Stock Market Predication

S

Tata Steel Stock Predication

Feynn Labs

Harish P
Task 1

What Is the Stock Market?

The term stock market refers to several exchanges in which shares of publicly held companies are bought and sold. Such financial activities are conducted through formal exchanges and via over-the-counter (OTC) marketplaces that operate under a defined set of regulations.

Both “stock market” and “stock exchange” are often used interchangeably. Traders in the stock market buy or sell shares on one or more of the stock exchanges that are part of the overall stock market.

Understanding the Stock Market:

The stock market allows buyers and sellers of securities to meet, interact, and transact. The markets allow for price discovery for shares of corporations and serve as a barometer for the overall economy. Buyers and sellers are assured of a fair price, high degree of liquidity, and transparency as market participants compete in the open market.

The first stock market was the London Stock Exchange which began in a coffeehouse, where traders met to exchange shares, in 1773. The first stock exchange in the United States began in Philadelphia in 1790. The Buttonwood Agreement, so named because it was signed under a buttonwood tree, marked the beginning of New York’s Wall Street in 1792. The agreement was signed by 24 traders and was the first American organization of its kind to trade in securities. The traders renamed their venture the New York Stock and Exchange Board in 1817.

A stock market is a regulated and controlled environment. In the United States, the main regulators include the Securities and Exchange Commission (SEC) and the Financial Industry Regulatory Authority (FINRA)

The earliest stock markets issued and dealt in paper-based physical share certificates. Today, stock markets operate electronically.

SEBI’s Role:

While trading in the Indian stock market, investors and traders have to execute trades while abiding by rules. This is to promote fairness. SEBI’s role is to carry out functions that meet with the tenets of SEBI regulations and these functions include the following:

SEBI regulates Capital Markets through certain measures it takes.

Protects the interests of traders and investors, thereby, promoting fairness in the stock exchange.

SEBI regulates how the security markets and stock exchanges function.

SEBI regulates how transfer agents, stock brokers and merchant bankers, etc, function.

SEBI handles the registration activity of new brokers, financial advisors, etc.

SEBI encourages the formation of Self-regulatory Organizations.

Process for Predicting Tata Steel's Stock Price:

1. Data collection:
    - Collect historical data on Tata Steel stock prices
    - Obtain data on company financials, market trends, and geopolitical events
2. Data pre-processing:
    - Clean the data and handle missing values
    - Transform the data into a format that can be used by machine learning algorithms
    - Normalize or scale the data if necessary
    - Perform feature engineering to extract relevant features
3. Feature selection:
    - Identify the most important features using techniques such as correlation analysis, mutual information, or PCA
4. Machine learning model selection:
    - Choose an appropriate machine learning algorithm for stock price prediction
    - Some popular algorithms include linear regression, decision trees, random forests, and neural networks
5. Model training:
    - Train the machine learning model using the historical data
    - Use cross-validation techniques to evaluate model performance and optimize hyperparameters
6. Model testing:
    - Test the model on new data to evaluate its performance
    - Use metrics such as mean squared error (MSE) or root mean squared error (RMSE) to evaluate model accuracy
7. Deployment:
    - Deploy the machine learning model to generate stock price predictions
    - Use the model to generate predictions for different time horizons, such as daily, weekly, or monthly
    - Consider other factors such as market trends, geopolitical events, and company performance when making investment decisions

Tata Steel Stock Price Predication.

Importing the modules

In [139…
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
```

In [139…
```
df=pd.read_csv("TATASTEEL.csv")
```

In [139…
```
df
```

Out[1396]:

	Date	Open	High	Low	Close	Adj Close	Volume
0	22-04-2022	130.160004	131.154999	127.375000	127.754997	56.789009	55848230
1	25-04-2022	125.000000	126.294998	121.279999	122.035004	54.246387	72310460
2	26-04-2022	123.500000	124.169998	122.500000	123.360001	54.835365	44812710
3	27-04-2022	122.699997	125.500000	121.040001	124.669998	55.417679	71209740
4	28-04-2022	125.949997	126.864998	124.300003	126.084999	56.046669	51295830
...	...	...	...	...	...	...	...
243	17-04-2023	107.599998	108.300003	106.750000	107.150002	107.150002	24642162
244	18-04-2023	107.199997	107.900002	106.849998	107.599998	107.599998	25476339
245	19-04-2023	107.849998	110.400002	107.750000	108.099998	108.099998	62100775
246	20-04-2023	108.199997	108.699997	107.400002	108.000000	108.000000	18337948
247	21-04-2023	108.000000	108.099998	105.349998	106.150002	106.150002	37638506

248 rows × 7 columns

In [139…
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 248 entries, 0 to 247
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   Date       248 non-null    object 
 1   Open       248 non-null    float64
 2   High       248 non-null    float64
 3   Low        248 non-null    float64
 4   Close      248 non-null    float64
 5   Adj Close  248 non-null    float64
 6   Volume     248 non-null    int64  
dtypes: float64(5), int64(1), object(1)
memory usage: 13.7+ KB
```

In [139…
```
df.describe()
```

Out[1398]:

	Open	High	Low	Close	Adj Close	Volume
count	248.000000	248.000000	248.000000	248.000000	248.000000	2.480000e+02
mean	107.010665	108.273145	105.397621	106.690443	97.093797	6.178698e+07
std	9.335336	9.392647	9.143153	9.271932	21.576094	3.290602e+07
min	84.000000	85.595001	82.699997	83.809998	44.093662	7.480281e+06
25%	102.737503	103.942499	101.128752	102.298752	92.620001	3.801461e+07
50%	107.000000	108.150002	105.424999	106.765003	105.125000	5.501907e+07
75%	111.825003	112.912501	110.187502	111.449997	109.162501	7.549948e+07
max	132.729996	133.000000	127.800003	129.520004	123.550003	2.738835e+08

check if there were any null values

In [139…
```
df.duplicated().sum()
```

Out[1399]:
```
0
```

Check if there were any duplicate values

In [140…
```
df.duplicated().sum()
```

Out[1400]:
```
0
```

Split Date into Year, Month and Date

In [140…
```
df["date"]=df["Date"].str[0:2]
```

In [140…
```
df["month_num"]=df["Date"].str[3:5]
```

In [140…
```
df["Year"]=df["Date"].str[6:10]
```

In [140…
```
def month_name(num):
    if num == "01":
        return "January"
    elif num == "02":
        return "February"
    elif num == "03":
        return "March"
    elif num == "04":
        return "April"
    elif num == "05":
        return "May"
    elif num == "06":
        return "June"
    elif num == "07":
        return "July"
    elif num == "08":
        return "August"
    elif num == "09":
        return "September"
    elif num == "10":
        return "October"
    elif num == "11":
        return "November"
    elif num == "12":
        return "December"
    else:
        return "Invalid Month"
```

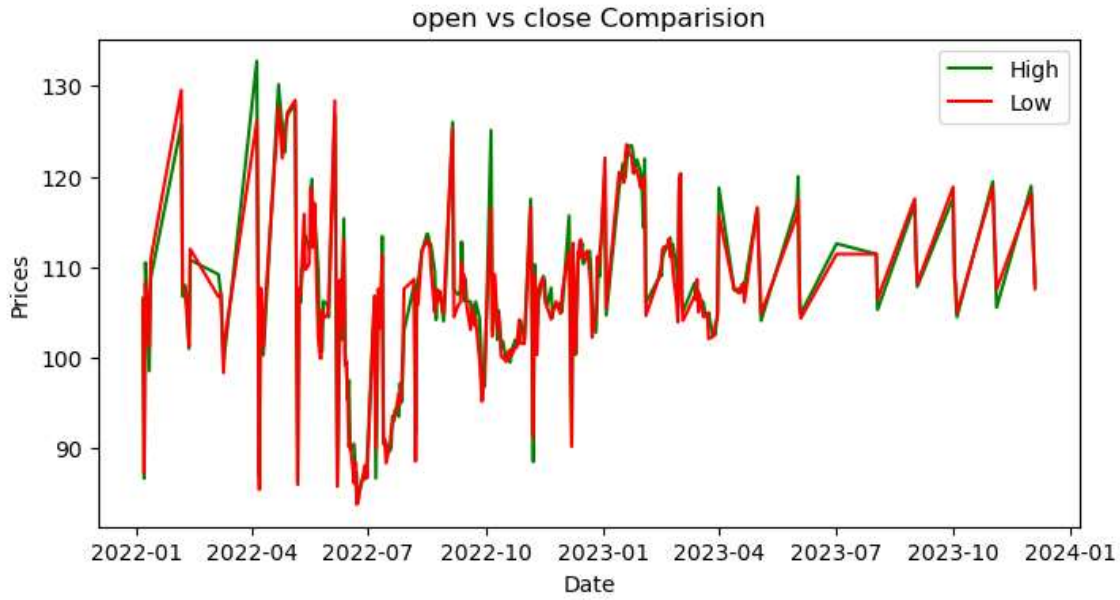
In [140…
```
df["month_name"]=df["Date"].str[3:5].apply(month_name)
```

In [140…
```
df['Date'] = pd.to_datetime(df['Date'])
```

Exploratory Data Analysis

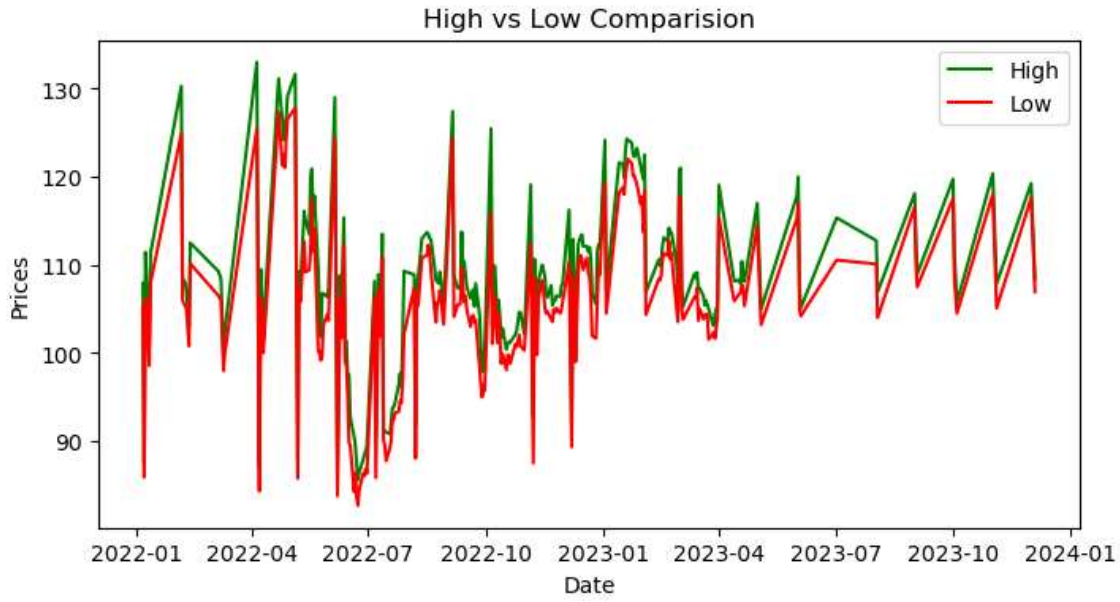
In [140…
```
plt.figure(figsize=(8,4))
sns.lineplot(x=df["Date"],y=df["Open"],color="g", label="High")
sns.lineplot(x=df["Date"],y=df["Close"],color="r", label="Low")
plt.ylabel("Prices")
plt.title("open vs close Comparision")
plt.legend()
```

Out[1407]:
```
<matplotlib.legend.Legend at 0x1e6889a7280>
```



In [140…
```
plt.figure(figsize=(8,4))
sns.lineplot(x=df["Date"],y=df["High"],color="g", label="High")
sns.lineplot(x=df["Date"],y=df["Low"],color="r", label="Low")
plt.ylabel("Prices")
plt.title("High vs Low Comparision")
plt.legend()
```

Out[1408]: <matplotlib.legend.Legend at 0x1e6894b83a0>



open price by Date

In [140…
```
sorted_df=df.sort_values(by="date")
```

In [141…
```
sorted_df
```

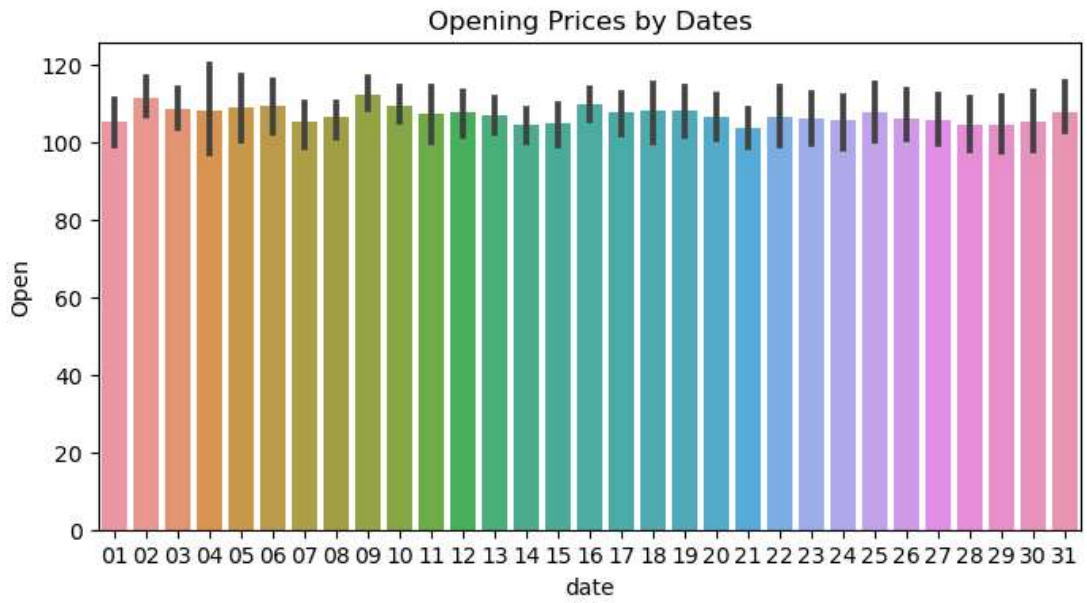
Out[1410]:

	Date	Open	High	Low	Close	Adj Close	Volume	date	month_num	Year	month_name
131	2022-01-11	98.550003	101.449997	98.550003	101.199997	101.199997	69031178	01	11	2022	November
27	2022-01-06	106.550003	107.900002	105.330002	106.355003	47.276394	94055290	01	06	2022	June
215	2023-01-03	104.699997	107.099998	104.500000	105.500000	105.500000	76175319	01	03	2023	March
70	2022-01-08	110.500000	111.449997	106.150002	108.250000	108.250000	125365761	01	08	2022	August
152	2022-01-12	108.699997	111.300003	108.000000	110.750000	110.750000	91702642	01	12	2022	December
...	...	...	...	...	...	...	...	...	...	...	...
48	2022-06-30	88.150002	89.449997	86.330002	86.705002	86.705002	84672090	30	06	2022	June
235	2023-03-31	105.000000	105.500000	104.099998	104.500000	104.500000	34579415	31	03	2023	March
26	2022-05-31	104.500000	107.190002	103.699997	105.565002	46.925224	121248850	31	05	2022	May
130	2022-10-31	101.900002	102.150002	100.300003	101.550003	101.550003	38715841	31	10	2022	October
194	2023-01-31	118.750000	120.449997	117.750000	119.699997	119.699997	43986769	31	01	2023	January

248 rows × 11 columns

In [141…
```
plt.figure(figsize=(8,4))
sns.barplot(x=sorted_df["date"],y=sorted_df["Open"])
plt.title("Opening Prices by Dates")
```

Out[1411]: Text(0.5, 1.0, 'Opening Prices by Dates')

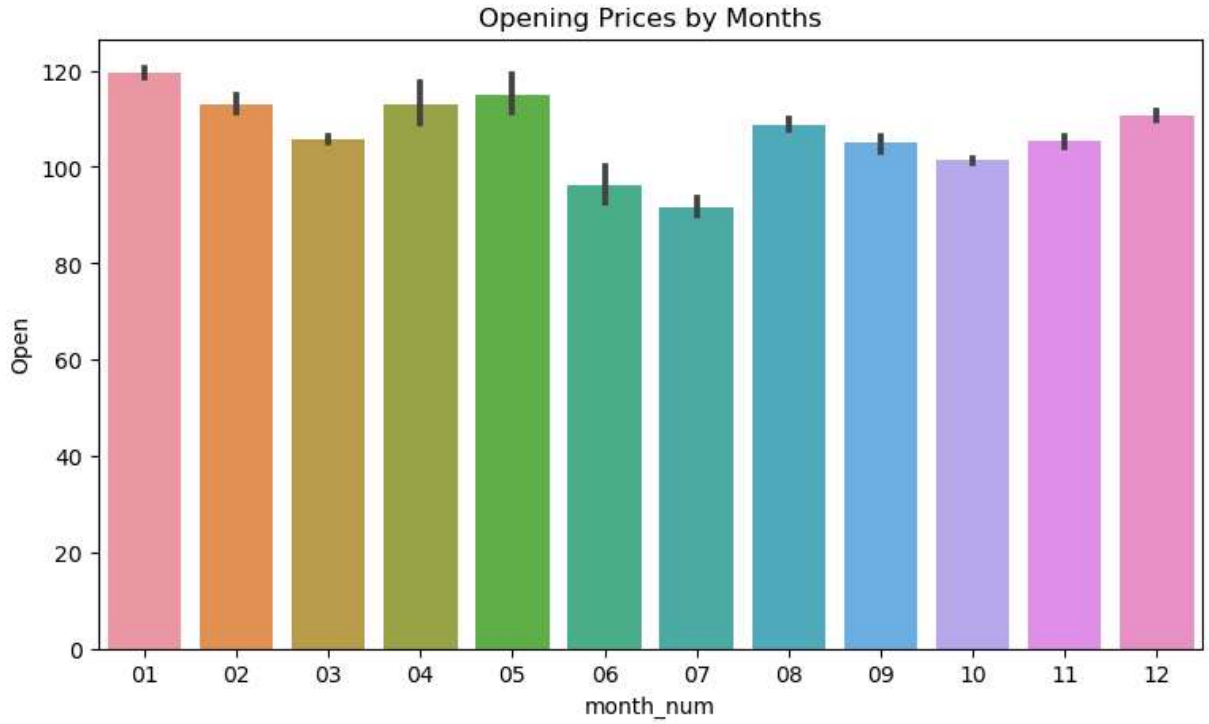


Open price by month

In [141…
```
sorted_df=df.sort_values(by="month_num")
```

In [141…
```
plt.figure(figsize=(9,5))
sns.barplot(x=sorted_df["month_num"],y=sorted_df["Open"])
plt.title("Opening Prices by Months")
```

Out[1413]: Text(0.5, 1.0, 'Opening Prices by Months')



Coreleation of features

In [141…
```
plt.figure(figsize=(9,5))
sns.heatmap(df.corr(),annot=True)
```

Out[1414]: <AxesSubplot:>

In [141…
```
df.drop(["High","Low","Close","Adj Close","Volume","Date"],axis=1,inplace=True)
```

In [141…
```
df=df.sample(frac=1)#To shuffle the all the rows
```

In [141…
```
df
```

Out[1417]:

	Open	date	month_num	Year	month_name
121	99.750000	17	10	2022	October
137	105.199997	10	11	2022	November
179	116.900002	09	01	2023	January
34	103.000000	10	06	2022	June
187	121.800003	19	01	2023	January
...	...	...	...	...	...
2	123.500000	26	04	2022	April
97	106.800003	12	09	2022	September
129	104.099998	28	10	2022	October
106	106.199997	23	09	2022	September
112	99.599998	03	10	2022	October

248 rows × 5 columns

Divide and Split data for the model Buliding

In [141…
```
x=df.drop(["Open","month_name"],axis=1)
```

In [141…
```
y=df["Open"]
```

In [142…
```
from sklearn.model_selection import train_test_split
```

In [142…
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.30,random_state=5)
```

Import the models

In [142…
```
from sklearn.ensemble import AdaBoostRegressor
from sklearn.ensemble import GradientBoostingRegressor
```

Model 1

In [142…
```
model1=AdaBoostRegressor()
model1.fit(x_train,y_train)
y_pred=model1.predict(x_test)
```

In [142…
```
from sklearn.metrics import r2_score
```

In [142…
```
r2_score(y_test,y_pred)
```

Out[1425]:
```
0.7470129000433503
```

Model 2

In [142…
```
model2=GradientBoostingRegressor()
model2.fit(x_train,y_train)
y_pred=model2.predict(x_test)
r2_score(y_test,y_pred)
```

Out[1426]:
```
0.9317646589235277
```

In [142…
```
from sklearn.metrics import mean_absolute_error,mean_absolute_percentage_error
```

In [142…
```
mean_absolute_error(y_test,y_pred)
```

Out[1428]:
```
1.8874360201434897
```

In [142…
```
mean_absolute_percentage_error(y_test,y_pred)
```

Out[1429]:
```
0.017909802631837304
```

Conclusion:

In the case of Tata Steel, a machine learning model can be developed to predict the stock prices based on historical data, company financials, market trends, and geopolitical events.

The process of developing a machine learning model for stock price prediction involves several steps, including data collection, data preprocessing, feature selection, machine learning model selection, model training, model testing, and deployment. The accuracy of the predictions will depend on the quality and relevance of the data used, as well as the choice of machine learning algorithm and hyperparameters.

It's important to note that stock prices are influenced by a variety of factors that are difficult to predict, such as macroeconomic trends, geopolitical events, and company performance. Therefore, while machine learning can be a useful tool for predicting stock prices, it's important to consider other factors and perform a comprehensive analysis before making investment decisions.