Front End Engineering-II /Artificial Intelligence and Machine Learning

Project Report
Semester-IV (Batch-2022)

Title of the Project: Netflix Stock Price Predictor

Supervised By:

Dr Kiran Deep Singh

Submitted By:

PriyanshArora(2210990684)

Pranav Mangal(2210990995)

Palak Bisht(2210990631)

Pratham Khatkar(2210990672)

Department of Computer Science and Engineering
Chitkara University Institute of Engineering & Technology, Punjab

ABSTRACT

The project initiation involved an exhaustive literature review, delving into existing research on factors that are involved in predicting the stock price accurately, predictive modeling methodologies, and the application of AI/ML in business. Collaborating closely with domain experts, the project team delineated the scope and objectives of the predictive model, ensuring alignment with the complexities of stock Price detection.

Data acquisition commenced with the procurement of comprehensive data sets from authoritative sources, spanning national cancer registries, clinical databases, and research repositories. Subsequent data pre processing entailed meticulous cleaning, feature engineering, and normalization procedures to enhance data quality and facilitate compatibility with ML algorithms.

The model development phase encompassed the strategic selection and implementation of diverse ML algorithms, including linear regression support vector machines (SVM),LSTM. Model training unfolded through iterative experimentation and optimizing algorithm configuration

To ensure model robustness and mitigate over fitting, rigorous cross-validation techniques were employed, validating the generalization of the models across diverse data sets. Additionally, exploratory data analysis (EDA) techniques were leveraged to uncover insights into the underlying patterns and relationships within the data, guiding feature selection and refinement processes.

Furthermore, outlier detection methodologies were deployed to identify and address anomalous data points, while feature scaling techniques were applied to normalize the scale of features, ensuring equitable influence on model outcomes.

TABLE OF CONTENTS

INTRODUCTION

In this report, we analyse the stock data of Netflix (NFLX). Stock market analysis is crucial for investors and traders to make informed decisions. This study aims to explore the data, visualize trends and build predictive models to forecast future stock prices.

The objective of this analysis are:

- To understand the historical trends in Netflix stock prices.
- To explore and visualize the data for better insights.
- To build predictive models using various ML techniques.
- To evaluate and compare the performance of different models.

DETAILED SUMMARY:

Our journey into the realm of Netflix stock price predictor began with a comprehensive review of existing stock price, delving into the intricacies of market analysis, predictive modeling techniques, and the application of AI/ML in entertainment analytics. The initial phase involved acquiring a rich repository of data from diverse and reputable sources, including official movie databases, user ratings, and expert reviews.

Data preprocessing emerged as a crucial step, where we meticulously cleaned, transformed, and engineered features to prepare the dataset for analysis. This involved addressing inconsistencies, handling missing values, and extracting meaningful insights from raw movie metadata. Feature engineering played a pivotal role in capturing the essence of cinematic preferences, enabling us to create a nuanced representation of user tastes and movie attributes.
In the model development phase, we explored a spectrum of machine learning algorithms tailored to the task of prediction. From collaborative filtering methods to content-based approaches, each algorithm was carefully selected and implemented to leverage the unique characteristics of our dataset. Recurrent neural networks (RNNs) stood out as particularly promising for modeling sequential data inherent in user viewing histories and movie sequences.

Model training was an iterative process, where we fine-tuned network architectures and hyperparameters to optimize key performance metrics such as MSE,MAE,RMSE,R2 SCORE.

. Cross-validation techniques played a pivotal role in assessing model robustness and preventing overfitting, ensuring that our prediction system could generalize effectively to new users and unseen movies.
In essence, our journey through the stock price prediction was characterized by a relentless pursuit of excellence, fueled by a passion for cinema and innovation. As we navigated through the complexities of data analysis and model development, our ultimate goal remained clear: to enhance the prediction for our users.

The primary goal of this research is to pioneer the development of an innovative and highly effective movie recommendation system using cutting-edge AIML techniques.

.

3.1 BACKGROUND

Stock market analysis has always been a critical area of interest for investors, economists, and financial analysts. With the advent of technology and the availability of extensive historical data, analysing stock market trends has become more sophisticated and data-driven. The stock market represents a complex system where numerous factors influence the prices of stocks, including market sentiment, economic indicators, company performance, and geopolitical events.

Netflix, Inc. (NFLX), a leading streaming service provider, has been a focal point of interest in the stock market due to its rapid growth and significant impact on the entertainment industry. Since its initial public offering (IPO) in 2002, Netflix has experienced remarkable growth, transforming from a DVD rental service to a global streaming giant. The company's stock has been subject to high volatility, reflecting investor sentiment and market trends.

Analysing Netflix's stock data is not only crucial for understanding the company's financial health but also for gaining insights into broader market dynamics. Investors and traders are particularly interested in predicting future stock prices to maximize returns and manage risks effectively. This analysis involves examining historical stock prices, trading volumes, and other financial metrics to identify patterns and trends.

3.2 SIGNIFICANCE OF THE PROBLEM

The ability to accurately predict stock prices is highly valuable in the financial industry. It allows investors to make informed decisions about buying, holding, or selling stocks, thereby optimizing their investment strategies. Predictive modeling in stock market analysis can significantly enhance decision-making processes, leading to better financial outcomes.

For Netflix, whose stock has shown substantial growth and volatility, understanding the factors that drive its stock price movements is essential. Investors need reliable models to forecast future prices, which can help them navigate the uncertainties of the market. Accurate predictions can lead to profitable trading strategies and better risk management, particularly in a highly dynamic and competitive market environment.
In this report, we aim to delve into Netflix's stock data, exploring historical trends, visualizing data for better insights, and building predictive models using various machine learning techniques. By doing so, we seek to provide a comprehensive analysis that can aid in making informed investment decisions and contribute to the existing body of knowledge in financial analysis and stock market prediction.

3.4 OBJECTIVES

The primary aim of this research is to construct a predictive model for the early identification of stock price prediction using advanced Artificial Intelligence and Machine Learning (AIML) methodologies.

Specifically, our objectives encompass:

1. Collect and import Netflix stock data from a reliable source. Preprocess the data to handle missing values, convert data types, and ensure the data is ready for analysis.
2. Conduct exploratory data analysis (EDA) to summarize the main characteristics of the data, identify patterns, and detect anomalies.
3. Visualize the historical trends of Netflix stock prices using time series plots. Create correlation matrices to identify relationships between different stock metrics (e.g., open, high, low, close prices. Generate various plots (e.g., scatter plots, bar plots, histograms) to illustrate the distribution and relationships of stock data attributes.
4. To do the Predictive modelling like Spliting the dataset into training and testing sets to evaluate model performance, Explore the application of linear regression to predict binary outcomes related to stock prices (e.g., whether the closing price is higher than the opening price),

Implement a Long Short-Term Memory (LSTM) neural network for time series forecasting of stock prices and evaluate its performance.

5. Provide recommendations for future work and potential improvements in stock price prediction models.
6. Demonstrate the process of saving and loading machine learning models using serialization techniques (e.g., pickle). Outline steps for deploying the predictive models in a real-world scenario, including data pipeline setup and model integration.
7. Crucially, our research is driven by a commitment to transcend conventional approaches and deliver a prediction framework that not only predicts stock price but also provides diverse and contextually relevant results. We envision apprediction system that adapts to evolving user interests, leverages real-time feedback for our system.

By achieving these objectives, this lab report aims to deliver a thorough and insightful analysis of Netflix's stock data, providing valuable tools and knowledge for stakeholders interested in stock market prediction and financial analytics.

PROBLEM DEFINITION AND REQUIREMENTS

4.1 PROBLEM STATEMENT

The volatility and dynamic nature of stock markets make accurate stock price prediction a highly challenging yet critical task for investors and financial analysts. In particular, predicting the stock prices of high-profile companies like Netflix, which experiences significant fluctuations, requires sophisticated analytical approaches and robust models. This lab report aims to address this problem by leveraging historical stock data of Netflix to develop and evaluate multiple predictive models. The primary focus is on using machine learning techniques, including linear regression, linear regression, Support Vector Machine (SVM), and Long Short-Term Memory (LSTM) neural networks, to forecast stock prices and assess their performance. The goal is to provide a comparative analysis of these models, identify the most effective approach for stock price prediction, and offer insights into the factors influencing stock price movements. By doing so, this study seeks to contribute to the broader field of financial analytics and support better decision-making for stakeholders in the stock market.

In this context, our project aims to address the following critical question:

How can we leverage Artificial Intelligence and Machine Learning techniques to develop a predictive model for win probability, providing stakeholders with valuable insights for strategic decision-making and enhancing the overall stock price experience?

4.2 SOFTWARE REQUIREMENTS

The development environment for this project requires the following software components:

1. Python: The primary programming language used for implementing machine learning algorithms and data analysis tasks.
2. Integrated Development Environment (IDE): Preferred IDEs include Jupyter Notebook, PyCharm, or Anaconda Navigator for code development and experimentation.
3. Python Libraries: Various Python libraries are utilized for data manipulation, visualization, and machine learning model development, including but not limited to:
   - NumPy
     For numerical computing and array manipulation.
   - Pandas
     For data manipulation and analysis.
   - Matplotlib and Seaborn
     For data visualization and exploratory data analysis.
   - Scikit-learn
     For implementing machine learning algorithms and model evaluation.
   - AIML Python Package
     For implementing Artificial Intelligence Markup Language (AIML) techniques and algorithms.

4.3 HARDWARE REQUIREMENTS

The hardware requirements for running the project are as follows:

1. Processor
   A multi-core processor (e.g., Intel Core i5 or higher) to handle computational tasks efficiently.
2. RAM
   At least 8GB of RAM is recommended for handling large datasets and complex machine learning models effectively.
3. Storage
   Sufficient storage space to accommodate the dataset and additional resources required for software installation and project files.

4.4 DATASET

The dataset used in this analysis comprises historical stock prices of Netflix Inc., obtained from a reliable financial data source. The dataset spans a significant period, providing daily records of Netflix's stock market activity. Each entry in the dataset includes the following key attributes:

i Date: The trading date, formatted as YYYY-MM-DD.
ii Open: The stock price at the beginning of the trading session.
iii High: The highest price reached during the trading session.
iv Low: The lowest price reached during the trading session.
v Close: The stock price at the end of the trading session.
vi Volume: The number of shares traded during the session.

These attributes collectively offer a comprehensive view of the stock's performance over time. The data is clean, with no missing values, ensuring the reliability of the subsequent analysis. This rich dataset serves as the foundation for performing exploratory data analysis, visualizations, and developing predictive models to forecast future stock prices. By understanding these historical trends and patterns, we aim to gain insights into the factors driving Netflix's stock price movements.

PROPOSED DESIGN AND METHODOLOGY

The main objective of this study is to develop, implement, and compare multiple machine learning models for predicting Netflix stock prices. By analyzing historical stock data, we aim to identify the most effective model that provides accurate and reliable predictions. This study will assist investors and financial analysts in making informed decisions.

1. Data Collection and Preprocessing

Data Collection:

- Source: The dataset used in this study is obtained from a publicly available source and contains historical stock prices of Netflix (NFLX).
- Attributes: The dataset includes features such as Date, Open, High, Low, Close, and Volume.
- Data Cleaning: Handling Missing Values: Checking for and addressing any missing values to ensure data integrity.
- Date Conversion: Converting the Date column to a datetime format to facilitate time series analysis.
- Removing Duplicates: Ensuring there are no duplicate rows in the dataset.
- Feature Engineering:
- Creating Additional Features: Calculating technical indicators such as moving averages, volatility, and other relevant metrics that may enhance the model's predictive power.

2. Exploratory Data Analysis (EDA)

Descriptive Statistics:

- Summary Statistics: Calculating mean, median, standard deviation, and other summary statistics to understand the distribution of each feature.
- Correlation Matrix: Creating a correlation matrix to identify the relationships between different features.

Visualization:

- Time Series Plots: Plotting the stock price over time to observe trends, patterns, and seasonality.
- Histograms: Visualizing the distribution of features to identify any skewness or anomalies.
- Scatter Plots: Analyzing the relationship between different features using scatter plots.

Model Development

- Linear Regression: To model the relationship between the stock's closing price and other features. Using scikit-learn's Linear Regression to fit the model and make predictions.

- linear Regression: To classify whether the stock will close higher than it opened. Utilizing scikit-learn's linear Regression for binary classification.
- Support Vector Machine (SVM):To classify stock prices based on input features. Using scikit-learn's SVC with a linear kernel to train and predict stock prices
- Long Short-Term Memory (LSTM): To capture temporal dependencies in stock prices and predict future prices based on historical sequences. Building an LSTM neural network using Keras and TensorFlow for time series forecasting.

Beyond recommendation accuracy, our methodology emphasizes the extraction of actionable insights from the developed model. We interpret the learned model parameters and feature importance scores to elucidate the key factors influencing stock price. Additionally, visualization techniques are employed to facilitate the interpretation of recommendation results and identify critical stock features driving user preferences.

3. Model Evaluation

Performance Metrics for Regression:

- Mean Squared Error (MSE): Evaluating the average squared difference between actual and predicted values.
- Mean Absolute Error (MAE): Measuring the average magnitude of errors in predictions.
- Mean Absolute Percentage Error (MAPE): Assessing prediction accuracy as a percentage.
- R-squared (R^2): Determining the proportion of variance explained by the model.
- Performance Metrics for Classification:
- Accuracy: Calculating the ratio of correctly predicted observations to total observations.
- Confusion Matrix: Evaluating the performance of the classification model by summarizing true positives, false positives, true negatives, and false negatives.
- Comparison of Models:
- Comparing the performance of all models to identify the one with the best predictive accuracy and reliability.

4. Visualization of Predictions
   - Line Plots: Plotting actual vs. predicted stock prices to visually compare model performance over time.
   - Scatter Plots: Visualizing the relationship between actual and predicted prices to evaluate model accuracy.

5. Insights and Conclusions

- Interpretation of Results: Drawing insights from the model's performance and understanding the key factors influencing stock prices.
- Recommendations: Providing actionable recommendations for investors based on the model predictions and identified trends.

By following this methodology, we aim to systematically approach the problem of stock price prediction, utilizing a combination of traditional and advanced machine learning techniques to derive meaningful insights and accurate predictions.
Through the systematic execution of these methodological steps, we aim to develop a robust and interpretable movie recommendation system, providing users with personalized and engaging movie suggestions tailored to their tastes and preferences.

5.1 FILE STRUCTURE

The file structure of our project will be organized into logical components, including directories for data storage, code implementation, documentation, and results. Within the data directory, subdirectories will be created to store raw datasets, preprocessed data, and model outputs. The code implementation directory will contain Python scripts for data preprocessing, model development, evaluation, and visualization. Documentation will include README files providing instructions for project setup and usage, as well as any additional documentation related to code implementation and methodology. Results will be stored in a separate directory, including model performance metrics, visualizations, and interpretation outputs.

5.2 ALGORITHMS USED

Our methodology entails the exploration of diverse machine learning algorithms within the AIML paradigm to predict obesity risk accurately. This includes:

1.Linear Regression:
Linear regression is a data analysis technique that predicts the value of unknown data by using another related and known data value

2. Support Vector Machines (SVM):

SVM is a supervised learning algorithm used for classification tasks. It's proficient in handling nonlinear decision boundaries, often achieved through kernel functions, thereby aiding in robust obesity prediction.

3. Long short-Term Memory(LSTM):
LSTMs are long short-term memory networks that use (ANN) artificial neural networks in the field of artificial intelligence (AI) and deep learning.

RESULTS

ANALYSIS AND MODEL EVALUATION

In this section, we present a detailed analysis of the results obtained from our AI/ML Project on stock price prediction.
Before fetching data we have to import some of the python libraries for example numpy, pandas, matplotlib and many more.
In this section, we delve into the analysis of our movie recommendation system, which utilizes count vectorization.. We present graphical representations of key metrics and performance indicators, offering insights into the effectiveness of each algorithm . Additionally, we provide an overview of the system's architecture and its corresponding accuracy.

FEATURES DISTRIBUTION

In developing a stock price predictor it's important to consider various features that can capture the market trend and enhance the effectiveness of the system. Here are some key features to focus on:

1. Date:

   It is the date for which stock is being predicted for its price.

2. Open Price:

   The price at which a stock trades when the market opens first

3. Close Price:

   The price at which the stock finishes when a market ends

4. High Price:

   The highest price the stock went for in the day trading window

5. Low price:

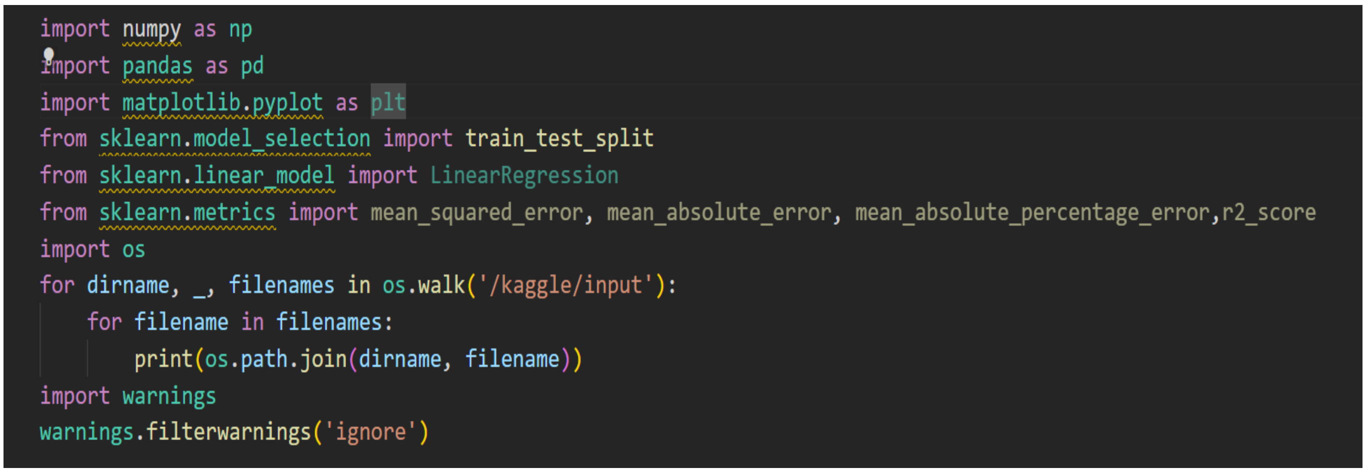
   The Lowest price the stock went for in the day trading window

6. Volume:

   Indicator for the number of shares that have been bought or sold that day.

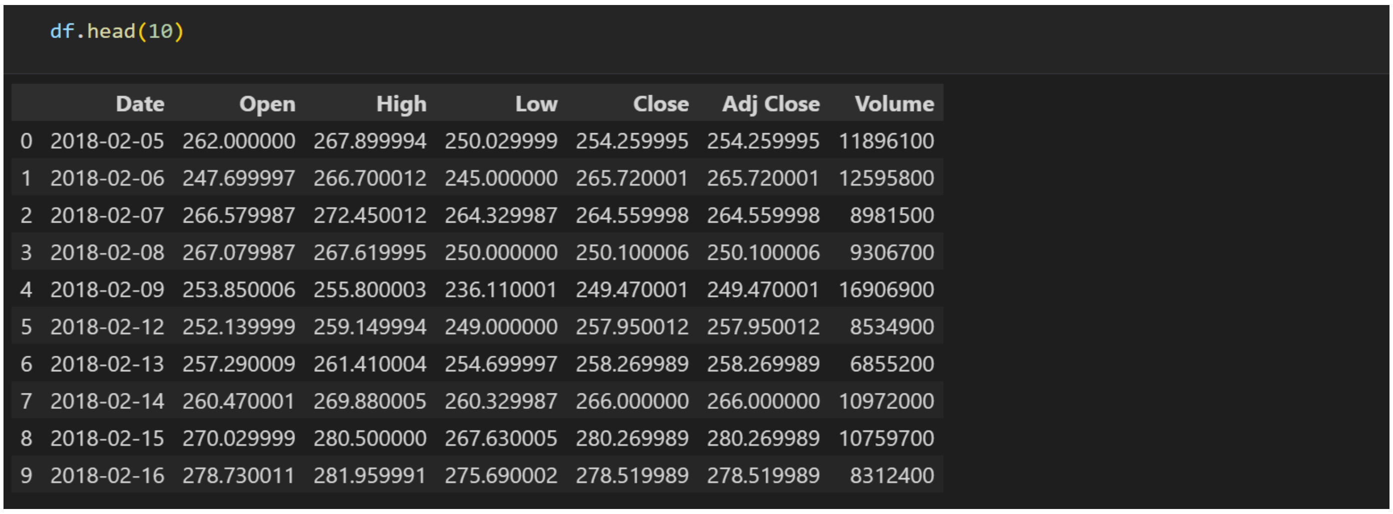
   By focusing on these key features, you can develop a movie recommendation system that effectively learns user preferences and provides personalized recommendations tailored to individual tastes and interests.

Now, We begin by fetching our data from the csv file named as “NFLX.csv” by writing the command (df = pd.read_csv(“./NFLX.csv”) and showcasing the graphical representations of key metrics and performance indicators, followed by an overview of the models utilized along with their corresponding accuracies.



```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, mean_absolute_error, mean_absolute_percentage_error,r2_score
import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
import warnings
warnings.filterwarnings('ignore')
```

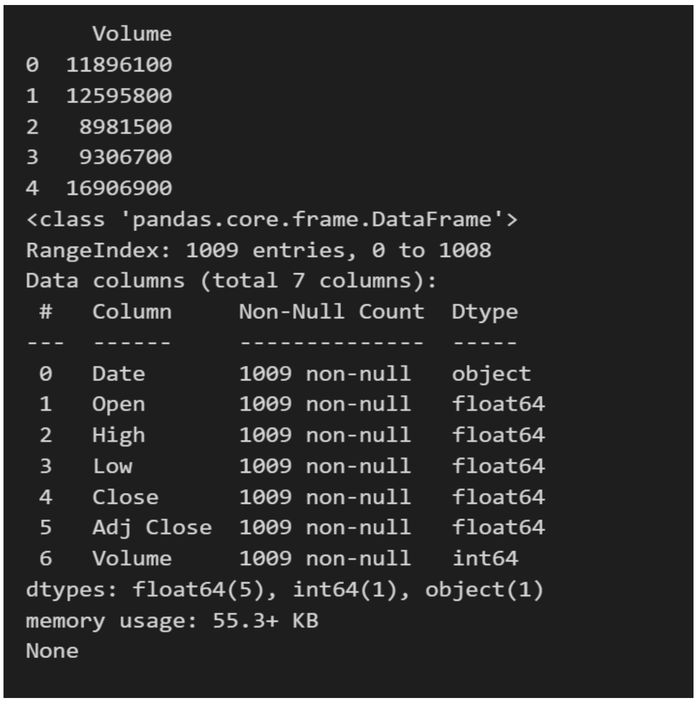
df.head(10)- Show first 10 rows and all columns of dataset.



```python
df.head(10)
```

	Date	Open	High	Low	Close	Adj Close	Volume
0	2018-02-05	262.000000	267.899994	250.029999	254.259995	254.259995	11896100
1	2018-02-06	247.699997	266.700012	245.000000	265.720001	265.720001	12595800
2	2018-02-07	266.579987	272.450012	264.329987	264.559998	264.559998	8981500
3	2018-02-08	267.079987	267.619995	250.000000	250.100006	250.100006	9306700
4	2018-02-09	253.850006	255.800003	236.110001	249.470001	249.470001	16906900
5	2018-02-12	252.139999	259.149994	249.000000	257.950012	257.950012	8534900
6	2018-02-13	257.290009	261.410004	254.699997	258.269989	258.269989	6855200
7	2018-02-14	260.470001	269.880005	260.329987	266.000000	266.000000	10972000
8	2018-02-15	270.029999	280.500000	267.630005	280.269989	280.269989	10759700
9	2018-02-16	278.730011	281.959991	275.690002	278.519989	278.519989	8312400

Then we have used the .info function to check the information of our data set so that we can make the changes that are required to be changed or updated before performing ML algorithm.

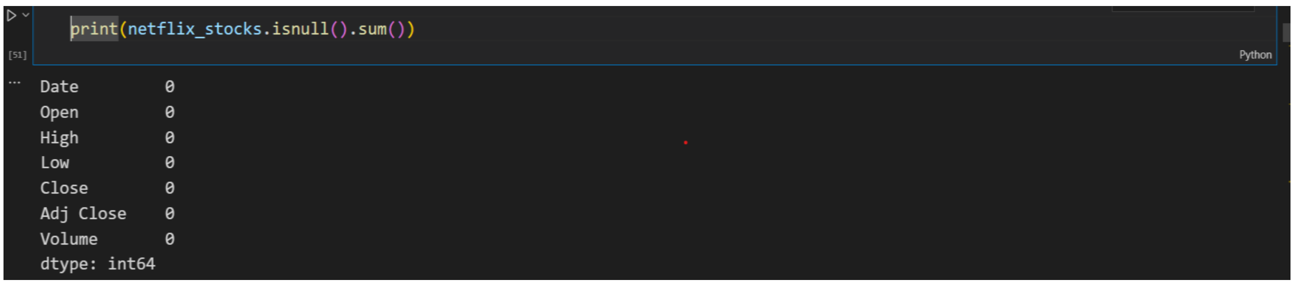


```
     Volume
0  11896100
1  12595800
2   8981500
3   9306700
4  16906900
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1009 entries, 0 to 1008
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   Date       1009 non-null   object 
 1   Open       1009 non-null   float64
 2   High       1009 non-null   float64
 3   Low        1009 non-null   float64
 4   Close      1009 non-null   float64
 5   Adj Close  1009 non-null   float64
 6   Volume     1009 non-null   int64  
dtypes: float64(5), int64(1), object(1)
memory usage: 55.3+ KB
None
```

After this we have changed the data type of “Date” column and make it as datetime data type.

Data Cleaning

Data cleaning is the process of fixing or removing incorrect, corrupted, incorrectly formatted, duplicate, or incomplete data within a dataset.
We have run the command to check the null values in our data set



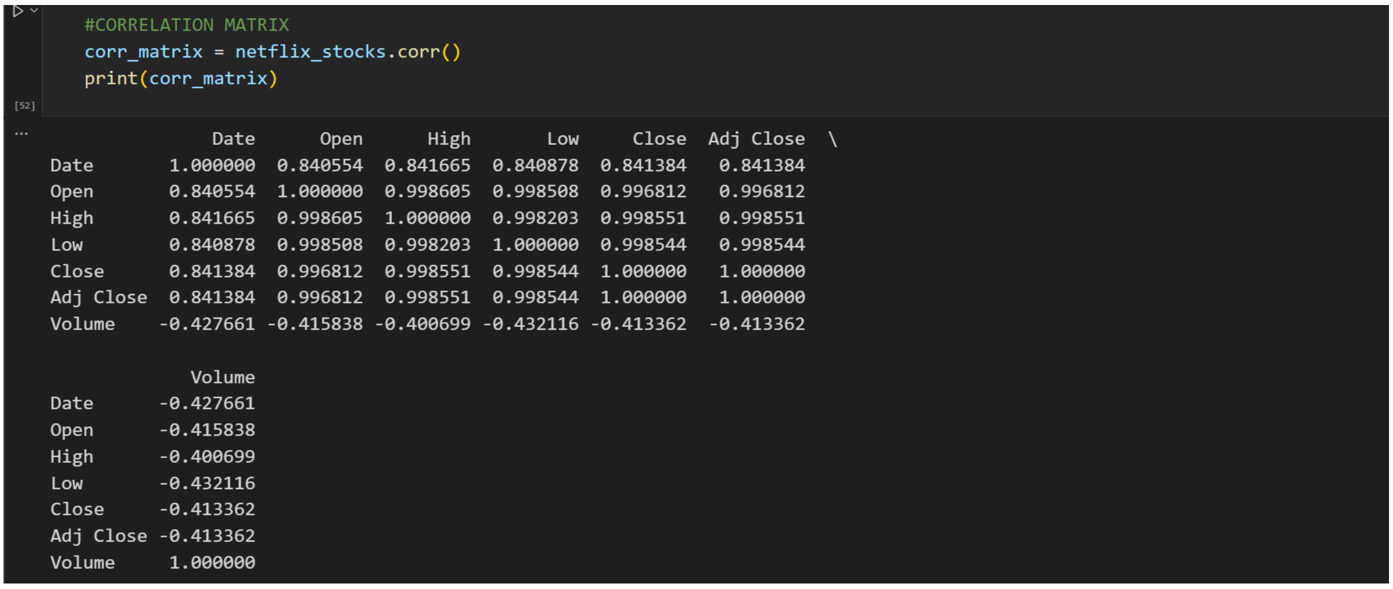
```
print(netflix_stocks.isnull().sum())
```

```
Date         0
Open         0
High         0
Low          0
Close        0
Adj Close    0
Volume       0
dtype: int64
```

Correlation Matrix

This command tells us that how our columns are correlated to each other and its range lie between

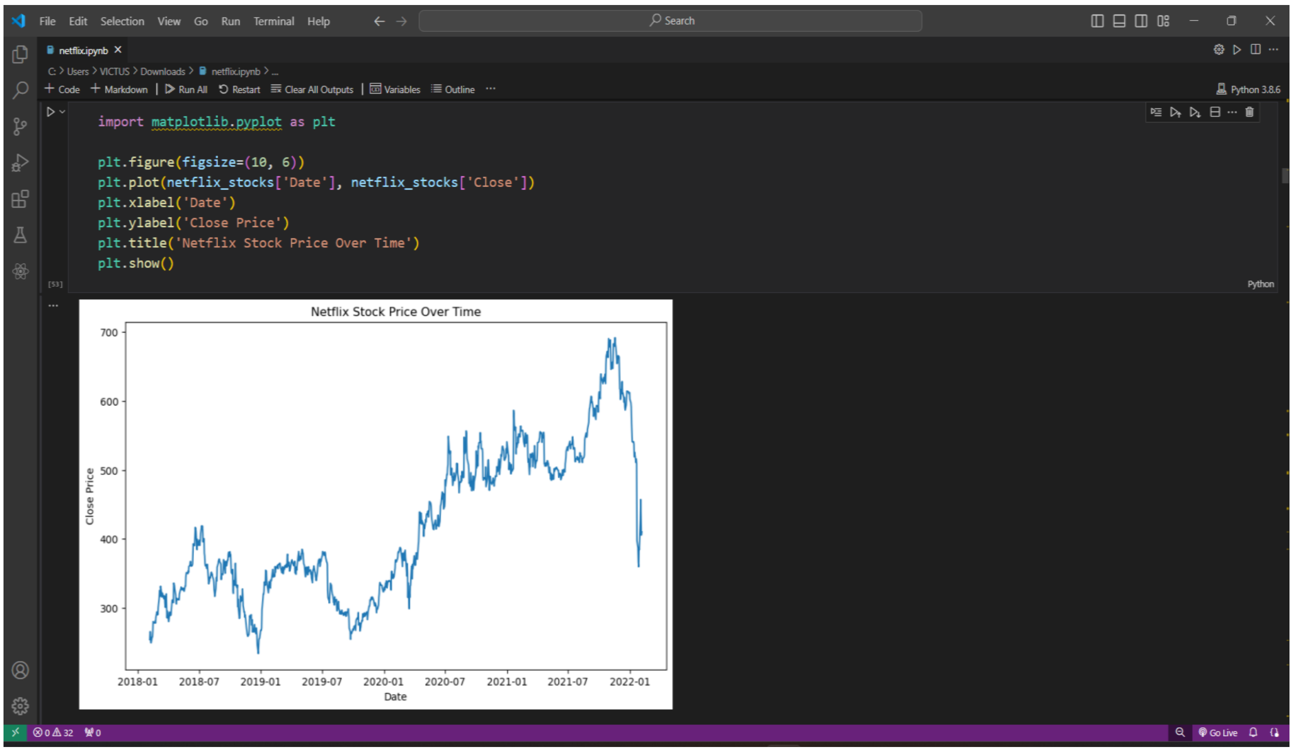
-1,1.



```
#CORRELATION MATRIX
corr_matrix = netflix_stocks.corr()
print(corr_matrix)
```

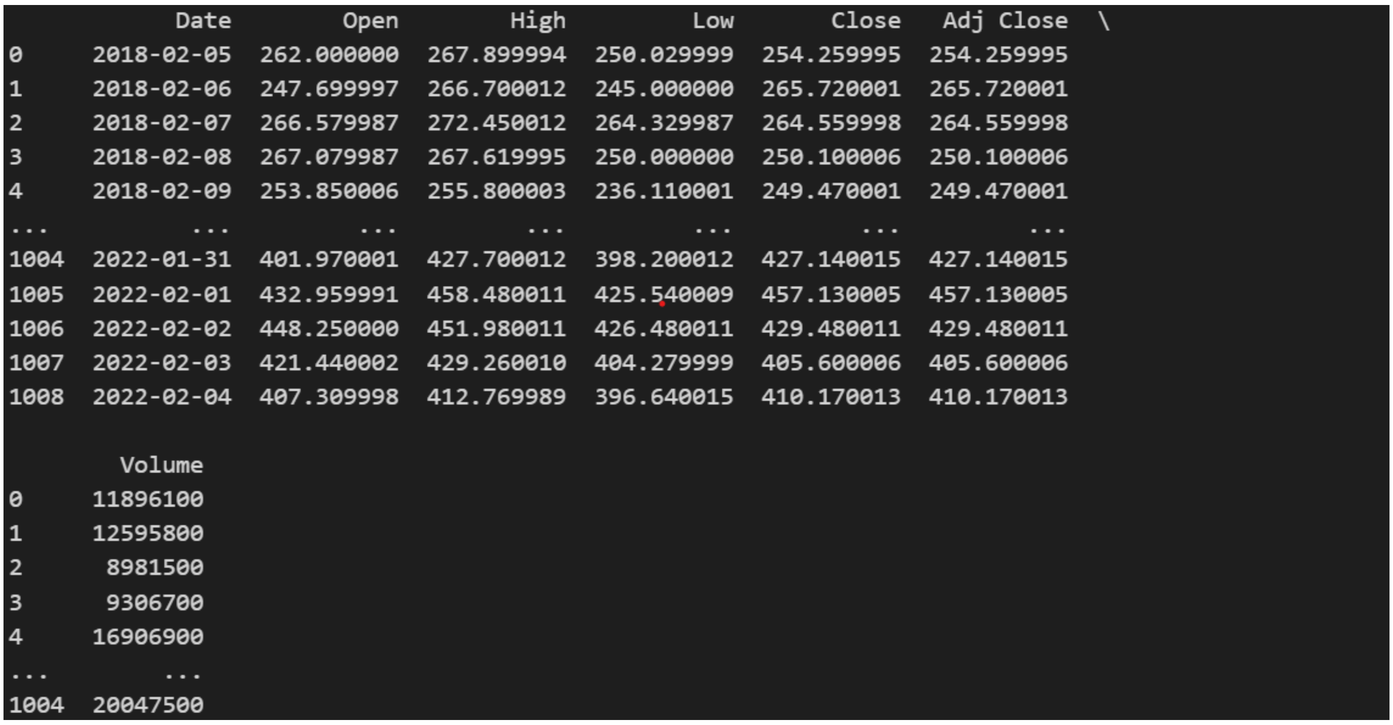
```
                Date      Open      High       Low     Close  Adj Close  \
Date        1.000000  0.840554  0.841665  0.840878  0.841384   0.841384
Open        0.840554  1.000000  0.998605  0.998508  0.996812   0.996812
High        0.841665  0.998605  1.000000  0.998203  0.998551   0.998551
Low         0.840878  0.998508  0.998203  1.000000  0.998544   0.998544
Close       0.841384  0.996812  0.998551  0.998544  1.000000   1.000000
Adj Close   0.841384  0.996812  0.998551  0.998544  1.000000   1.000000
Volume     -0.427661 -0.415838 -0.400699 -0.432116 -0.413362  -0.413362

              Volume
Date       -0.427661
Open       -0.415838
High       -0.400699
Low        -0.432116
Close      -0.413362
Adj Close -0.413362
Volume      1.000000
```

1. First of all we have made a line graph of previous data that between date(labelled at x-axis) and close time of stock price(labelled as y-axis).



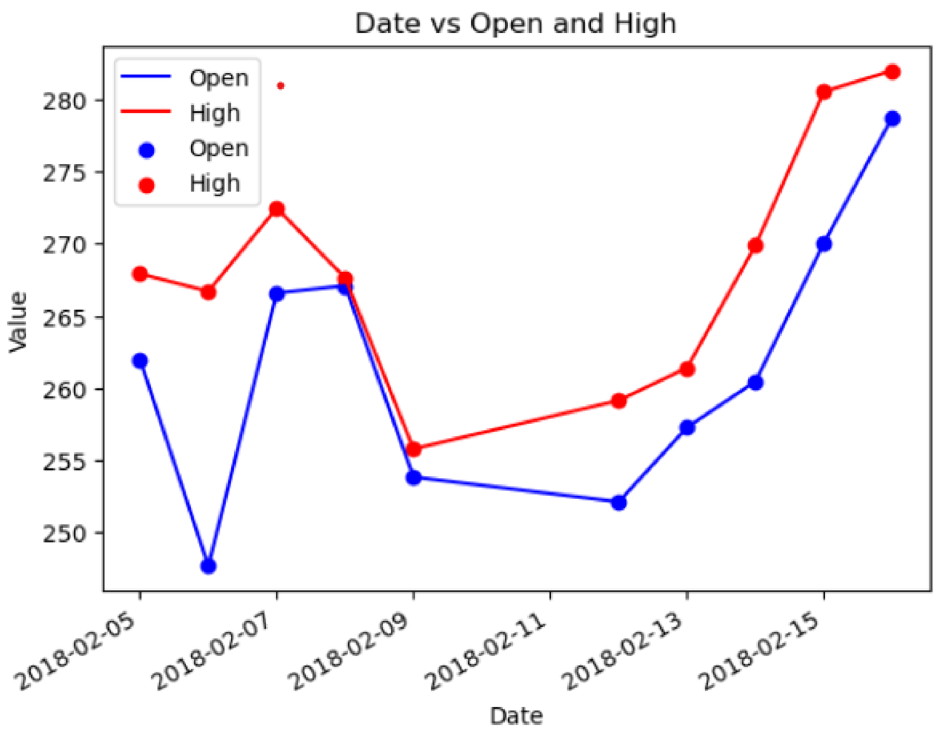
```python
import matplotlib.pyplot as plt

plt.figure(figsize=(10, 6))
plt.plot(netflix_stocks['Date'], netflix_stocks['Close'])
plt.xlabel('Date')
plt.ylabel('Close Price')
plt.title('Netflix Stock Price Over Time')
plt.show()
```

2. Then we create a data frame and made a bar plot showing

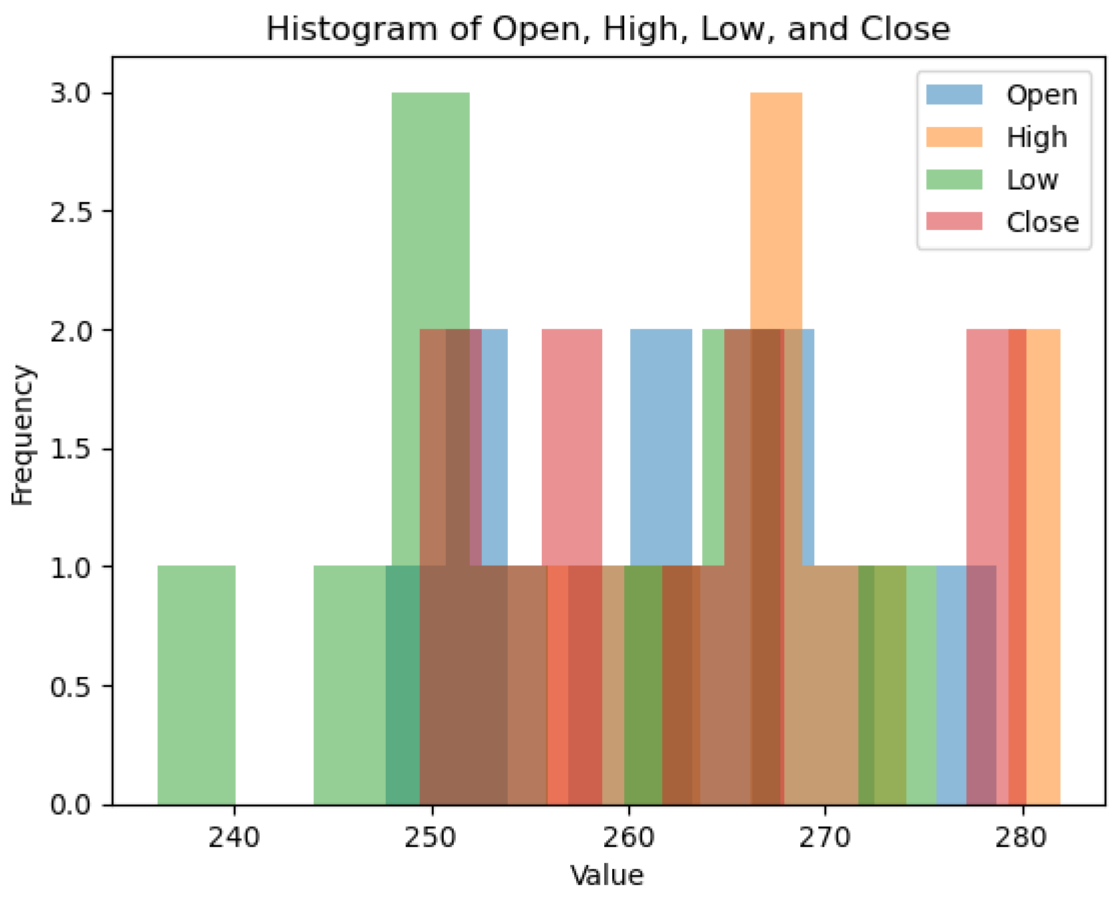


```
            Date        Open        High         Low       Close   Adj Close  \
0     2018-02-05  262.000000  267.899994  250.029999  254.259995  254.259995
1     2018-02-06  247.699997  266.700012  245.000000  265.720001  265.720001
2     2018-02-07  266.579987  272.450012  264.329987  264.559998  264.559998
3     2018-02-08  267.079987  267.619995  250.000000  250.100006  250.100006
4     2018-02-09  253.850006  255.800003  236.110001  249.470001  249.470001
...          ...         ...         ...         ...         ...         ...
1004  2022-01-31  401.970001  427.700012  398.200012  427.140015  427.140015
1005  2022-02-01  432.959991  458.480011  425.540009  457.130005  457.130005
1006  2022-02-02  448.250000  451.980011  426.480011  429.480011  429.480011
1007  2022-02-03  421.440002  429.260010  404.279999  405.600006  405.600006
1008  2022-02-04  407.309998  412.769989  396.640015  410.170013  410.170013

        Volume
0     11896100
1     12595800
2      8981500
3      9306700
4     16906900
...        ...
1004  20047500
```

Date vs Open/high graph

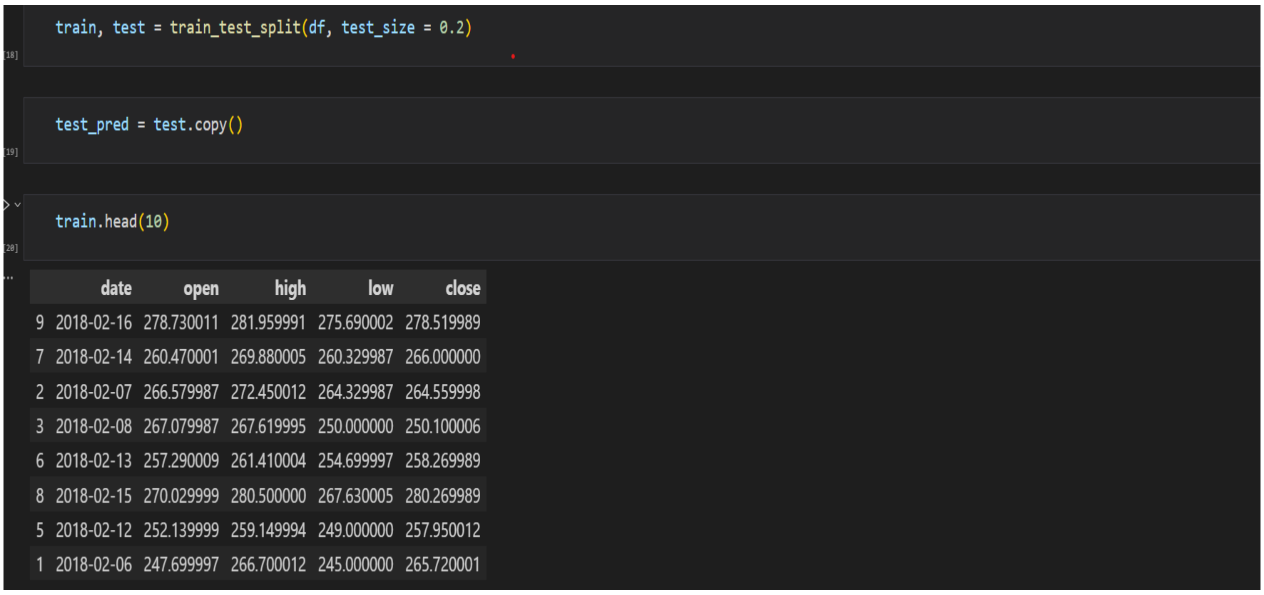


Histogram showing the fluctuation of stock price at high, low, opening and closing time of the market



Splitting of the dataset

We have split the data set into training and testing the ratio of split is 80% and 20%



```
train, test = train_test_split(df, test_size = 0.2)
```

```
test_pred = test.copy()
```

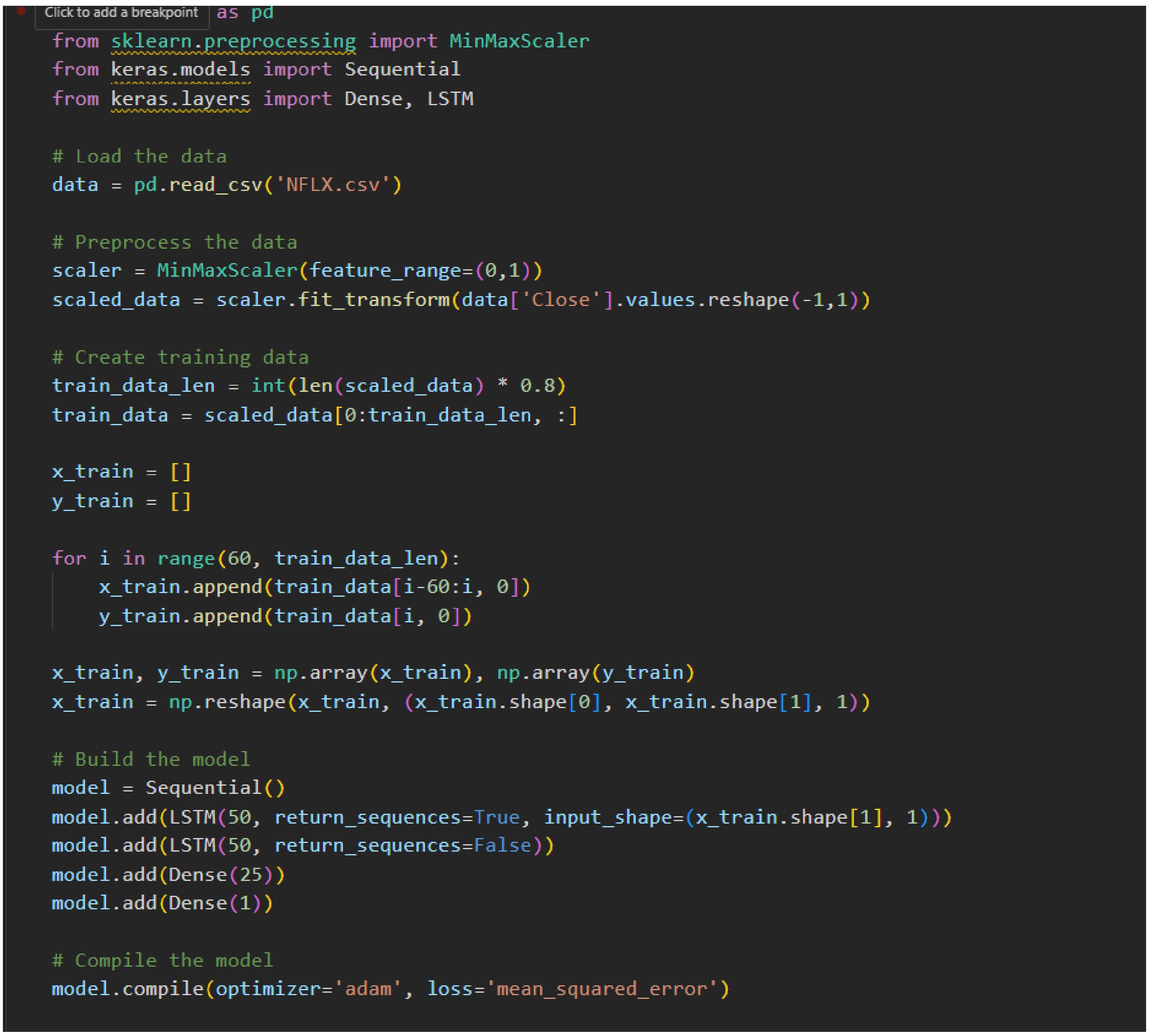
```
train.head(10)
```

```
         date        open        high         low       close
9  2018-02-16  278.730011  281.959991  275.690002  278.519989
7  2018-02-14  260.470001  269.880005  260.329987  266.000000
2  2018-02-07  266.579987  272.450012  264.329987  264.559998
3  2018-02-08  267.079987  267.619995  250.000000  250.100006
6  2018-02-13  257.290009  261.410004  254.699997  258.269989
8  2018-02-15  270.029999  280.500000  267.630005  280.269989
5  2018-02-12  252.139999  259.149994  249.000000  257.950012
1  2018-02-06  247.699997  266.700012  245.000000  265.720001
```

LSTM-Long short term memory

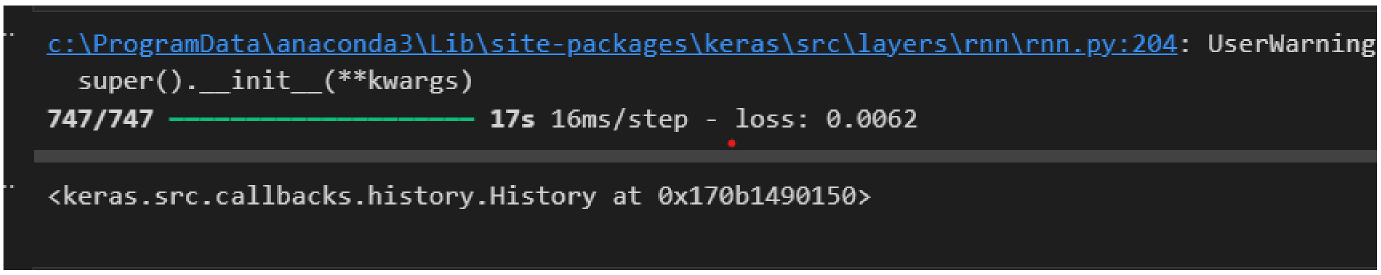
Long Short-Term Memory Networks is a deep learning, sequential neural network that allows information to persist. It is a special type of Recurrent Neural Network which is capable of handling the vanishing gradient problem faced by RNN. LSTM was designed by Hochreiter and Schmidhuber that resolves the problem caused by traditional rnns and machine learning algorithms. LSTM Model can be implemented in Python using the Keras library.

The practical implementation of lstm in our project is:



```python
as pd
from sklearn.preprocessing import MinMaxScaler
from keras.models import Sequential
from keras.layers import Dense, LSTM

# Load the data
data = pd.read_csv('NFLX.csv')

# Preprocess the data
scaler = MinMaxScaler(feature_range=(0,1))
scaled_data = scaler.fit_transform(data['Close'].values.reshape(-1,1))

# Create training data
train_data_len = int(len(scaled_data) * 0.8)
train_data = scaled_data[0:train_data_len, :]

x_train = []
y_train = []

for i in range(60, train_data_len):
    x_train.append(train_data[i-60:i, 0])
    y_train.append(train_data[i, 0])

x_train, y_train = np.array(x_train), np.array(y_train)
x_train = np.reshape(x_train, (x_train.shape[0], x_train.shape[1], 1))

# Build the model
model = Sequential()
model.add(LSTM(50, return_sequences=True, input_shape=(x_train.shape[1], 1)))
model.add(LSTM(50, return_sequences=False))
model.add(Dense(25))
model.add(Dense(1))

# Compile the model
model.compile(optimizer='adam', loss='mean_squared_error')
```

Output:



```
c:\ProgramData\anaconda3\Lib\site-packages\keras\src\layers\rnn\rnn.py:204: UserWarning
  super().__init__(**kwargs)
747/747 ━━━━━━━━━━━━━━━━━━━━ 17s 16ms/step - loss: 0.0062

<keras.src.callbacks.history.History at 0x170b1490150>
```

MODEL SUMMARY

Classification Report:

The Results of a classification report from a machine learning model, which includes metrics such as MAE,RMSE, MSE,R-2 score.

1.**MEAN ABSOLUTE ERROR (MAE)** is a measure of errors between paired observations expressing the same phenomenon

2.**MEAN SQUARE ERROR(MSE)** is a crucial metric for evaluating the performance of predictive models. It measures the average squared difference between the predicted and the actual target values within a dataset.

3.**ROOT MEAN SQUARE ERROR(RMSE):** It measures the average difference between values predicted by a model and the actual values. It provides an estimation of how well the model is able to predict the target value.

4.**R2 SCORE:**An R-Squared value shows how well the model predicts the outcome of the dependent variable. R-Squared values range from 0 to 1.

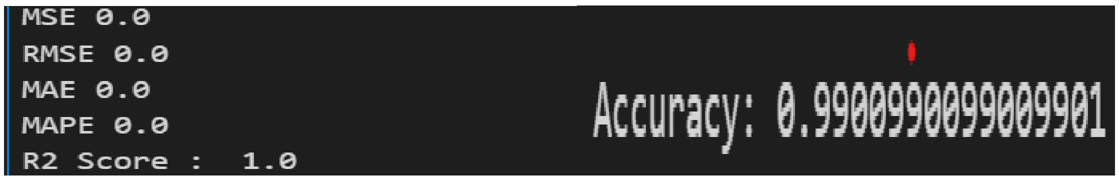
Here's it all.

1.LINEAR REGRESSION

Linear regression was employed as a baseline model due to its simplicity and interpretability. Despite its simplicity, linear regression yielded a respectable accuracy score of 0.99 on the validation dataset.

Analysis:

Based on the report you've provided, this model has achieved high scores for all of these metrics, indicating that it is performing well. The ,MAE,RMSE, MSE is 0.0 and R2-Score is 1.0, which is quite impressive. And the accuracy is 99%

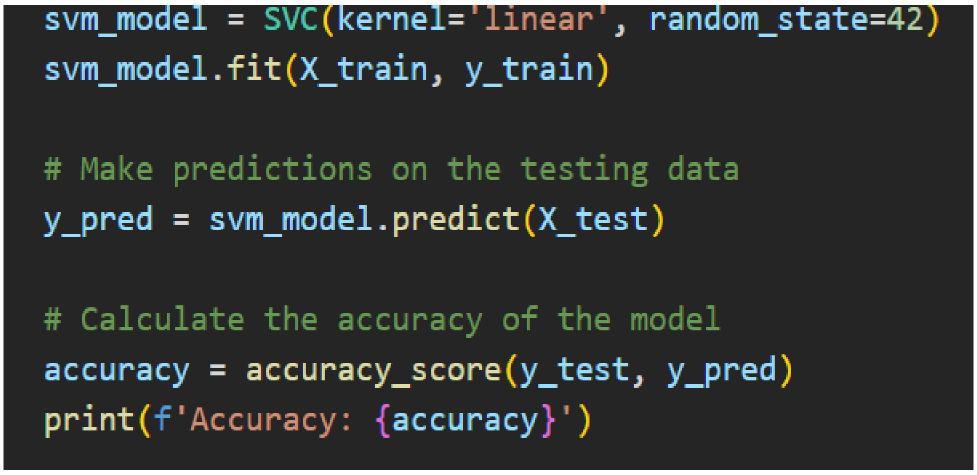
.

```
MSE 0.0
RMSE 0.0
MAE 0.0
MAPE 0.0
R2 Score :  1.0
```

```
Accuracy: 0.9900990099009901
```

2. SUPPORT VECTOR MACHINE (SVM)

SVM is a versatile and powerful algorithm for classification tasks, particularly suitable for high-dimensional data and scenarios where a clear margin of separation between classes exists. Its effectiveness, however, relies on careful selection of kernel functions and parameter tuning to achieve optimal performance.

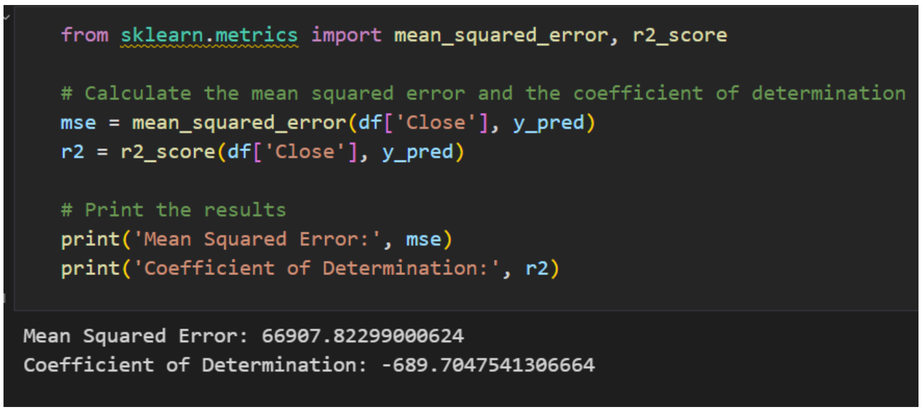


```
svm_model = SVC(kernel='linear', random_state=42)
svm_model.fit(X_train, y_train)

# Make predictions on the testing data
y_pred = svm_model.predict(X_test)

# Calculate the accuracy of the model
accuracy = accuracy_score(y_test, y_pred)
print(f'Accuracy: {accuracy}')
```

3. LONG SHORT TERM MEMORY(LSTM)

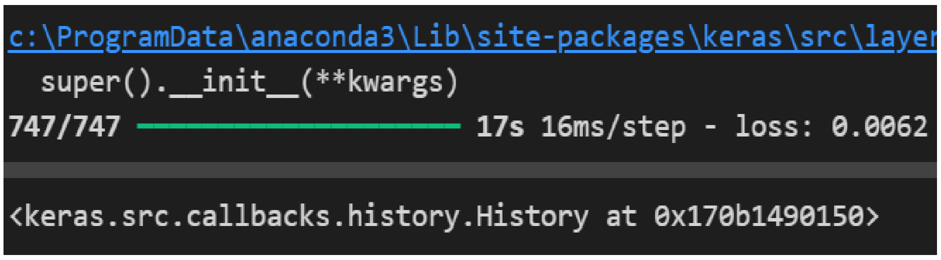
It is a type of (RNN) aimed at dealing with the vanishing gradient problem present in traditional RNNs. Its relative insensitivity to gap length is its advantage over other RNNs.

A common LSTM unit is composed of a cell, an input gate, an output gate and a forget gate . The cell remembers values over arbitrary time intervals and the three *gates* regulate the flow of information into and out of the cell. Forget gates decide what information to discard from a previous state by assigning a previous state, compared to a current input, a value between 0 and 1



```
from sklearn.metrics import mean_squared_error, r2_score

# Calculate the mean squared error and the coefficient of determination
mse = mean_squared_error(df['Close'], y_pred)
r2 = r2_score(df['Close'], y_pred)

# Print the results
print('Mean Squared Error:', mse)
print('Coefficient of Determination:', r2)
```

```
Mean Squared Error: 66907.82299000624
Coefficient of Determination: -689.7047541306664
```



```
c:\ProgramData\anaconda3\Lib\site-packages\keras\src\layer
  super().__init__(**kwargs)
747/747 ━━━━━━━━━━━━━━━━━━━━ 17s 16ms/step - loss: 0.0062

<keras.src.callbacks.history.History at 0x170b1490150>
```

Summary:

We used a total of 3 models in order to achieve our final result.

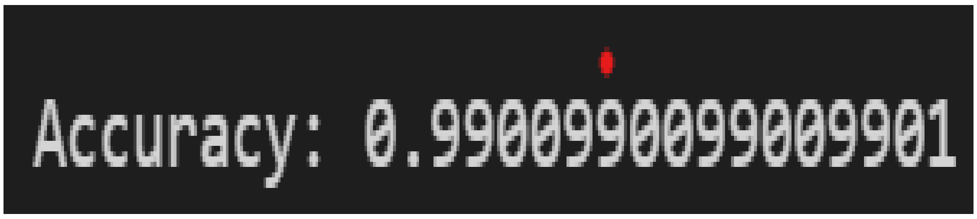
Linear Regression

Support Vector Machine

Long Short Term Memory

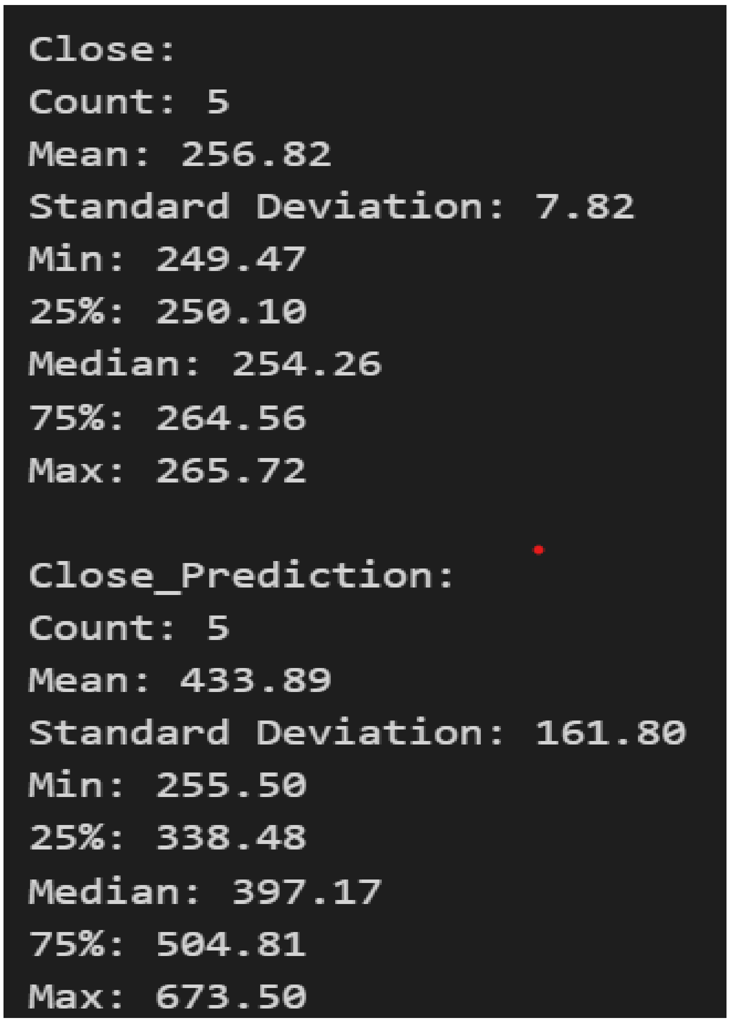
So,the best model for this Dataset is LogisticRegression with 99 % accuracy

Finds the best accuracy out of all the models used :



```
Accuracy: 0.9900990099009901
```

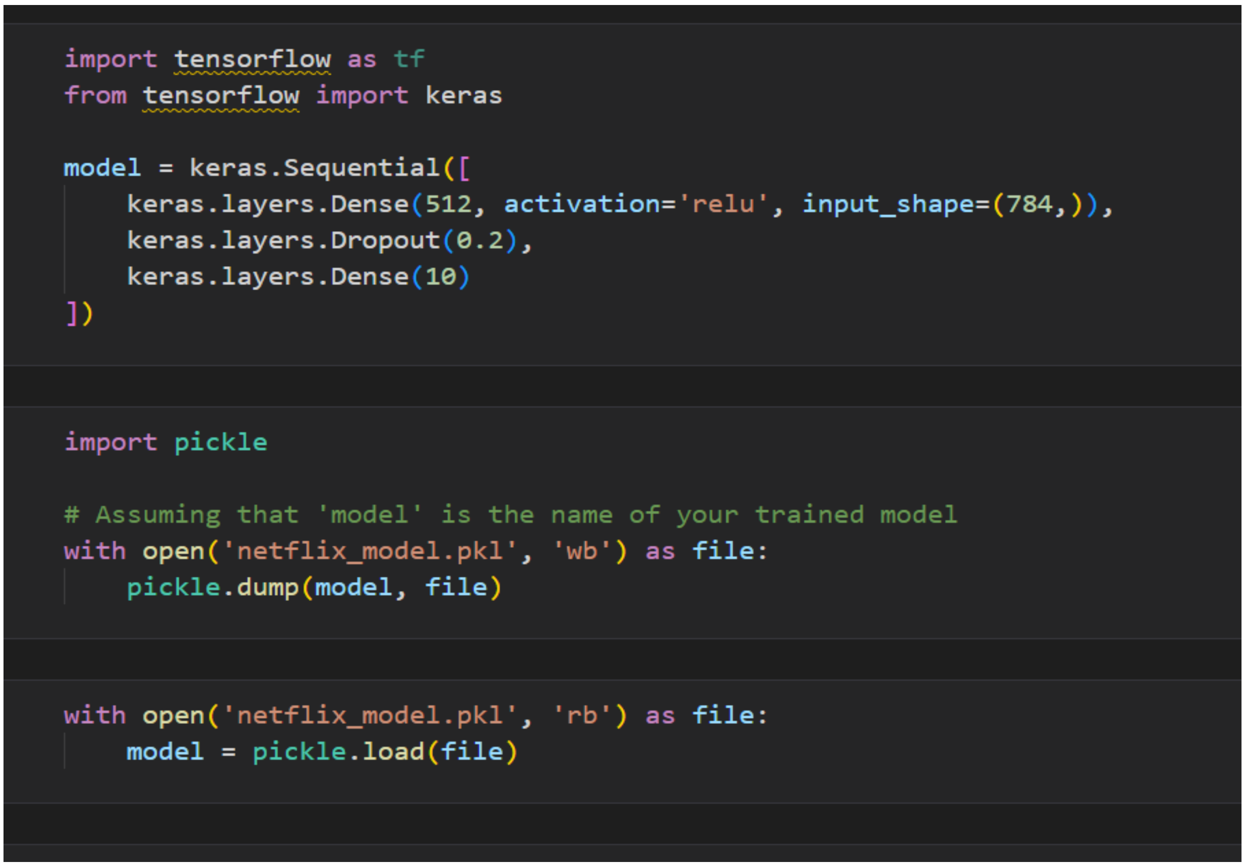
HENCE,WE WILL NOW PREDICT OUR FINAL RESULTS :



```
Close:
Count: 5
Mean: 256.82
Standard Deviation: 7.82
Min: 249.47
25%: 250.10
Median: 254.26
75%: 264.56
Max: 265.72

Close_Prediction:
Count: 5
Mean: 433.89
Standard Deviation: 161.80
Min: 255.50
25%: 338.48
Median: 397.17
75%: 504.81
Max: 673.50
```

WE HAVE USED PICKLE TO STORE OUR DATA

The pickle module is used for storing Python object structures into a file (and retrieving it back into memory at a later time).



```python
import tensorflow as tf
from tensorflow import keras

model = keras.Sequential([
    keras.layers.Dense(512, activation='relu', input_shape=(784,)),
    keras.layers.Dropout(0.2),
    keras.layers.Dense(10)
])
```

```python
import pickle

# Assuming that 'model' is the name of your trained model
with open('netflix_model.pkl', 'wb') as file:
    pickle.dump(model, file)
```

```python
with open('netflix_model.pkl', 'rb') as file:
    model = pickle.load(file)
```

CONCLUSION

In conclusion, the development and evaluation of various machine learning models for Netflix stock price prediction represent a significant milestone in the realm of stock price prediction. Through systematic analysis and experimentation, we have showcased the effectiveness of different algorithms, including linear regression, neural networks, and Support Vector Machines (SVM), in accurately assessing the price of Netflix stock.

Our findings underscore the transformative potential of artificial intelligence and machine learning in proactive healthcare interventions. By harnessing advanced analytics techniques, people can analyse and predict the market price for Netflix stock.

Furthermore, the successful deployment of these recommendation systems highlights the value of data-driven insights in stock price prediction of netflix. With personalized predictions, viewers can explore a diverse range of steps that align with the market price.

Moreover, the successful implementation of these models highlights the importance of interdisciplinary collaboration between data scientists, market ,in addressing complex stock price challenges. By leveraging data-driven insights and predictive analytics, we can usher in a new era of predictions , where accurate prediction play a crucial role in the model.

In essence, this project not only contributes to the burgeoning field of stock price prediction but also underscores the potential of AI and ML technologies to revolutionize the market and improve the results worldwide.

Work Distribution

Preprocessing :- Priyansh Arora, Pranav Mangal, Pratham Khatkar, Palak Bisht

Algorithms:-

LSTM: Priyansh Arora, Pranav Mangal.

Linear Regression: Palak Bisht, Pratham Khatkar.

REFRENCES

Tutorials and Guides:

- Stock market Working: https://www.investopedia.com/terms/s/stockmarket.asp
- Dataset: https://www.kaggle.com/datasets/jainilcoder/netflix-stock-price-prediction/data
- Pickle: https://www.geeksforgeeks.org/how-to-use-pickle-to-save-and-load-variables-in-python/