SUMMER INTERNSHIP REPORT

Area Of Online Internship	AL/ML/DL
Intern Name	**AKASH S**
Name Of Institution	**INDIAN INSTITUTE OF TECHNOLOGY, INDORE**
Faculty Mentor Name	**Dr. Vimal Bhatia**
Duration	**1 MONTHS (01/06/2021 TO 30/06/2021)**
Date Of Submission	**26/06/2021**

Table Of Contents

Introduction:

Linear Regression Using Machine Learning.

Linear regression is one of the easiest and most popular Machine Learning algorithms. It is a statistical method that is used for predictive analysis. Linear regression makes predictions for continuous/real or numeric variables such as **sales, salary, age, product price,** etc.

Linear Regression is a machine learning algorithm based on **supervised learning**. It performs a **regression task**. Regression models a target prediction value based on independent variables. It is mostly used for finding out the relationship between variables and forecasting. Different regression models differ based on – the kind of relationship between dependent and independent variables, they are considering and the number of independent variables being used.

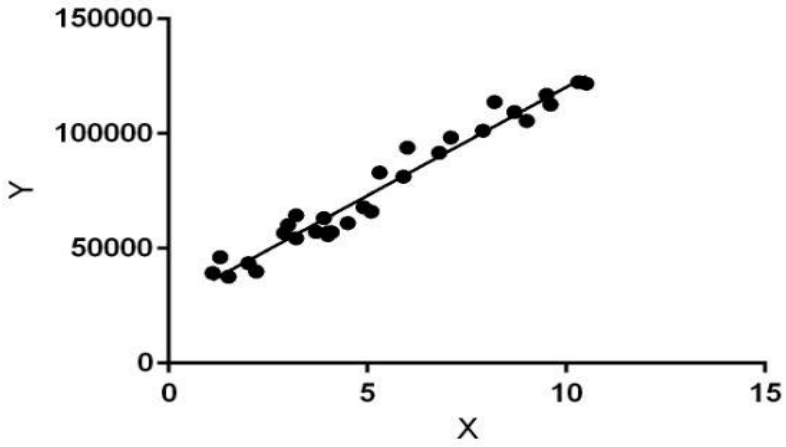


Figure. 1 Reference[2]

Linear regression performs the task to predict a dependent variable value (y) based on a given independent variable (x). So, this regression technique finds out a linear relationship between x (input) and y(output). Hence, the name is Linear Regression.
In the figure above, X (input) is the work experience and Y (output) is the salary of a person. The regression line is the best fit line for our model.

Linear Regression is the basic form of regression analysis. It assumes that there is a linear relationship between the dependent variable and the predictor(s). In regression, we try to calculate the best fit line which describes the relationship between the predictors and predictive/dependent variable.

Hypothesis function for Linear Regression :

$$y = \theta_1 + \theta_2.x$$

Equation 1 Reference [2]

While training the model we are given :
x: input training data (univariate – one input variable(parameter))
y: labels to data (supervised learning)
When training the model – it fits the best line to predict the value of y for a given value of x.
The model gets the best regression fit line by finding the best θ_1 and θ_2 values.
θ_1: intercept
θ_2: coefficient of x

There are four assumptions associated with a linear regression model:

1. **Linearity**: The relationship between independent variables and the mean of the dependent variable is linear.
2. **Homoscedasticity**: The variance of residuals should be equal.
3. **Independence**: Observations are independent of each other.
4. **Normality**: For any fixed value of an independent variable, the dependent variable is normally distributed.

Problem:

Understanding the Problem Statement

Problem - Predict the stock market price of next few days using previous stock market data (equity or indices) using machine learning or Deep learning.
1. Use News headlines as Data for prediction.
2. Use previous Equity data of Day open, close, low, high for prediction.
3. Any other stock Relative data.
Note: Try to improve the accuracy of the previously built model or implement it from scratch

---

Stock Prices Prediction Using Machine Learning and Deep Learning Techniques (with Python codes)

Stock market analysis is divided into two parts – Fundamental Analysis and Technical Analysis.

- Fundamental Analysis involves analyzing the company's future profitability on the basis of its current business environment and financial performance.
- Technical Analysis, on the other hand, includes reading the charts and using statistical figures to identify the trends in the stock market.

There are multiple variables in the dataset – date, open, high, low, last, close, total_trade_quantity, and turnover.

- The columns *Open* and *Close* represent the starting and final price at which the stock is traded on a particular day.
- *High*, *Low* and *Last* represent the maximum, minimum, and last price of the share for the day.
- *Total Trade Quantity* is the number of shares bought or sold in the day and *Turnover (Lacs)* is the turnover of the particular company on a given date.

k-Nearest Neighbours

Another interesting ML algorithm that one can use here is kNN (k nearest neighbours). Based on the independent variables, kNN finds the similarity between new data points and old data points. Let me explain this with a simple example.

Consider the height and age for 11 people. On the basis of given features ('Age' and 'Height'), the table can be represented in a graphical format as shown below

ID	Age	Height	Weight
1	45	5	77
2	26	5.11	47
3	30	5.6	55
4	34	5.9	59
5	40	4.8	72
6	36	5.8	60
7	19	5.3	40
8	28	5.8	60
9	23	5.5	45
10	32	5.6	58
11	38	5.5	?

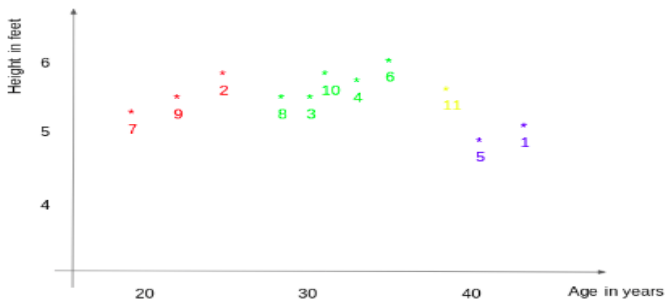


Table 1 reference[1]

Figure. 2 Reference[1]

To determine the weight for ID #11, kNN considers the weight of the nearest neighbors of this ID. The weight of ID #11 is predicted to be the average of it's neighbors. If we consider three neighbours (k=3) for now, the weight for ID#11 would be = (77+72+60)/3 = 69.66 kg.

Table 2

ID	Height	Age	Weight
1	5	45	77
5	4.8	40	72
6	5.8	36	60

Solution

From the given dataset the following predictions

Google Colab Link:

https://colab.research.google.com/drive/1oiBOEeYGQ6-kAkPAiGTqWkoGtjyqNCPz?usp=sharing

Python source codes for stock prediction:

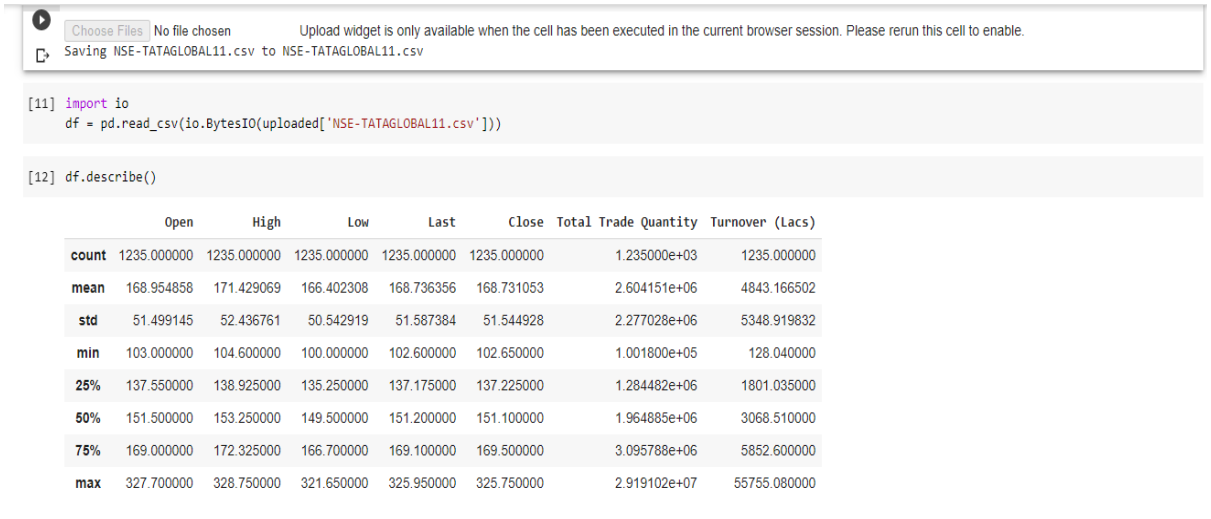
```
#import packages
import pandas as pd
import numpy as np

#to plot within notebook
import matplotlib.pyplot as plt
%matplotlib inline

#setting figure size
from matplotlib.pylab import rcParams
rcParams['figure.figsize'] = 20,10

#for normalizing data
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler(feature_range=(0, 1))

from google.colab import files
uploaded=files.upload()
```



Choose Files No file chosen Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

Saving NSE-TATAGLOBAL11.csv to NSE-TATAGLOBAL11.csv

```
[11] import io
     df = pd.read_csv(io.BytesIO(uploaded['NSE-TATAGLOBAL11.csv']))
```

```
[12] df.describe()
```

	Open	High	Low	Last	Close	Total Trade Quantity	Turnover (Lacs)
count	1235.000000	1235.000000	1235.000000	1235.000000	1235.000000	1.235000e+03	1235.000000
mean	168.954858	171.429069	166.402308	168.736356	168.731053	2.604151e+06	4843.166502
std	51.499145	52.436761	50.542919	51.587384	51.544928	2.277028e+06	5348.919832
min	103.000000	104.600000	100.000000	102.600000	102.650000	1.001800e+05	128.040000
25%	137.550000	138.925000	135.250000	137.175000	137.225000	1.284482e+06	1801.035000
50%	151.500000	153.250000	149.500000	151.200000	151.100000	1.964885e+06	3068.510000
75%	169.000000	172.325000	166.700000	169.100000	169.500000	3.095788e+06	5852.600000
max	327.700000	328.750000	321.650000	325.950000	325.750000	2.919102e+07	55755.080000

Figure 3 Describe the given Data set. Reference[3]

```
[13] df.head()
```

	Date	Open	High	Low	Last	Close	Total Trade Quantity	Turnover (Lacs)
0	2018-10-08	208.00	222.25	206.85	216.00	215.15	4642146.0	10062.83
1	2018-10-05	217.00	218.60	205.90	210.25	209.20	3519515.0	7407.06
2	2018-10-04	223.50	227.80	216.15	217.25	218.20	1728786.0	3815.79
3	2018-10-03	230.00	237.50	225.75	226.45	227.60	1708590.0	3960.27
4	2018-10-01	234.55	234.60	221.05	230.30	230.90	1534749.0	3486.05

Figure . 4 Head of the data set. Reference [given data set]

```
df.tail()
```

Date	Date	Open	High	Low	Last	Close	Total Trade Quantity	Turnover (Lacs)
2013-10-14	2013-10-14	160.85	161.45	157.70	159.3	159.45	1281419.0	2039.09
2013-10-11	2013-10-11	161.15	163.45	159.00	159.8	160.05	1880046.0	3030.76
2013-10-10	2013-10-10	156.00	160.80	155.85	160.3	160.15	3124853.0	4978.80
2013-10-09	2013-10-09	155.70	158.20	154.15	155.3	155.55	2049580.0	3204.49
2013-10-08	2013-10-08	157.00	157.80	155.20	155.8	155.80	1720413.0	2688.94

Figure 5 Tail of the dataset. Reference [given data set]

```
#setting index as date
df['Date'] = pd.to_datetime(df.Date,format='%Y-%m-%d')
df.index = df['Date']


#plot
plt.figure(figsize=(16,8))
plt.plot(df['Close'], label='Close Price history')
```

```
[<matplotlib.lines.Line2D at 0x7f97172a88d0>]
```

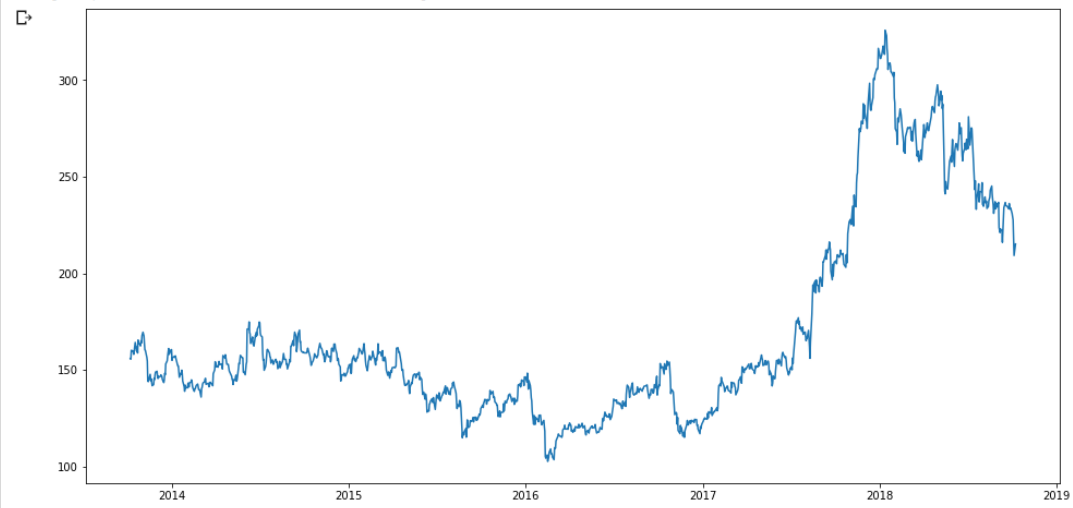


Figure 6 price history. Reference[colab link]

```
# setting the index as date
df['Date'] = pd.to_datetime(df.Date,format='%Y-%m-%d')
df.index = df['Date']

#creating dataframe with date and the target variable
data = df.sort_index(ascending=True, axis=0)
new_data = pd.DataFrame(index=range(0,len(df)),columns=['Date', 'Close'])

for i in range(0,len(data)):
     new_data['Date'][i] = data['Date'][i]
     new_data['Close'][i] = data['Close'][i]

# splitting into train and validation
train = new_data[:987]
valid = new_data[987:]

x_train.shape
(987, 1)

# making predictions
preds = []
for i in range(0,valid.shape[0]):
    a = train['Close'][len(train)-248+i:].sum() + sum(preds)
    b = a/248
    preds.append(b)

#plot
valid['Predictions'] = 0
valid['Predictions'] = preds
plt.plot(train['Close'])
plt.plot(valid[['Close', 'Predictions']])
```

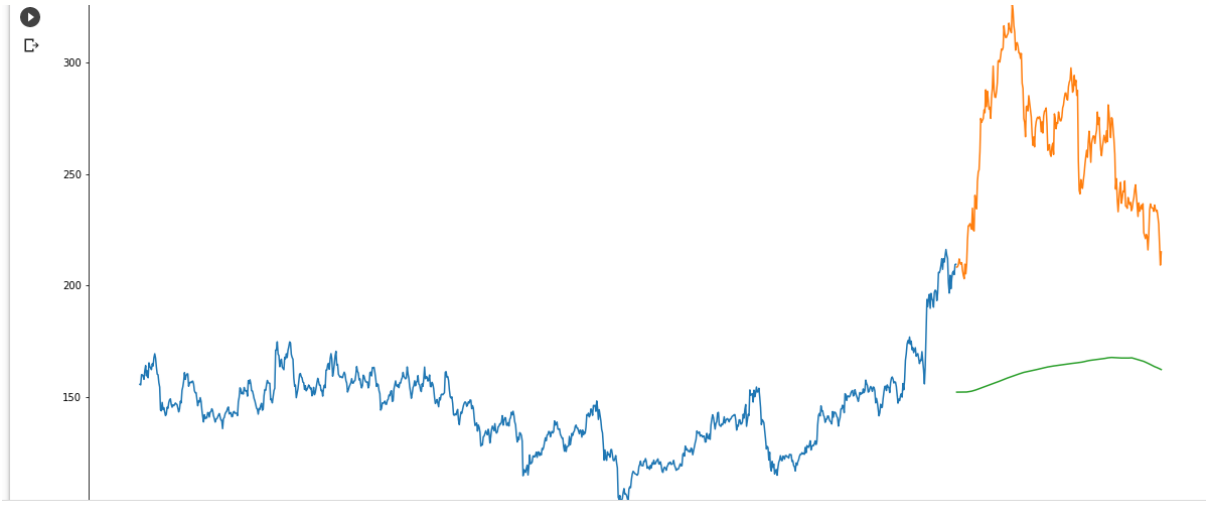


Figure 7 close prediction. Reference [colal link]

```
#setting index as date values
df['Date'] = pd.to_datetime(df.Date,format='%Y-%m-%d')
df.index = df['Date']

#sorting
data = df.sort_index(ascending=True, axis=0)

#creating a separate dataset
```

```
new_data = pd.DataFrame(index=range(0,len(df)),columns=['Date', 'Close'])


for i in range(0,len(data)):
    new_data['Date'][i] = data['Date'][i]
    new_data['Close'][i] = data['Close'][i]

# checking the results (RMSE value)
rms=np.sqrt(np.mean(np.power((np.array(valid['Close'])-preds),2)))
print('\n RMSE value on validation set:')
print(rms)

RMSE value on validation set:
104.51415465984348
```

Result:

Final predictions and accuracy is..

```
RMSE value on validation set:
104.51415465984348
```

Formula using for this machine learning:

$$minimize \frac{1}{n} \sum_{i=1}^{n} (pred_i - y_i)^2$$

Equation 2 reference [2]

$$J = \frac{1}{n} \sum_{i=1}^{n} (pred_i - y_i)^2$$

Equation 3 reference [2]

Conclusion:

Two techniques have been utilized in this paper: LSTM and Regression, on the given dataset. Both the techniques have shown an improvement in the accuracy of predictions, thereby yielding positive results. Use of recently introduced machine learning techniques in the prediction of stocks have yielded promising results and thereby marked the use of them in profitable exchange schemes.

It has led to the conclusion that it is possible to predict stock market with more accuracy and efficiency using machine learning techniques. In the future, the stock market prediction system can be further improved by utilizing a much bigger dataset than the one being utilized currently. This would help to increase the accuracy of our prediction models.

Furthermore, other models of Machine Learning could also be studied to check for the accuracy rate resulted by them

Application of machine learning

- However, with the advent of Machine Learning and its robust algorithms, the latest market analysis and Stock Market Prediction developments have started incorporating such techniques in understanding the stock market data.

- In short, Machine Learning Algorithms are being used widely by many organisations in analysing and predicting stock values. This article shall go through a simple Implementation of analysing and predicting a Popular Worldwide Online Retail Store's stock values using several Machine Learning Algorithms in Python.

- The predicted values are of the same range as the observed values in the train set (there is an increasing trend initially and then a slow decrease). In the next section, we will look at two commonly used machine learning techniques – Linear Regression and kNN, and see how they perform on our stock market data.

- We will implement a mix of machine learning algorithms to **predict** the future **stock** price of this company, starting **with** simple algorithms like averaging and linear regression, and then move on to advanced techniques like Auto ARIMA and LSTM.

References:

1 . https://towardsdatascience.com/machine-learning-techniques-applied-to-stock-price-prediction-6c1994da8001

2 . https://www.geeksforgeeks.org/ml-linear-regression/

3 . https://www.analyticsvidhya.com/blog/2018/10/predicting-stock-price-machine-learningnd-deep-learning-techniques-python/

4.https://www.researchgate.net/publication/331345883_Stock_Market_Prediction_Using_Machine_Learning/link/5cb4487a299bf120976663be/download

5 .https://data-flair.training/blogs/stock-price-prediction-machine-learning-project-in-python/