STOCK MARKET PREDICTION

A Course Project report submitted

in partial fulfillment of requirement for the award of degree

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE & ENGINEERING

by

N.Vivek naik - 2003A51188

Under the guidance of

Mr. S NARESH KUMAR

Assistant Professor, Department of CSE.

Department of Computer Science and Artificial Intelligence

Department of Computer Science and Artificial Intelligence

CERTIFICATE

This is to certify that this project entitled **"STOCK MARKET PREDICTION**" is the bonafied work carried out by **N.VIVEK NAIK(2003A51188)**as a Course Project for the partial fulfillment to award the degree **BACHELOR OF TECHNOLOGY** in **COMPUTER SCIENCE & ENGINEERING** during the academic year 2021-2022 under our guidance and Supervision.

Mr. S Naresh Kumar
Assistant Professor ,
S R University,
Ananthasagar, Warangal.

Dr. M.Sheshikala
Assoc. Prof. & HOD (CSE),
S R University,
Ananthasagar, Warangal.

ACKNOWLEDGEMENT

We express our thanks to Course co-coordinator **Mr. S.Naresh Kumar, Asst. Prof.** for guiding us from the beginning through the end of the Course Project. We express our gratitude to Head of the department CS&AI, **Dr. M.Sheshikala, Associate Professor** for encouragement, support and insightful suggestions. We truly value their consistent feedback on our progress, which was always constructive and encouraging and ultimately drove us to the right direction.

We wish to take this opportunity to express our sincere gratitude and deep sense of respect to our beloved Dean, School of Computer Science and Artificial Intelligence, **C. V. Guru Rao**, for his continuous support and guidance to complete this project in the institute.

Finally, we express our thanks to all the teaching and non-teaching staff of the department for their suggestions and timely support.

ABSTRACT

Machine learning (ML) is a technology that gives the systems the ability to learn on its own through real-world interactions and generalizing from examples without being explicitly programmed as in the case of rule-based programming. Machine Learning can play a key role in a wide range of critical applications. In machine learning, Linear Regression (LR) is a basic technique by which a linear trend can be obtained. But Support Vector Machines (SVMs) have advanced features such as high accuracy and predictability. In this paper we survey the pros and cons of using both these techniques to predict values and compare both algorithms.

Keywords: Prediction, Datasets, Linear Regression, Support Vector Machines, Machine Learning

Contents

Table of Contents

INTRODUCTION

The financial market is a dynamic and composite system where people can buy and sell currencies, stocks, equities and derivatives over virtual platforms supported by brokers. The stock market allows investors to own shares of public companies through trading either by exchange or over the counter markets. This market has given investors the chance of gaining money and having a prosperous life through investing small initial amounts of money, low risk compared to the risk of opening new business or the need of high salary career. Stock markets are affected by many factors causing the uncertainty and high volatility in the market. Although humans can take orders and submit them to the market, automated trading systems (ATS) that are operated by the implementation of computer programs can perform better and with higher momentum in submitting orders than any human. However, to evaluate and control the performance of ATSs, the implementation of risk strategies and safety measures applied based on human judgements are required. Many factors are incorporated and considered when developing an ATS, for instance, trading strategy to be adopted, complex mathematical functions that reflect the state of a specific stock, machine learning algorithms that enable the prediction of the future stock value, and specific news related to the stock being analysed. Time-series prediction is a common technique widely used in many real-world applications such as weather forecasting and financial market prediction. It uses the continuous data in a period of time to predict the result in the next time unit. Many timeseries prediction algorithms have shown their effectiveness in practice. One of these methods was to fit a straight line of the form $y = mx + c$ to a graph such that the line passes through the maximum number of data points of the given dataset. Mathematically speaking, on plotting the values of the dataset on a graph, fit a straight line through the points such that the square of the distance between each point and the line is minimum. This line, called the hypothesis is used to predict the y value for any given x. This prediction technique is called Linear Regression and the formula used is called the Least Squares method. This technique is widely known to statisticians and has also been used as one of the basic concepts of ML. The hypothesis function of Linear Regression has the general form,

$$y = h\theta(x) = \theta 0 + \theta 1 x \quad (1)$$

Note that this is like the equation of a straight line. The values of $\theta 0$ and $\theta 1$ is given to $h\theta(x)$ to get the estimated output y. Note that the effort is to get various values of $\theta 0$ and $\theta 1$ to try to find values which provide the best possible “fit” or the most representative “straight line” through the data points mapped on the x-y plane.

PROBLEM STATEMENT

The stock market appears in the news every day. You hear about it every time it reaches a new high or a new low. The rate of investment and business opportunities in the Stock market can increase if an efficient algorithm could be devised to predict the short term price of an individual stock.

Previous methods of stock predictions involve the use of Artificial Neural Networks and Convolution Neural Networks which has an error loss at an average of 20%.

In this report, we will see if there is a possibility of devising a model using Recurrent Neural Network which will predict stock price with a less percentage of error. And if the answer turns to be **YES**, we will also see how reliable and efficient will this model be.

OBJECTIVES DESCRIPTION

Stock market prediction seems a complex problem because there are many factors that have yet to be proper use of machine learning techniques, one can relate previous data to the current data and train the machine to learn from it and make appropriate assumptions. Machine learning as such has many models but this paper focuses on two most important of them and made the predictions using them.

METHODOLOGY

A. Environment

This survey has been performed using a statistical language such as R, with the RStudio development environment. It is therefore easier to view the observations later by plotting them using functions predefined in R. Since Linear Regression and SVMs are standard algorithms used in almost all Data Science fields, the built-in functions in PYTHON packages are being used for this purpose.

B. Time-Series Forecasting

A time series is a series of data points indexed (or listed or graphed) in time order. Most commonly, a time series is a sequence taken at successive equally spaced points in time. Thus, it is a sequence of discrete-time data. Time series analysis comprises methods for analysing time series data in order to extract meaningful statistics and other characteristics of the data. Time series forecasting is the use of a model to predict future values based on previously observed values. In statistics, prediction is a part of statistical inference. When information is transferred across time, often to specific points in time, the process is known as forecasting

C Sliding-Window Method

In time series prediction, the time series are typically expanded into three or higher-dimensional space to exploit the information that is implicit in them. Given a sequence of numbers for a time series dataset, the data can be restructured to look like a supervised learning problem. This can be done by using previous time steps as input variables and using the next time step as the output variable .

3. LITERATURE SURVEY

RELATED WORK

PYTHON - Python is an interpreted, high-level, general-purpose programming language. Python's design philosophy emphasizes code readability with its notable use of significant whitespace. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects. Python is dynamically typed AND supports multiple programming paradigms, including procedural, object-oriented, and functional programming.

JUPYTER Lab - Project Jupyter is a non-profit organization created to develop open-source software, open-standards, and services for interactive computing across dozens of programming languages.

SYSTEM STUDY

This survey is done to comprehend the need and prerequisite of the general population, and to do as such, we went through different sites and applications and looked for the fundamental data. Based on these data, we made an audit that helped us get new thoughts and make different arrangements for our task. We reached the decision that there is a need of such application and felt that there is a decent extent of progress in this field

DESIGN

Requirement specifications

software requirements

Python

Jupyter

Libraries

- Numpy
- Pandas
- Matplotlib
- Scikit-learn

Hardware requirements

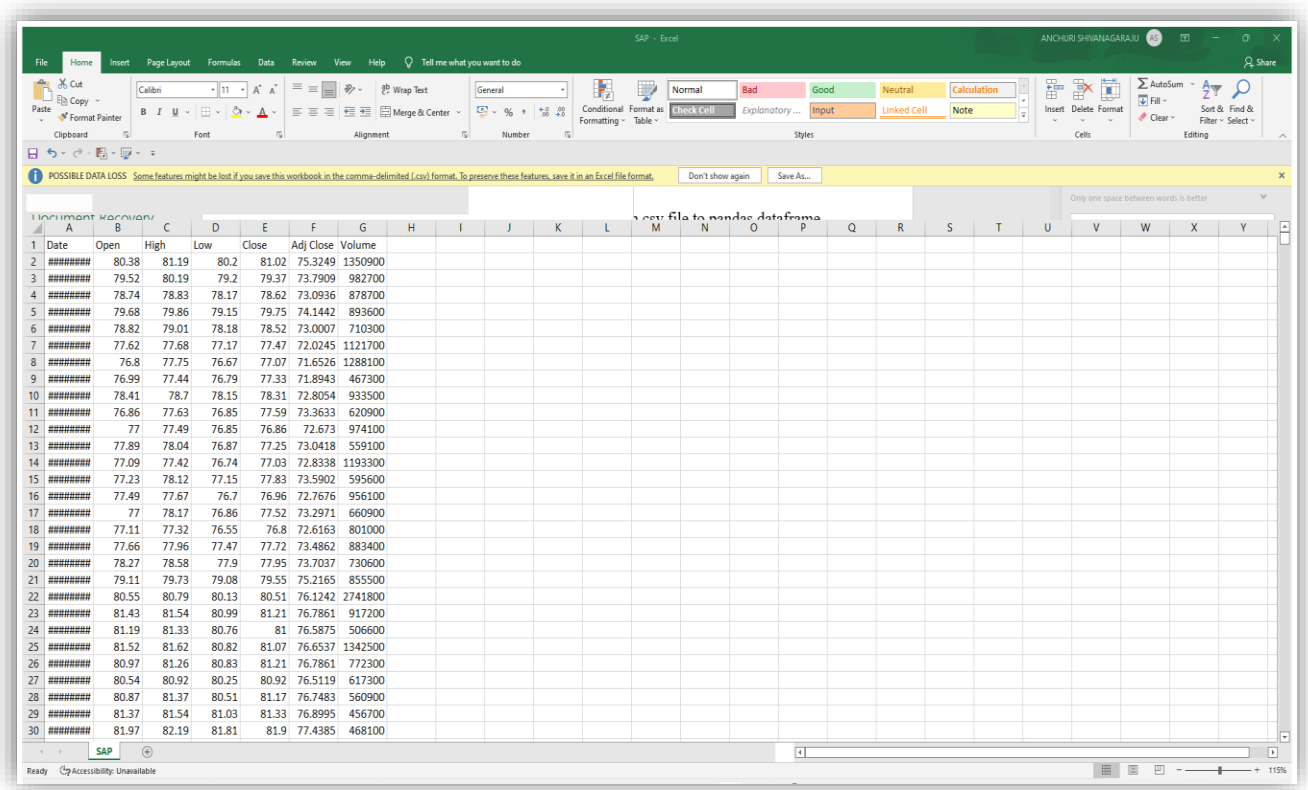
Laptop with basic hardware

IMPLEMENTATION

Modules

1. Download data set in csv form
2. Upload to folder
3. Open Jupyter
4. Loading the data from csv file to pandas dataframe
5. Understanding of the dataset
6. Checking missing values
7. Checking distribution of categorical values
8. Encoding certain columns(if required)
9. Data Visualization
10. Deploy linear regression model manually
11. Set the best possible equation
12. Obtain the mean error for comparison and further study
13. If possible find which feature variables is effecting the target variable

DATA SET :



	A	B	C	D	E	F	G
1	Date	Open	High	Low	Close	Adj Close	Volume
2	########	80.38	81.19	80.2	81.02	75.3249	1350900
3	########	79.52	80.19	79.2	79.37	73.7909	982700
4	########	78.74	78.83	78.17	78.62	73.0936	878700
5	########	79.68	79.86	79.15	79.75	74.1442	893600
6	########	78.82	79.01	78.18	78.52	73.0007	710300
7	########	77.62	77.68	77.17	77.47	72.0245	1121700
8	########	76.8	77.75	76.67	77.07	71.6526	1288100
9	########	76.99	77.44	76.79	77.33	71.8943	467300
10	########	78.41	78.7	78.15	78.31	72.8054	933500
11	########	76.86	77.63	76.85	77.59	73.3633	620900
12	########	77	77.49	76.85	76.86	72.673	974100
13	########	77.89	78.04	76.87	77.25	73.0418	559100
14	########	77.09	77.42	76.74	77.03	72.8338	1193300
15	########	77.23	78.12	77.15	77.83	73.5902	595600
16	########	77.49	77.67	76.7	76.96	72.7676	956100
17	########	77	78.17	76.86	77.52	73.2971	660900
18	########	77.11	77.32	76.55	76.8	72.6163	801000
19	########	77.66	77.96	77.47	77.72	73.4862	883400
20	########	78.27	78.58	77.9	77.95	73.7037	730600
21	########	79.11	79.73	79.08	79.55	75.2165	855500
22	########	80.55	80.79	80.13	80.51	76.1242	2741800
23	########	81.43	81.54	80.99	81.21	76.7861	917200
24	########	81.19	81.33	80.76	81	76.5875	506600
25	########	81.52	81.62	80.82	81.07	76.6537	1342500
26	########	80.97	81.26	80.83	81.21	76.7861	772300
27	########	80.54	80.92	80.25	80.92	76.5119	617300
28	########	80.87	81.37	80.51	81.17	76.7483	560900
29	########	81.37	81.54	81.03	81.33	76.8995	456700
30	########	81.97	82.19	81.81	81.9	77.4385	468100

LOADING DATA :

In [15]:
```
df = pd.read_csv("sap.csv")
df.head(20)
```

Out[15]:

	Date	Open	High	Low	Close	Adj Close	Volume
0	2016-04-27	80.379997	81.190002	80.199997	81.019997	75.324928	1350900
1	2016-04-28	79.519997	80.190002	79.199997	79.370003	73.790909	982700
2	2016-04-29	78.739998	78.830002	78.169998	78.620003	73.093628	878700
3	2016-05-02	79.680000	79.860001	79.150002	79.750000	74.144196	893600
4	2016-05-03	78.820000	79.010002	78.180000	78.519997	73.000656	710300
5	2016-05-04	77.620003	77.680000	77.169998	77.470001	72.024467	1121700
6	2016-05-05	76.800003	77.750000	76.669998	77.070000	71.652580	1288100
7	2016-05-06	76.989998	77.440002	76.790001	77.330002	71.894310	467300
8	2016-05-09	78.410004	78.699997	78.150002	78.309998	72.805420	933500
9	2016-05-10	76.860001	77.629997	76.849998	77.589996	73.363281	620900
10	2016-05-11	77.000000	77.489998	76.849998	76.860001	72.673042	974100
11	2016-05-12	77.889999	78.040001	76.870003	77.250000	73.041801	559100
12	2016-05-13	77.089996	77.419998	76.739998	77.029999	72.833786	1193300
13	2016-05-16	77.230003	78.120003	77.150002	77.830002	73.590210	595600
14	2016-05-17	77.489998	77.669998	76.699997	76.959999	72.767601	956100
15	2016-05-18	77.000000	78.169998	76.860001	77.519997	73.297089	660900
16	2016-05-19	77.110001	77.320000	76.550003	76.800003	72.616318	801000
17	2016-05-20	77.660004	77.959999	77.470001	77.720001	73.486198	883400
18	2016-05-23	78.269997	78.580002	77.900002	77.949997	73.703659	730600
19	2016-05-24	79.110001	79.730003	79.080002	79.550003	75.216515	855500

DATA CLEANING

In [4]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1258 entries, 0 to 1257
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   Date       1258 non-null   object 
 1   Open       1258 non-null   float64
 2   High       1258 non-null   float64
 3   Low        1258 non-null   float64
 4   Close      1258 non-null   float64
 5   Adj Close  1258 non-null   float64
 6   Volume     1258 non-null   int64  
dtypes: float64(5), int64(1), object(1)
memory usage: 68.9+ KB
```

In [5]:
```
df.describe()
```

Out[5]:

	Open	High	Low	Close	Adj Close	Volume
count	1258.000000	1258.000000	1258.000000	1258.000000	1258.000000	1.258000e+03
mean	113.877051	114.605533	113.128331	113.908013	110.992397	8.590167e+05
std	19.456975	19.617020	19.234336	19.433727	20.830699	6.328945e+05
min	72.349998	72.360001	71.389999	72.019997	68.096703	1.179000e+05
25%	103.049999	103.922499	102.412503	103.314999	99.348825	5.167750e+05
50%	113.420002	113.990002	112.645001	113.279999	109.620148	7.136500e+05
75%	125.545002	126.559998	124.590001	125.750000	124.164991	9.943500e+05
max	168.089996	169.300003	166.289993	169.020004	169.020004	1.129020e+07

In [6]:
```
df.columns
```

Out[6]:
```
Index(['Date', 'Open', 'High', 'Low', 'Close', 'Adj Close', 'Volume'], dtype='object')
```

DATA VISUALIZTION :

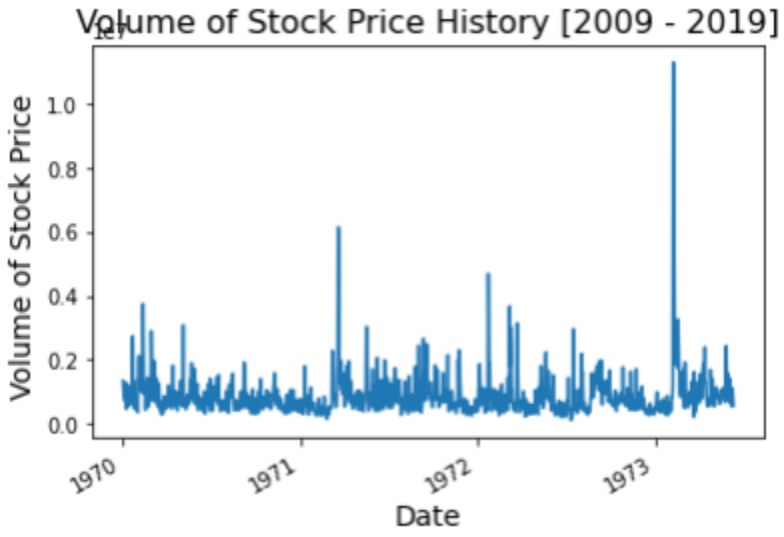
Graphs based models on every possible aspects.

Date vs Open

In [20]:
```
import matplotlib.dates as mdates

years = mdates.YearLocator() # Get every year
yearsFmt = mdates.DateFormatter('%Y') # Set year format

# Create subplots to plot graph and control axes
fig, ax = plt.subplots()
ax.plot(df['Date'], df['Open'])

# Format the ticks
ax.xaxis.set_major_locator(years)
ax.xaxis.set_major_formatter(yearsFmt)

# Set figure title
plt.title('Open Stock Price History [2009 - 2019]', fontsize=16)
# Set x label
plt.xlabel('Date', fontsize=14)
# Set y label
plt.ylabel('Opening Stock Price in $', fontsize=14)

# Rotate and align the x labels
fig.autofmt_xdate()

# Show plot
plt.show()
```

Date vs Volume

In [21]:
```
import matplotlib.dates as mdates

years = mdates.YearLocator() # Get every year
yearsFmt = mdates.DateFormatter('%Y') # Set year format

# Create subplots to plot graph and control axes
fig, ax = plt.subplots()
ax.plot(df['Date'], df['Volume'])

# Format the ticks
ax.xaxis.set_major_locator(years)
ax.xaxis.set_major_formatter(yearsFmt)

# Set figure title
plt.title('Volume of Stock Price History [2009 - 2019]', fontsize=16)
# Set x label
plt.xlabel('Date', fontsize=14)
# Set y label
plt.ylabel('Volume of Stock Price ', fontsize=14)

# Rotate and align the x labels
fig.autofmt_xdate()

# Show plot
plt.show()
```

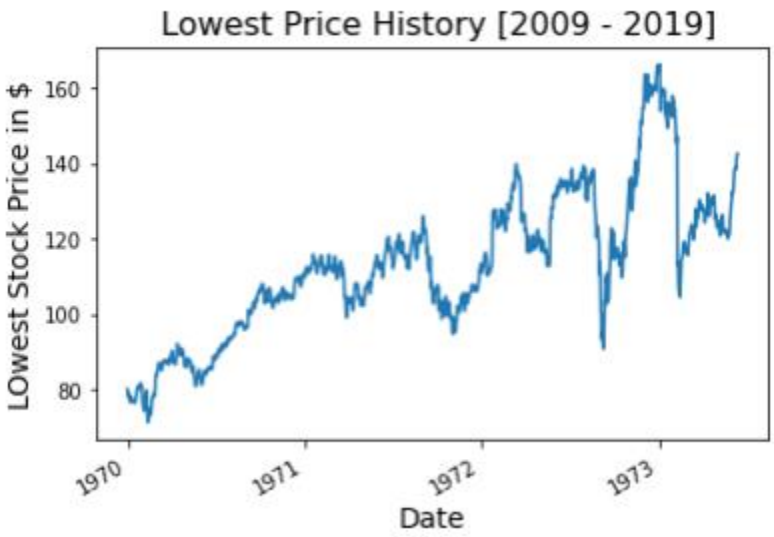


Date vs High

In [22]:
```
import matplotlib.dates as mdates

years = mdates.YearLocator() # Get every year
yearsFmt = mdates.DateFormatter('%Y') # Set year format

# Create subplots to plot graph and control axes
fig, ax = plt.subplots()
ax.plot(df['Date'], df['High'])

# Format the ticks
ax.xaxis.set_major_locator(years)
ax.xaxis.set_major_formatter(yearsFmt)

# Set figure title
plt.title('Highest Stock Price History [2009 - 2019]', fontsize=16)
# Set x label
plt.xlabel('Date', fontsize=14)
# Set y label
plt.ylabel('Highest Stock Price in $', fontsize=14)

# Rotate and align the x labels
fig.autofmt_xdate()

# Show plot
plt.show()
```

Date vs Low

In [23]:
```
import matplotlib.dates as mdates

years = mdates.YearLocator() # Get every year
yearsFmt = mdates.DateFormatter('%Y') # Set year format

# Create subplots to plot graph and control axes
fig, ax = plt.subplots()
ax.plot(df['Date'], df['Low'])

# Format the ticks
ax.xaxis.set_major_locator(years)
ax.xaxis.set_major_formatter(yearsFmt)

# Set figure title
plt.title(' Lowest Price History [2009 - 2019]', fontsize=16)
# Set x label
plt.xlabel('Date', fontsize=14)
# Set y label
plt.ylabel('LOwest Stock Price in $', fontsize=14)

# Rotate and align the x labels
fig.autofmt_xdate()

# Show plot
plt.show()
```



Results :

In [18]:
```
from sklearn.model_selection import train_test_split
# Split data into train and test set: 80% / 20%
train, test = train_test_split(df, test_size=0.20)
```

In [9]:
```
from sklearn.linear_model import LinearRegression
# Reshape index column to 2D array for .fit() method
X_train = np.array(train.index).reshape(-1, 1)
y_train = train['Close']
# Create LinearRegression Object
model = LinearRegression()
# Fit linear model using the train data set
model.fit(X_train, y_train)
```

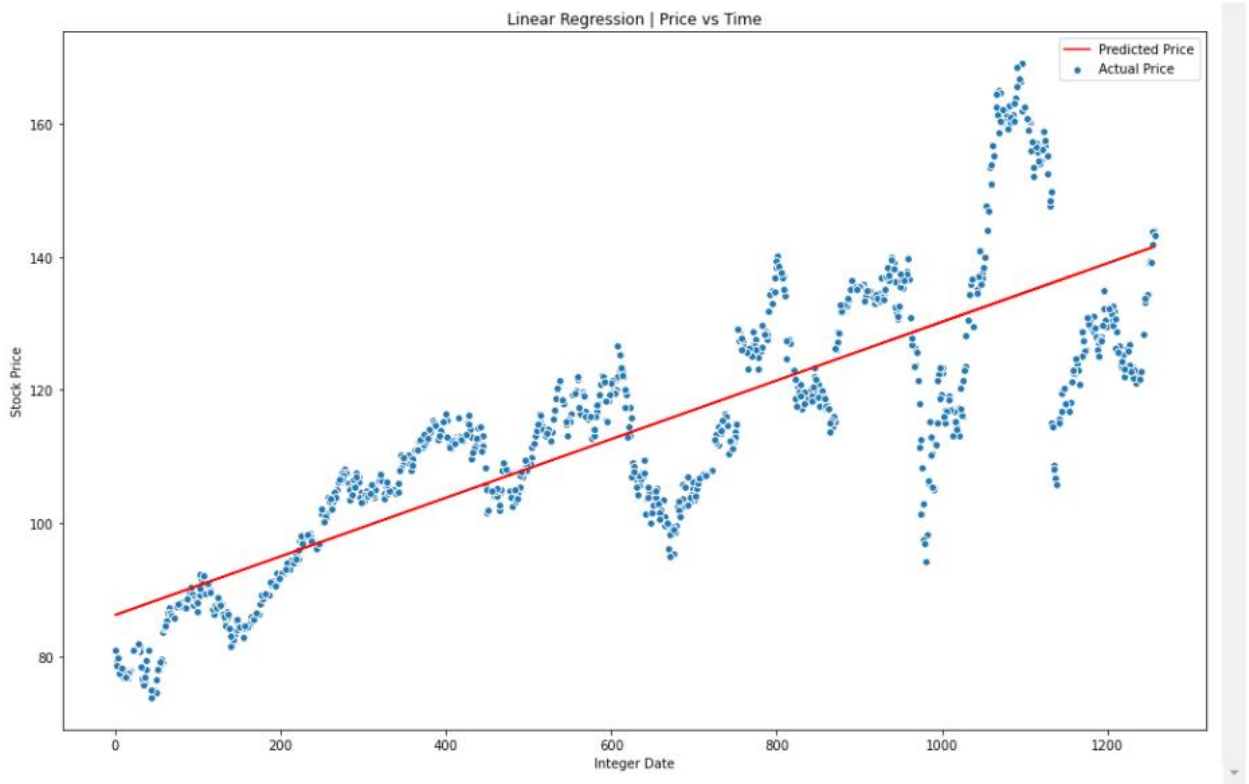
Out[9]:
```
LinearRegression()
```

In [24]:
```
# The coefficient
print('Slope: ', np.asscalar(np.squeeze(model.coef_)))
# The Intercept
print('Intercept: ', model.intercept_)
```

```
Slope:  0.043958127564923774
Intercept:  86.26105402827619

C:\Users\SATHIS~1\AppData\Local\Temp/ipykernel_18732/4094384673.py:2: DeprecationWarning: np.asscalar(a) is deprecated
since NumPy v1.16, use a.item() instead
  print('Slope: ', np.asscalar(np.squeeze(model.coef_)))
```

Conclusion

In [11]:
```
# Train set graph
plt.figure(1, figsize=(16,10))
plt.title('Linear Regression | Price vs Time')
plt.scatter(X_train, y_train, edgecolor='w', label='Actual Price')
plt.plot(X_train, model.predict(X_train), color='r', label='Predicted Price')
plt.xlabel('Integer Date')
plt.ylabel('Stock Price')
plt.legend()
plt.show()
```



Linear Regression The first step that was performed was to fetch the values or to download them as a CSV file. In this literature, the stock prices of the Coca-Cola company within the date range were obtained. The obtained data-frame had two columns namely, Date and Close which were initially plotted onto the graph using the plot() functions. A linear model was later fit to this graph and displayed and observations were made.

Conclusion and Future Scope

Predicting stock market returns is a challenging task due to consistently changing stock values which are dependent on multiple parameters which form complex patterns. The historical dataset available on company's website consists of only few features like high, low, open, close, adjacent close value of stock prices, volume of shares traded etc., which are not sufficient enough. To obtain higher accuracy in the predicted price value new variables have been created using the existing variables. ANN is used for predicting the next day closing price of the stock and for a comparative analysis, RF is also implemented. The comparative analysis based on RMSE, MAPE and MBE values clearly indicate that ANN gives better prediction of stock prices as compared to RF. Results show that the best values obtained by ANN model gives RMSE (0.42), MAPE (0.77) and MBE (0.013). For future work, deep learning models could be developed which consider financial news articles along with financial parameters such as a closing price, traded volume, profit and loss statements etc., for possibly better results.

BIBLIOGRAPHY

Alpharithms.com ,Predicting Prices

Git Hub.com Analysis Stock Prices

Chatterjee. *Regression Analysis by Example, 5th Edition*. 5th ed., Wiley, 2012.

Guyon, Isabelle. A Scaling Law for the Validation-Set Training-Set Size Ratio. *In AT & T Bell Laboratories*. (1997) doi:10.1.1.33.1337