Stock Price Prediction

Phase 2 Submission Document

INTRODUCTION:

- ✓ In recent years, the financial industry has witnessed a surge in the adoption of artificial intelligence and deep learning techniques to enhance decision-making processes, particularly in the domain of stock price prediction.

- ✓ Deep learning, a subset of machine learning, has proven to be exceptionally effective in handling complex and non-linear data patterns, making it a promising approach for forecasting stock prices.

- ✓ This research aims to explore the application of deep learning techniques, particularly recurrent neural networks (RNNs) and convolutional neural networks (CNNs), in predicting stock prices.

- ✓ RNNs excel at capturing sequential dependencies in time-series data, which is crucial in modeling stock price movements over time.

CNNs, with their ability to extract hierarchical features, can be employed to analyse various technical indicators and sentiment data.

- ✓ Briefly introduce the real estate market and the importance of accurate Stock price prediction.

- ✓ Emphasize the need for more advanced deep learning techniques like CNN-LSTM or attention mechanisms for improved accuracy in predicting stock prices.

Content for Project Phase 2 :

Consider exploring more advanced deep learning techniques like CNN-LSTM or attention mechanisms for improved accuracy in predicting stock prices.

Data Source:

A good data source for stock price prediction using deep learning should be Accurate, Complete, Covering the geographic area of interest, Accessible.

Dataset Link: https://www.kaggle.com/datasets/prasoonkottarathil/microsoft-lifetime-stocks-dataset

Date	Open	High	Low	Close	Adj Close	Volume
13/03/1986	0.08854	0.10156	0.08854	0.09722	0.06255	1031788800
14/03/1986	0.09722	0.10243	0.09722	0.10069	0.06478	308160000
17/03/1986	0.10069	0.1033	0.10069	0.10243	0.0659	133171200
18/03/1986	0.10243	0.1033	0.09896	0.09983	0.06422	67766400
19/03/1986	0.09983	0.10069	0.09722	0.09809	0.06311	47894400
20/03/1986	0.09809	0.09809	0.09462	0.09549	0.06143	58435200
21/03/1986	0.09549	0.09722	0.09115	0.09288	0.05976	59990400
24/03/1986	0.09288	0.09288	0.08941	0.09028	0.05808	65289600
25/03/1986	0.09028	0.09201	0.08941	0.09201	0.0592	32083200
26/03/1986	0.09201	0.09549	0.09115	0.09462	0.06087	22752000
27/03/1986	0.09462	0.09635	0.09462	0.09635	0.06199	16848000
31/03/1986	0.09635	0.09635	0.09375	0.09549	0.06143	12873600
01/04/1986	0.09549	0.09549	0.09462	0.09462	0.06087	11088000
02/04/1986	0.09462	0.09722	0.09462	0.09549	0.06143	27014400
03/04/1986	0.09635	0.09896	0.09635	0.09635	0.06199	23040000
04/04/1986	0.09635	0.09722	0.09635	0.09635	0.06199	26582400
07/04/1986	0.09635	0.09722	0.09288	0.09462	0.06087	16560000
08/04/1986	0.09462	0.09722	0.09462	0.09549	0.06143	10252800
09/04/1986	0.09549	0.09809	0.09549	0.09722	0.06255	12153600
10/04/1986	0.09722	0.09896	0.09549	0.09809	0.06311	13881600
11/04/1986	0.09896	0.10156	0.09896	0.09983	0.06422	17222400
14/04/1986	0.09983	0.10156	0.09983	0.10069	0.06478	12153600
15/04/1986	0.10069	0.10069	0.09722	0.10069	0.06478	9302400

Data Collection and Preprocessing:

- ✓ Data Preparation: Gathering historical stock price data, economic indicators, news sentiment scores, and other relevant features for model training and testing.
- ✓ Model Architecture: Designing and implementing deep learning models, including RNNs and CNNs, to learn and predict stock price movements.

Exploratory Data Analysis (EDA):

- ✓ EDA provides valuable insights into the data, and the visualizations help to communicate these insights effectively.
- ✓ By performing EDA, we can identify trends, patterns, and relationships that may not be immediately apparent from the data.
- ✓ This knowledge can then be used to inform further analysis and decision-making.

Feature Engineering:

- ✓ Create new features or transform existing ones to capture valuable information.
- ✓ Utilize domain knowledge to engineer features that may impact stock prices, such as proximity to schools, transportation, or crime rates.
- ✓ Explain the process of creating new features or transforming existing ones.
- ✓ Showcase domain-specific feature engineering, such as proximity scores or composite indicators.
- ✓ Emphasize the impact of engineered features on model performance.

Advanced Regression Techniques:

1) **Convolutional Neural Networks (CNNs):** Can take technical indicators' 2-D images as inputs and predict the stock price.

2) **Long Short Term Memory Networks (LSTMs):** Models are extremely powerful time-series models. They can predict an arbitrary number of steps into the future.

3) **Recurrent Neural Networks (RNNs):** is used on time-series data ofthe stocks.

4) **Generative Adversarial Networks (GANs):** Used for image andvideo generation, and even for style transfer.

5) **Radial Basis Function Networks (RBFNs):** A particular type ofArtificial Neural Network used for function approximation problems.

6) **Self-Supervised learning:** Learning from unlabelled data using pretext tasks, e.g., contrastive learning.

Model Evaluation and Selection:

- ✓ Split the dataset into training and testing sets.
- ✓ Evaluate models using appropriate metrics
- ✓ Use cross-validation techniques to tune hyperparameters and ensure model stability.
- ✓ Compare the results with traditional Convolutional Neural Networks to highlight improvements.
- ✓ Select the best-performing model for further analysis.

Model Interpretability:

- ✓ Explain how to interpret feature importance from attention mechanisms for improved accuracy in predicting stock prices.

- ✓ Discuss the insights gained from feature importance analysis and their relevance to stock price prediction.
- ✓ Interpret feature importance from ensemble models advanced deep learning techniques like CNN-LSTM or attention mechanisms for improved accuracy in predicting stock prices.

.Deployment and Prediction:

- ✓ Deploy the chosen regression model to predict house prices.
- ✓ Develop a user-friendly interface for users to input property features and receive price predictions.

Program:

Stock Price Prediction

```
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 import metrics

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

In[1]:

```
import numpy as np

import matplotlib.pyplot as plt

import pandas as pd

dataset_train = pd.read_csv('D:\data\msft.csv')

print('shape is = {}'.format(dataset_train.shape))

print(dataset_train.head())
```

Out[1]:

```
shape is = (8525, 7)
        Date      Open      High       Low     Close  Adj Close      Volume
0  3/13/1986  0.088542  0.101563  0.088542  0.097222   0.062549  1031788800
1  3/14/1986  0.097222  0.102431  0.097222  0.100694   0.064783   308160000
2  3/17/1986  0.100694  0.103299  0.100694  0.102431   0.065899   133171200
3  3/18/1986  0.102431  0.103299  0.098958  0.099826   0.064224    67766400
4  3/19/1986  0.099826  0.100694  0.097222  0.098090   0.063107    47894400
```

In[2]:

```
Print(df.shape)
```

Out[2]:

```
(8525, 7)
```

In[3]:

```
Print(df.describe())
```

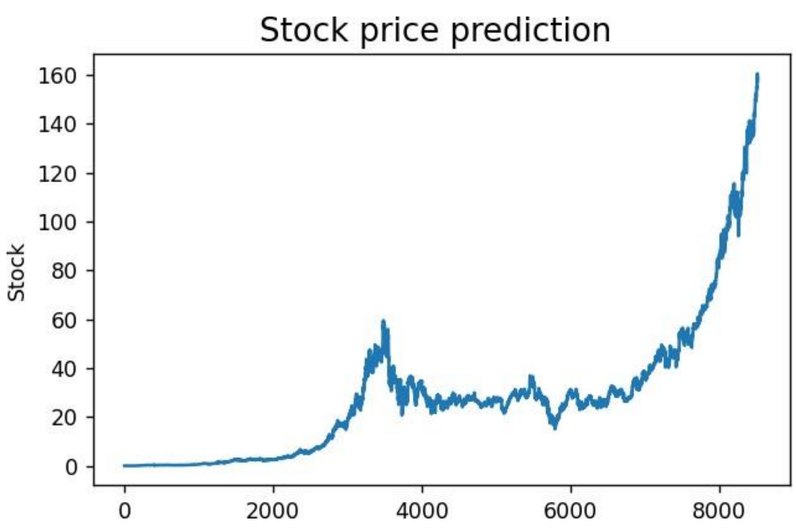
Out[3]:

```
              Open         High  ...    Adj Close        Volume
count  8525.000000  8525.000000  ...  8525.000000  8.525000e+03
mean     28.220247    28.514473  ...    23.417934  6.045692e+07
std      28.626752    28.848988  ...    28.195330  3.891225e+07
min       0.088542     0.092014  ...     0.058081  2.304000e+06
25%       3.414063     3.460938  ...     2.196463  3.667960e+07
50%      26.174999    26.500000  ...    18.441576  5.370240e+07
75%      34.230000    34.669998  ...    25.392508  7.412350e+07
max     159.449997   160.729996  ...   160.619995  1.031789e+09

[8 rows x 6 columns]
```

In[4]:

```
df = pd.read_csv('D:\data\msft.csv')
print(df.head())
plt.figure(figsize=(15,5))
plt.plot(df['Close'])
plt.title('Stock price prediction ', fontsize=15)
plt.ylabel('Stock')
plt.show()
```

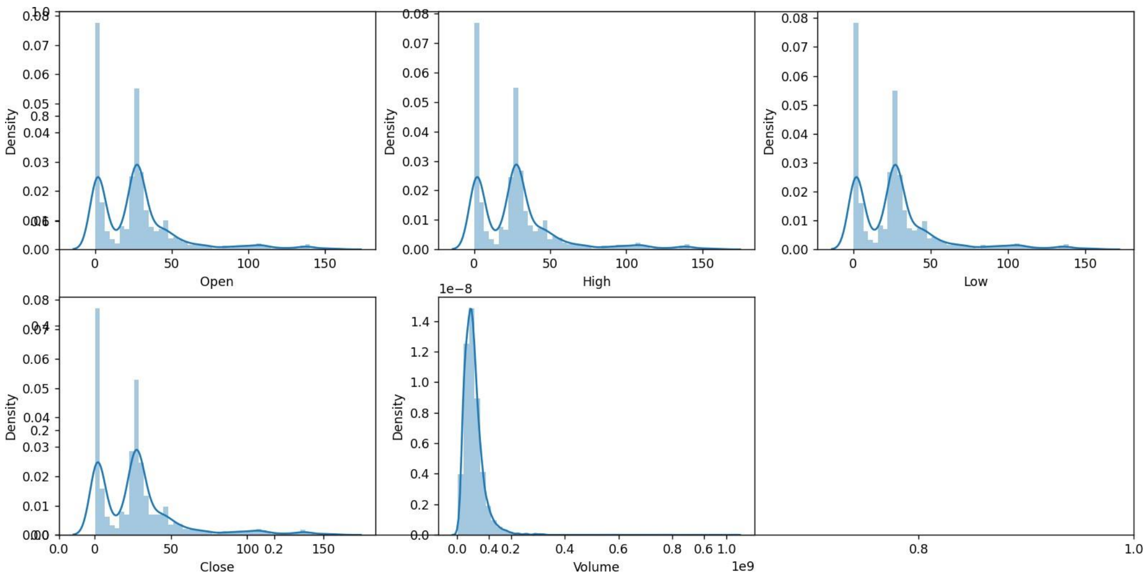
Out[4]:



In[5]:

```
df = pd.read_csv('D:\data\msft.csv')
print(df.head())
features = ['Open', 'High', 'Low', 'Close', 'Volume']
plt.subplots(figsize=(20,10))
for i, col in enumerate(features):
      plt.subplot(2,3,i+1)
      sb.distplot(df[col])
plt.show()
```

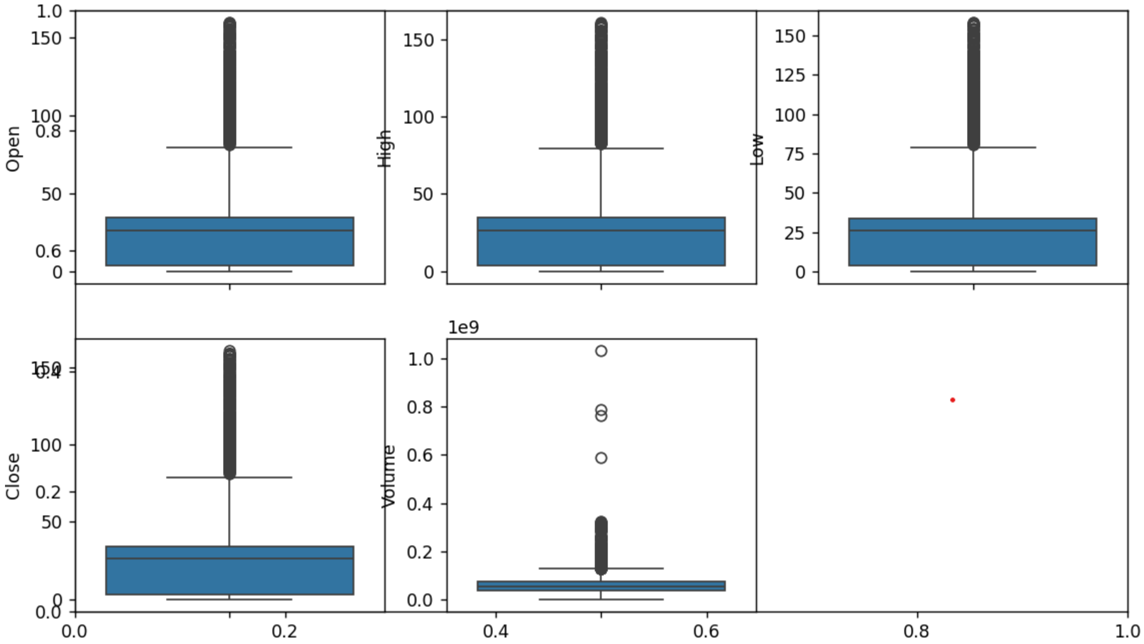
Out[5]:



In[6]:

```
df = pd.read_csv('D:\data\msft.csv')
df.head()
features = ['Open', 'High', 'Low', 'Close', 'Volume']
plt.subplots(figsize=(20,10))
for i, col in enumerate(features):
   plt.subplot(2,3,i+1)
   sb.boxplot(df[col])
plt.show()
```

Out[6]:



In[7]:

```
df = pd.read_csv('D:\data\msft.csv')
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')
print(df.head())
```

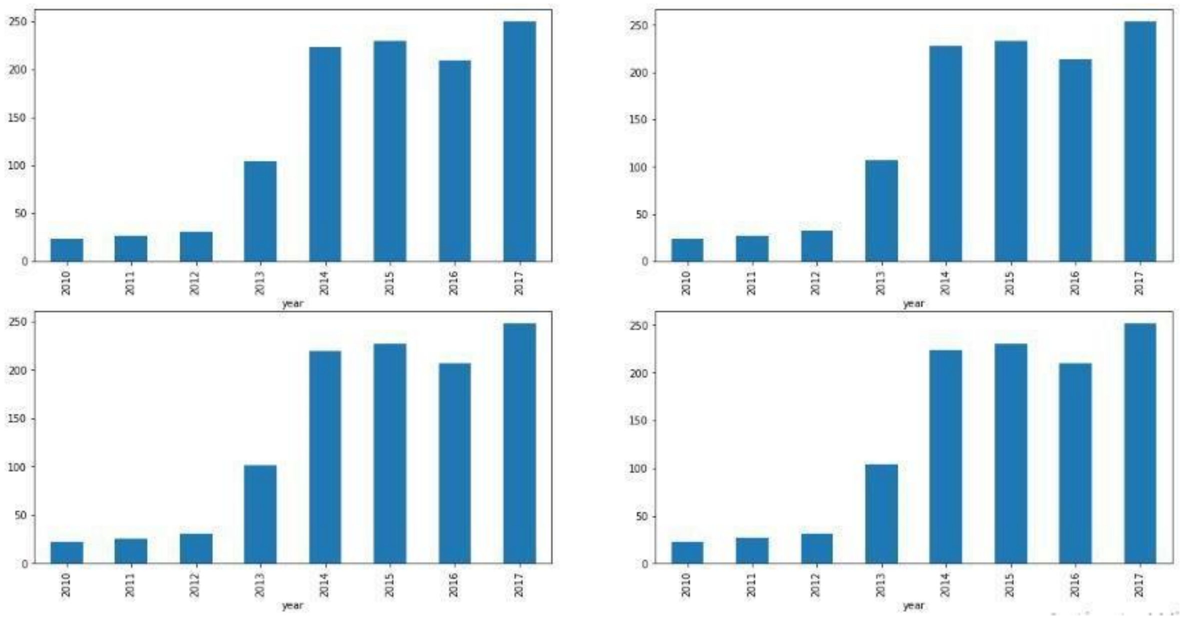
Out[7]:

```
C:\Users\JeevamathiSrinivasan\PycharmProjects\pythonProject\venv\Scripts\python.e
        Date      Open      High       Low  ...      Volume  day  month  year
0  3/13/1986  0.088542  0.101563  0.088542  ...  1031788800   13      3  1986
1  3/14/1986  0.097222  0.102431  0.097222  ...   308160000   14      3  1986
2  3/17/1986  0.100694  0.103299  0.100694  ...   133171200   17      3  1986
3  3/18/1986  0.102431  0.103299  0.098958  ...    67766400   18      3  1986
4  3/19/1986  0.099826  0.100694  0.097222  ...    47894400   19      3  1986

[5 rows x 10 columns]
```

In[8]:

```
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()
```

Out[8]:



In[9]:

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

df.head()
```

Out[9]:

	Date	Open	High	Low	Close	Volume	day	month	year	is_quarter_end
0	6/29/2010	19.000000	25.00	17.540001	23.889999	18766300	29	6	2010	1
1	6/30/2010	25.790001	30.42	23.299999	23.830000	17187100	30	6	2010	1
2	7/1/2010	25.000000	25.92	20.270000	21.959999	8218800	1	7	2010	0
3	7/2/2010	23.000000	23.10	18.709999	19.200001	5139800	2	7	2010	0
4	7/6/2010	20.000000	20.00	15.830000	16.110001	6866900	6	7	2010	0

In[10]:

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

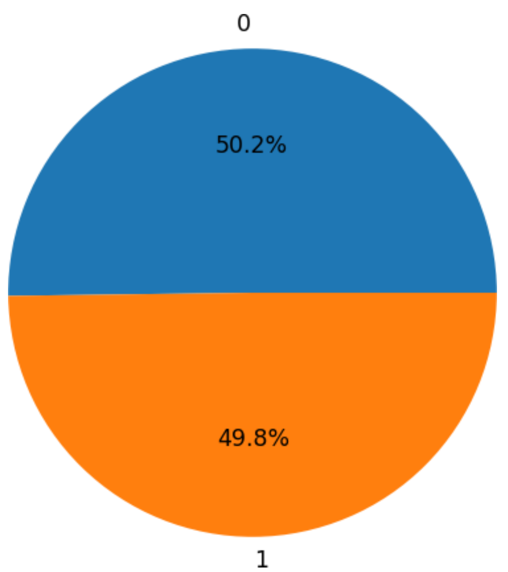
Out[10]:

	Open	High	Low	Close	Volume	day	month	year
is_quarter_end								
0	130.813739	133.182620	128.257229	130.797709	4.461581e+06	15.686501	6.141208	2013.353464
1	135.679982	137.927032	133.455777	135.673269	3.891084e+06	15.657244	7.584806	2013.314488

In[11]:

```
df = pd.read_csv('D:\data\msft.csv')
print(df.head())
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()
```

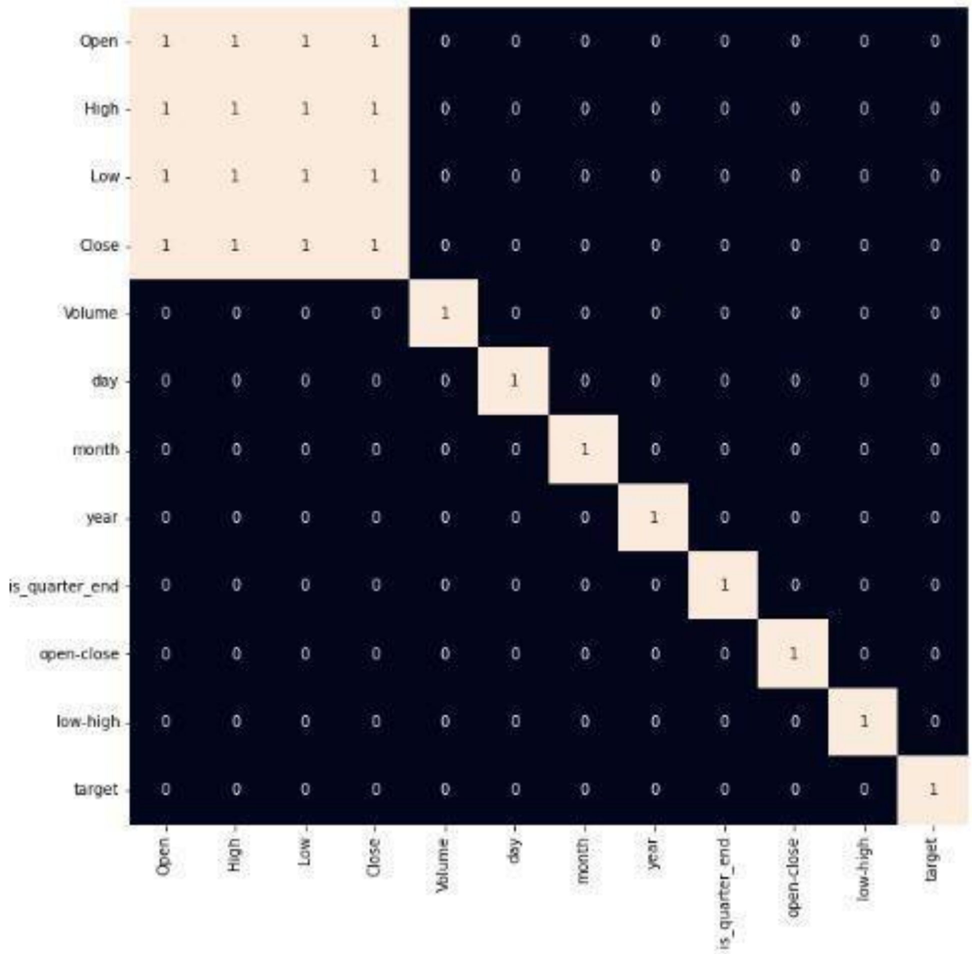
Out[11]:



In[12]:

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

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

plt.show()
```

Out[12]:



In[13]:

```
models = [LogisticRegression(), SVC(

kernel='poly', probability=True), XGBClassifier()]

for i in range(3):

  models[i].fit(X_train, Y_train)
```

```
print(f'{models[i]} : ')

print('Training Accuracy : ', metrics.roc_auc_score(

Y_train, models[i].predict_proba(X_train)[:,1]))

print('Validation Accuracy : ', metrics.roc_auc_score(

Y_valid, models[i].predict_proba(X_valid)[:,1]))

print()
```

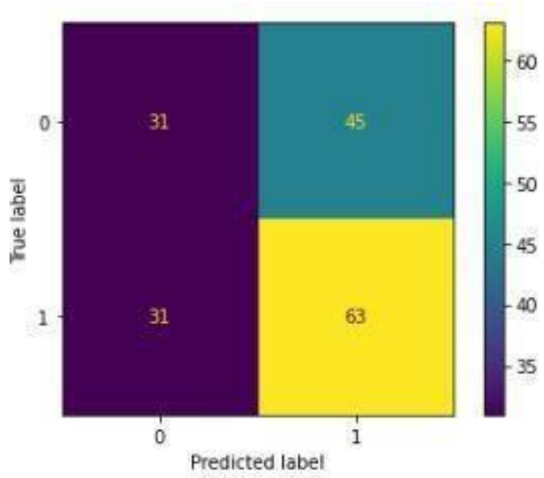
Out[13]:

```
LogisticRegression() :
Training Accuracy :  0.5191709844559586
Validation Accuracy :  0.5435330347144457

SVC(kernel='poly', probability=True) :
Training Accuracy :  0.4718091537132988
Validation Accuracy :  0.4451987681970885

XGBClassifier() :
Training Accuracy :  0.7829611398963732
Validation Accuracy :  0.5706187010078387
```

In[14]:

```
metrics.plot_confusion_matrix(models[0], X_valid, Y_valid)

plt.show()
```

Out[14]:



In[15]:

```
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)
```

Out[15]:

```
(1522, 3) (170, 3)
```

Conclusion and Future Work (Phase 2):

Project Conclusion:

- In the Phase 2 conclusion, we will summarize the key findings and insights from the advanced regression techniques. We will reiterate the impact of these techniques on improving the accuracy and robustness of stock price predictions.
- Future Work: We will discuss potential avenues for future work, such as incorporating additional data sources exploring deep learning models for prediction, or expanding the project into a web application with more features and interactivity