STOCK PRICE PREDICTION

A Project-II Report

Submitted in partial fulfillment of requirement of the

Degree of

BACHELOR OF TECHNOLOGY in COMPUTER SCIENCE & ENGINEERING

BY

Samarth Jain

(EN19CS3L1021)

Under the Guidance of

Dr. Kailash Chandra Bandhu

Department of Computer Science & Engineering

Faculty of Engineering

MEDI-CAPS UNIVERSITY, INDORE- 453331

May 2022

Report Approval

The project work **“Stock Price Prediction** is hereby approved as a creditable study of an engineering/computer application subject carried out and presented in a manner satisfactory to warrant its acceptance as prerequisite for the Degree for which it has been submitted.

It is to be understood that by this approval the undersigned do not endorse or approved any statement made, opinion expressed, or conclusion drawn there in; but approve the “Project Work-II” only for the purpose for which it has been submitted.

Internal Examiner

Name:

Designation:

Affiliation:

External Examiner

Name:

Designation

Affiliation

Declaration

I/We hereby declare that the project entitled **"Stock Price Prediction"** submitted in partial fulfillment for the award of the degree of Bachelor of Technology in 'Computer Science & Engineering' completed under the supervision of **Dr. Kailash Chandra Bandhu,** Professor of Department of Computer Science & Engineering, Medi-Caps University, Indore is an authentic work.

Further, I declare that the content of this Project work-II, in full or in parts, have neither been taken from any other source nor have been submitted to any other Institute or University for the award of any degree or diploma.

Samarth Jain (EN19CS3L1021)

Date 04/05/2022

<u>Certificate</u>

I/We, Prof. Vimal Bhatia (Professor IIT Indore), Dr. Kailash Chandra Bandhu(Professor Medi-caps University ,Indore) certify that the project entitled **“Stock Price Prediction”** submitted in partial fulfillment for the award of the degree of Bachelor of Technology by **Samarth Jain** is the record carried out by him under my guidance and that the work has not formed the basis of award of any other degree elsewhere.

[illegible]
04.04.2022

______________________	______________________
Dr. Kailash Chandra Bandhu	**Prof. Vimal Bhatia**
Computer Science Department	Electrical Engineering Department
Medi-Caps University, Indore	IIT Indore

____

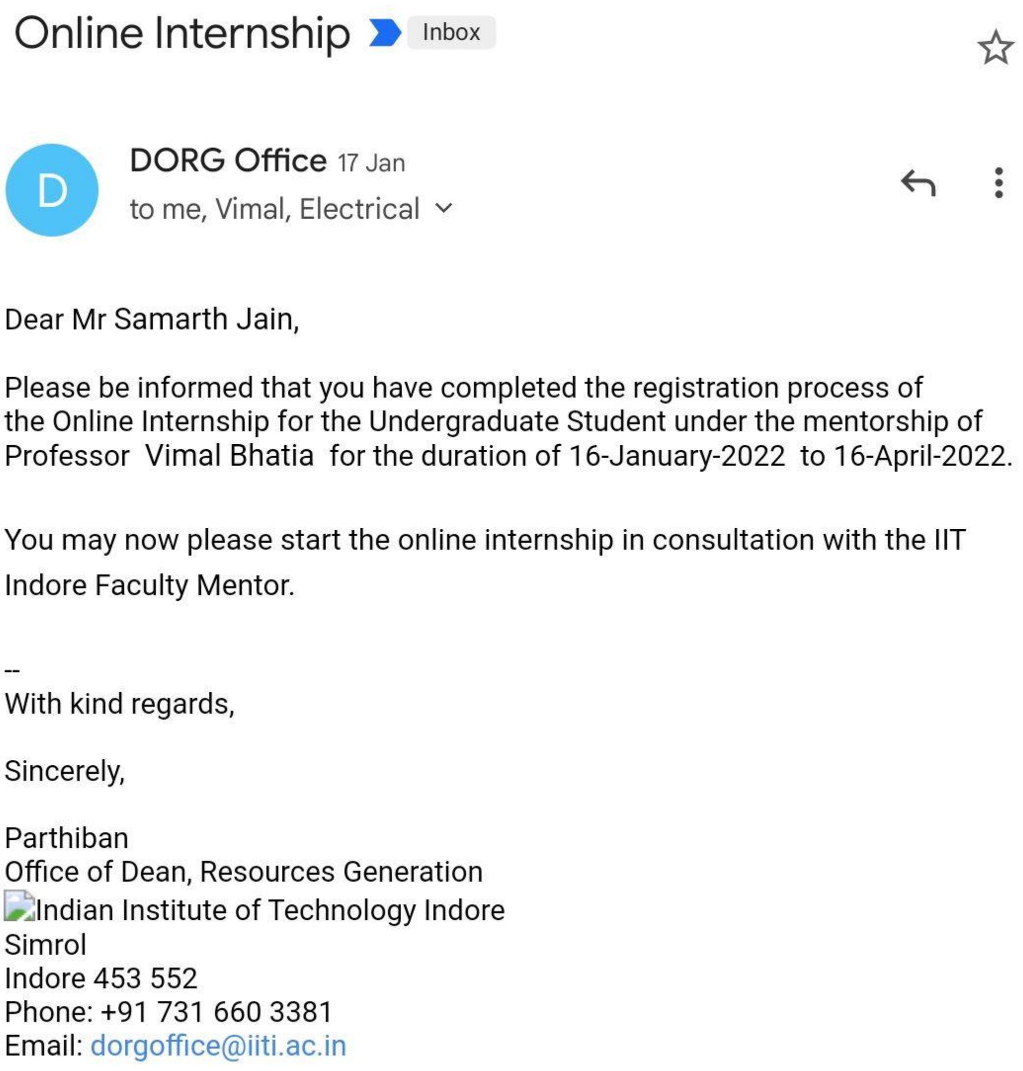
Dr. Pramod S Nair

Head of the Department

Computer Science & Engineering

Medi-Caps University, Indore

Offer Letter of the Project work-II



Online Internship Inbox

DORG Office 17 Jan

to me, Vimal, Electrical

Dear Mr Samarth Jain,

Please be informed that you have completed the registration process of the Online Internship for the Undergraduate Student under the mentorship of Professor Vimal Bhatia for the duration of 16-January-2022 to 16-April-2022.

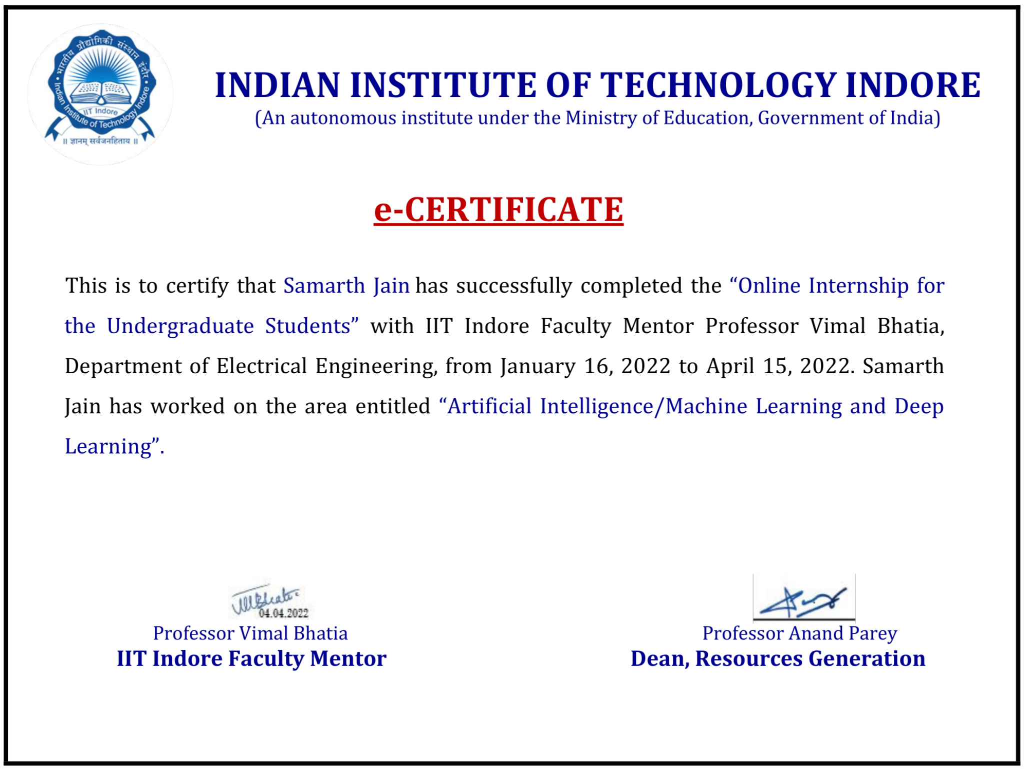
You may now please start the online internship in consultation with the IIT Indore Faculty Mentor.

--
With kind regards,

Sincerely,

Parthiban
Office of Dean, Resources Generation
Indian Institute of Technology Indore
Simrol
Indore 453 552
Phone: +91 731 660 3381
Email: dorgoffice@iiti.ac.in

Completion Certificate/Letter



INDIAN INSTITUTE OF TECHNOLOGY INDORE

(An autonomous institute under the Ministry of Education, Government of India)

e-CERTIFICATE

This is to certify that Samarth Jain has successfully completed the "Online Internship for the Undergraduate Students" with IIT Indore Faculty Mentor Professor Vimal Bhatia, Department of Electrical Engineering, from January 16, 2022 to April 15, 2022. Samarth Jain has worked on the area entitled "Artificial Intelligence/Machine Learning and Deep Learning".

[illegible] 04.04.2022

Professor Vimal Bhatia
IIT Indore Faculty Mentor

[illegible]

Professor Anand Parey
Dean, Resources Generation

Acknowledgements

I would like to express my deepest gratitude to Honorable Chancellor, **Shri R C Mittal,** who has provided me with every facility to successfully carry out this project, and my profound indebtedness to **Prof. (Dr.) Dilip K. Patnaik,** Vice Chancellor, Medi-Caps University, whose unfailing support, and enthusiasm has always boosted up my morale. I also thank **Prof. (Dr.) Suresh Jain, Dean**, Faculty of Engineering, Medi-Caps University, for giving me a chance to work on this project. I would also like to thank my Head of the Department **Dr. Pramod S Nair** for his continuous encouragement for betterment of the project.

I express my heartfelt gratitude to my external Guide **Prof. Vimal Bhatia** Professor of Department in Electrical Engineering, IIT Indore as well as my Internal Guide, **Dr. Kailash Chandra Bandhu**, Professor of Department in Computer Science Engineering, MU, without whose continuous help and support, this project would ever have reached to the completion.

It is their help and support, due to which we became able to complete the design and technical report.

Without their support this report would not have been possible.

Samarth Jain (EN19CS3L1021)
B.Tech. IV Year
Department of Computer Science & Engineering
Medi-Caps University, Indore

Abstract

In Stock Market Prediction, the aim is to predict the future value of the financial stocks of a company. The recent trend in stock market prediction technologies is the use of machine learning which makes predictions based on the values of current stock market indices by training on their previous values. Machine learning itself employs different models to make prediction easier and authentic. We will be making use of Regression and LSTM based Machine learning to predict stock values. Factors considered are open, close, low, high and volume.

Keywords: Stock, Prediction, Finance, Machine Learning, LSTM.

Table Of Contents

List of Figures

Abbreviations

Abbr.	Meaning
AI	Artificial Intelligence
ML	Machine Learning
DL	Deep Learning
RL	Reinforcement Learning
LSTM	Long Short Term Memory
RNN	Recurrent Neural Network
ANN	Artificial Neural Network
MAE	Mean Average Error
MSE	Mean Square Error
SVM	Support Vector Machine

1.1 Introduction to AI/ML/Deep learning

Artificial intelligence (AI)

At its most basic level, the field of artificial intelligence uses computer science and data to enable problem solving in machines. While we don't yet have human-like robots trying to take over the world, we do have examples of AI all around us. These could be as simple as a computer program that can play chess, or as complex as an algorithm that can predict the RNA structure of a virus to help develop vaccines.

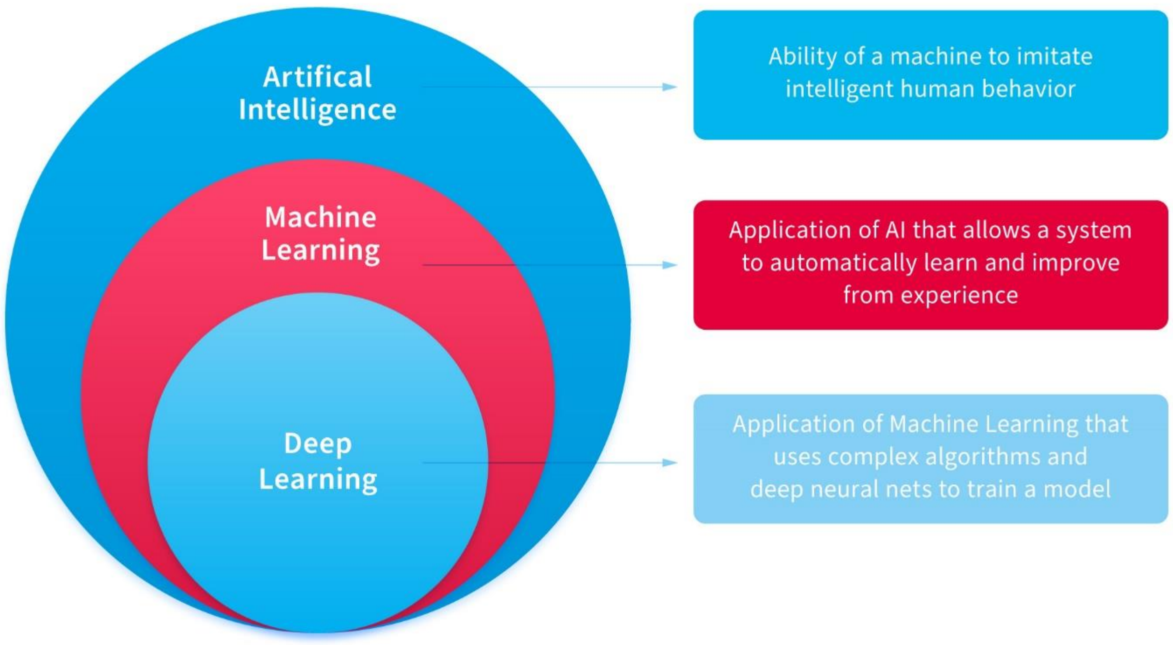

Artifical Intelligence
Ability of a machine to imitate intelligent human behavior
Machine Learning
Application of AI that allows a system to automatically learn and improve from experience
Deep Learning
Application of Machine Learning that uses complex algorithms and deep neural nets to train a model

Fig. 01 Artificial Intelligence

Machine learning

It is an application of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed. Machine learning focuses on the development of computer programs that can access data and use it learn for themselves.

❖ Features of Machine Learning

✓ Machine Learning is computing-intensive and generally requires a large amount of training data.

✓ It involves repetitive training to improve the learning and decision making of algorithms.

✓ As more data gets added, Machine Learning training can be automated for learning new data patterns and adapting its algorithm.

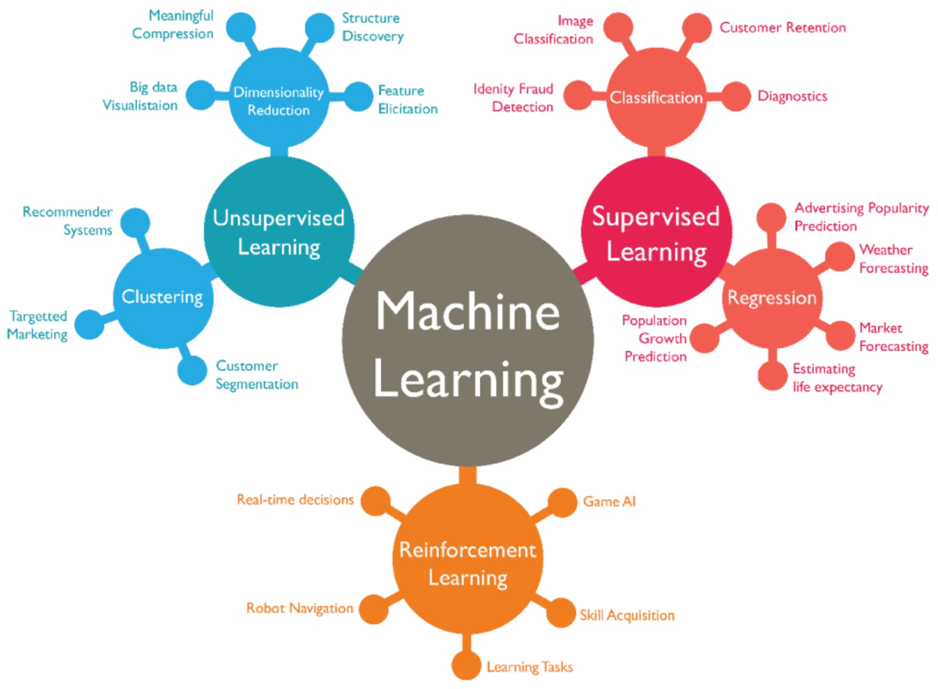


Fig. 02 Machine Learning

Deep learning

Machine learning and deep learning are both types of AI. In short, machine learning is AI that can automatically adapt with minimal human interference. Deep learning is a subset of machine learning that uses artificial neural networks to mimic the learning process of the human brain.

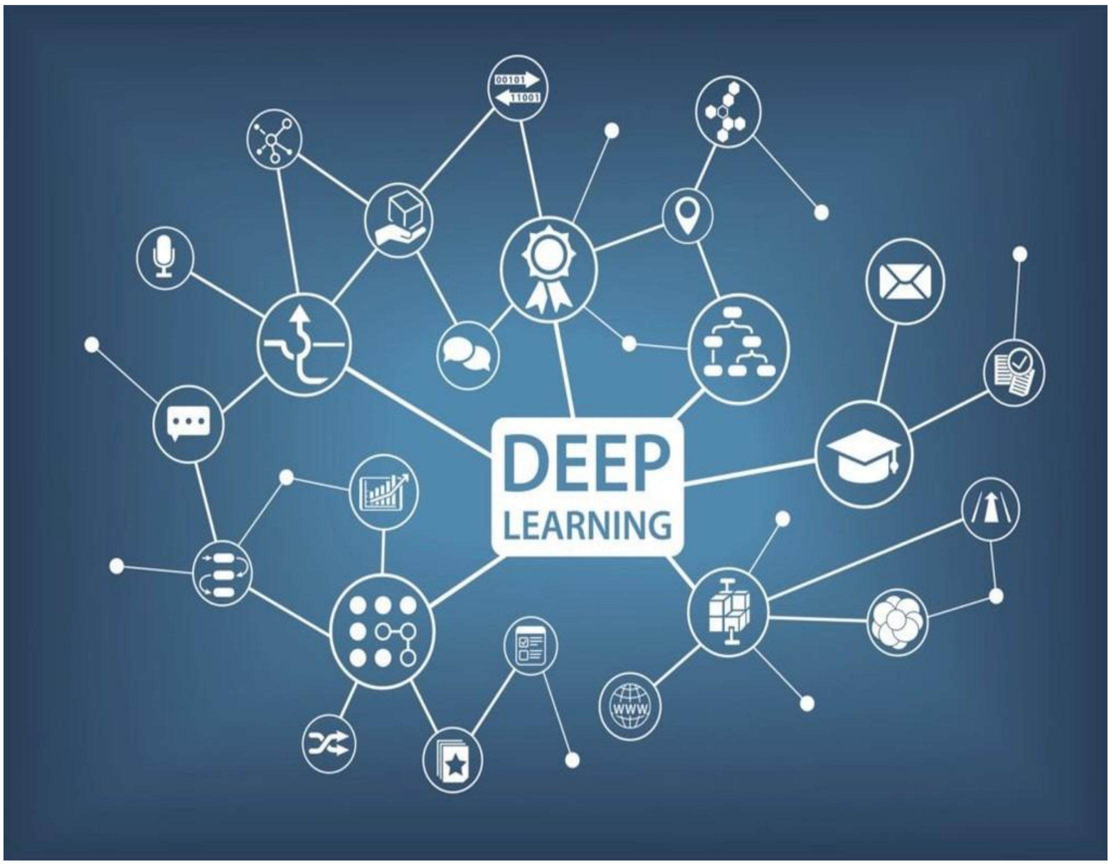


Fig. 03 Deep Learning

1.2 Objectives of Project

The Objective of this project is to train the algorithm and predict the price of stock and display by the graph. For example, it will show the spike up if the price is going to increase and down for the decrease in it in the graph.

1.3 Significance

Stock market prediction aims to determine the future movement of the stock value of a financial exchange. The accurate prediction of share price movement will lead to more profit investors can make.

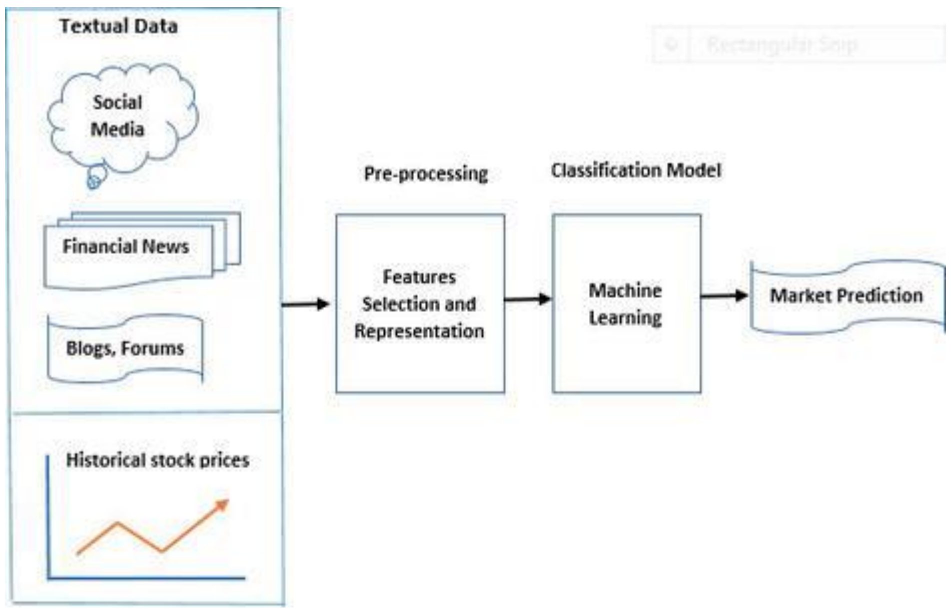


Fig. 04 Prediction Model

1.4 Chapter Scheme

Techniques of Machine Learning-

The types of machine learning algorithms differ in their approach, the type of data they input and output, and the type of task or problem that they are intended to solve. Broadly Machine Learning can be categorized into four categories.

Supervised Learning-

Supervised learning is the machine learning task of learning a function that maps an input to an output based on example input-output pairs. It infers a function from labelled training data consisting of a set of training examples.

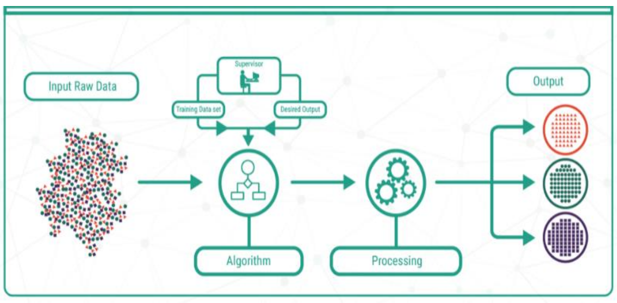


Fig. 05 Supervised Learning

In supervised learning, each example is a pair consisting of an input object (typically a vector) and a desired output value (also called the supervisory signal).

Supervised Learning Flow

✓ Data Preparation

- Clean data
- Label data (x, y)
- Feature Engineering

- Reserve 80% of data for Training (Train_X) and 20% for Evaluation(Train_E)

✓ Training Step

- Design algorithmic logic
- Train the model with Train X
- Derive the relationship between x and y, that is, y = f(x)

✓ Evaluation or Test Step

- Evaluate or test with Train E
- If accuracy score is high, you have the final learned algorithm y = f(x)
- If accuracy score is low, go back to training step

✓ Production Deployment

- Use the learned algorithm y = f(x) to predict production data.
- The algorithm can be improved by more training data, capacity, or algo redesign.

Testing the Algorithms

✓ Once the algorithm is trained, test it with test data (a set of data instances that do not appear in the training set).

✓ A well-trained algorithm can predict well for new test data

Unsupervised Learning-

Unsupervised learning is the training of machine using information that is neither classified nor labelled and allowing the algorithm to act on that information without guidance.

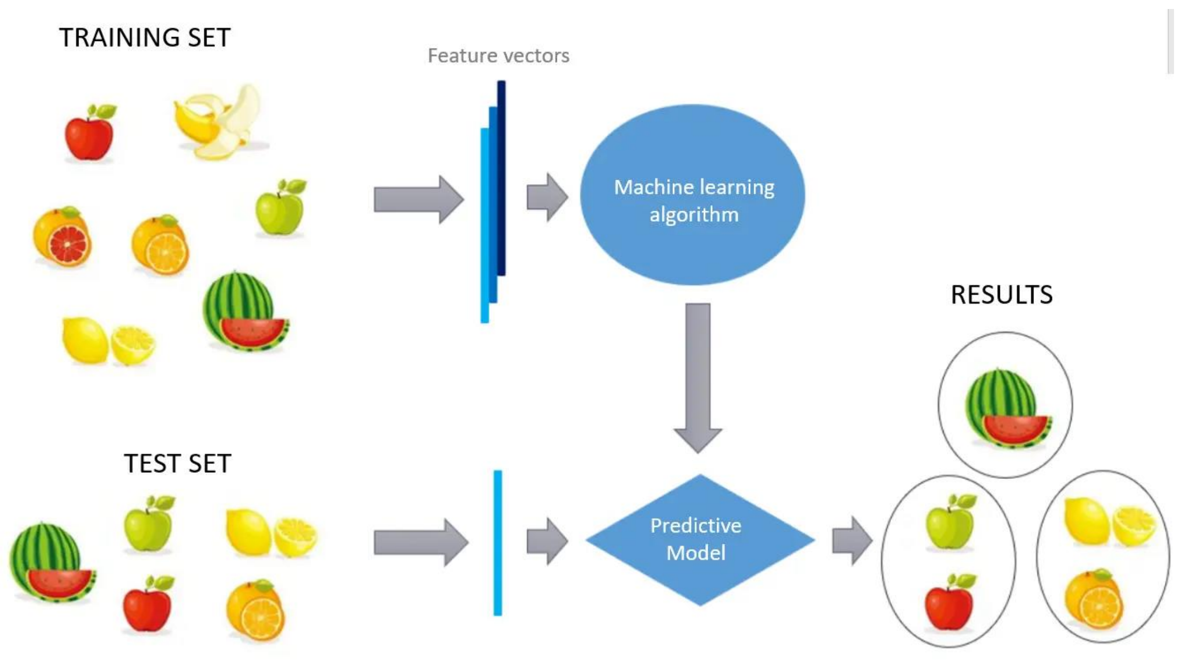


Fig.06 Unsupervised Learning

Reinforcement Learning (RL)-

Reinforcement learning enables a machine to interact with an environment. A simple example is repeatedly playing a video game, and providing a reward whenthe algorithm takes the desired action. By repeating the process thousands or millions of times, the machine can eventually learn from its experience.

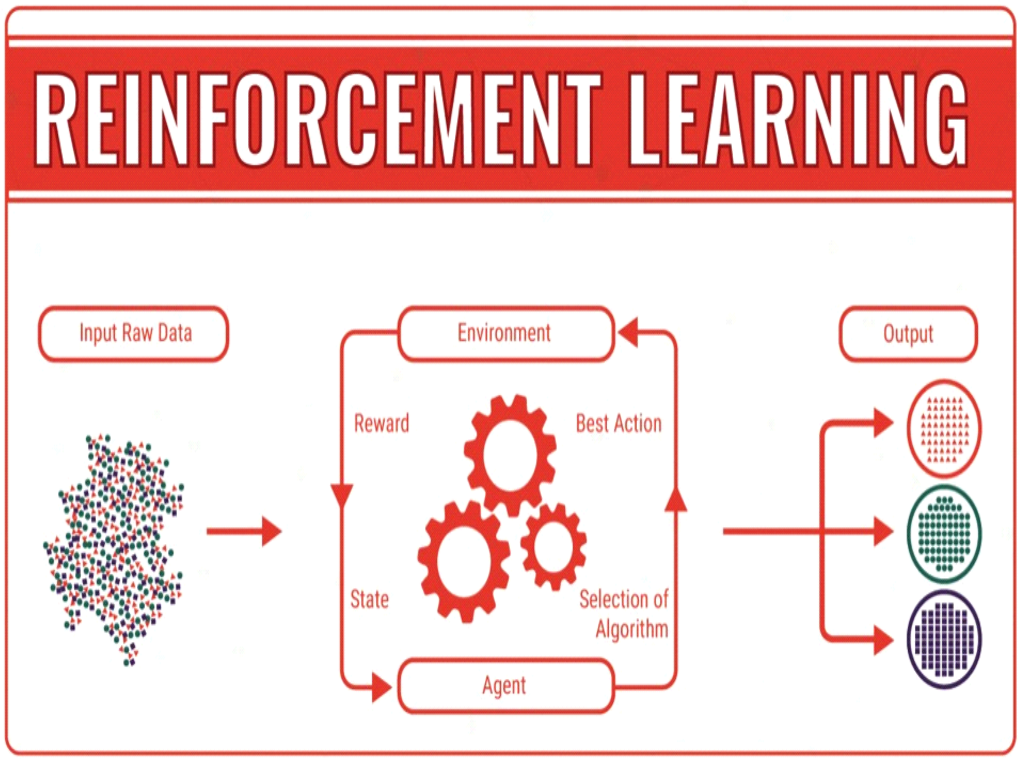


Fig. 07 Reinforcement Learning

Regression-

In statistical modeling, regression analysis is a set of statistical processes forestimating the relationships among variables.

It includes many techniques for modeling and analyzing several variables, whenthe focus is on the relationship between a dependent variable and one or more independent variables (or 'predictors').

More specifically, regression analysis helps one understand how the typical value of the dependent variable (or 'criterion variable') changes when any one of the independent variables is varied, while the other independent variables are heldfixed.

- **Linear Regression**

Linear regression is a linear approach for modeling the relationship between ascalar dependent variable y and an independent variable x. where x, y, w are vectors of real numbers and w is a vector of weightparameters.

The equation is also written as: $y = wx + b$

where b is the bias or the value of output for zero input.

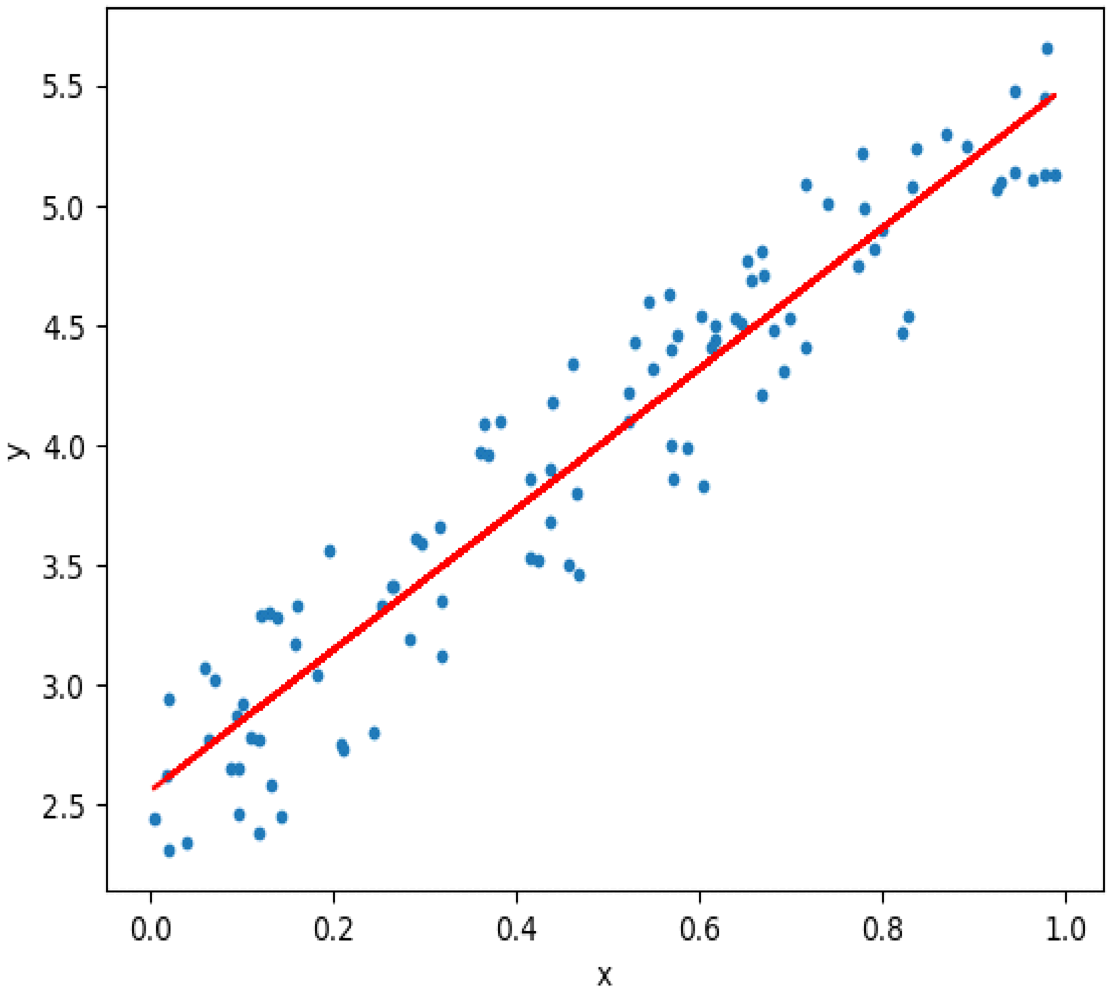


Fig.08 Linear Regression

- **Multiple Linear Regression**

It is a statistical technique used to predict the outcome of a response variable through several explanatory variables and model the relationships between them. The goal of multiple linear regression is to model the linear relationship between the explanatory (independent) variables and response (dependent) variables. In essence, multiple regression is the extension of ordinary least-squares (OLS) regression because it involves more than one explanatory variable.

It represents line fitment between multiple inputs and output, typically:

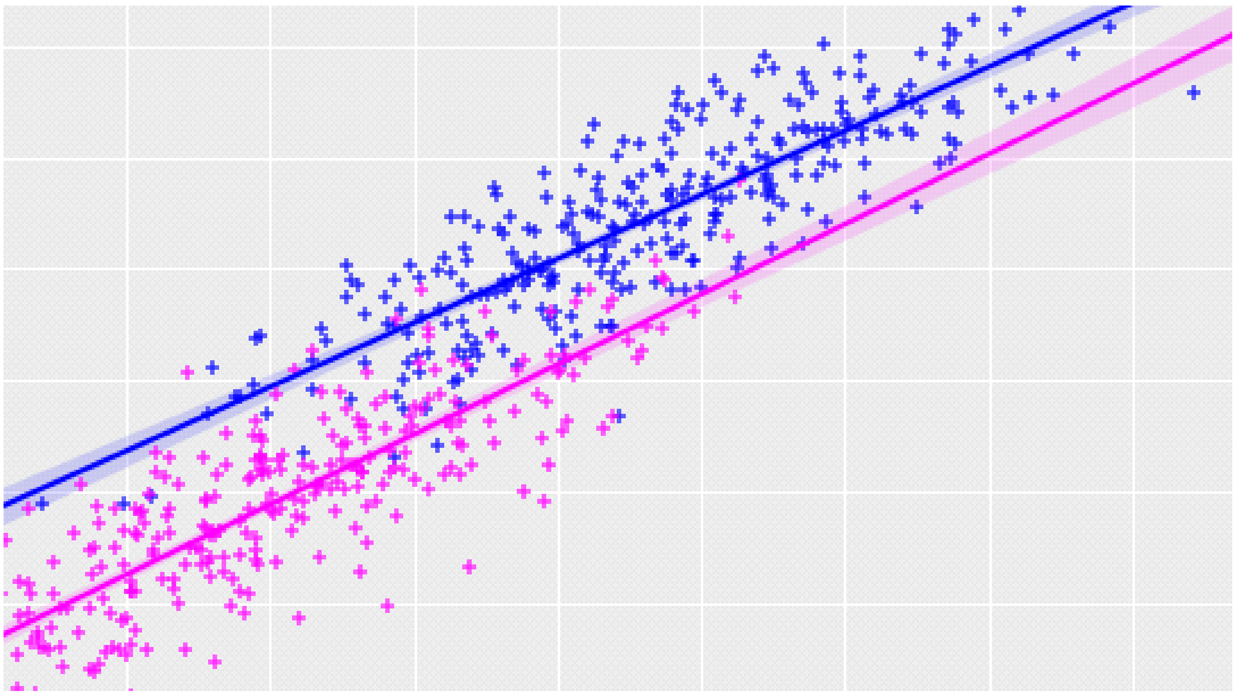


Fig.09 Multiple Linear Regression

- **Logistic Regression**

Logistic regression is a supervised classification is unique Machine Learning algorithms in Python that finds its use in estimating discrete values like 0/1, yes/no,and true/false. This is based on a given set of independent variables. We use a logistic function to predict the probability of an event and this gives us an output between 0 and 1. Although it says ‘regression’, this is actually a classification algorithm. Logistic regression fits data into a logit function and is also called logistic regression.

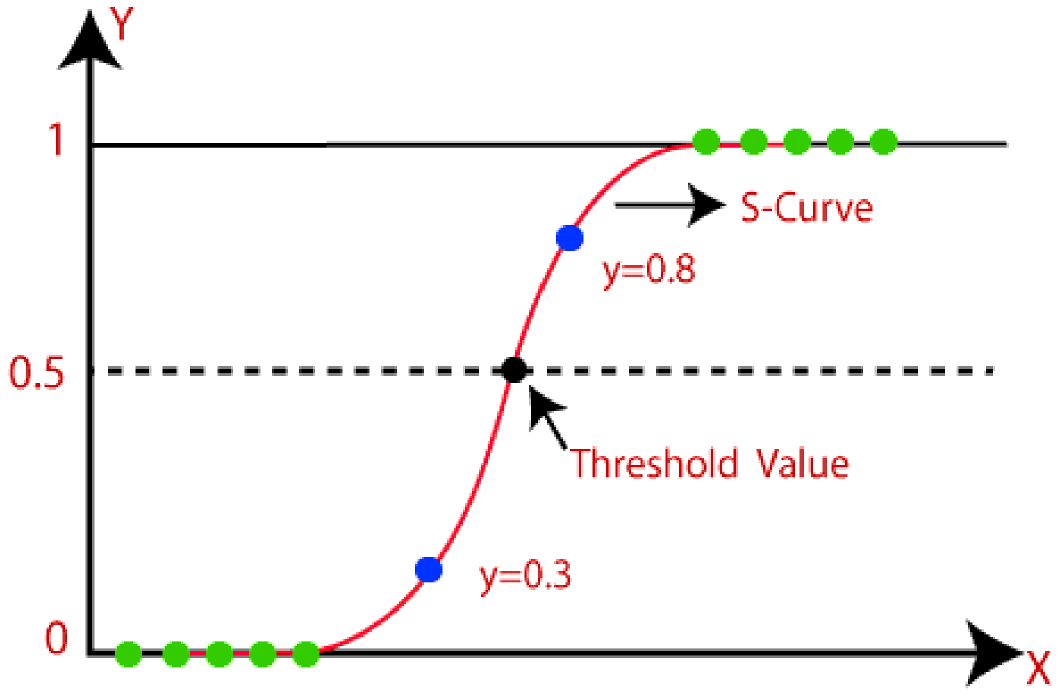


Fig.10 Logistic Regression

- **Decision Tree**

A decision tree falls under supervised Machine Learning Algorithms in Python andcomes of use for both classification and regression- although mostly for classification. This model takes an instance, traverses the tree, and compares important features with a determined conditional statement.

Whether it descends to the left child branch or the right depends on the result. Usually, more important features are closer to the root. Decision Tree, a Machine Learning algorithm in Python can work on both categorical and continuous dependent variables. Here, we split a population into two or more homogeneous sets. Tree models where the target variable can take a discrete set of values are called classification trees; in these tree structures, leaves represent class labels and branches represent conjunctions of features that lead to those class labels. Decisiontrees where the target variable can take continuous values (typically real numbers) are called regression trees.

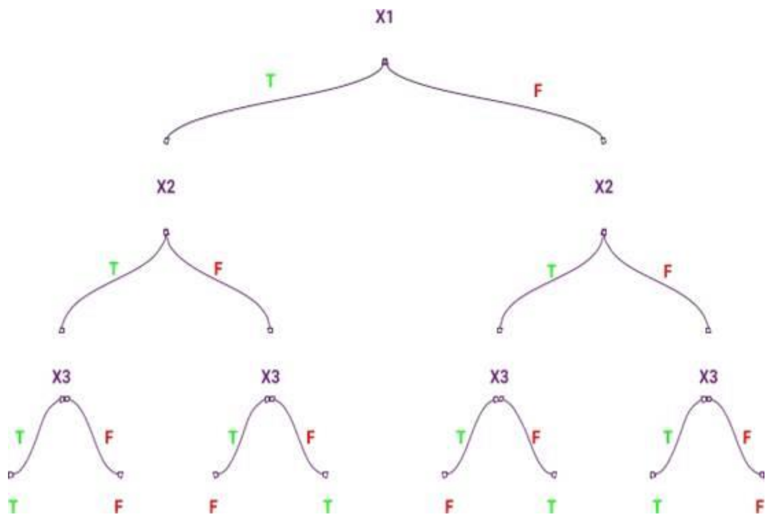


Fig.11 Decision Tree

• Support Vector Machine (SVM)

SVM is a supervised classification is one of the most important Machines Learning algorithms in Python, that plots a line that divides different categories of your data. In this ML algorithm, we calculate the vector to optimize the line. This is to ensure that the closest point in each group lies farthest from each other. While you will almost always find this to be a linear vector, it can be other than that. An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. In addition to performing linear classification, SVMs can efficiently perform a non-linear classification using what is called the kernel trick, implicitly mapping their inputs into high-dimensional feature spaces. When data are unlabeled, supervised learning is not possible, and an unsupervised learning approach is required, which attempts to find natural clustering of the data to groups, and then map new data to these formed groups.

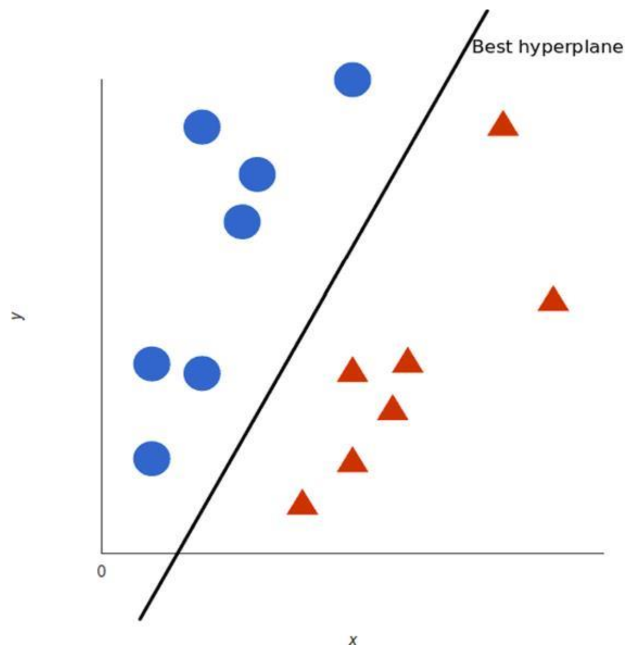


Fig.12 SVM

- **K-Means Algorithm**

k-Means is an unsupervised algorithm that solves the problem of clustering. It classifies data using a number of clusters. The data points inside a class are homogeneous and heterogeneous to peer groups. k-means clustering is a method ofvector quantization, originally from signal processing, that is popular for cluster analysis in data mining. k-means clustering aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster. k-means clustering is rather easy to apply to even large data sets, particularly when using heuristics such as Lloyd's algorithm. It often is used as a pre-processing step for other algorithms, for example to find a starting configuration. The problem is computationally difficult (NP-hard). k- means originates from signal processing, and still finds use in this domain. In cluster analysis, the k-means algorithm can be used to partition the input data set into k partitions (clusters). k-means clustering has been used as a feature learning (or dictionary learning) step, in either (semi-)supervised learning or unsupervised learning.

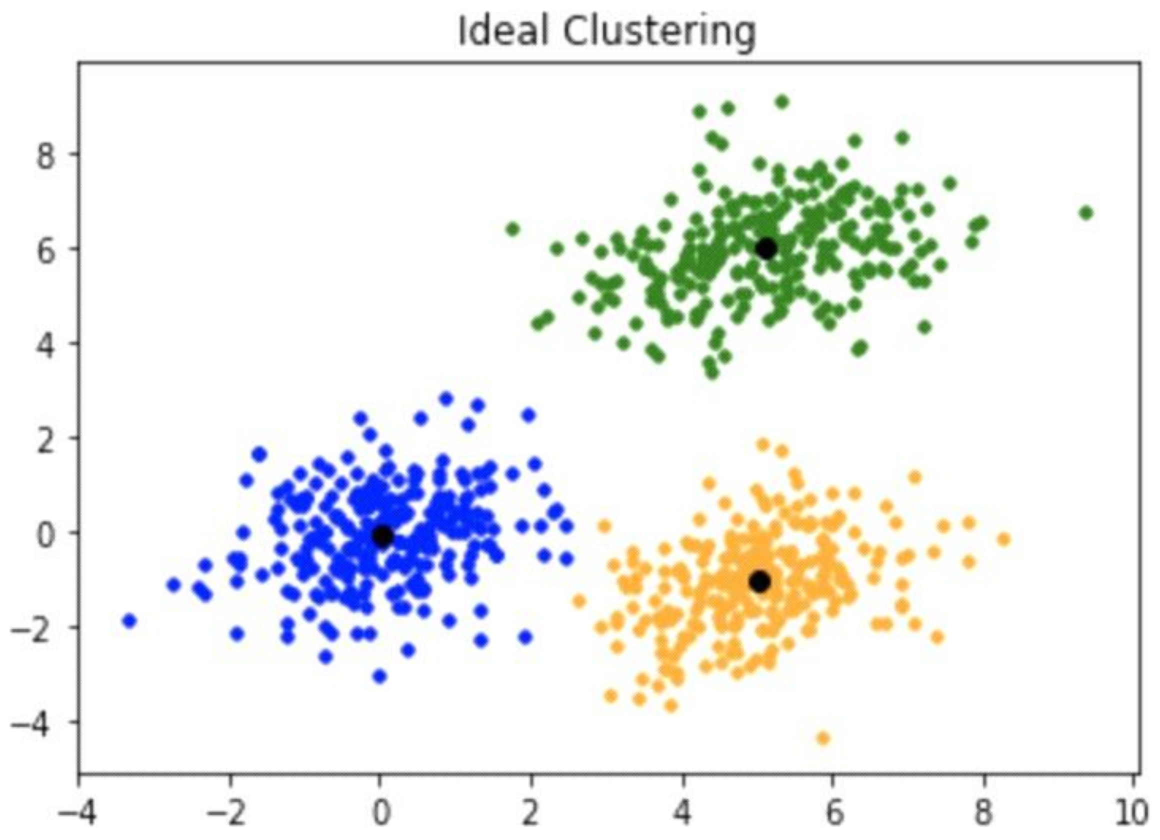


Fig.13 K-Means

- **Random Forest**

A random forest is an ensemble of decision trees. In order to classify every newobject based on its attributes, trees vote for class- each tree provides a classification. The classification
14 with the most votes wins in the forest. Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time andoutputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees.

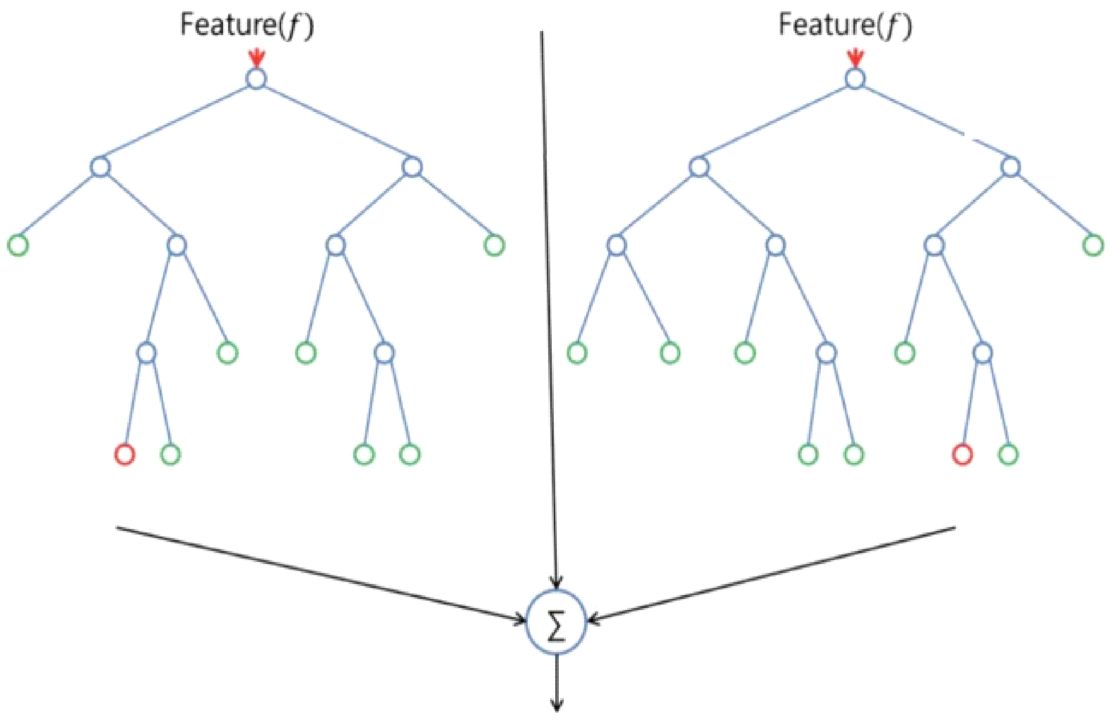


Fig. 14 Random Forest

Deep Learning

Deep Learning is a specialized form of Machine Learning that uses supervised,unsupervised, or semi-supervised learning to learn data representations.

It is similar to the structure and function of the human nervous system.

In deep learning, a computer model learns to perform classification tasks directly from images, text, or sound. Deep learning models can achieve state-of-the-art accuracy, sometimes exceeding human-level performance. Models are trained by using a large set of labeled data and neural network architectures that contain manylayers.

OR

Practically, **Deep Learning** is a subset of **Machine Learning that achieves great power and flexibility by learning to represent the world as nested hierarchy of concepts, with each concept defined in relation to simpler concepts, and more abstract representations computed in terms of less abstract ones.**

Deep Learning requires high-end machines contrary to traditional **Machine Learning** algorithms. GPU has become a integral part now to execute any **Deep Learning** algorithm.

Why Deep Learning

The vast availability of Big Data enables machines to be trained.

Experts havediscovered multi- layered learning networks that can be leveraged for deep learning as they learn in layers.

Scientists have figured out that high-performing graphics processing units (GPU)can be used for deep learning.

Artificial Neural Networks

The term "**Artificial Neural Network**" is derived from Biological neural networks that develop the structure of a human brain. Similar to the human brain that has neurons interconnected to one another, artificial neural networks also have neurons that are interconnected to one another in various layers of the networks. These neurons are known as nodes.

- Deep learning relies on multiple layers of training.

- Artificial Neural Network is a computing system made up of a number of simple,highly interconnected processing elements which process information by their dynamic state response to external inputs.

- It is an interconnected group of nodes akin to the vast network of layers ofneurons in a brain.

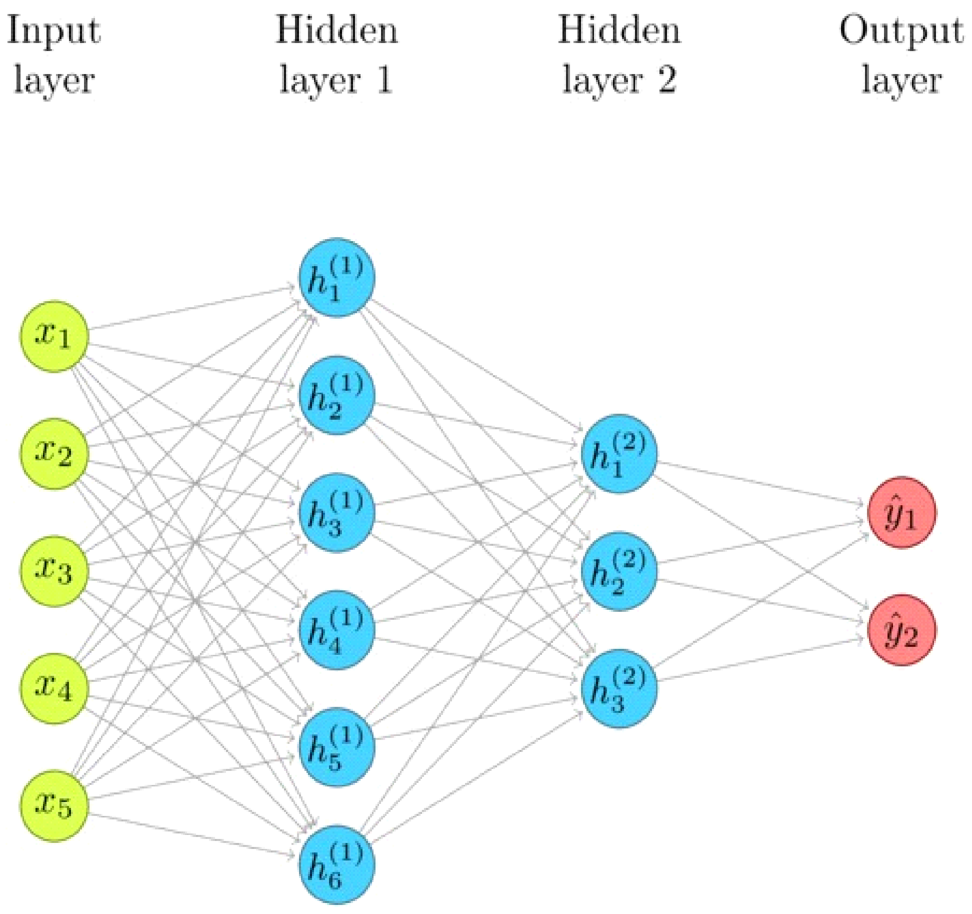

Input layer
Hidden layer 1
Hidden layer 2
Output layer
x_1
x_2
x_3
x_4
x_5
$h_1^{(1)}$
$h_2^{(1)}$
$h_3^{(1)}$
$h_4^{(1)}$
$h_5^{(1)}$
$h_6^{(1)}$
$h_1^{(2)}$
$h_2^{(2)}$
$h_3^{(2)}$
$\hat{y}_1$
$\hat{y}_2$

Fig.15 ANN

2.1 Experimental Set-up

In Machine learning we need data to train (learn) models, and to test how good the models are. The training data (call it set A) needs to be different (disjoint) from the test data (call it set B). Otherwise, we would be testing the learned model on data it had previously seen, and we would get a biased estimate of the model's generalized performance.

Most machine learning algorithms require choosing parameter values; very often this is done by setting aside some of the training data to evaluate the quality of different parameter settings. All together, we usually end up splitting our data into 3 disjoint sets:

- A1: training set = used for learning model
- A2: validation set = used for exploring and picking best parameters
- B: test set = used for testing model generalization perf, testing hypotheses

2.2 Procedures Adopted

Data Pre-processing

Machine Learning algorithms don't work so well with processing raw data. Before we can feed such data to an ML algorithm, we must pre-process it. We must apply some transformations on it. With data pre-processing, we convert raw data into a clean data set. To perform data this, there are 7 techniques –

1. Rescaling Data

   For data with attributes of varying scales, we can rescale attributes to possess the same scale. We rescale attributes into the range 0 to 1 and call it normalization. We use the MinMaxScaler class from sci-kit learn. This gives us values between 0 and 1.

2. Standardizing Data

   With standardizing, we can take attributes with a Gaussian distribution and different means and standard deviations and transform them into a standard Gaussian distribution with a mean of 0 and a standard deviation of 1.

3. Normalizing Data

   In this task, we rescale each observation to a length of 1 (a unit norm). For this, we use the Normalizer class.

4. Binarizing Data

Using a binary threshold, it is possible to transform our data by marking the values above it 1 and those equal to or below it, 0. For this purpose, we use the Binarize class.

5. Mean Removal

We can remove the mean from each feature to centre it on zero.

6. One Hot Encoding

When dealing with few and scattered numerical values, we may not need to store these. Then, we can perform One Hot Encoding. For k distinct values, we can transform the feature into a k-dimensional vector with one value of 1 and 0 as the rest values.

7. Label Encoding

Some labels can be words or numbers. Usually, training data is labelled with words to make it readable. Label encoding converts word labels into numbers to let algorithms work on them.

Algorithm

1. **LSTM**
2. **SEQUENTIAL**
3. **DENSE**
4. **DROPOUT**

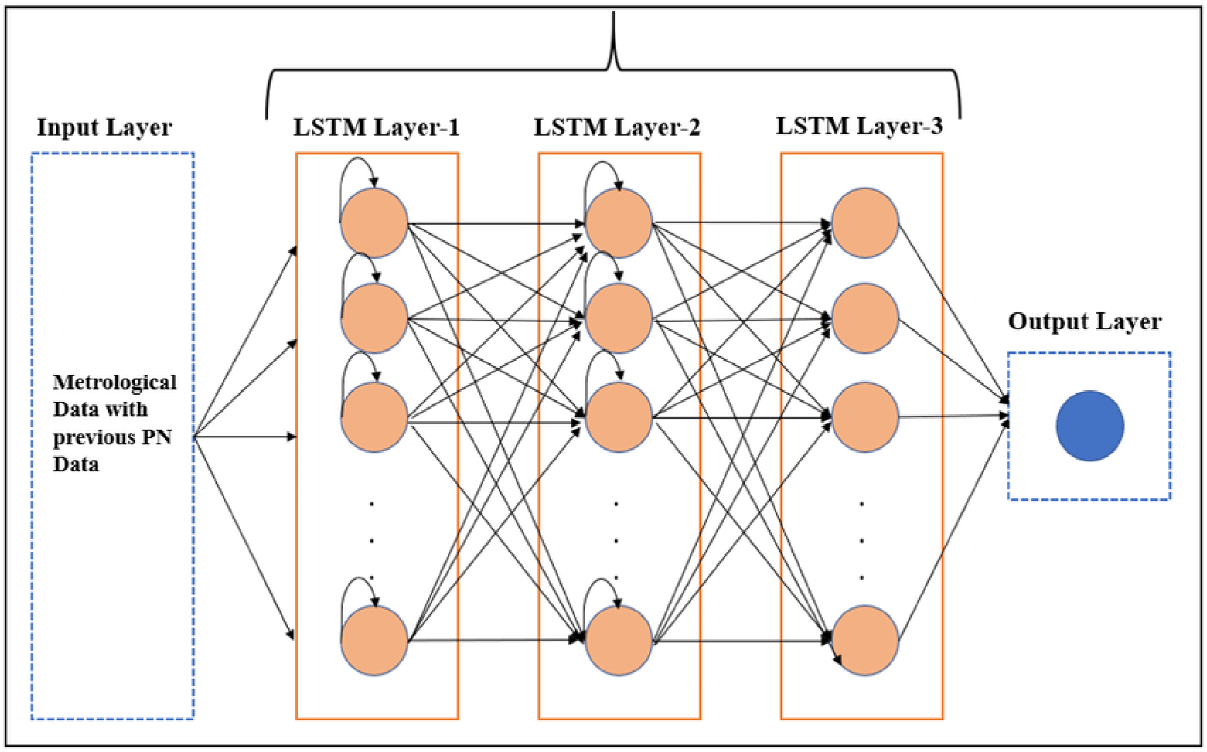


Fig.16 LSTM and Sequential

LSTM-

1. Long Short-Term Memory (LSTM) networks are a type of recurrent neural network capable of learning order dependence in sequence prediction problems.
2. This is a behavior required in complex problem domains like machine translation, speech recognition, and more.
3. LSTMs are a complex area of deep learning. It can be hard to get your hands around what LSTMs are, and how terms like bidirectional and sequence-to-sequence relate to the field.

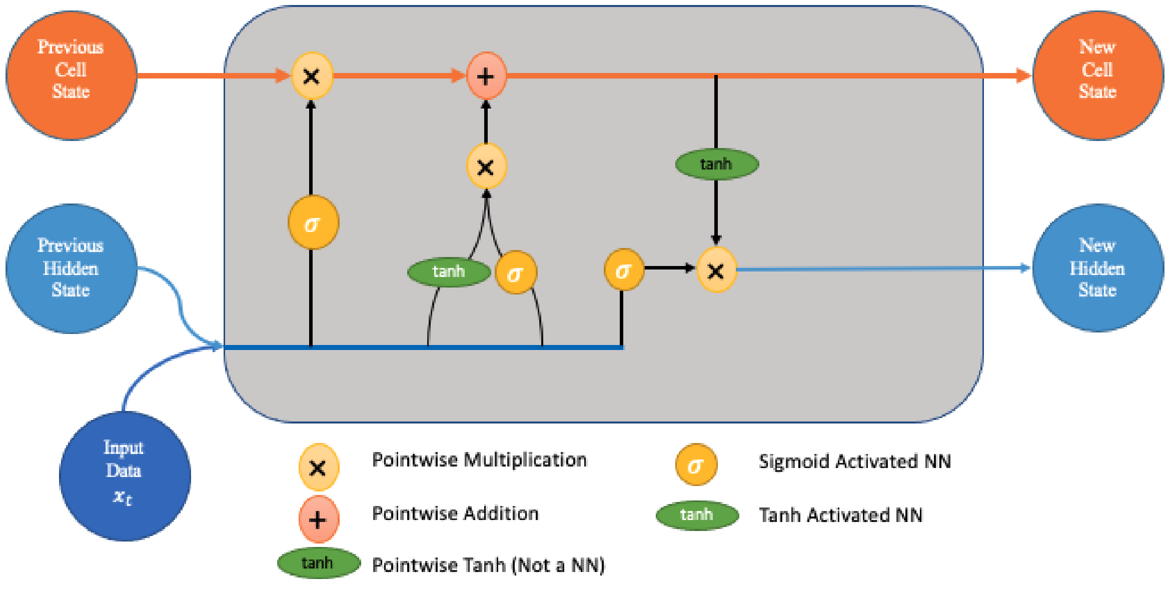


Fig.17 LSTM

Sequential-

- Machine learning models that input or output data sequences are known as sequence models. Text streams, audio clips, video clips, time-series data, and other types of sequential data are examples of sequential data. Recurrent Neural Networks (RNNs) are a well-known method in sequence models.

- The analysis of sequential data such as text sentences, time-series, and other discrete sequence data prompted the development of Sequence Models. These models are better suited to handle sequential data, whereas Convolutional Neural Networks are better suited to treat spatial data.

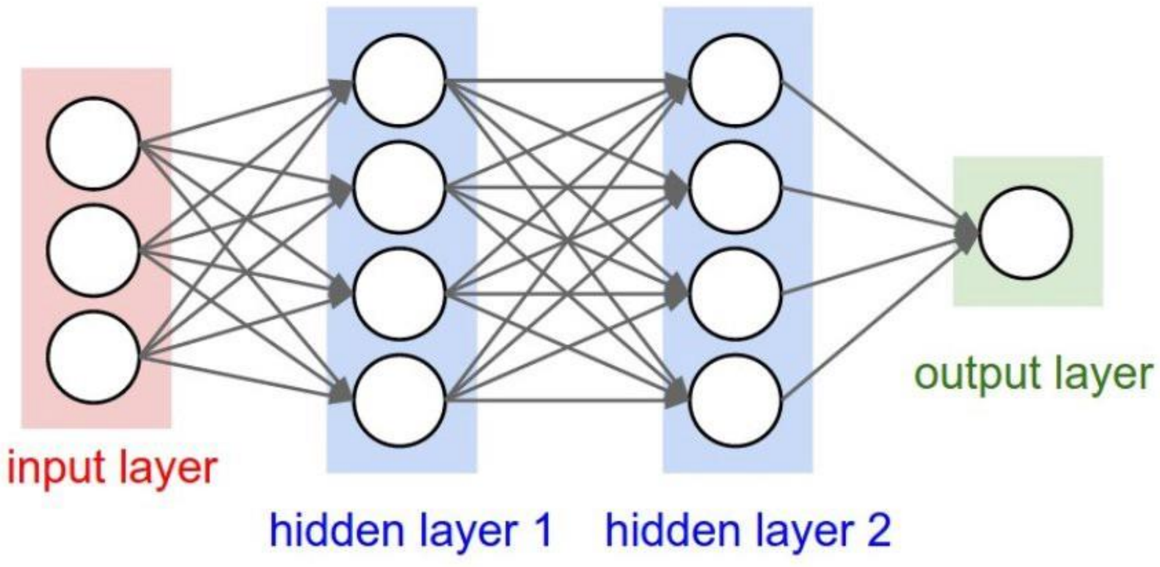


Fig.18 Sequential

DropOut-

- Dropout refers to ignoring units (i.e. neurons) during the training phase of certain set of neurons which is chosen at random. By “ignoring”, I mean these units are not considered during a particular forward or backward pass.
- More technically, At each training stage, individual nodes are either dropped out of the net with probability *1-p* or kept with probability *p*, so that a reduced network is left; incoming and outgoing edges to a dropped-out node are also removed.

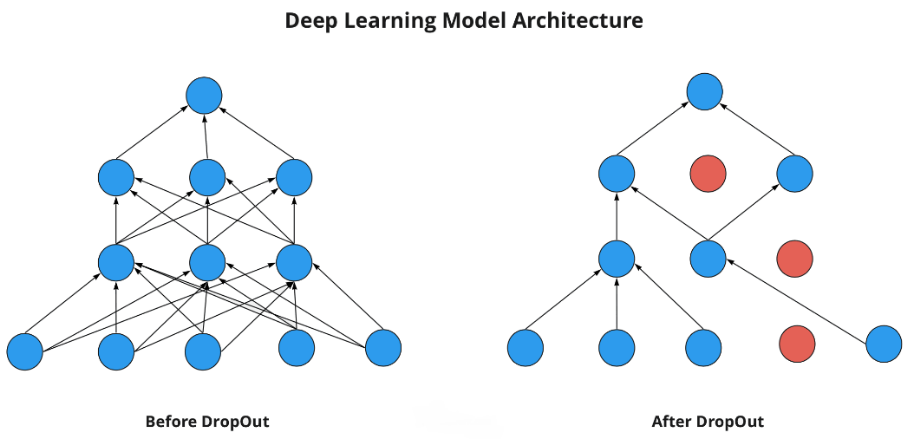


Fig.19 DropOut

Project Report

Problem Statement – 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

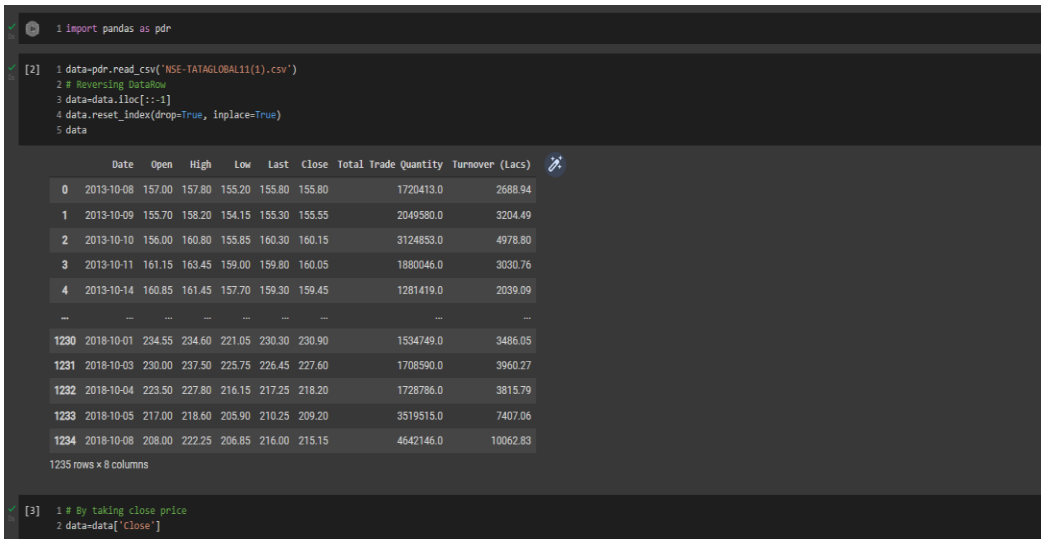
2. Use previous equity data of Day open, close, low, high for prediction.

3.Any other stock Relative data

Solution- We will use the Long Short-Term Memory (LSTM) method to create a Machine Learning model to forecast stock values. They are used to make minor changes to the information by multiplying and adding. Long Short-term memory (LSTM) is deep learning artificial RNN architecture. It can handle single data points (such as pictures) as well as full data sequences.

Code

Reading the dataset



```
import pandas as pdr
```

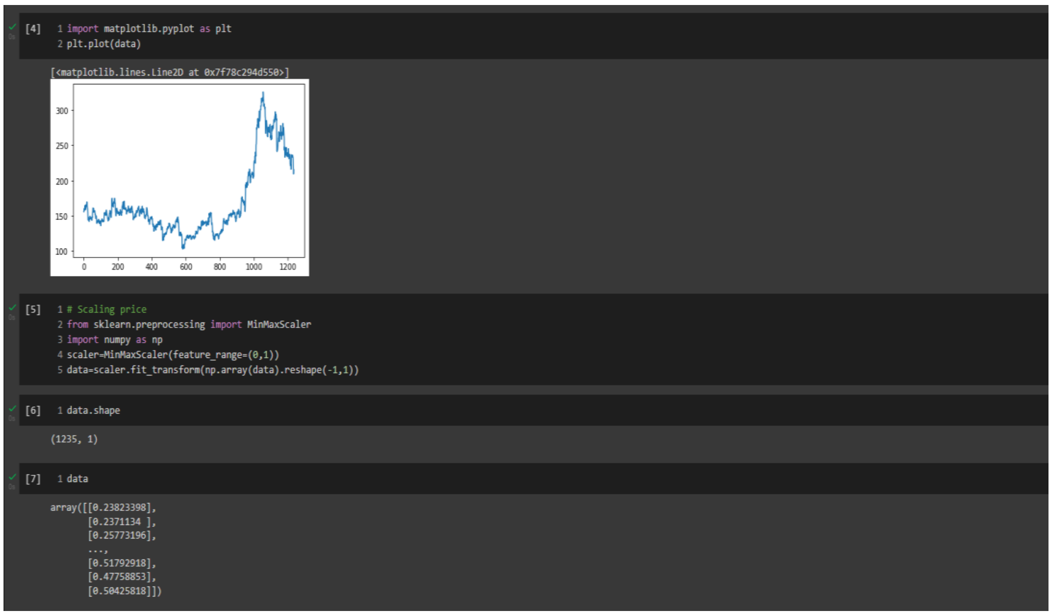
[2]
```
data=pdr.read_csv('NSE-TATAGLOBAL11(1).csv')
# Reversing DataRow
data=data.iloc[::-1]
data.reset_index(drop=True, inplace=True)
data
```

	Date	Open	High	Low	Last	Close	Total Trade Quantity	Turnover (Lacs)
0	2013-10-08	157.00	157.80	155.20	155.80	155.80	1720413.0	2688.94
1	2013-10-09	155.70	158.20	154.15	155.30	155.55	2049580.0	3204.49
2	2013-10-10	156.00	160.80	155.85	160.30	160.15	3124853.0	4978.80
3	2013-10-11	161.15	163.45	159.00	159.80	160.05	1880046.0	3030.76
4	2013-10-14	160.85	161.45	157.70	159.30	159.45	1281419.0	2039.09
...	...	...	...	...	...	...	...	...
1230	2018-10-01	234.55	234.60	221.05	230.30	230.90	1534749.0	3486.05
1231	2018-10-03	230.00	237.50	225.75	226.45	227.60	1708590.0	3960.27
1232	2018-10-04	223.50	227.80	216.15	217.25	218.20	1728786.0	3815.79
1233	2018-10-05	217.00	218.60	205.90	210.25	209.20	3519515.0	7407.06
1234	2018-10-08	208.00	222.25	206.85	216.00	215.15	4642146.0	10062.83

1235 rows × 8 columns

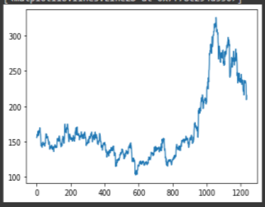
[3]
```
# By taking close price
data=data['Close']
```

Plotting the data



[4]
```
import matplotlib.pyplot as plt
plt.plot(data)
```

```
[<matplotlib.lines.Line2D at 0x7f78c294d550>]
```



[5]
```
# Scaling price
from sklearn.preprocessing import MinMaxScaler
import numpy as np
scaler=MinMaxScaler(feature_range=(0,1))
data=scaler.fit_transform(np.array(data).reshape(-1,1))
```

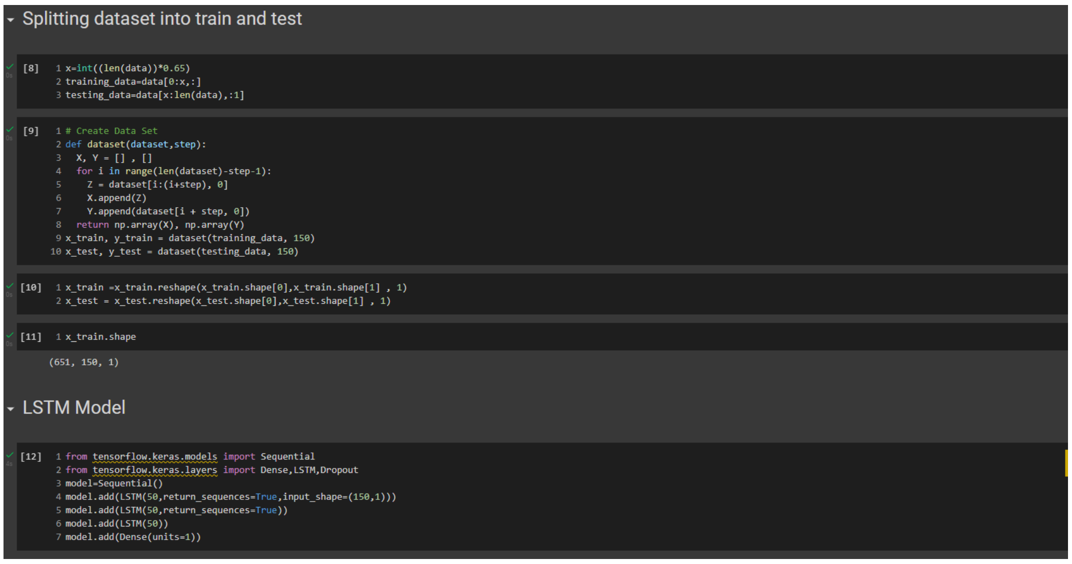
[6]
```
data.shape
```

```
(1235, 1)
```

[7]
```
data
```

```
array([[0.23823398],
       [0.2371134 ],
       [0.25773196],
       ...,
       [0.51792918],
       [0.47758853],
       [0.50425818]])
```

Splitting dataset into train data and test data



Splitting dataset into train and test

```
[8]
x=int((len(data))*0.65)
training_data=data[0:x,:]
testing_data=data[x:len(data),:1]
```

```
[9]
# Create Data Set
def dataset(dataset,step):
  X, Y = [] , []
  for i in range(len(dataset)-step-1):
    Z = dataset[i:(i+step), 0]
    X.append(Z)
    Y.append(dataset[i + step, 0])
  return np.array(X), np.array(Y)
x_train, y_train = dataset(training_data, 150)
x_test, y_test = dataset(testing_data, 150)
```

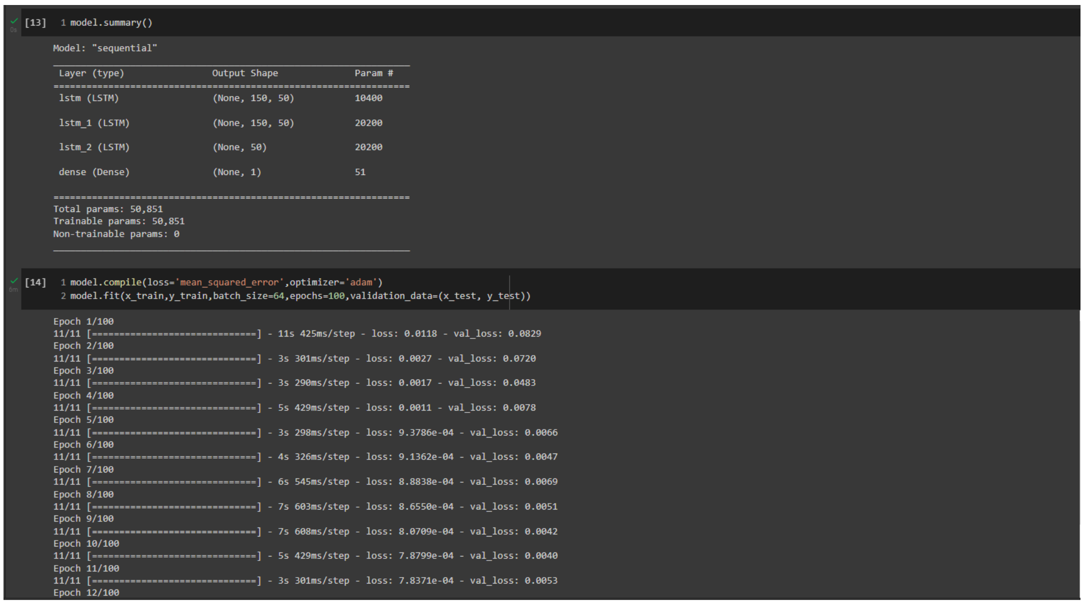
```
[10]
x_train =x_train.reshape(x_train.shape[0],x_train.shape[1] , 1)
x_test = x_test.reshape(x_test.shape[0],x_test.shape[1] , 1)
```

```
[11]
x_train.shape

(651, 150, 1)
```

LSTM Model

```
[12]
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense,LSTM,Dropout
model=Sequential()
model.add(LSTM(50,return_sequences=True,input_shape=(150,1)))
model.add(LSTM(50,return_sequences=True))
model.add(LSTM(50))
model.add(Dense(units=1))
```

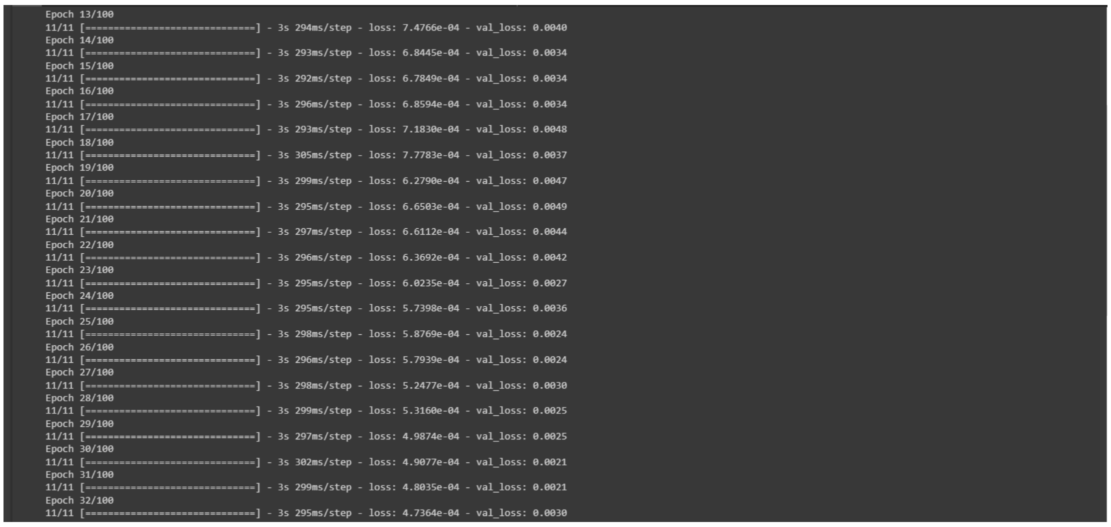
Applying LSTM Model



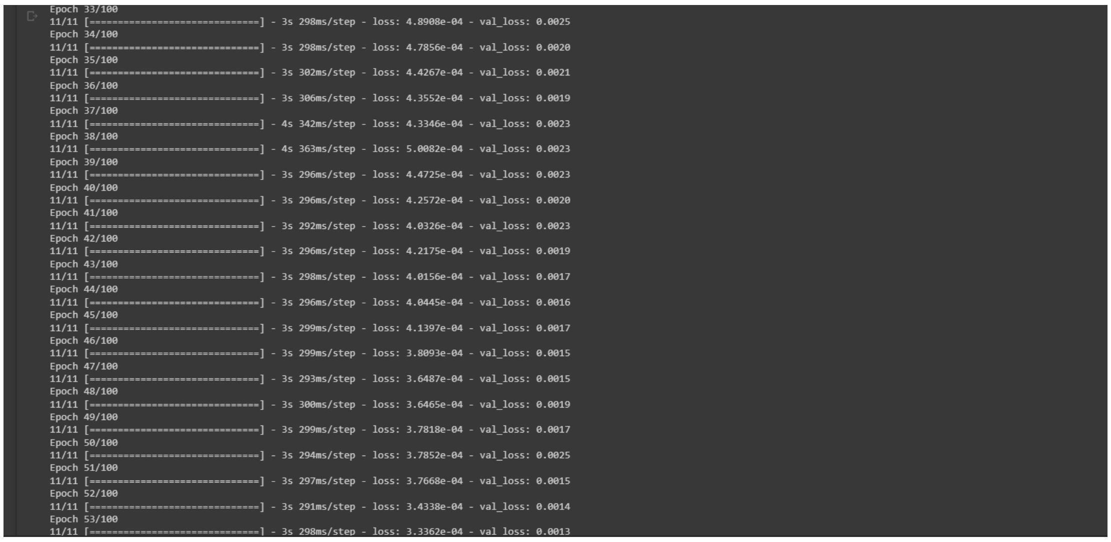
```
[13]
model.summary()

Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm (LSTM)                 (None, 150, 50)           10400

 lstm_1 (LSTM)               (None, 150, 50)           20200

 lstm_2 (LSTM)               (None, 50)                20200

 dense (Dense)               (None, 1)                 51

=================================================================
Total params: 50,851
Trainable params: 50,851
Non-trainable params: 0
_________________________________________________________________
```

```
[14]
model.compile(loss='mean_squared_error',optimizer='adam')
model.fit(x_train,y_train,batch_size=64,epochs=100,validation_data=(x_test, y_test))

Epoch 1/100
11/11 [==============================] - 11s 425ms/step - loss: 0.0118 - val_loss: 0.0829
Epoch 2/100
11/11 [==============================] - 3s 301ms/step - loss: 0.0027 - val_loss: 0.0720
Epoch 3/100
11/11 [==============================] - 3s 290ms/step - loss: 0.0017 - val_loss: 0.0483
Epoch 4/100
11/11 [==============================] - 5s 429ms/step - loss: 0.0011 - val_loss: 0.0078
Epoch 5/100
11/11 [==============================] - 3s 298ms/step - loss: 9.3786e-04 - val_loss: 0.0066
Epoch 6/100
11/11 [==============================] - 4s 326ms/step - loss: 9.1362e-04 - val_loss: 0.0047
Epoch 7/100
11/11 [==============================] - 6s 545ms/step - loss: 8.8838e-04 - val_loss: 0.0069
Epoch 8/100
11/11 [==============================] - 7s 603ms/step - loss: 8.6550e-04 - val_loss: 0.0051
Epoch 9/100
11/11 [==============================] - 7s 608ms/step - loss: 8.0709e-04 - val_loss: 0.0042
Epoch 10/100
11/11 [==============================] - 5s 429ms/step - loss: 7.8799e-04 - val_loss: 0.0040
Epoch 11/100
11/11 [==============================] - 3s 301ms/step - loss: 7.8371e-04 - val_loss: 0.0053
Epoch 12/100
```

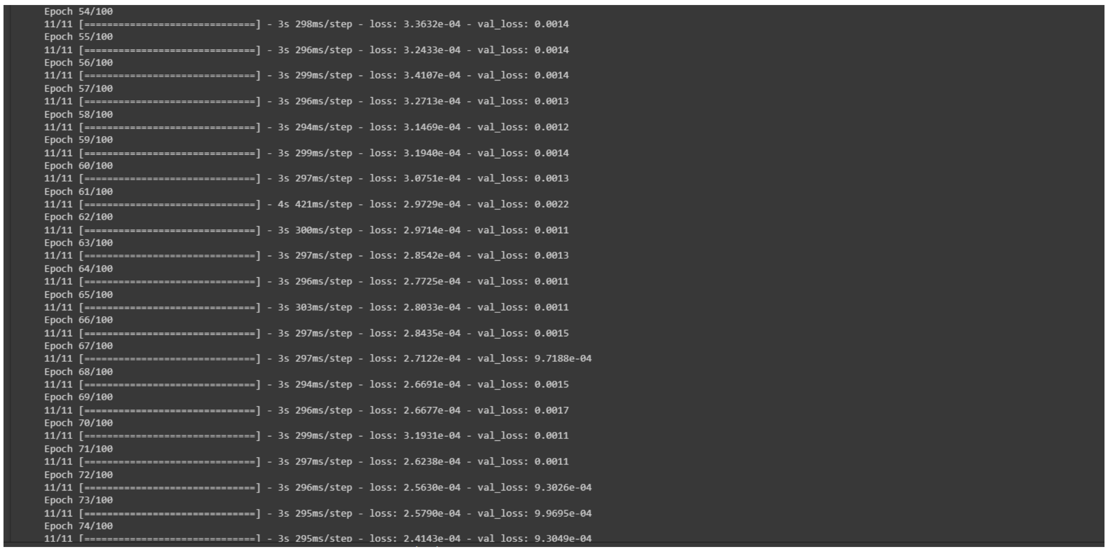
Fitting the train data



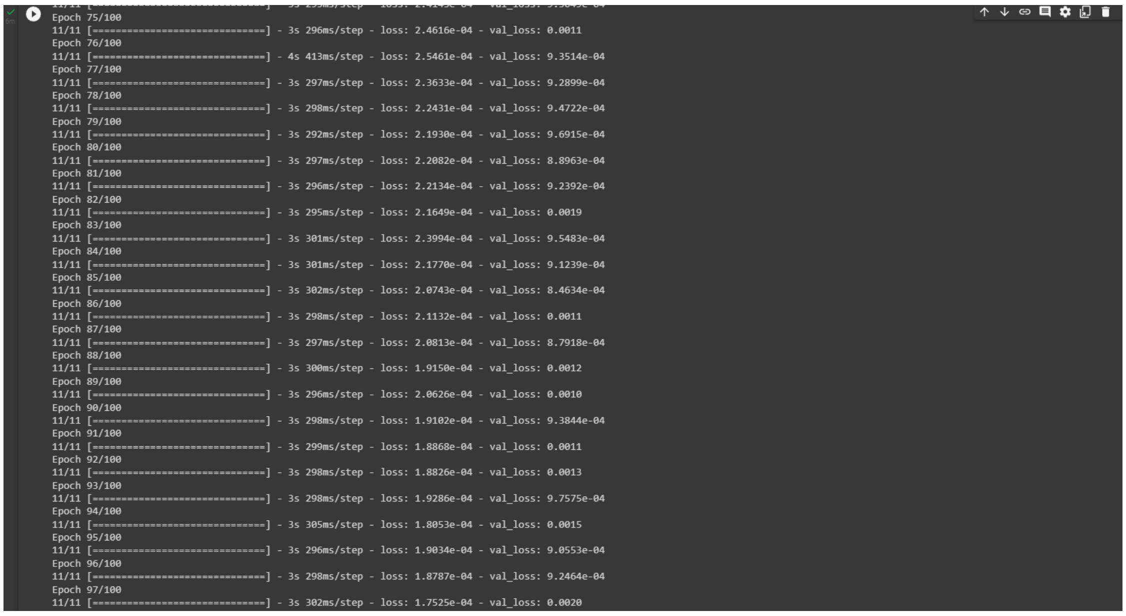
```
Epoch 13/100
11/11 [==============================] - 3s 294ms/step - loss: 7.4766e-04 - val_loss: 0.0040
Epoch 14/100
11/11 [==============================] - 3s 293ms/step - loss: 6.8445e-04 - val_loss: 0.0034
Epoch 15/100
11/11 [==============================] - 3s 292ms/step - loss: 6.7849e-04 - val_loss: 0.0034
Epoch 16/100
11/11 [==============================] - 3s 296ms/step - loss: 6.8594e-04 - val_loss: 0.0034
Epoch 17/100
11/11 [==============================] - 3s 293ms/step - loss: 7.1830e-04 - val_loss: 0.0048
Epoch 18/100
11/11 [==============================] - 3s 305ms/step - loss: 7.7783e-04 - val_loss: 0.0037
Epoch 19/100
11/11 [==============================] - 3s 299ms/step - loss: 6.2790e-04 - val_loss: 0.0047
Epoch 20/100
11/11 [==============================] - 3s 295ms/step - loss: 6.6503e-04 - val_loss: 0.0049
Epoch 21/100
11/11 [==============================] - 3s 297ms/step - loss: 6.6112e-04 - val_loss: 0.0044
Epoch 22/100
11/11 [==============================] - 3s 296ms/step - loss: 6.3692e-04 - val_loss: 0.0042
Epoch 23/100
11/11 [==============================] - 3s 295ms/step - loss: 6.0235e-04 - val_loss: 0.0027
Epoch 24/100
11/11 [==============================] - 3s 295ms/step - loss: 5.7398e-04 - val_loss: 0.0036
Epoch 25/100
11/11 [==============================] - 3s 298ms/step - loss: 5.8769e-04 - val_loss: 0.0024
Epoch 26/100
11/11 [==============================] - 3s 296ms/step - loss: 5.7939e-04 - val_loss: 0.0024
Epoch 27/100
11/11 [==============================] - 3s 298ms/step - loss: 5.2477e-04 - val_loss: 0.0030
Epoch 28/100
11/11 [==============================] - 3s 299ms/step - loss: 5.3160e-04 - val_loss: 0.0025
Epoch 29/100
11/11 [==============================] - 3s 297ms/step - loss: 4.9874e-04 - val_loss: 0.0025
Epoch 30/100
11/11 [==============================] - 3s 302ms/step - loss: 4.9077e-04 - val_loss: 0.0021
Epoch 31/100
11/11 [==============================] - 3s 299ms/step - loss: 4.8035e-04 - val_loss: 0.0021
Epoch 32/100
11/11 [==============================] - 3s 295ms/step - loss: 4.7364e-04 - val_loss: 0.0030
```



```
Epoch 33/100
11/11 [==============================] - 3s 298ms/step - loss: 4.8908e-04 - val_loss: 0.0025
Epoch 34/100
11/11 [==============================] - 3s 298ms/step - loss: 4.7856e-04 - val_loss: 0.0020
Epoch 35/100
11/11 [==============================] - 3s 302ms/step - loss: 4.4267e-04 - val_loss: 0.0021
Epoch 36/100
11/11 [==============================] - 3s 306ms/step - loss: 4.3552e-04 - val_loss: 0.0019
Epoch 37/100
11/11 [==============================] - 4s 342ms/step - loss: 4.3346e-04 - val_loss: 0.0023
Epoch 38/100
11/11 [==============================] - 4s 363ms/step - loss: 5.0082e-04 - val_loss: 0.0023
Epoch 39/100
11/11 [==============================] - 3s 296ms/step - loss: 4.4725e-04 - val_loss: 0.0023
Epoch 40/100
11/11 [==============================] - 3s 296ms/step - loss: 4.2572e-04 - val_loss: 0.0020
Epoch 41/100
11/11 [==============================] - 3s 292ms/step - loss: 4.0326e-04 - val_loss: 0.0023
Epoch 42/100
11/11 [==============================] - 3s 296ms/step - loss: 4.2175e-04 - val_loss: 0.0019
Epoch 43/100
11/11 [==============================] - 3s 298ms/step - loss: 4.0156e-04 - val_loss: 0.0017
Epoch 44/100
11/11 [==============================] - 3s 296ms/step - loss: 4.0445e-04 - val_loss: 0.0016
Epoch 45/100
11/11 [==============================] - 3s 299ms/step - loss: 4.1397e-04 - val_loss: 0.0017
Epoch 46/100
11/11 [==============================] - 3s 299ms/step - loss: 3.8093e-04 - val_loss: 0.0015
Epoch 47/100
11/11 [==============================] - 3s 293ms/step - loss: 3.6487e-04 - val_loss: 0.0015
Epoch 48/100
11/11 [==============================] - 3s 300ms/step - loss: 3.6465e-04 - val_loss: 0.0019
Epoch 49/100
11/11 [==============================] - 3s 299ms/step - loss: 3.7818e-04 - val_loss: 0.0017
Epoch 50/100
11/11 [==============================] - 3s 294ms/step - loss: 3.7852e-04 - val_loss: 0.0025
Epoch 51/100
11/11 [==============================] - 3s 297ms/step - loss: 3.7668e-04 - val_loss: 0.0015
Epoch 52/100
11/11 [==============================] - 3s 291ms/step - loss: 3.4338e-04 - val_loss: 0.0014
Epoch 53/100
11/11 [==============================] - 3s 298ms/step - loss: 3.3362e-04 - val_loss: 0.0013
```

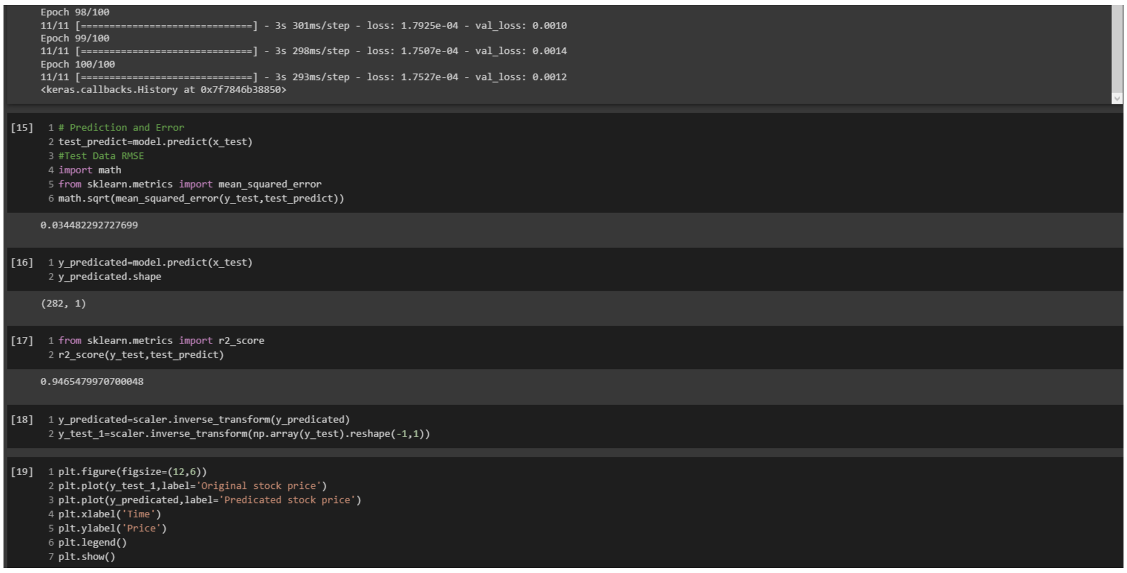


```
Epoch 54/100
11/11 [==============================] - 3s 298ms/step - loss: 3.3632e-04 - val_loss: 0.0014
Epoch 55/100
11/11 [==============================] - 3s 296ms/step - loss: 3.2433e-04 - val_loss: 0.0014
Epoch 56/100
11/11 [==============================] - 3s 299ms/step - loss: 3.4107e-04 - val_loss: 0.0014
Epoch 57/100
11/11 [==============================] - 3s 296ms/step - loss: 3.2713e-04 - val_loss: 0.0013
Epoch 58/100
11/11 [==============================] - 3s 294ms/step - loss: 3.1469e-04 - val_loss: 0.0012
Epoch 59/100
11/11 [==============================] - 3s 299ms/step - loss: 3.1940e-04 - val_loss: 0.0014
Epoch 60/100
11/11 [==============================] - 3s 297ms/step - loss: 3.0751e-04 - val_loss: 0.0013
Epoch 61/100
11/11 [==============================] - 4s 421ms/step - loss: 2.9729e-04 - val_loss: 0.0022
Epoch 62/100
11/11 [==============================] - 3s 300ms/step - loss: 2.9714e-04 - val_loss: 0.0011
Epoch 63/100
11/11 [==============================] - 3s 297ms/step - loss: 2.8542e-04 - val_loss: 0.0013
Epoch 64/100
11/11 [==============================] - 3s 296ms/step - loss: 2.7725e-04 - val_loss: 0.0011
Epoch 65/100
11/11 [==============================] - 3s 303ms/step - loss: 2.8033e-04 - val_loss: 0.0011
Epoch 66/100
11/11 [==============================] - 3s 297ms/step - loss: 2.8435e-04 - val_loss: 0.0015
Epoch 67/100
11/11 [==============================] - 3s 297ms/step - loss: 2.7122e-04 - val_loss: 9.7188e-04
Epoch 68/100
11/11 [==============================] - 3s 294ms/step - loss: 2.6691e-04 - val_loss: 0.0015
Epoch 69/100
11/11 [==============================] - 3s 296ms/step - loss: 2.6677e-04 - val_loss: 0.0017
Epoch 70/100
11/11 [==============================] - 3s 299ms/step - loss: 3.1931e-04 - val_loss: 0.0011
Epoch 71/100
11/11 [==============================] - 3s 297ms/step - loss: 2.6238e-04 - val_loss: 0.0011
Epoch 72/100
11/11 [==============================] - 3s 296ms/step - loss: 2.5630e-04 - val_loss: 9.3026e-04
Epoch 73/100
11/11 [==============================] - 3s 295ms/step - loss: 2.5790e-04 - val_loss: 9.9695e-04
Epoch 74/100
11/11 [==============================] - 3s 295ms/step - loss: 2.4143e-04 - val_loss: 9.3049e-04
```



```
Epoch 75/100
11/11 [==============================] - 3s 296ms/step - loss: 2.4616e-04 - val_loss: 0.0011
Epoch 76/100
11/11 [==============================] - 4s 413ms/step - loss: 2.5461e-04 - val_loss: 9.3514e-04
Epoch 77/100
11/11 [==============================] - 3s 297ms/step - loss: 2.3633e-04 - val_loss: 9.2899e-04
Epoch 78/100
11/11 [==============================] - 3s 298ms/step - loss: 2.2431e-04 - val_loss: 9.4722e-04
Epoch 79/100
11/11 [==============================] - 3s 292ms/step - loss: 2.1930e-04 - val_loss: 9.6915e-04
Epoch 80/100
11/11 [==============================] - 3s 297ms/step - loss: 2.2082e-04 - val_loss: 8.8963e-04
Epoch 81/100
11/11 [==============================] - 3s 296ms/step - loss: 2.2134e-04 - val_loss: 9.2392e-04
Epoch 82/100
11/11 [==============================] - 3s 295ms/step - loss: 2.1649e-04 - val_loss: 0.0019
Epoch 83/100
11/11 [==============================] - 3s 301ms/step - loss: 2.3994e-04 - val_loss: 9.5483e-04
Epoch 84/100
11/11 [==============================] - 3s 301ms/step - loss: 2.1770e-04 - val_loss: 9.1239e-04
Epoch 85/100
11/11 [==============================] - 3s 302ms/step - loss: 2.0743e-04 - val_loss: 8.4634e-04
Epoch 86/100
11/11 [==============================] - 3s 298ms/step - loss: 2.1132e-04 - val_loss: 0.0011
Epoch 87/100
11/11 [==============================] - 3s 297ms/step - loss: 2.0813e-04 - val_loss: 8.7918e-04
Epoch 88/100
11/11 [==============================] - 3s 300ms/step - loss: 1.9150e-04 - val_loss: 0.0012
Epoch 89/100
11/11 [==============================] - 3s 296ms/step - loss: 2.0626e-04 - val_loss: 0.0010
Epoch 90/100
11/11 [==============================] - 3s 298ms/step - loss: 1.9102e-04 - val_loss: 9.3844e-04
Epoch 91/100
11/11 [==============================] - 3s 299ms/step - loss: 1.8868e-04 - val_loss: 0.0011
Epoch 92/100
11/11 [==============================] - 3s 298ms/step - loss: 1.8826e-04 - val_loss: 0.0013
Epoch 93/100
11/11 [==============================] - 3s 298ms/step - loss: 1.9286e-04 - val_loss: 9.7575e-04
Epoch 94/100
11/11 [==============================] - 3s 305ms/step - loss: 1.8053e-04 - val_loss: 0.0015
Epoch 95/100
11/11 [==============================] - 3s 296ms/step - loss: 1.9034e-04 - val_loss: 9.0553e-04
Epoch 96/100
11/11 [==============================] - 3s 298ms/step - loss: 1.8787e-04 - val_loss: 9.2464e-04
Epoch 97/100
11/11 [==============================] - 3s 302ms/step - loss: 1.7525e-04 - val_loss: 0.0020
```

Importing Modules



```
Epoch 98/100
11/11 [==============================] - 3s 301ms/step - loss: 1.7925e-04 - val_loss: 0.0010
Epoch 99/100
11/11 [==============================] - 3s 298ms/step - loss: 1.7507e-04 - val_loss: 0.0014
Epoch 100/100
11/11 [==============================] - 3s 293ms/step - loss: 1.7527e-04 - val_loss: 0.0012
<keras.callbacks.History at 0x7f7846b38850>
```

```
[15]
# Prediction and Error
test_predict=model.predict(x_test)
#Test Data RMSE
import math
from sklearn.metrics import mean_squared_error
math.sqrt(mean_squared_error(y_test,test_predict))

0.034482292727699
```

```
[16]
y_predicated=model.predict(x_test)
y_predicated.shape

(282, 1)
```

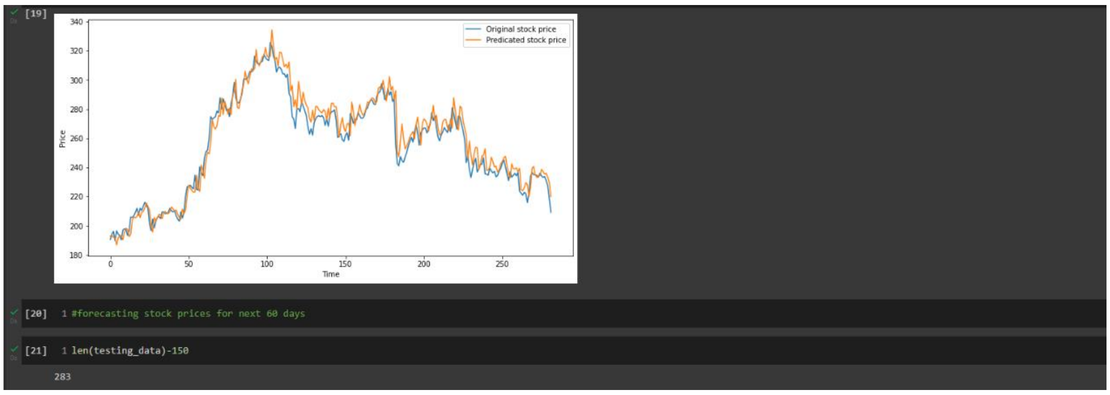
```
[17]
from sklearn.metrics import r2_score
r2_score(y_test,test_predict)

0.9465479970700048
```

```
[18]
y_predicated=scaler.inverse_transform(y_predicated)
y_test_1=scaler.inverse_transform(np.array(y_test).reshape(-1,1))
```

```
[19]
plt.figure(figsize=(12,6))
plt.plot(y_test_1,label='Original stock price')
plt.plot(y_predicated,label='Predicated stock price')
plt.xlabel('Time')
plt.ylabel('Price')
plt.legend()
plt.show()
```

Finding Mean Squared error and accuracy

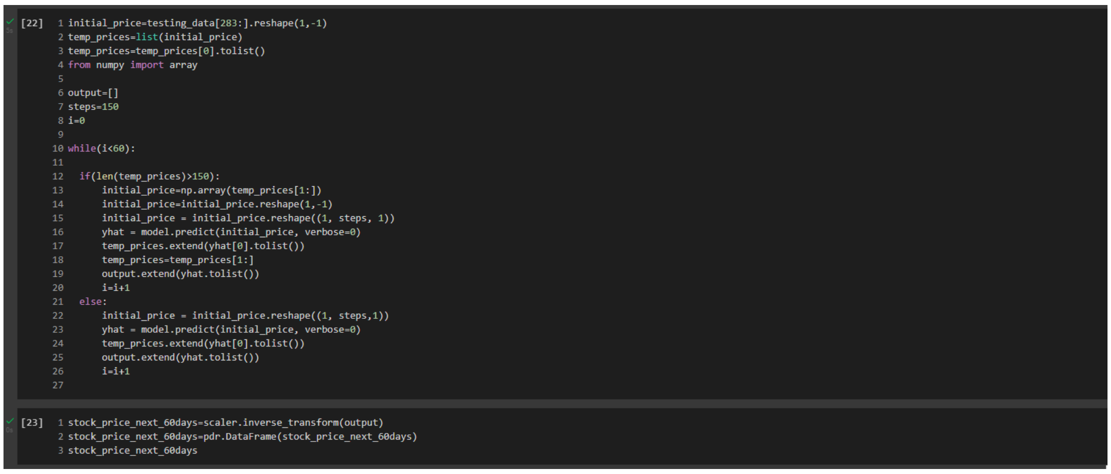
Plotting Original and Predicted Stock Price



```
[19]
```

```
[20] #forecasting stock prices for next 60 days
```

```
[21] len(testing_data)-150

283
```

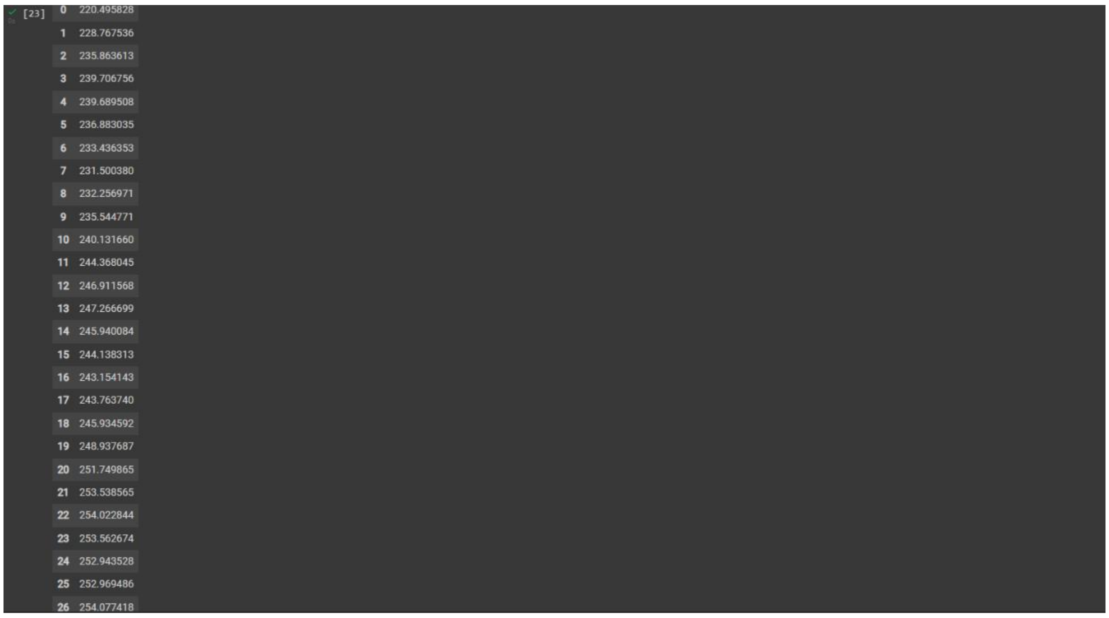


```
[22] initial_price=testing_data[283:].reshape(1,-1)
temp_prices=list(initial_price)
temp_prices=temp_prices[0].tolist()
from numpy import array

output=[]
steps=150
i=0

while(i<60):

  if(len(temp_prices)>150):
      initial_price=np.array(temp_prices[1:])
      initial_price=initial_price.reshape(1,-1)
      initial_price = initial_price.reshape((1, steps, 1))
      yhat = model.predict(initial_price, verbose=0)
      temp_prices.extend(yhat[0].tolist())
      temp_prices=temp_prices[1:]
      output.extend(yhat.tolist())
      i=i+1
  else:
      initial_price = initial_price.reshape((1, steps,1))
      yhat = model.predict(initial_price, verbose=0)
      temp_prices.extend(yhat[0].tolist())
      output.extend(yhat.tolist())
      i=i+1

```

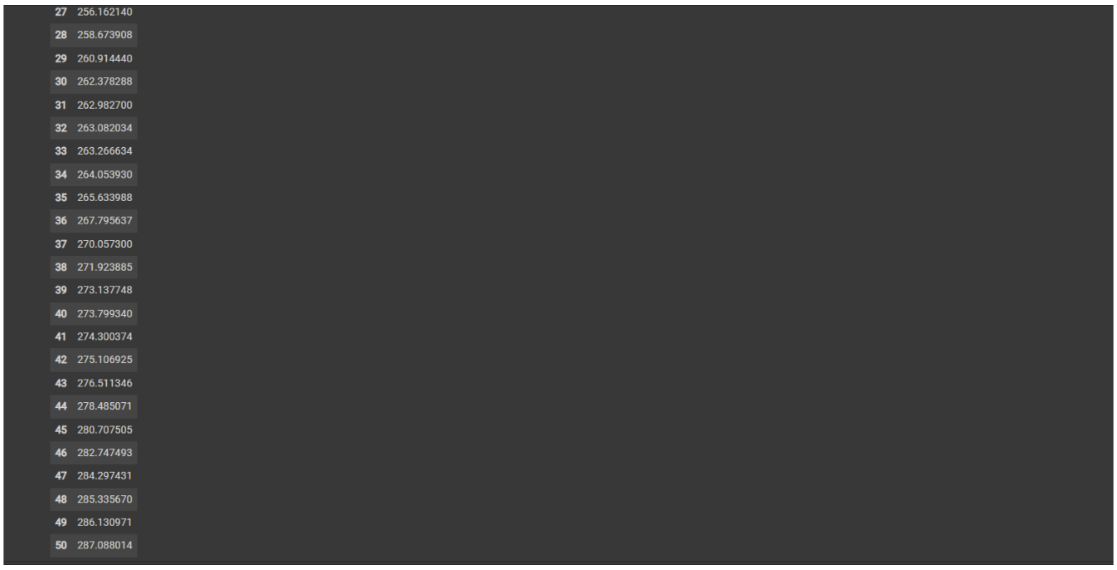
```
[23] stock_price_next_60days=scaler.inverse_transform(output)
stock_price_next_60days=pdr.DataFrame(stock_price_next_60days)
stock_price_next_60days
```

Predicting the stock price for next 60 days

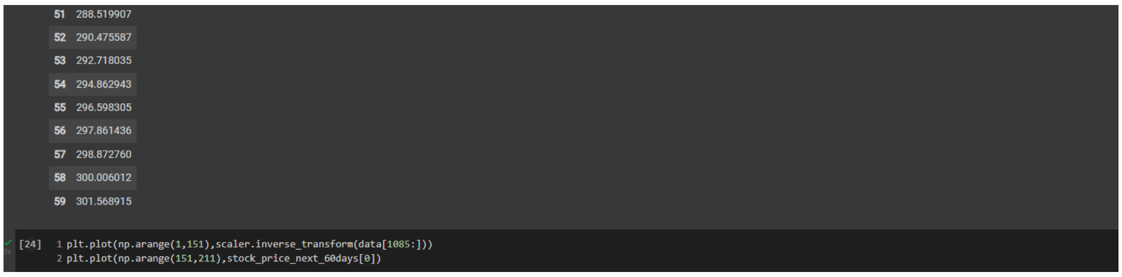
We can see **Ups** and **Downs** in the stock value



```
[23]  0  220.495828
      1  228.767536
      2  235.863613
      3  239.706756
      4  239.689508
      5  236.883035
      6  233.436353
      7  231.500380
      8  232.256971
      9  235.544771
     10  240.131660
     11  244.368045
     12  246.911568
     13  247.266699
     14  245.940084
     15  244.138313
     16  243.154143
     17  243.763740
     18  245.934592
     19  248.937687
     20  251.749865
     21  253.538565
     22  254.022844
     23  253.562674
     24  252.943528
     25  252.969486
     26  254.077418
```



```
     27  256.162140
     28  258.673908
     29  260.914440
     30  262.378288
     31  262.982700
     32  263.082034
     33  263.266634
     34  264.053930
     35  265.633988
     36  267.795637
     37  270.057300
     38  271.923885
     39  273.137748
     40  273.799340
     41  274.300374
     42  275.106925
     43  276.511346
     44  278.485071
     45  280.707505
     46  282.747493
     47  284.297431
     48  285.335670
     49  286.130971
     50  287.088014
```



```
     51  288.519907
     52  290.475587
     53  292.718035
     54  294.862943
     55  296.598305
     56  297.861436
     57  298.872760
     58  300.006012
     59  301.568915
```

```
[24]  plt.plot(np.arange(1,151),scaler.inverse_transform(data[1085:]))
      plt.plot(np.arange(151,211),stock_price_next_60days[0])
```

Chapter 4

Result

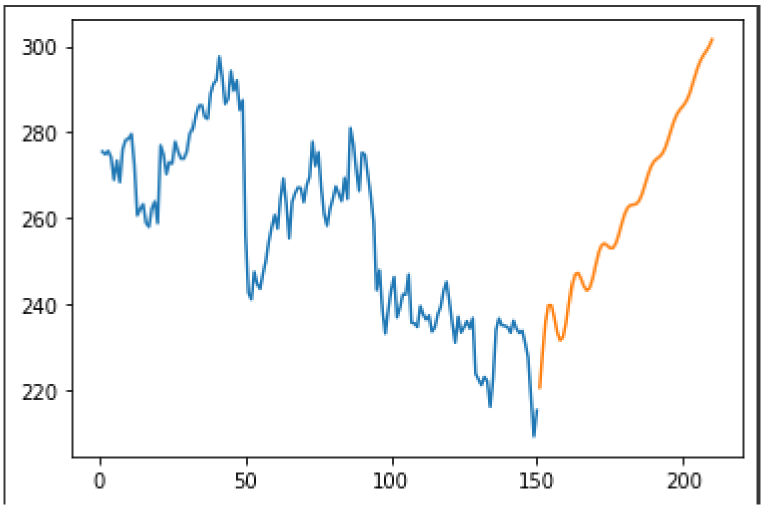


Fig. 20 Stock Price Graph

*Note: Result may change when you run the program

Chapter 5

Conclusion

With the introduction of Machine Learning and its strong algorithms, the most recent market research and Stock Market Prediction advancements have begun to include such approaches in analyzing stock market data.

The Opening Value of the stock, the Highest and Lowest values of that stock on the same days, as well as the Closing Value at the end of the day, are all indicated for each date. Furthermore, the total volume of the stocks in the market is provided, with this information, it is up to the job of a Machine Learning Data Scientist to look at the data and develop different algorithms that may help in finding appropriate stocks values.

Predicting the stock market was a time-consuming and laborious procedure a few years or even a decade ago. However, with the application of machine learning for stock market forecasts, the procedure has become much simpler. Machine learning not only saves time and resources but also outperforms people in terms of performance. It will always prefer to use a trained computer algorithm since it will advise you based only on facts, numbers, and data and will not factor in emotions or prejudice.

Chapter 6

Project Scope

This Project can predict the stock by training the algorithm and for that we are using dataset for the test data then apply the prediction on the test data. It needs a large dataset for the test data to make the prediction.

The Accuracy of this project is 94.65%.

Bibliography

References

- https://www.geeksforgeeks.org/ml-linear-regression/
- https://www.expertsystem.com/
- https://www.edureka.co/
- https://www.wikipedia.org/
- https://towardsdatascience.com/
- https://www.simplilearn.com/
- https://www.forbes.com
- https://www.cousera.org/
- https://www.google.com/
- https://www.medium.com/