Green University of Bangladesh

Department of Computer Science and Engineering (CSE)
Semester: (Spring, Year: 2023), B.Sc. in CSE (Day)

Stock Price Prediction

Course Title: Artificial Intelligence Lab
Course Code: CSE-316
Section: 203-D1

Students Details

Name	**ID**
Md. Montasir Rahman	202002003
Joy Pal	201002418

Submission Date: 26/06/2023
Course Teacher's Name: Md. Moshiur Rahman

[For teachers use only: Don't write anything inside this box]

Lab Project Status	
Marks:	**Signature:**
Comments:	**Date:**

Contents

Chapter 1

Introduction

1.1 Overview

A stock market prediction project aims to predict stock market trends using historical data and machine learning algorithms. The project involves collecting and preprocessing historical stock market data, choosing a suitable algorithm, training the model, testing the model, deploying it in a production environment, and monitoring its performance over time. The goal of the project is to accurately predict future stock market trends to help investors make informed decisions. The created prediction models can be used to forecast future stock values using either historical or real-time data. These forecasts can be used by traders and investors to spot possible market movements, gauge risk exposure, and decide whether to buy, sell, or hold stocks. By automatically executing transactions based on established rules and anticipated price movements, the models can also enable algorithmic trading systems.

1.2 Motivation

- To help investors make informed decisions by predicting future stock market trends.
- By predicting the stock market trends, investors can make better decisions on when to buy or sell stocks, which can lead to better returns on their investment.
- It can also help financial institutions and companies make strategic decisions based on future stock market trends.
- AI-based stock market prediction can assist in managing risks associated with investing.
- It advances academic study and innovation in the fields of finance and machine learning to create AI stock market prediction models.

1.3 Problem Definition

1.3.1 Problem Statement

- Create a machine learning model that is accurate and trustworthy for forecasting stock market prices.
- Utilize historical stock market data to train the prediction model, including data on price, volume, and other pertinent variables.
- Address the challenge of predicting stock market prices, which are influenced by a multitude of factors, including market trends, economic indicators, news sentiment, and company-specific information.
- To address the challenge of accurately predicting future stock market trends.

1.3.2 Complex Engineering Problem

Table 1.1: Summary of the attributes touched by the mentioned projects

Name of the P Attributess	**Explain how to address**
P1: Depth of knowledge required	Trading Mechanism, Statistical Concepts, Proficiency in Algorithms, Preprocessing Techniques
P2: Range of conflicting requirements	Accuracy vs. interpretability, Scalability vs. real-time processing, Complexity vs. simplicity
P3: Depth of analysis required	—-
P4: Familiarity of issues	Data Quality, Risk Management, Features selection, and model evaluation.
P5: Extent of applicable codes	—-
P6: Extent of stakeholder involvement and conflicting requirements	—-
P7: Interdependence	Preprocessing, Data collection, Model training, optimization

1.4 Design Goals/Objectives

- To learn the various algorithm to detect the accurate prediction
- To develop an AI model to predict future stock market trends accurately.
- Improve decision-making.
- Analyze complex data.
- Increase efficiency.
- Reduce financial risk.

1.5 Application [1]

- Trading and Investment: Traders and investors can use machine learning-based stock market prediction models to make informed decisions on buying, selling, or holding stocks. These models can provide insights into potential price movements, identify investment opportunities, and assist in optimizing trading strategies.
- Market Analysis and Research: Financial analysts, researchers, and organizations can utilize stock market prediction models to gain insight into market dynamics, examine historical trends, and comprehend the effects of various events on stock prices. This data can be used to create research reports, assist investing strategies, and advance market understanding.
- Financial Forecasting and Planning: Stock market prediction models can aid in the forecasting and planning of a person's finances as well as those of their companies and institutions. These models help inform judgments about investing strategies, budgeting, and financial goal-setting by offering forecasts of future stock values.

Chapter 2

Design/Development/Implementation of the Project

2.1 Introduction

The project focuses on utilizing machine learning techniques to predict stock market prices, offering a valuable tool for traders, investors, and financial institutions. By analyzing historical market data, incorporating relevant indicators, and employing advanced algorithms, the aim is to develop accurate and reliable prediction models that can assist in making informed investment decisions.

2.2 Project Details

Importing Python libraries make it very easy for us to handle the data and perform typical and complex tasks with a single line of code. The dataset we will use here to perform the analysis and build a predictive model is British American Tobacco Price data. We will use OHLC('Open', 'High', 'Low', 'Close') data from 1997 to 2023. we got to know that there are 1692 rows of data available and for each row, we have 7 columns. Then we use EDA, which is an approach to analyzing the data using visual techniques. It is used to discover trends, and patterns, or to check assumptions with the help of statistical summaries and graphical representations. While performing the EDA of that dataset, we will analyze how prices of the stock have moved over the period of time and how the end of the quarters affects the prices of the stock. Then we check for the null values if any are present in the data frame. Then we apply the distribution plot and box plot method of OHLC data. Then we apply different types of Algorithms to analyze the dataset.

2.3 Implementation

2.3.1 Importing Libraries

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sb

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from xgboost import XGBClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn import metrics
from sklearn.metrics import ConfusionMatrixDisplay,
precision_score, recall_score, f1_score

import warnings
warnings.filterwarnings('ignore')
```

2.3.2 Importing Dataset

```
df = pd.read_csv('/content/BAT.csv')
df.head()
```

2.3.3 Exploratory Data Analysis

```
plt.figure(figsize=(15,5))
plt.plot(df[' Close'])
plt.title('BAT Close Price.', fontsize=15)
plt.ylabel('Price in Taka.')
plt.show()
```

2.3.4 Feature Engineering

```
splitted = df['Date'].str.split('/', expand=True)

df['day'] = splitted[1].astype('int')
df['month'] = splitted[0].astype('int')
df['year'] = splitted[2].astype('int')
```

```
df.head()

df['is_quarter_end'] = np.where(df['month']%3==0,1,0)
df.head()

data_grouped = df.groupby('year').mean()
plt.subplots(figsize=(20,10))

for i, col in enumerate([' Open', ' High', ' Low', ' Close']):
  plt.subplot(2,2,i+1)
  data_grouped[col].plot.bar()
plt.show()

df.groupby('is_quarter_end').mean()

df['open-close'] = df[' Open'] - df[' Close']
df['low-high'] = df[' Low'] - df[' High']
df['target'] = np.where(df[' Close'].shift(-1) > df[' Close'], 1, 0)

plt.pie(df['target'].value_counts().values,
        labels=[0, 1], autopct='%1.1f%%')
plt.show()

plt.figure(figsize=(10, 10))

sb.heatmap(df.corr() > 0.9, annot=True, cbar=False)
plt.show()
```

2.3.5 Data Splitting and Normalization

```
features = df[['open-close', 'low-high', 'is_quarter_end']]
target = df['target']

scaler = StandardScaler()
features = scaler.fit_transform(features)

X_train, X_valid, Y_train, Y_valid = train_test_split(
    features, target, test_size=0.1, random_state=2022)
print(X_train.shape, X_valid.shape)
```

2.3.6 Model Development and Evaluation

```
models = [LogisticRegression(), SVC(kernel='poly', probability=True),
XGBClassifier(), RandomForestClassifier(n_estimators=10, criterion="entropy")]

for i in range(4):
    models[i].fit(X_train, Y_train)
    print(f'{models[i]}:')
    Y_train_pred = models[i].predict(X_train)
    Y_valid_pred = models[i].predict(X_valid)

    ConfusionMatrixDisplay.from_estimator(models[0], X_valid, Y_valid)
plt.show()
```

2.3.7 F-Measure, Precision and Recall

```
train_precision = precision_score(Y_train, Y_train_pred)
    train_recall = recall_score(Y_train, Y_train_pred)
    train_f_measure = f1_score(Y_train, Y_train_pred)

    valid_precision = precision_score(Y_valid, Y_valid_pred)
    valid_recall = recall_score(Y_valid, Y_valid_pred)
    valid_f_measure = f1_score(Y_valid, Y_valid_pred)

    print('Training Precision:', train_precision)
    print('Training Recall:', train_recall)
    print('Training F-measure:', train_f_measure)
    print('Validation Precision:', valid_precision)
    print('Validation Recall:', valid_recall)
    print('Validation F-measure:', valid_f_measure)
    print()
```

2.4 Algorithms

2.4.1 Logistic Regression

- Step 1: Initialize the parameters
- Step 2: Define the sigmoid function
- Step 3: Define the cost function
- Step 4: Gradient Descent
- Step 5: Prediction

2.4.2 Support Vector Machine (SVM)

- Step 1: Load the training data
- Step 2: Preprocess the data (if required)
- Step 3: Define the SVM model
- Step 4: Train the SVM model
- Step 5: Make predictions
- Step 6: Evaluate the model

2.4.3 XGBoost

- Step 1: Load the training data
- Step 2: Preprocess the data (if required)
- Step 3: Define the XGBoost model
- Step 4: Train the XGBoost model
- Step 5: Make predictions
- Step 6: Evaluate the model

2.4.4 Random Forest

- Step 1: Load the training data
- Step 2: Preprocess the data (if required)
- Step 3: Define the Random Forest model
- Step 4: Train the Random Forest model
- Step 5: Make predictions
- Step 6: Evaluate the model

Chapter 3

Performance Evaluation

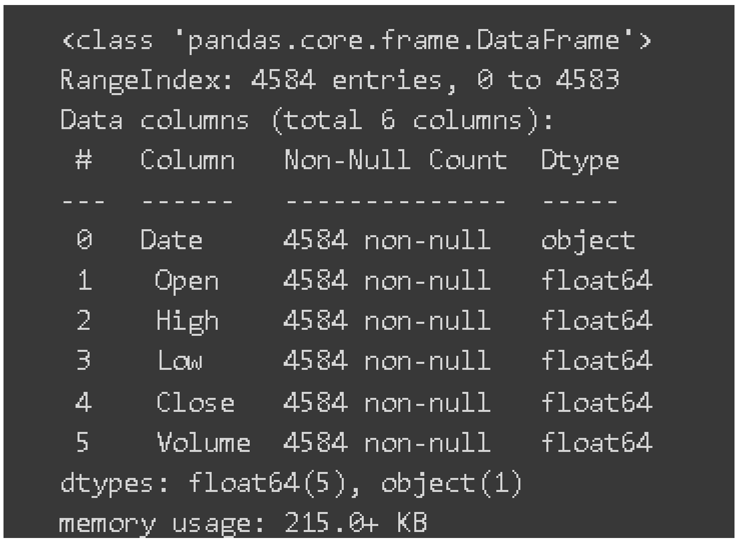
3.1 Simulation Environment/ Simulation Procedure

We have used Google Colaboratory to simulate our program.

Figure 3.1: Figure name

3.2 Results Analysis/Testing

3.2.1 df info



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4584 entries, 0 to 4583
Data columns (total 6 columns):
 #   Column   Non-Null Count  Dtype
---  ------   --------------  -----
 0   Date     4584 non-null   object
 1    Open    4584 non-null   float64
 2    High    4584 non-null   float64
 3    Low     4584 non-null   float64
 4    Close   4584 non-null   float64
 5    Volume  4584 non-null   float64
dtypes: float64(5), object(1)
memory usage: 215.0+ KB
```

Figure 3.2: df info output

3.2.2 BAT Close Price Chart

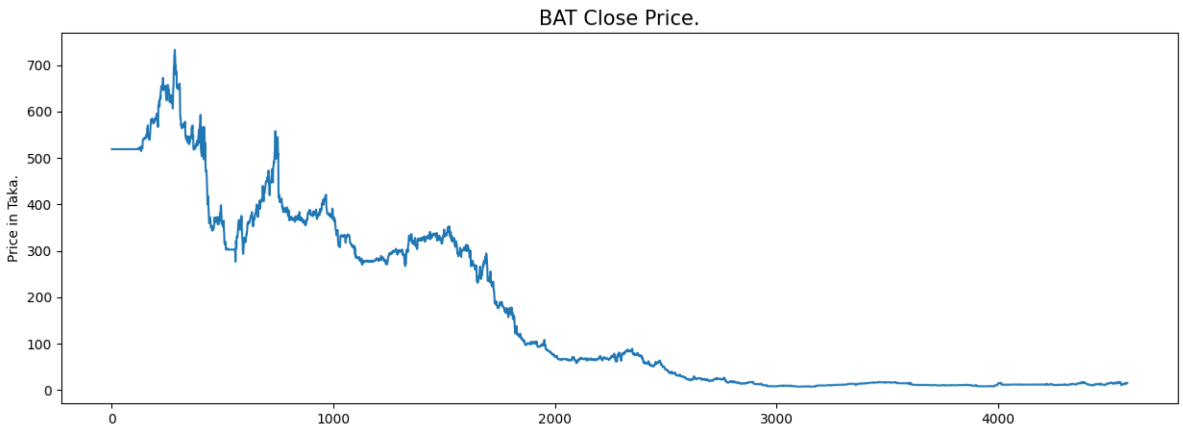


Figure 3.3: BAT Close Price Chart

3.2.3 Distribution Plot

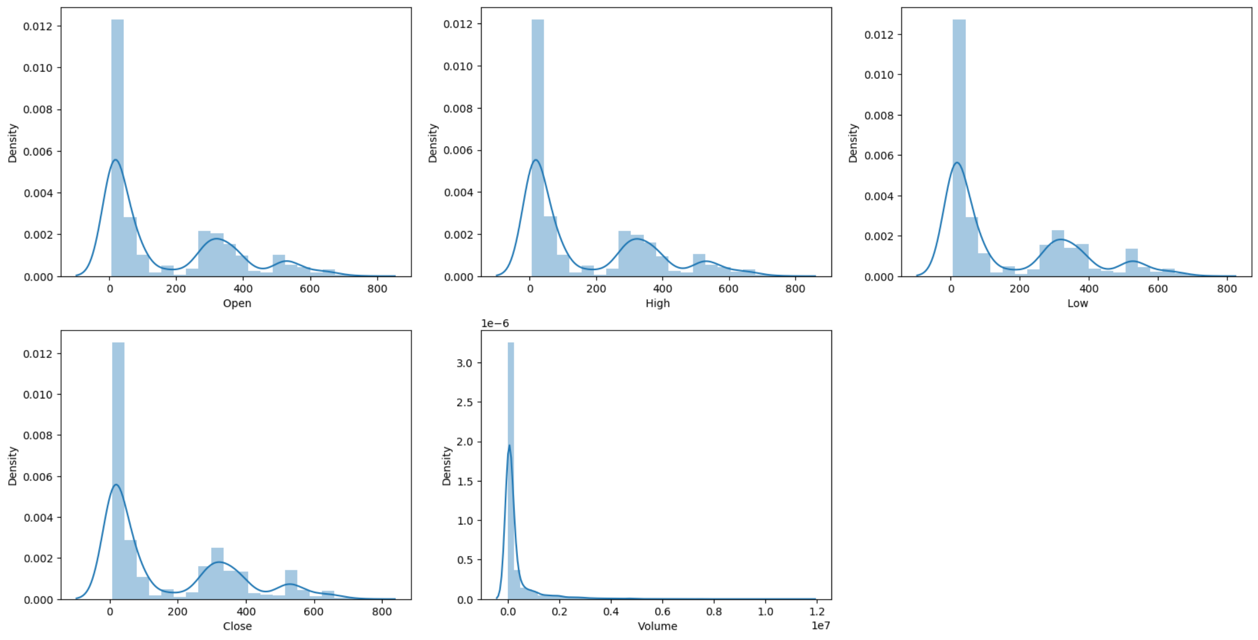


Figure 3.4: Distribution Plot of the Continuous Variable

3.2.4 Box Plot

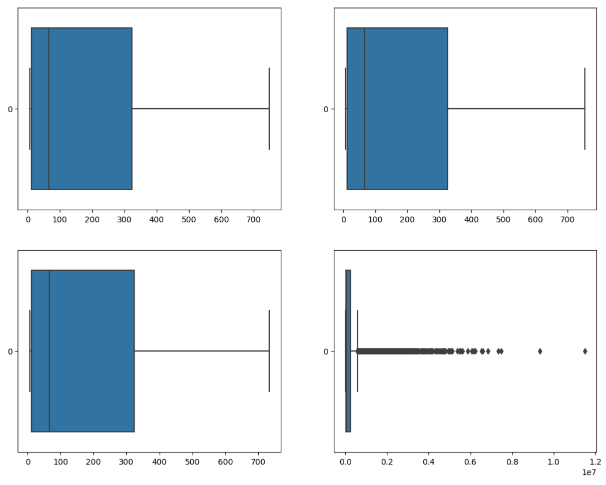


Figure 3.5: Box Plot of the Continuous Variable

3.2.5 Bar Graph

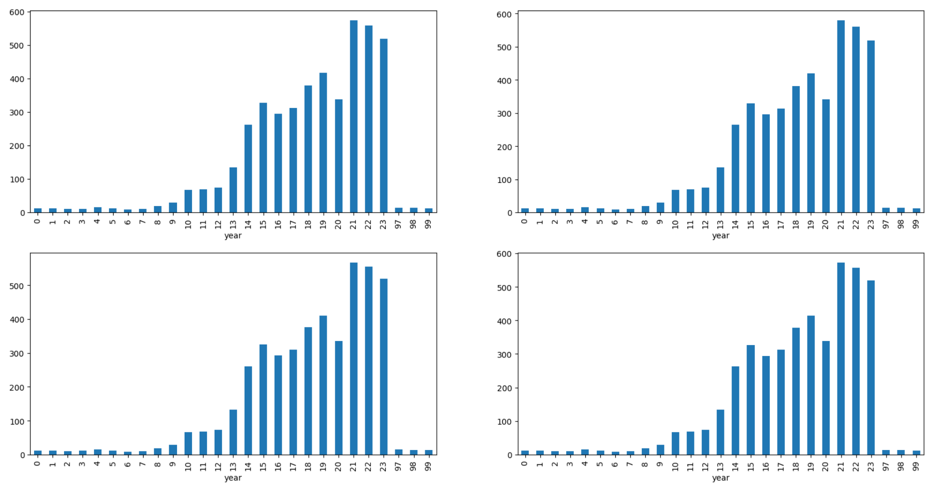


Figure 3.6: Bar Graph of OHLC

3.2.6 Pie Chart

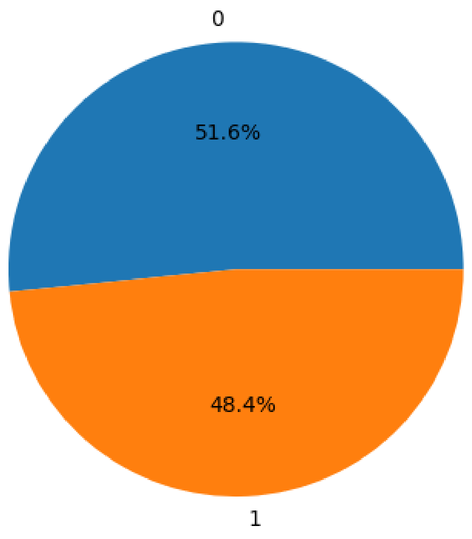
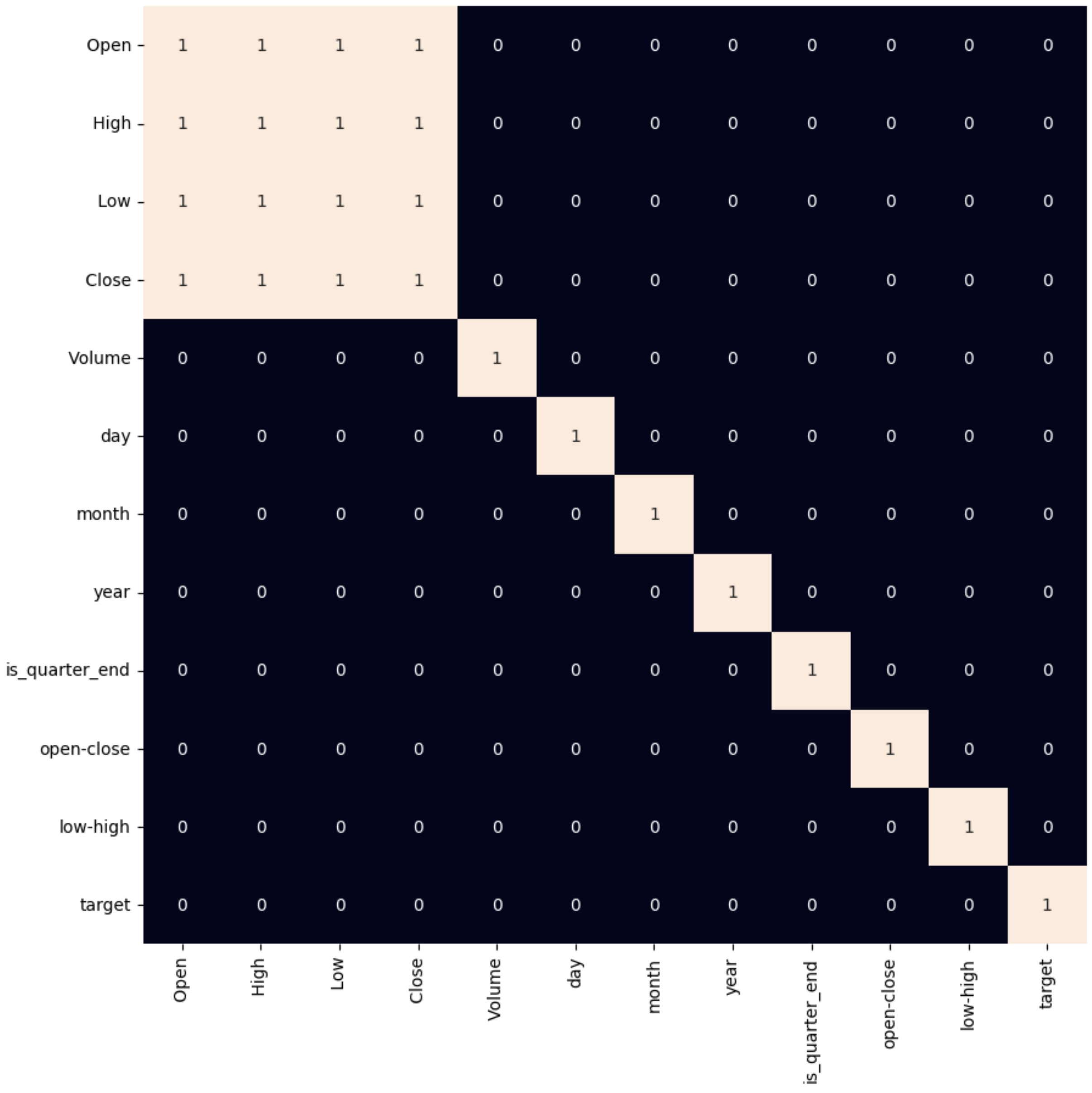


Figure 3.7: Bar Graph of OHLC

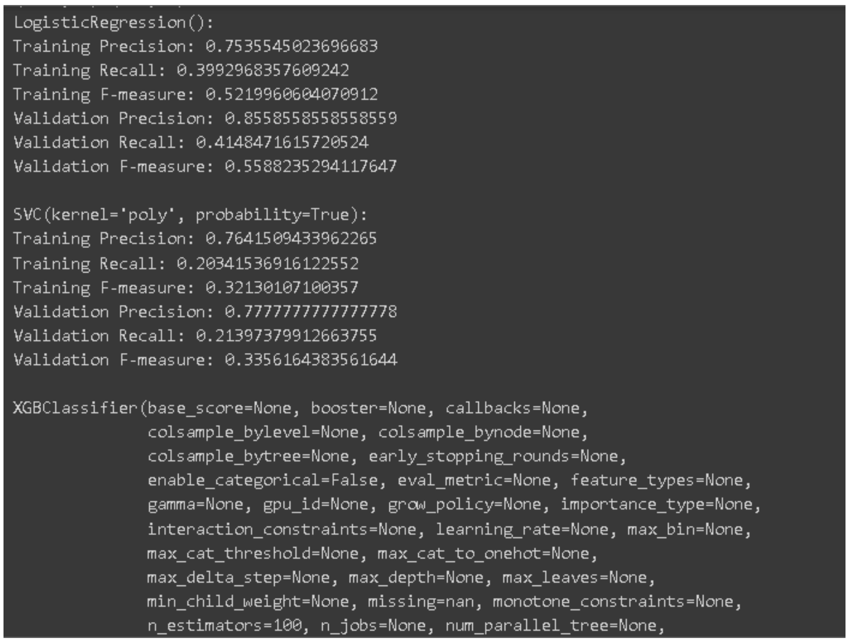
3.2.7 Heatmap



	Open	High	Low	Close	Volume	day	month	year	is_quarter_end	open-close	low-high	target
Open	1	1	1	1	0	0	0	0	0	0	0	0
High	1	1	1	1	0	0	0	0	0	0	0	0
Low	1	1	1	1	0	0	0	0	0	0	0	0
Close	1	1	1	1	0	0	0	0	0	0	0	0
Volume	0	0	0	0	1	0	0	0	0	0	0	0
day	0	0	0	0	0	1	0	0	0	0	0	0
month	0	0	0	0	0	0	1	0	0	0	0	0
year	0	0	0	0	0	0	0	1	0	0	0	0
is_quarter_end	0	0	0	0	0	0	0	0	1	0	0	0
open-close	0	0	0	0	0	0	0	0	0	1	0	0
low-high	0	0	0	0	0	0	0	0	0	0	1	0
target	0	0	0	0	0	0	0	0	0	0	0	1

Figure 3.8: Heatmap of the correlation between the features

3.2.8 Evaluation of the Model 1



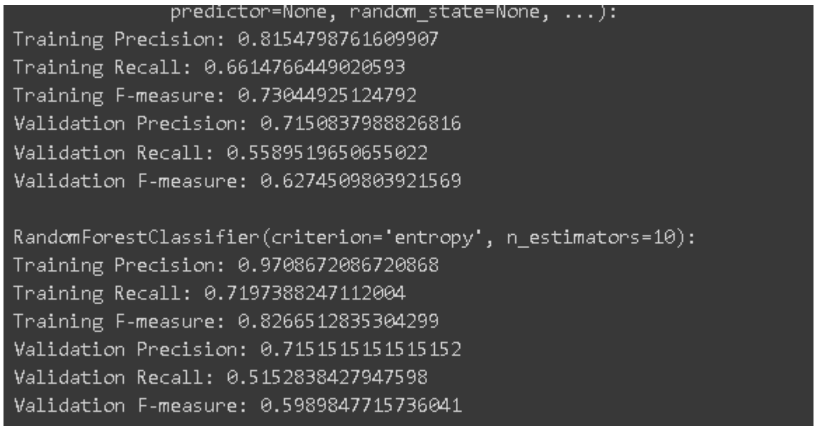
```
LogisticRegression():
Training Precision: 0.7535545023696683
Training Recall: 0.3992968357609242
Training F-measure: 0.5219960604070912
Validation Precision: 0.8558558558558559
Validation Recall: 0.4148471615720524
Validation F-measure: 0.5588235294117647

SVC(kernel='poly', probability=True):
Training Precision: 0.7641509433962265
Training Recall: 0.20341536916122552
Training F-measure: 0.32130107100357
Validation Precision: 0.7777777777777778
Validation Recall: 0.21397379912663755
Validation F-measure: 0.3356164383561644

XGBClassifier(base_score=None, booster=None, callbacks=None,
              colsample_bylevel=None, colsample_bynode=None,
              colsample_bytree=None, early_stopping_rounds=None,
              enable_categorical=False, eval_metric=None, feature_types=None,
              gamma=None, gpu_id=None, grow_policy=None, importance_type=None,
              interaction_constraints=None, learning_rate=None, max_bin=None,
              max_cat_threshold=None, max_cat_to_onehot=None,
              max_delta_step=None, max_depth=None, max_leaves=None,
              min_child_weight=None, missing=nan, monotone_constraints=None,
              n_estimators=100, n_jobs=None, num_parallel_tree=None,
```

Figure 3.9: Heatmap of the correlation between the features

3.2.9 Evaluation of the Model 2



```
              predictor=None, random_state=None, ...):
Training Precision: 0.8154798761609907
Training Recall: 0.6614766449020593
Training F-measure: 0.73044925124792
Validation Precision: 0.7150837988826816
Validation Recall: 0.5589519650655022
Validation F-measure: 0.6274509803921569

RandomForestClassifier(criterion='entropy', n_estimators=10):
Training Precision: 0.9708672086720868
Training Recall: 0.7197388247112004
Training F-measure: 0.8266512835304299
Validation Precision: 0.7151515151515152
Validation Recall: 0.5152838427947598
Validation F-measure: 0.5989847715736041
```

Figure 3.10: Heatmap of the correlation between the features

3.2.10 Confusion Matrix

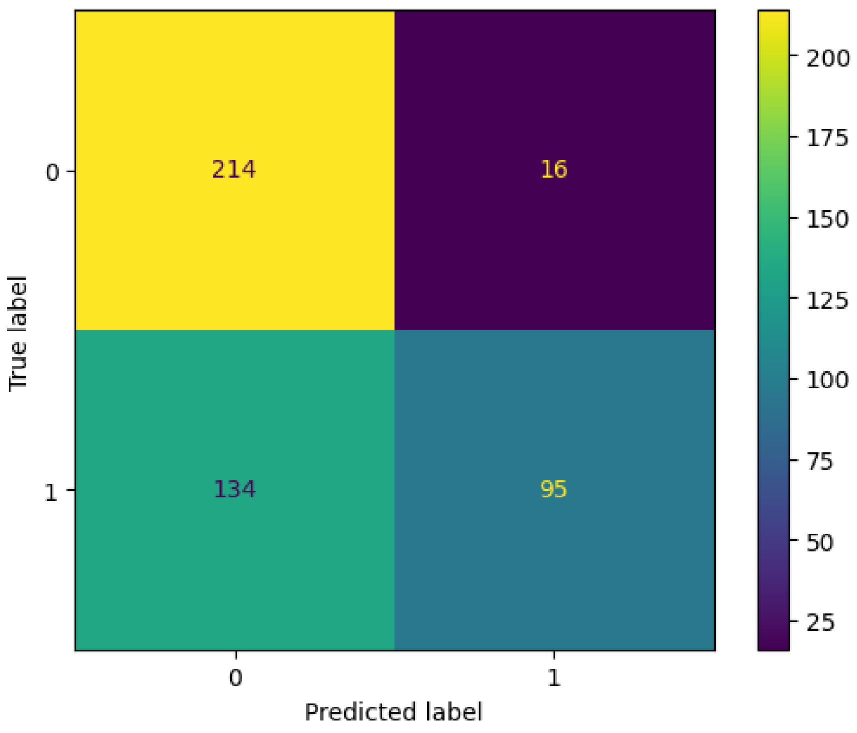


Figure 3.11: Heatmap of the correlation between the features

Chapter 4

Conclusion

4.1 Discussion

In this project, we tried to predict the stock price of BAT by analyzing its previous data sets and used several machine learning algorithms to predict whether the stock should be bought or not by the customers.

4.2 Limitations

- The project uses a limited set of features, namely ’open-close’, ’low-high’, and ’is quarter end’, to predict stock prices. However, there might be other relevant features, such as technical indicators, market sentiment, or external factors, that could improve the predictive power of the models.
- The project calculates evaluation metrics such as precision, recall, and F-measure for both training and validation sets. While these metrics provide insights into model performance, it’s also crucial to consider additional metrics like accuracy, ROC curve, or mean absolute error (MAE) to have a comprehensive understanding of the model’s predictive ability.
- The code fits multiple models to the data without performing proper model selection or assessing overfitting. It’s important to evaluate the models on unseen data, perform cross-validation, or consider more advanced techniques like ensemble learning to mitigate overfitting and improve generalization.

4.3 Scope of Future Work

- We can explore additional features or indicators that could capture relevant information about the stock price movements. This could include technical indicators, market sentiment data, news sentiment analysis, or fundamental factors. Experimenting with different combinations of features may improve the predictive power of the models.
- We can consider implementing more advanced machine learning models or algorithms specifically designed for time series analysis. Examples include recurrent neural networks (RNNs), long short-term memory (LSTM) networks, or convolutional neural networks (CNNs). These models can capture temporal dependencies and patterns in the data, which may lead to better predictions.
- We could incorporate time series analysis techniques such as autoregressive integrated moving average (ARIMA) models or Prophet forecasting models. These models are specifically designed for time series data and can capture seasonality, trends, and other temporal patterns.

References

[1] Omid C Farokhzad and Robert Langer. Impact of nanotechnology on drug delivery. *ACS nano*, 3(1):16–20, 2009.