)	26

Total params: 352,859 (1.35 MB)
Trainable params: 117,619 (459.45 KB)
Non-trainable params: 0 (0.00 B)
Optimizer params: 235,240 (918.91 KB)

Figure 8: Summary of the model , showing different types of parameters used.

4 Prediction, Evaluation, and Visualization

The trained model is used to predict values for the test data set. The predicted values are then transformed back to their original scale using the inverse transformation of the scaler. Similarly, the actual test values are also inverse-transformed for accurate comparison.

The Root Mean Squared Error (RMSE) is computed to evaluate the model's performance. This metric is calculated as the square root of the mean squared differences between the inverse-transformed predictions and actual test values, providing a measure of the prediction accuracy. The root mean squared error was calculated to be approximately 9.71 .

To visualize the results, a pandas dataframe was created containing two columns: the original test data values and the predicted values. The index of this dataframe corresponds to the appropriate dates from the original dataset, allowing for an effective time-series comparison.

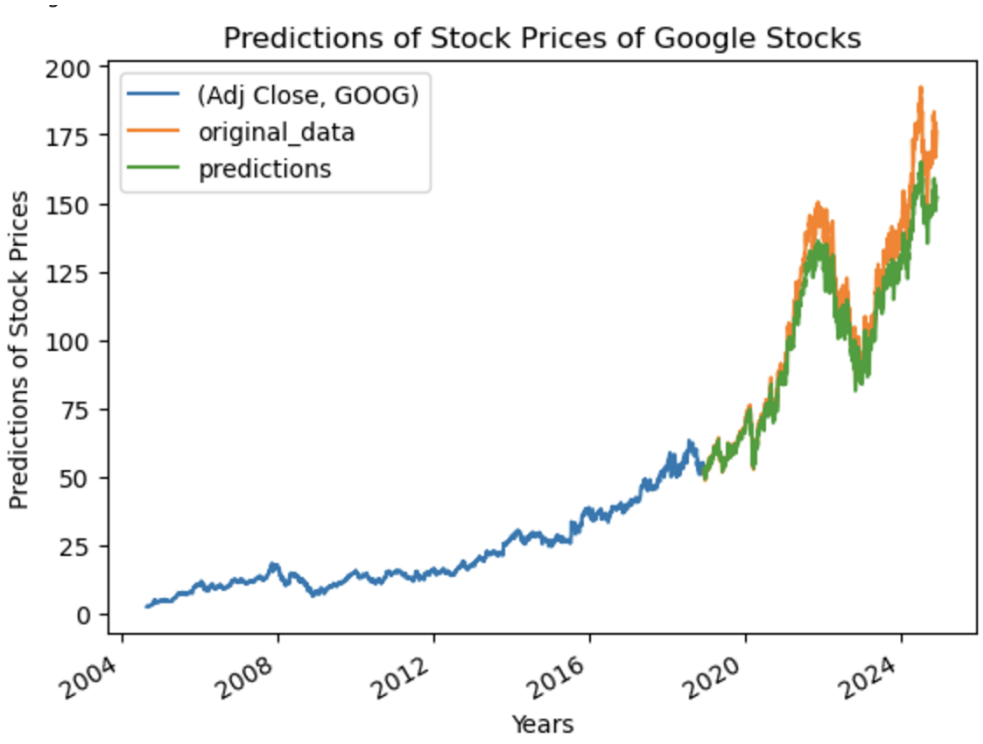

Figure 9: Comparison of original test data and predicted stock prices using the trained model

5 Data Availability:

After installing Yahoo Finance as yfinance , click on the following link to access the data : Google stock prices dataset.