QS STARS RATED FOR EXCELLENCE 2012

FPT Education

FPT UNIVERSITY

Analyze and predict stock prices using LSTM

Group 9:

Le Van Tra

Phan Duc Hien

Supervisor: Mr. Nguyen Quoc Trung

DSP391m

Summer Semester - 2024

- Ho Chi Minh City, July 1, 2024 –

This page is intentionally left blank

I. Introduction

1. Overview

1.1 Project Information

- Project name: Analyze and predict stock prices using LSTM.
- Group name: Group 7.

1.2 Project Team

a. Supervisor

Full Name	Email	Mobile	Title
Nguyen Quoc Trung	trungnq46@fe.edu.vn trungnq46@fpt.edu.vn	0979350707	Lecturer

Table 1. Supervisor contact information

b. Team Members

Full Name	Email	Mobile	Role
Le Van Tra	tralvse160069@fpt.edu.vn	0342603821	Leader
Phan Duc Hien	Hienpdse173074@fpt.edu.vn	0962064670	Member

Table 2. Team member contact information

2. Background and Context

The financial market is a complex system where billions of transactions take place every day. Stock prices are influenced by a multitude of factors, including economic indicators, company earnings reports, and global events. Predicting stock prices accurately is a challenging task due to the inherent volatility and randomness in the market.

Traditional methods of stock price prediction have relied on fundamental and technical analysis. Fundamental analysis involves evaluating a company’s financial statements, industry position, and market trends. Technical analysis, on the other hand, focuses on patterns in trading data such as price and volume. While these methods have their merits, they often fall short in predicting future prices with high accuracy.

With the advent of machine learning and artificial intelligence, new methods for stock price prediction have emerged. Long Short-Term Memory (LSTM) is one such method. LSTM is a type of recurrent neural network (RNN) that can learn and remember patterns over time, making it particularly suited for time-series data like stock prices.

Our project, “Analyze and Predict Stock Prices Using LSTM”, aims to leverage the power of LSTM to predict stock prices. We believe that by incorporating LSTM, we can capture the temporal dependencies in stock price data and make more accurate predictions.

This proposal outlines our approach to building and training an LSTM model for stock price prediction. We will detail the data we plan to use, the preprocessing steps required, the architecture of our LSTM model, and how we plan to evaluate its performance.

II. Data

1. Data overview:

This dataset contains historical prices of Tesla's stock (TSLA) from 2010 to the present. The data was collected from Yahoo Finance (https://finance.yahoo.com) using the yfinance library. After collecting the data, it was preprocessed, removing some features and adding new ones. The final dataset consists of 3,519 rows and 7 columns in CSV format.

Our dataset includes the following common features:

Date: The trading date of Tesla's stock (TSLA), formatted as MM-DD-YYYY.

Open: The stock price at the beginning of the trading session, formatted as price (dollar)/share.

High: The highest stock price reached during the trading session, formatted as price (dollar)/share.

Low: The lowest stock price reached during the trading session, formatted as price (dollar)/share.

Close: The stock price at the end of the trading session, formatted as price (dollar)/share.

Volume: The number of shares traded during the session, formatted as shares.

Volatility: The degree of price fluctuation of the stock between sessions, formatted as price (dollar)/share.

Among these, "Open" is the target feature we aim to predict. "Close" and "Volatility" will be used as features to predict "Open."

2. Data Cleaning and Preprocessing:

We used the pandas library for data cleaning and preprocessing:

Step 1: Load Data: We import the dataset stored in CSV format into a DataFrame.

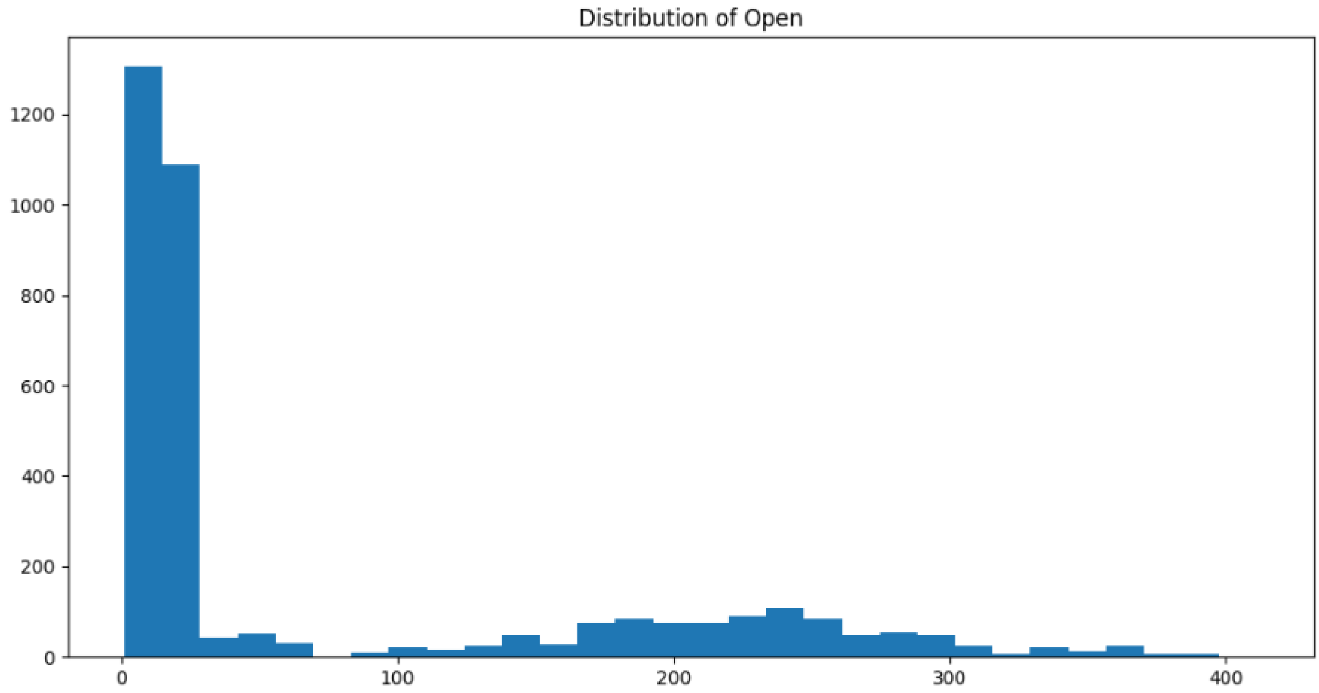
Step 2: Format Date: We ensure the data in the "Date" column is in the correct format (MM-DD-YYYY) using the **‘dt.date’** function.

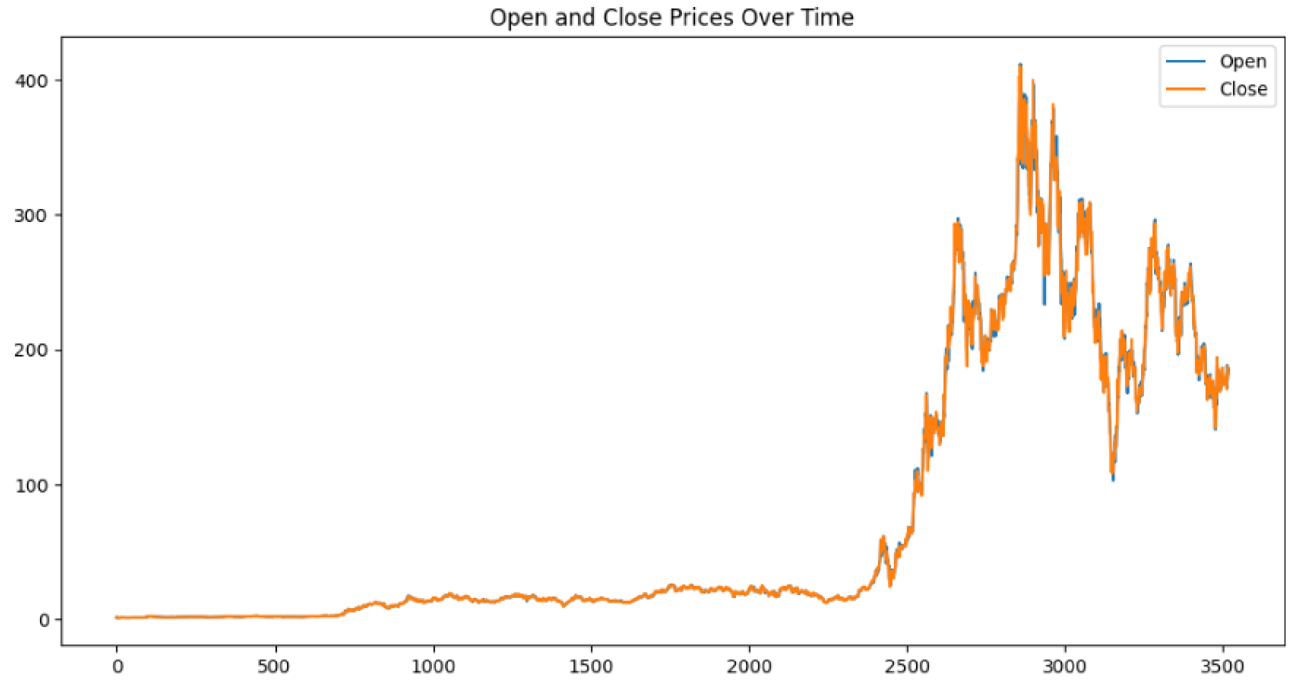
Step 3: Create Volatility Column: We create an additional column, "Volatility," which represents the price fluctuation between two days by subtracting the "Open" value of the previous row from the "Open" value of the current row. We use the **'diff()'** function for this (e.g., **tesla_data['Volatility'] = tesla_data['Open'].diff()**).

Step 4: Remove Unnecessary Columns: We delete columns with no values, such as "Dividends" and "Stock Splits," using the **'drop()'** function.

Step 5: Save Cleaned Data: We save the cleaned data into a new CSV file using the **'to_csv'** function.

3. Exploratory and Analysis(EDA):





Based on the distribution chart of Tesla's stock prices over time, with the price (dollars per share) on the X-axis and frequency on the Y-axis, as well as the "Open and Close Prices Over Time" chart with time on the X-axis and price on the Y-axis, we can observe that historically, the majority of prices fluctuated between $1 and $20 from 2010 to the end of 2019, showing slow growth. However, starting in 2020, the price of TSLA shares began to experience explosive growth. When looking at Tesla's history, we can see that this growth was driven by the success of the Model 3 and government support for electric vehicles. By 2021, Tesla's stock price had peaked, making Elon Musk the richest person in the world

III. Methodology

1. Model Development:

1.1 Model and Training Data Strategy

Model selection: LSTM (Long Short-Term Memory)

LSTM (Long Short-Term Memory) is a type of recurrent neural network (RNN) introduced by Hochreiter and Schmidhuber in 1997. LSTM is designed to address the "vanishing gradient" and "exploding gradient" problems commonly encountered in traditional RNNs when processing long sequences of data.

Architecture of LSTM The LSTM architecture includes the following main components:

Cell State: Considered the long-term memory of LSTM. It stores information across many time steps. Forget Gate: Decides which information from the cell state will be discarded. Input Gate: Decides which information will be added to the cell state. Output Gate: Decides which part of the cell state will be output as the output of the current time step.

Data Splitting Strategy:

To effectively train and evaluate our model, we split the dataset into two parts: training and testing sets. Based on standard practice and our experience, we used an 80/20 split, where 80% of the data is allocated to the training set and 20% to the testing set. This approach ensures that our model has sufficient data to learn from while also having enough data to evaluate its performance and generalizability.

The training set is used to train the model, allowing it to learn patterns and relationships within the data. The testing set is then used to assess the model's performance on unseen data, helping us evaluate its predictive accuracy and identify any potential overfitting.

By using this strategy, we aim to create a balanced approach that maximizes the model's learning potential while providing a reliable assessment of its performance.

Feature Engineering and Selection:

We chose to predict the opening price of the stock to enable buyers to make informed and strategic decisions regarding their stock purchases and sales. We selected the closing price ("Close") and the price volatility ("Volatility") as features to make our predictions.

This decision is based on several reasons:

Historical Significance: The closing price reflects the final consensus value of the stock for a trading day, providing insights into the stock's overall performance and market sentiment. It often serves as a benchmark for future predictions.

Market Trends: Price volatility measures the degree of variation in the stock price over time, indicating the market's stability and potential risk. High volatility often correlates with greater uncertainty and potential for larger price swings, which can be crucial for predicting the opening price.

Correlation: The closing price and volatility are likely to have a significant correlation with the opening price. By analyzing these features, we can capture important patterns and trends that influence the stock's next opening price.

1.2 Training procedure

We utilized LSTM (Long Short-Term Memory) and GRU (Gated Recurrent Unit) models for training and comparison. Each model was trained for 200 epochs with the Adam optimizer and Mean Absolute Error (MAE) as the loss function. We saved the model with the lowest loss during training.

Our training procedure involved the following steps:

Model Setup:

We set up both LSTM and GRU models to capture temporal dependencies in the stock price data.

Both models were compiled using the Adam optimizer and MAE as the loss function, which is suitable for regression tasks.

Training and Validation:

Each model was trained for 200 epochs to ensure sufficient learning and convergence.

Feature Input Variations:

We trained each model twice with different feature inputs:

Close Only: The first training used only the "Close" price as the input feature.

Volatility Only: The second training used only the "Volatility" as the input feature.

Evaluation and Comparison:

After training, we evaluated each model's performance by plotting the MAE and printing the MAE values for both feature input variations.

2. Model Evaluation:

To evaluate the performance of our model, we employed the Mean Absolute Error (MAE) as our primary metric.

The Mean Absolute Error is a popular metric used in regression problems and is particularly useful in our case of stock price prediction. It is calculated as the average of the absolute differences between the actual and predicted values. Mathematically, it can be represented as:

$$\text{MAE} = \frac{1}{n}\sum_{1}^{n}|y_i - \hat{y}_i||$$

where: y_i represents the actual value, $\hat{y}_i$ represents the predicted value, and n is the total number of data points.

The MAE gives us a straightforward measure of how far off our predictions are on average. A lower MAE indicates a better fit of the model to the data. One of the key advantages of using MAE is that it does not penalize large errors as severely as metrics like Mean Squared Error (MSE), making it more robust to outliers in the data.

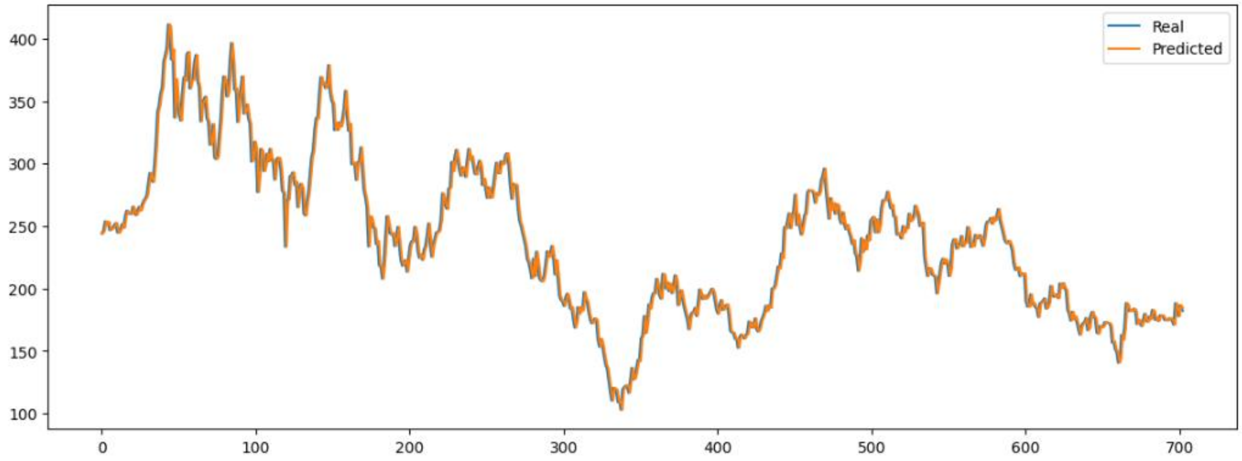
In stock price prediction, a model with a lower MAE will be more reliable as it indicates that the model's predictions are, on average, closer to the actual stock prices. This is crucial for investors who rely on these predictions to make informed investment decisions.

3. Results and Visualization:

Comparison of Mean Absolute Error (MAE) for Different Models and Inputs:

	Open	Volatility
LSTM with 5 days input	13,0044	7,2108
LSTM with 10 days input	21,0494	7,2201
LSTM with 30 days input	12,6189	7,2402
GRU	11,0997	7,2386

Comparison of Predicted and Actual Stock Prices:



Upon completion of the model training, we compared the Mean Absolute Error (MAE) of our models. Our findings indicate that the use of Long Short-Term Memory (LSTM) and Volatility as input variables significantly improves the accuracy of our predictions, resulting in a lower loss.

The LSTM model, with its ability to remember patterns over long sequences, proved to be particularly effective in capturing the temporal dependencies inherent in stock price data. This, coupled with the inclusion of Volatility - a key indicator of price variations - as an input, allowed our model to better understand and predict future stock prices.

The lower MAE achieved by this approach demonstrates its superiority over other models we tested. The reduced loss indicates that our predictions were, on average, closer to the actual stock prices, which is a highly desirable attribute in the context of stock price prediction.

IV. Conclution and Recommendation

1. Conclution:

Our project has demonstrated the effectiveness of using Long Short-Term Memory (LSTM) models and Volatility as input variables for predicting stock prices. The results, as indicated by the lower Mean Absolute Error (MAE), show that our approach provides more accurate predictions compared to other models we tested.

The use of LSTM takes advantage of the temporal dependencies in stock price data, while the inclusion of Volatility as an input variable allows the model to account for price variations, a critical factor in stock price movements.

These findings have significant implications for investors and financial analysts who rely on accurate stock price predictions for making informed investment decisions. The improved accuracy of our model could potentially lead to more profitable investment strategies and better risk management.

However, it's important to note that stock price prediction is inherently uncertain and influenced by a multitude of factors. While our model has shown promising results, it should be used in conjunction with other tools and knowledge for the best outcomes.

In future work, we plan to explore other types of neural networks and input variables to further improve the accuracy of our predictions. We also aim to test our model on different stock markets to assess its generalizability.

2. Recommendation:

Based on our findings, we recommend the following strategies to further improve the accuracy of stock price predictions:

Incorporate Multiple Feature Inputs: Our study used 'Close' and 'Volatility' as input variables. However, stock prices are influenced by a multitude of factors. Therefore, incorporating additional feature inputs such as 'EPS', 'P/E', 'ROA', 'Volume', and other technical indicators could potentially enhance the model's predictive power.

Ensemble of Models: An ensemble approach, which averages the predictions from multiple models, can often yield better results. This is because different models may capture different patterns in the data, and averaging their predictions can help to balance out their individual weaknesses.

By implementing these recommendations, we believe that the accuracy of stock price predictions can be significantly improved, thereby aiding investors in making more informed and profitable investment decisions.