Two Way Relay Network

A Project-II Report

Submitted in partial fulfillment of requirement of the

Degree of

BACHELOR OF TECHNOLOGY in COMPUTER SCIENCE & ENGINEERING

BY

Anushk Jain

Enrollment Number:
EN18CS301046

Under the Guidance of
Dr. Nitika Vats Doohan

Department of Computer Science & Engineering
Faculty of Engineering
MEDI-CAPS UNIVERSITY, INDORE- 453331

May 2022

Report Approval

The project work **“Two Way Relay Network”** 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 Report” only for the purpose for which it has been submitted.

Internal Examiner
Name: Dr. Nitika Vats Doohan
Designation: Professor
Computer Science Engineering
Medi-Caps University, Indore

External Examiner
Name:
Designation
Affiliation

Declaration

I hereby declare that the project entitled **"Two Way Relay Network"** submitted in partial fulfillment for the award of the degree of Bachelor of Technology in 'Computer Science & Engineering' completed under the supervision of **Dr. Nitika Vats Doohan,** Faculty of Engineering, Medi-Caps University Indore is an authentic work.

Further, I/we declare that the content of this Project work, 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.

Anushk Jain

(EN18CS301046)

Date :- 10/05/2022

Certificate

I/We, **Professor Vimal Bhatia (Professor IIT Indore), Dr. Nitika Vats Doohan (Professor Medi-Caps University Indore)** certify that the project entitled **"Two Way Relay Network"** submitted in partial fulfillment for the award of the degree of Bachelor of Technology by **Anushk Jain (EN18CS301046)** is the record carried out by him/them under my/our guidance and that the work has not formed the basis of award of any other degree elsewhere.

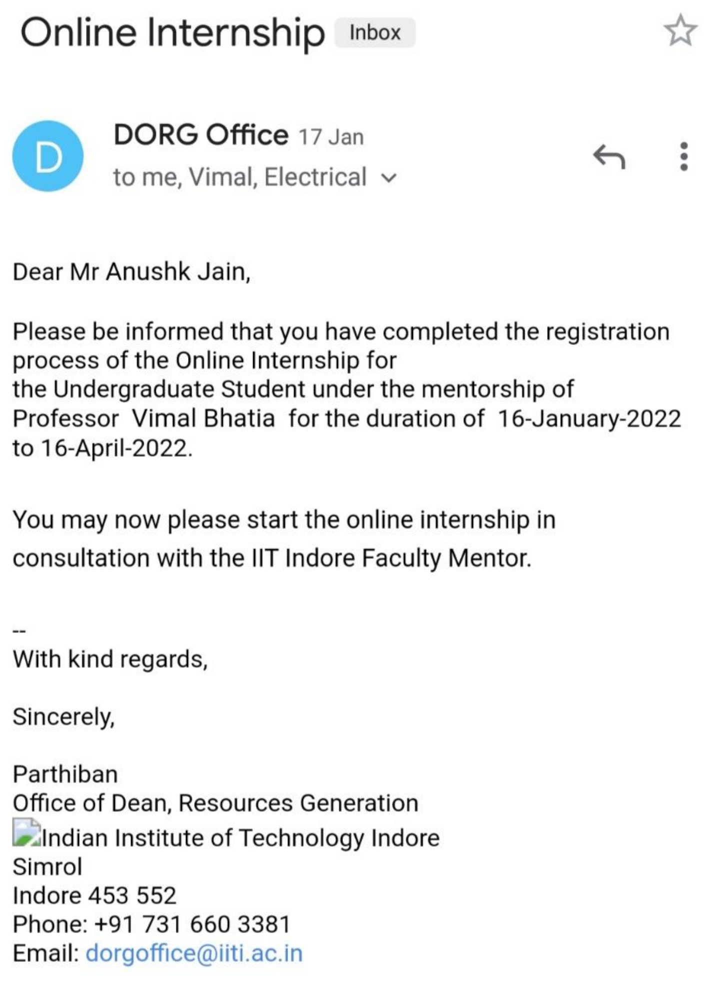
[illegible] 04.04.2022

Dr. Nitika Vats Doohan
Computer Science Department
Medi-Caps University, Indore

Prof. Vimal Bhatia
Electrical Engineering Department
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 Anushk 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.

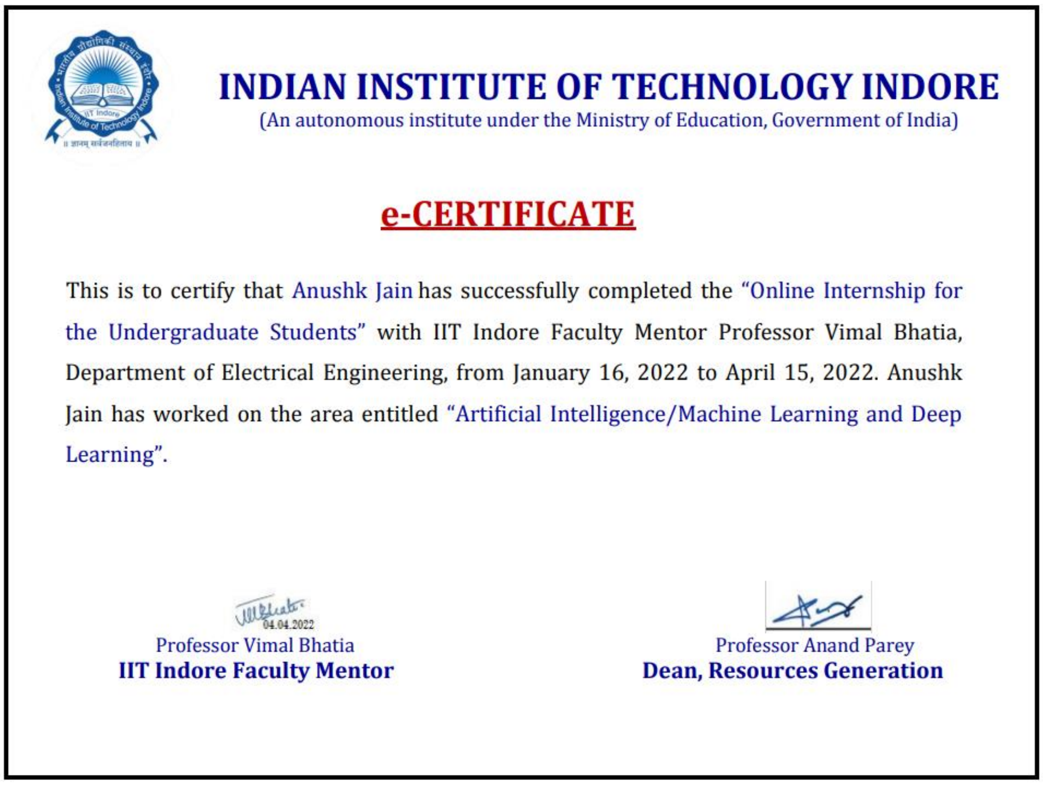
--
With kind regards,

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

Offer Letter

Completion Certificate



INDIAN INSTITUTE OF TECHNOLOGY INDORE

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

e-CERTIFICATE

This is to certify that Anushk 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. Anushk 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

Completion Certificate

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 Internal Guide, **Prof. Vimal Bhatia** Professor of Department in Electrical Engineering , IIT Indore as well as my Internal Guide, **Dr. Nitika Vats Doohan**, Professor, Department of 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.

Anushk Jain (EN18CS301046)
B.Tech. IV Year
Department of Computer Science & Engineering
Medi-Caps University, Indore

Abstract

This training has introduced us to Machine Learning a-nd Deep learning . Now, weknow that Machine Learning is a technique of training machines to perform the activities a human brain can do, albeit bit faster and better than an average human- being. We will see that machines can be trained to perform human activities in several areas and can aid humans in living better lives. Machine learning is quicklygrowing field in computer science. It has applications in nearly every other field ofstudy and is already being implemented commercially because machine learning can solve problems too difficult or time consuming for humans to solve. To describe machine learning in general terms, a variety models are used to learn patterns in data and make accurate predictions based on the patterns it observes. In the Two-way Relay network the relay suffers from non-linearity which causes bad communication we'll try to fix this problem by ML model and predict the approximate accurate value of the time-switching factor (α) to design an energy efficient system.

Table of Contents

List of Figures

Abbreviations

Abbr.	Meaning
AI	Artificial Intelligence
ML	Machine Learning
DL	Deep Learning
RL	Reinforcement Learning
RNN	Recurrent Neural Network
ANN	Artificial Neural Network
MAE	Mean Average Error
MSE	Mean Square Error

Chapter 1

Introduction to Artificial Intelligence/ Machine Learning/ Deep Learning

What is Artificial Intelligence?

Artificial intelligence is a branch of computer science that aims to create intelligentmachines. It has become an essential part of the technology industry.

Research associated with artificial intelligence is highly technical and specialized. The core problems of artificial intelligence include programming computers for certain traits such as:

- Knowledge
- Reasoning
- Problem solving
- Perception
- Learning
- Planning
- Ability to manipulate and move objects

- An intelligent entity created by humans.
- Capable of performing tasks intelligently without being explicitly instructed.
- Capable of thinking and acting rationally and humanely.

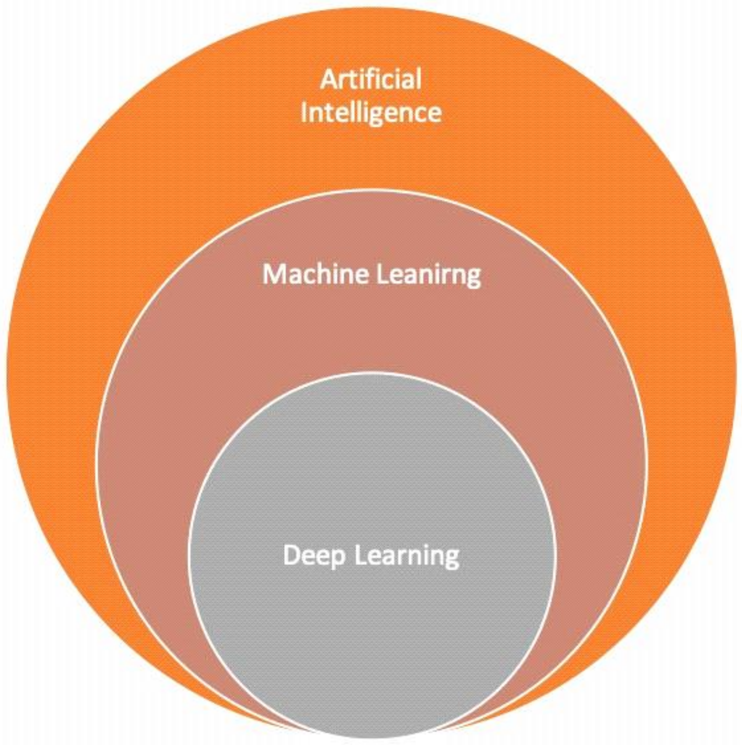


Fig. 01 Artificial Intelligence

What is Machine Learning ?

Machine learning is a subfield of artificial intelligence (AI). The goal of machinelearning generally is to understand the structure of data and fit that data into models that can be understood and utilized by people.

Although machine learning is a field within computer science, it differs from traditional computational approaches. In traditional computing, algorithms are setsof explicitly programmed instructions used by computers to calculate or problem solve. Machine learning algorithms instead allow for computers to train on data inputs and use statistical analysis in order to output values that fall within a specific range. Because of this, machine learning facilitates computers in building models from sample data in order to automate decision-making processes based ondata inputs.

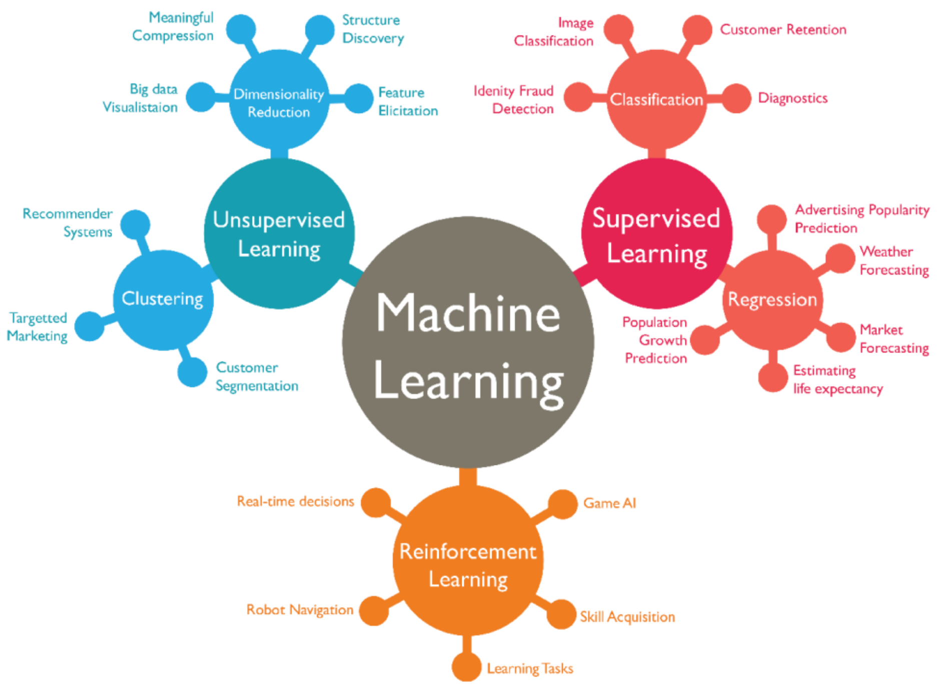


Fig. 02 Machine Learning

What is Deep Learning?

Deep Learning is a subset of Machine Learning that uses mathematical functions tomap the input to the output. These functions can extract non-redundant informationor patterns from the data, which enables them to form a relationship between the input and the output.

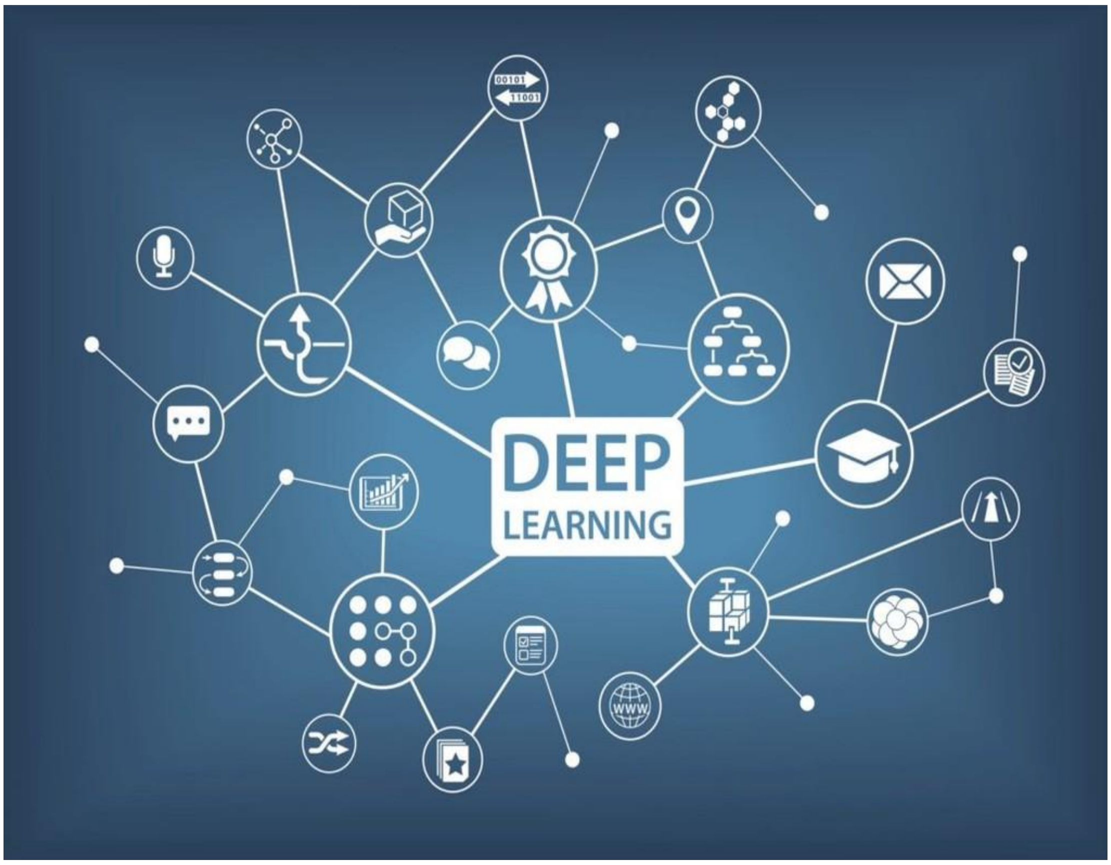


Fig. 03 Deep Learning

<u>Techniques of Machine Learning</u>

- **SUPERVISED LEARNING**

Use supervised learning if you know in advance what you want to teach a machine.This typically requires exposing the algorithm to a huge set of training data, lettingthe model examine the output, and adjusting the parameters until getting the desired results. You can then test the machine by letting it make predictions for a "validation data set", or in other words, new unseen data.

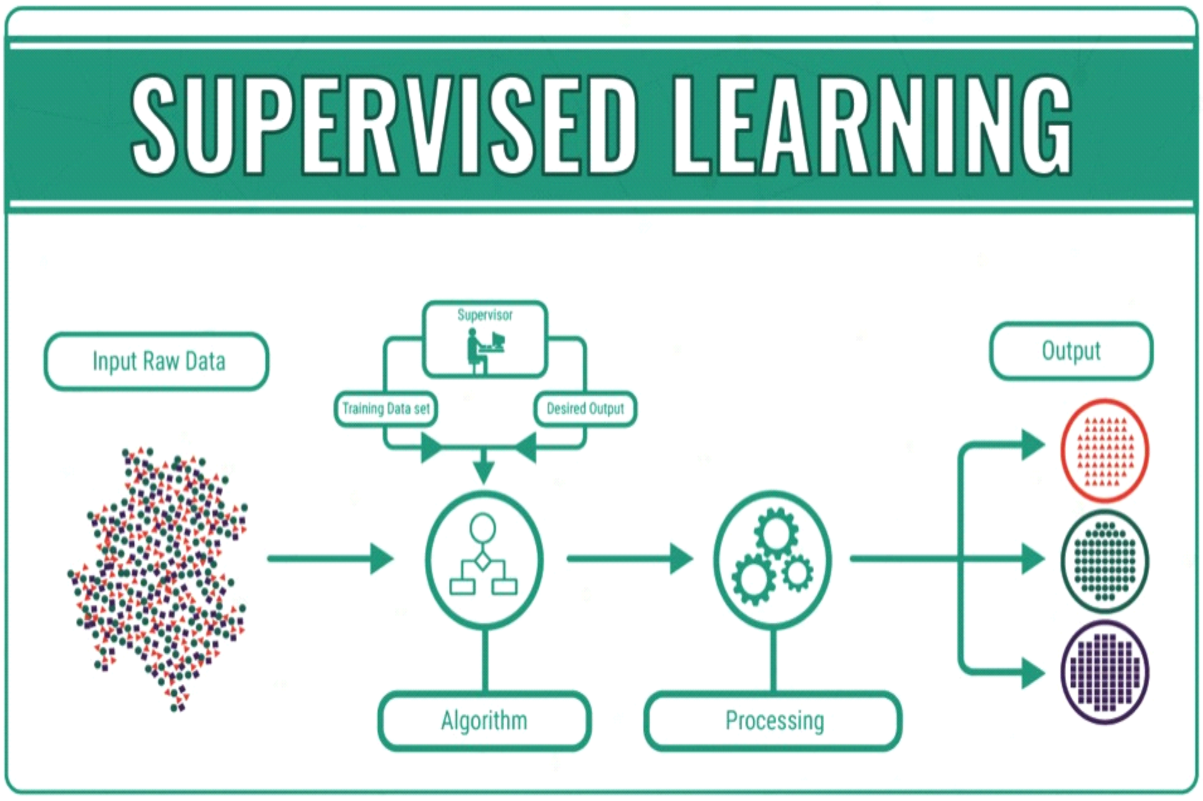


Fig. 04 Supervised Learning

• UNSUPERVISED LEARNING

Unsupervised learning enables a machine to explore a set of data. After the initial exploration, the machine tries to identify hidden patterns that connect different variables. This type of learning can help turn data into groups, based only on statistical properties. Unsupervised learning does not require training on large data sets, and so it is much faster and easier to deploy, compared to supervised learning.

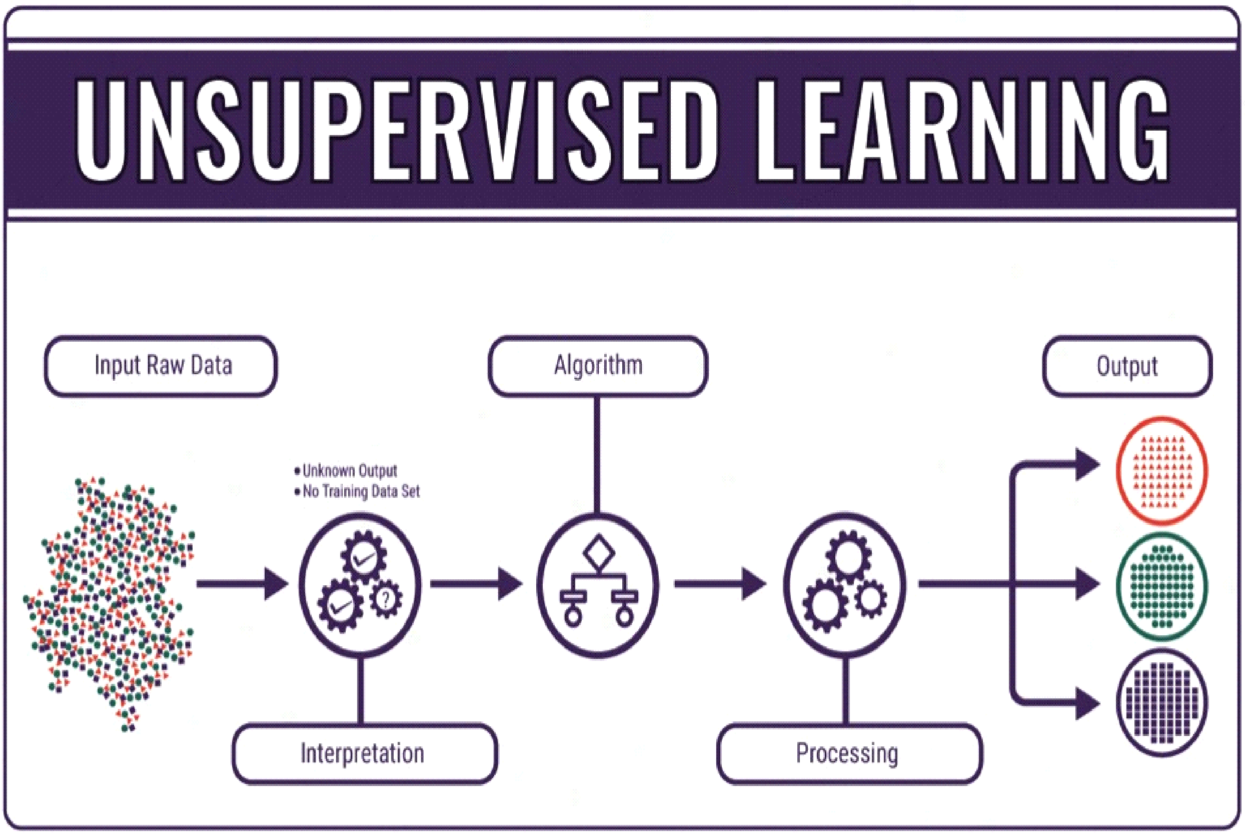


Fig. 05 Unsupervised Learning

- **SEMI-SUPERVISED LEARNING**

Semi-supervised learning combines techniques from unsupervised and supervisedlearning. For example, manually labeling some of the data can provide the algorithm with an example on how the rest of the data set should be grouped.

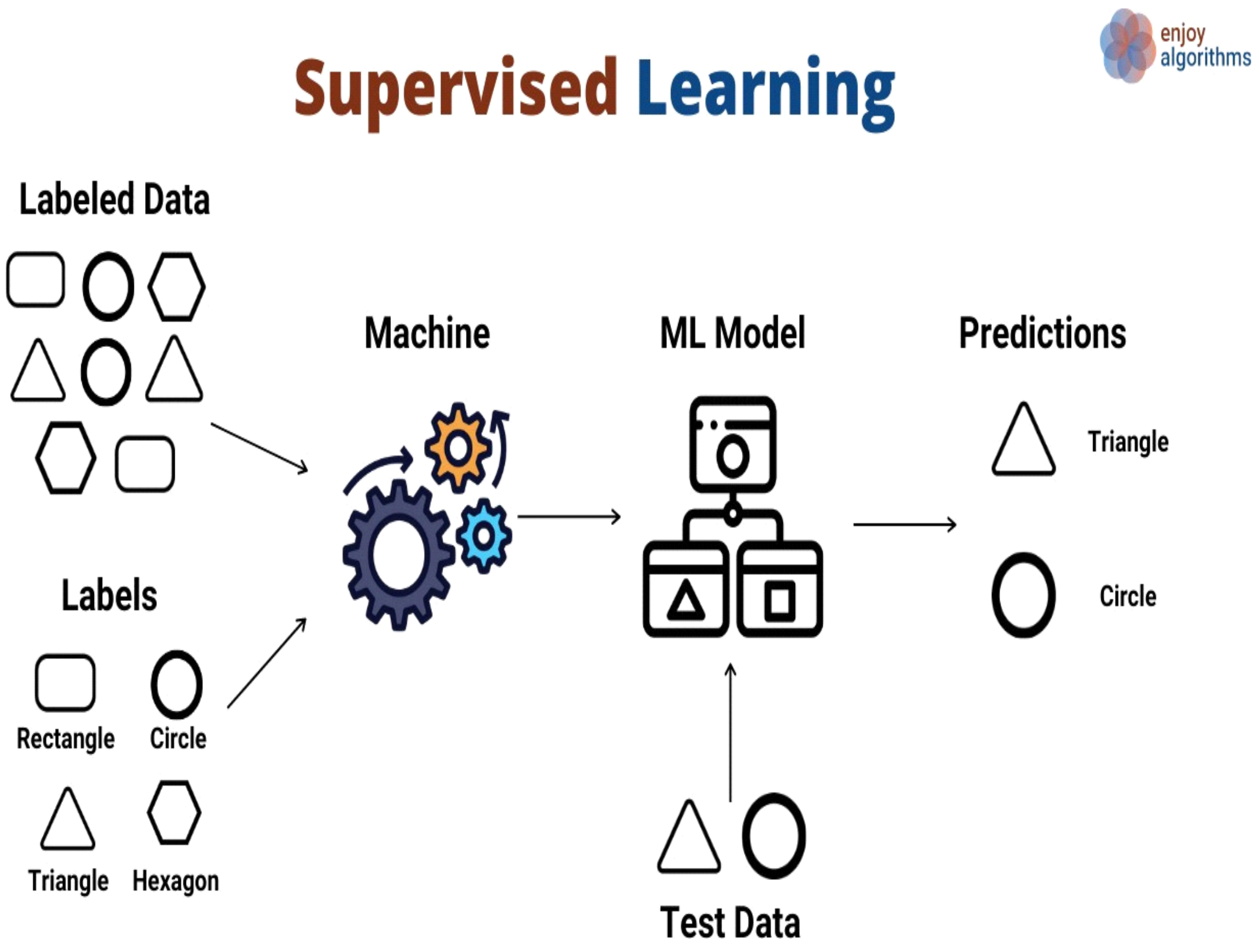


Fig. 06 Semi-Supervised Learning

• 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.

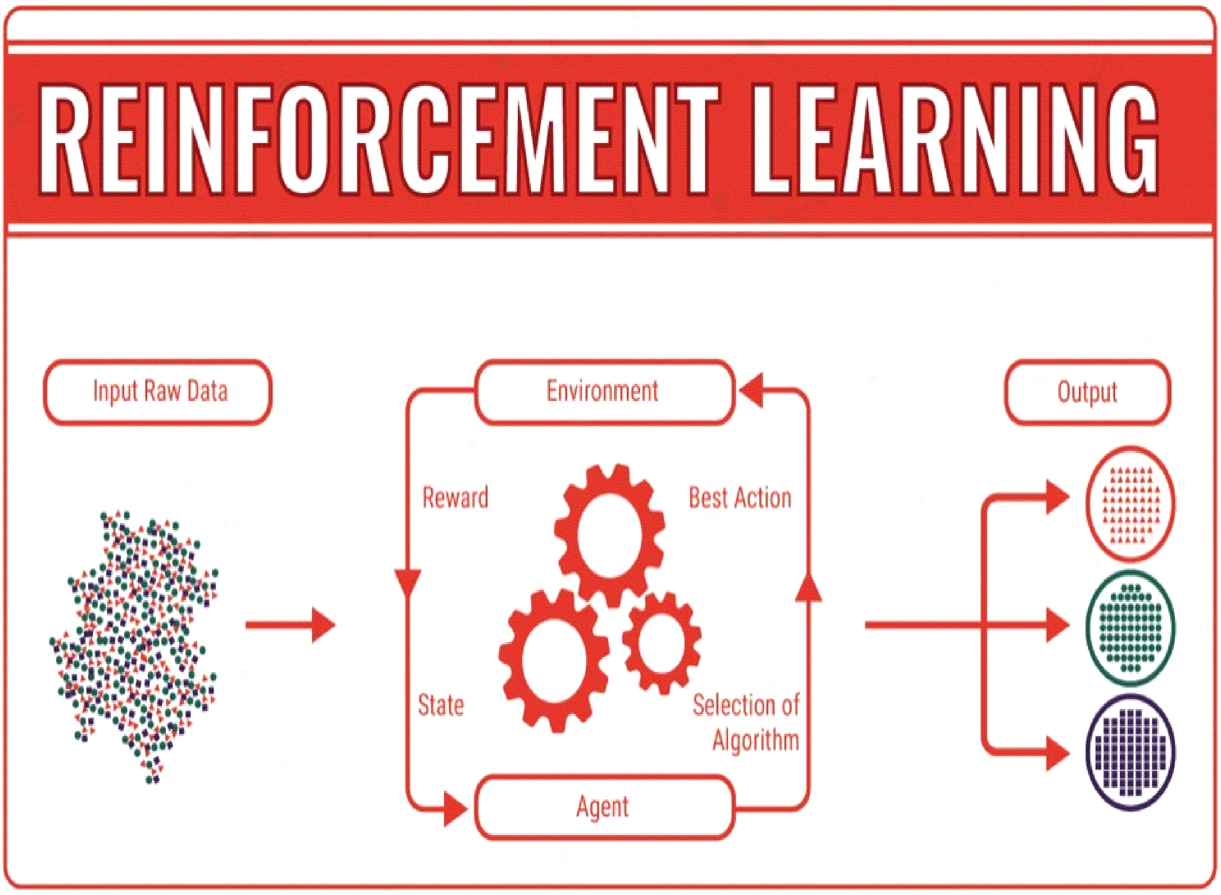


Fig. 07 Reinforcement Learning

Data Preprocessing

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

- **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 scikitlearn. This gives us values between 0 and 1.

- **Standardizing Data**

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

- **Normalizing Data**

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

- **Binarizing Data**

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

- **Mean Removal**

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

- **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 therest values.

- **Label Encoding**

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

Algorithms

Supervised 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.

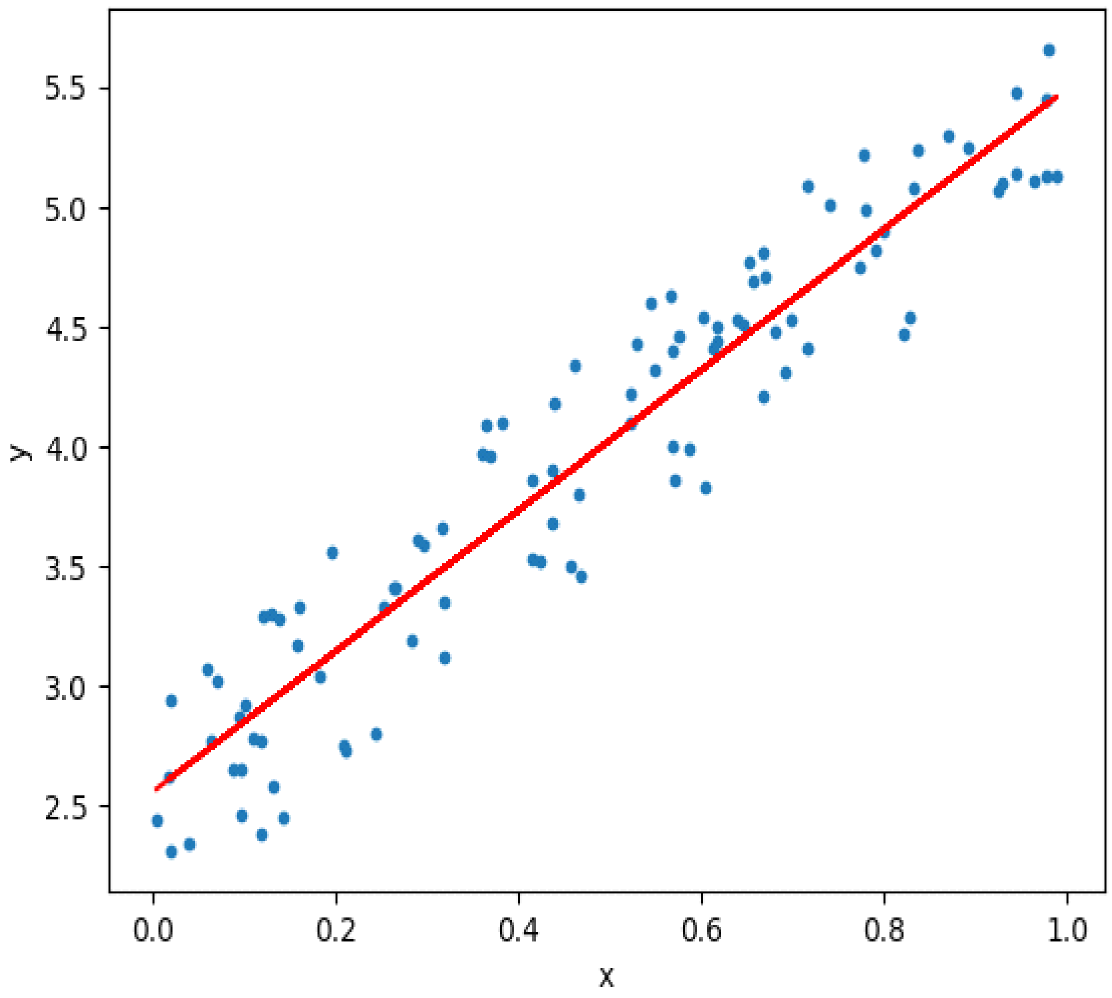


Fig. 08 Linear Regression

- **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:

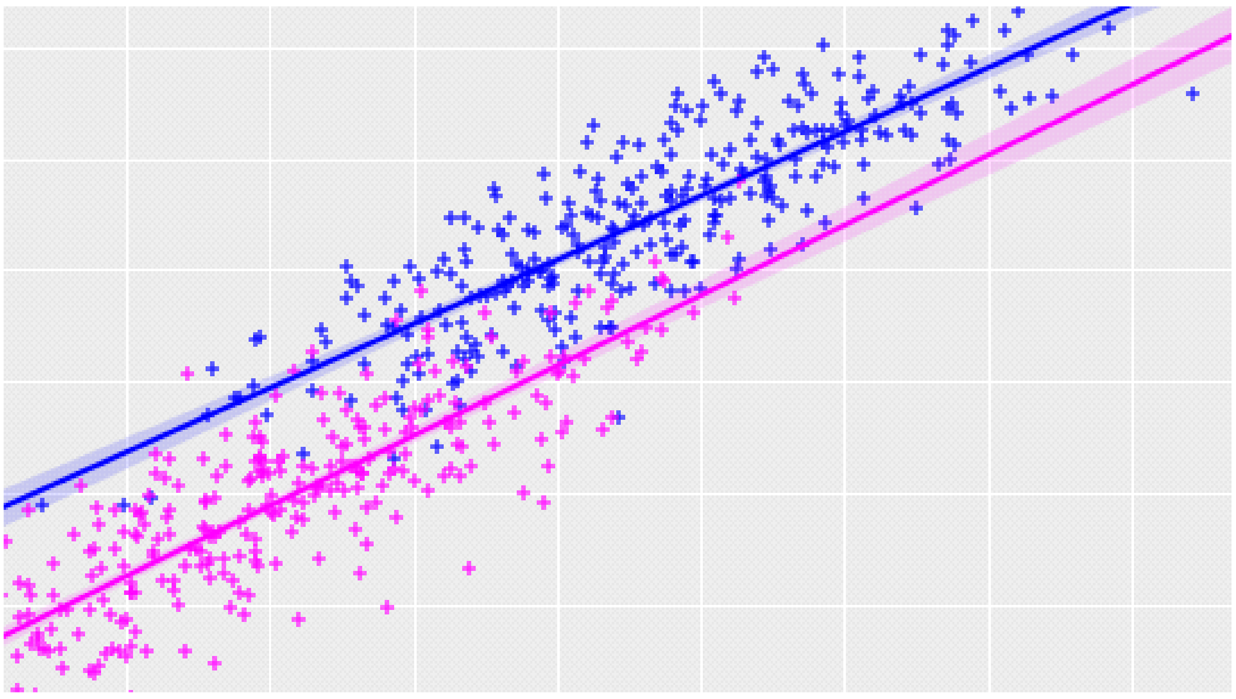


Fig. 09 Multiple Linear Regression

1.3 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.

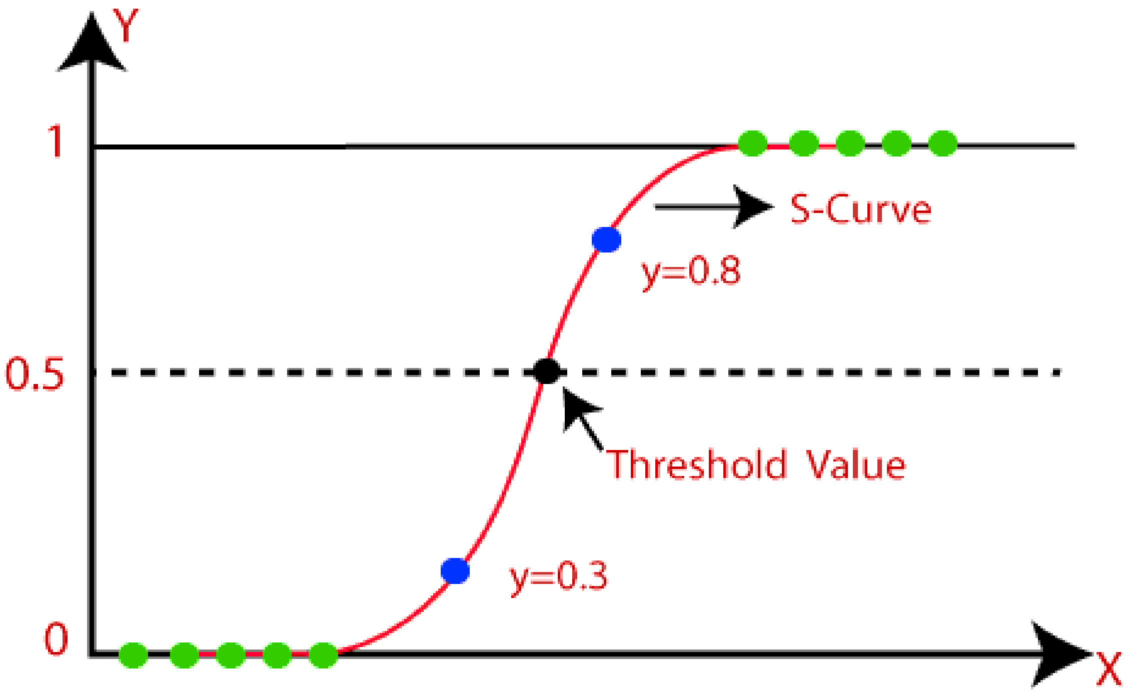


Fig. 10 Logistic Regression

- **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.

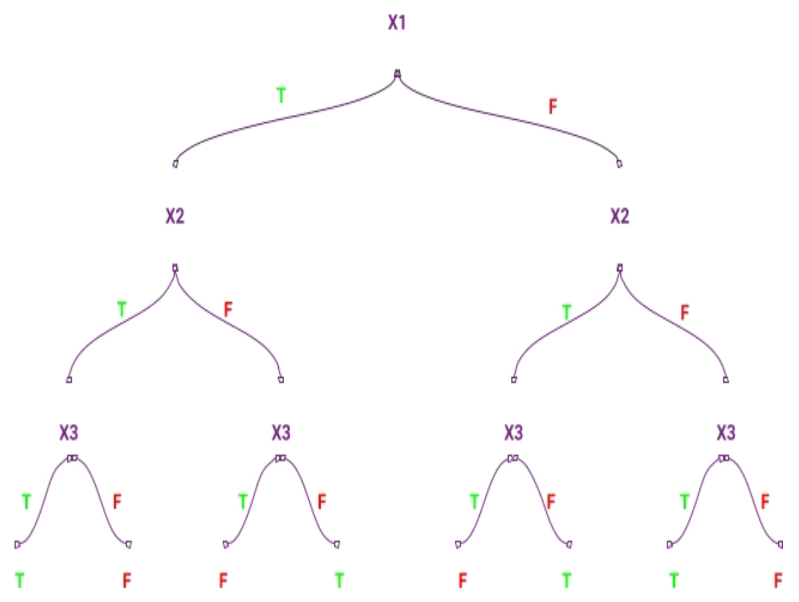


Fig. 11 Decision Tree

• 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.

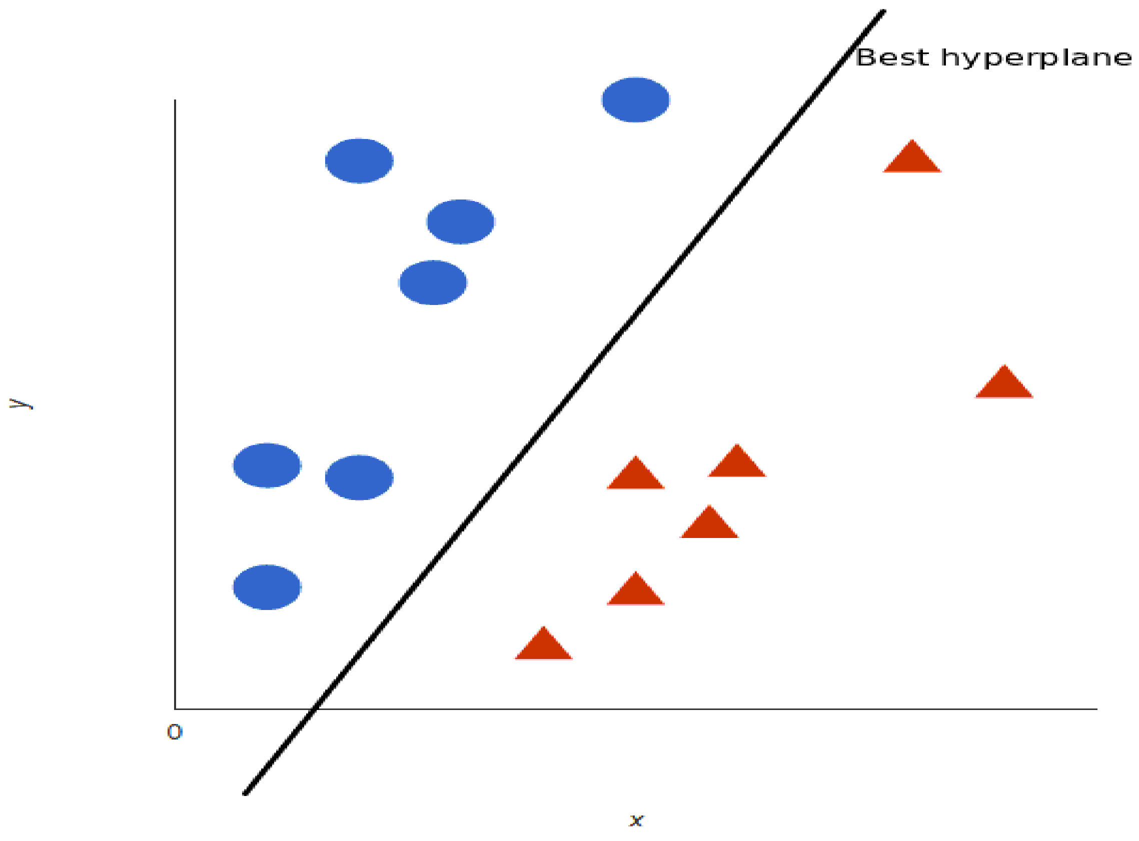


Fig. 12 Support Vector Machine

- **Naïve Bayes Algorithm**

It is a classification method which is based on Bayes' theorem. This assumes independence between predictors. A Naive Bayes classifier will assume that a feature in a class is unrelated to any other. Consider a fruit. This is an apple if it isround, red, and 2.5 inches in diameter. A Naive Bayes classifier will say these characteristics independently contribute to the probability of the fruit being an apple.

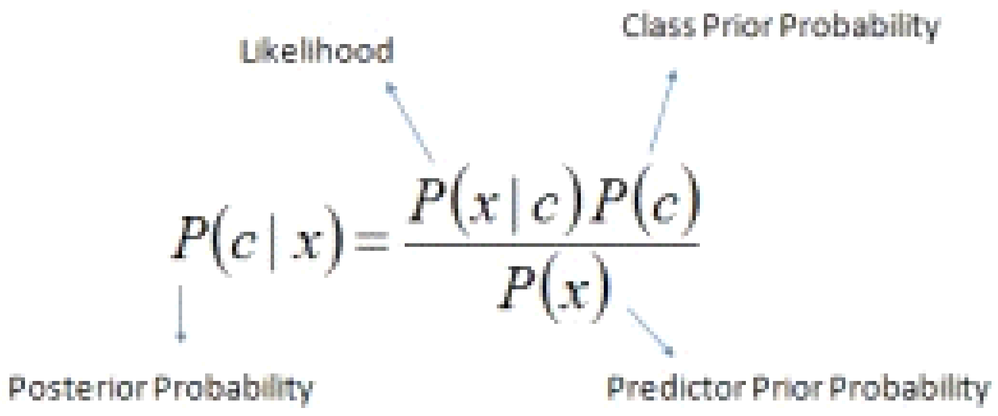
Why is it called Naïve Bayes?

The Naïve Bayes algorithm is comprised of two words Naïve and Bayes, Which can be described as:

- **Naïve**: It is called Naïve because it assumes that the occurrence of a certain feature is independent of the occurrence of other features. Such as if the fruit is identified on the bases of color, shape, and taste, then red, spherical, and sweet fruit is recognized as an apple. Hence each feature individually contributes to identify that it is an apple without depending on each other.
- **Bayes**: It is called Bayes because it depends on the principle of Bayes Theorem

Bayes' Theorem:

- Bayes' theorem is also known as **Bayes' Rule** or **Bayes' law**, which is used to determine the probability of a hypothesis with prior knowledge. It depends on the conditional probability.
- The formula for Bayes' theorem is given as:


Likelihood
Class Prior Probability
$P(c \mid x) = \frac{P(x \mid c)P(c)}{P(x)}$
Posterior Probability
Predictor Prior Probability

$$P(c \mid X) = P(x_1 \mid c) \times P(x_2 \mid c) \times \cdots \times P(x_n \mid c) \times P(c)$$

Fig. 13 Naïve Bayes’ Theorem

Where,

P(A|B) is Posterior probability: Probability of hypothesis A on the observed event B.

P(B|A) is Likelihood probability: Probability of the evidence given that the probability of a hypothesis is true.

P(A) is Prior Probability: Probability of hypothesis before observing the evidence.

P(B) is Marginal Probability: Probability of Evidence.

- **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.

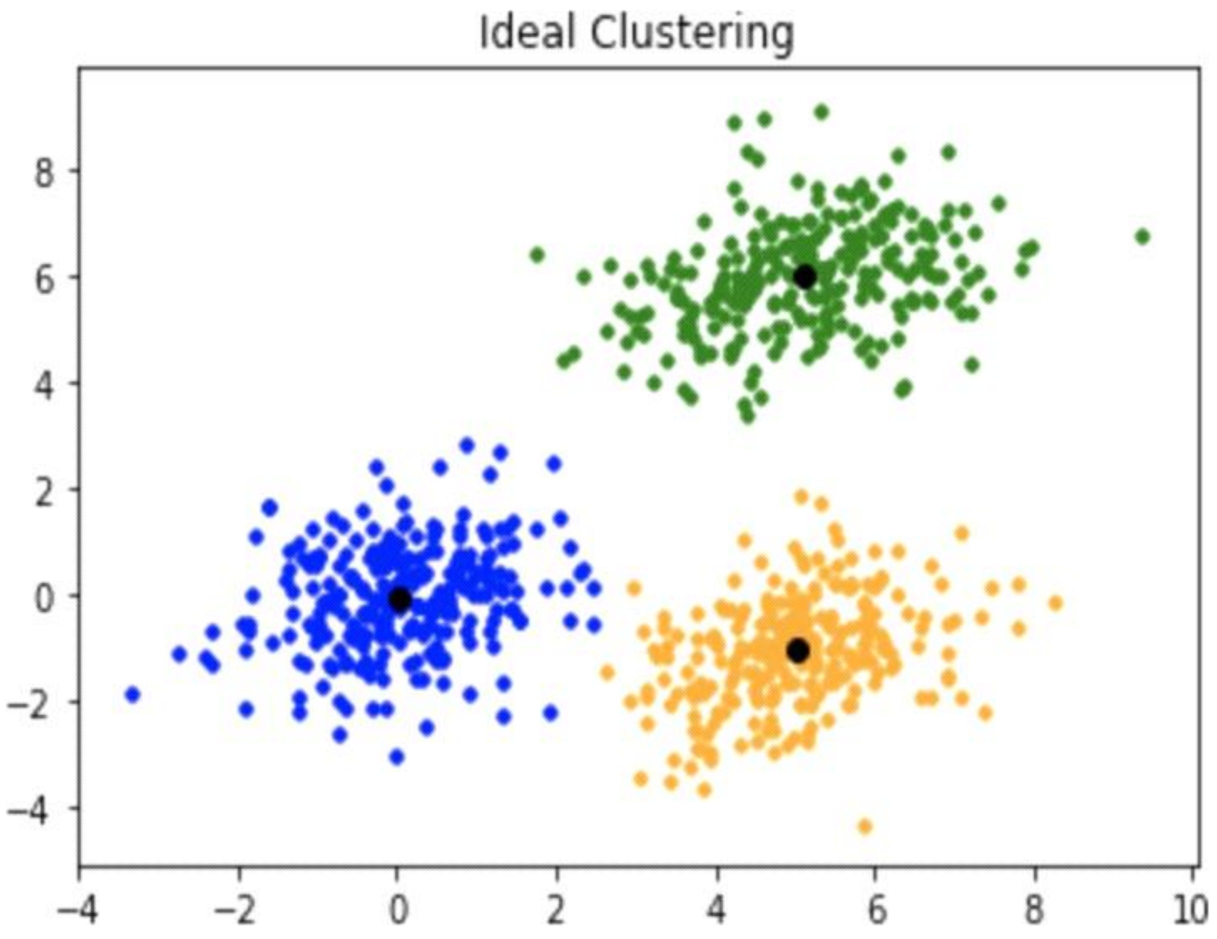


Fig. 14 K-Means Algorithm

- **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.

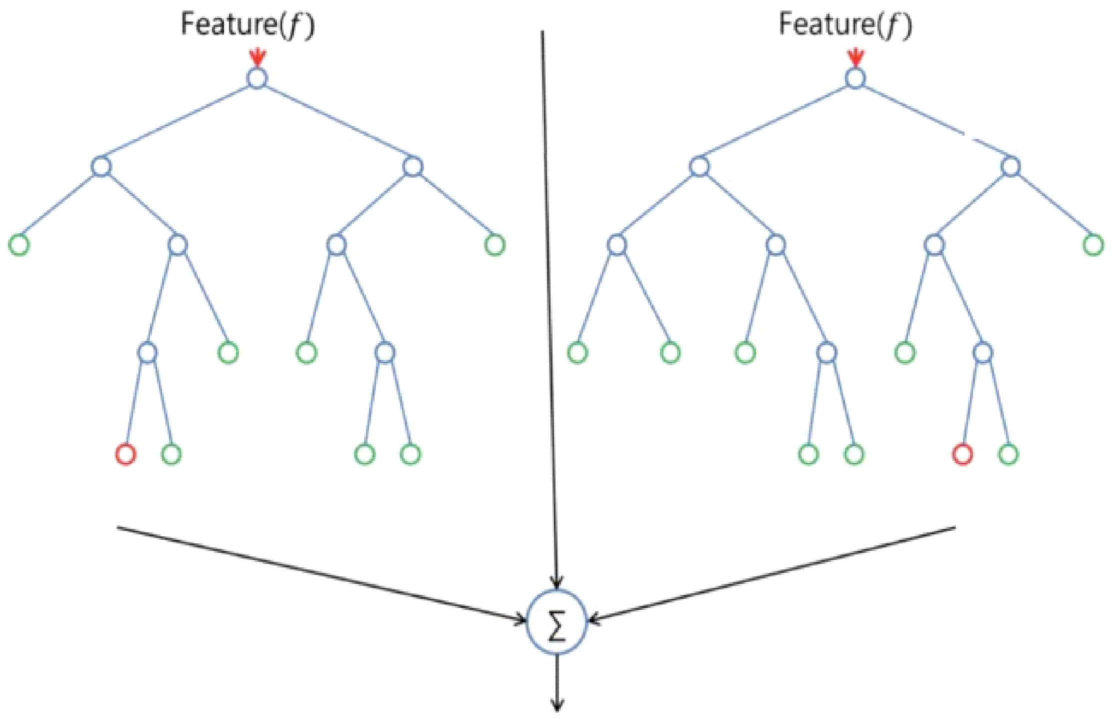


Fig. 15 Random Forest

<u>**Deep Learning**</u>

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.

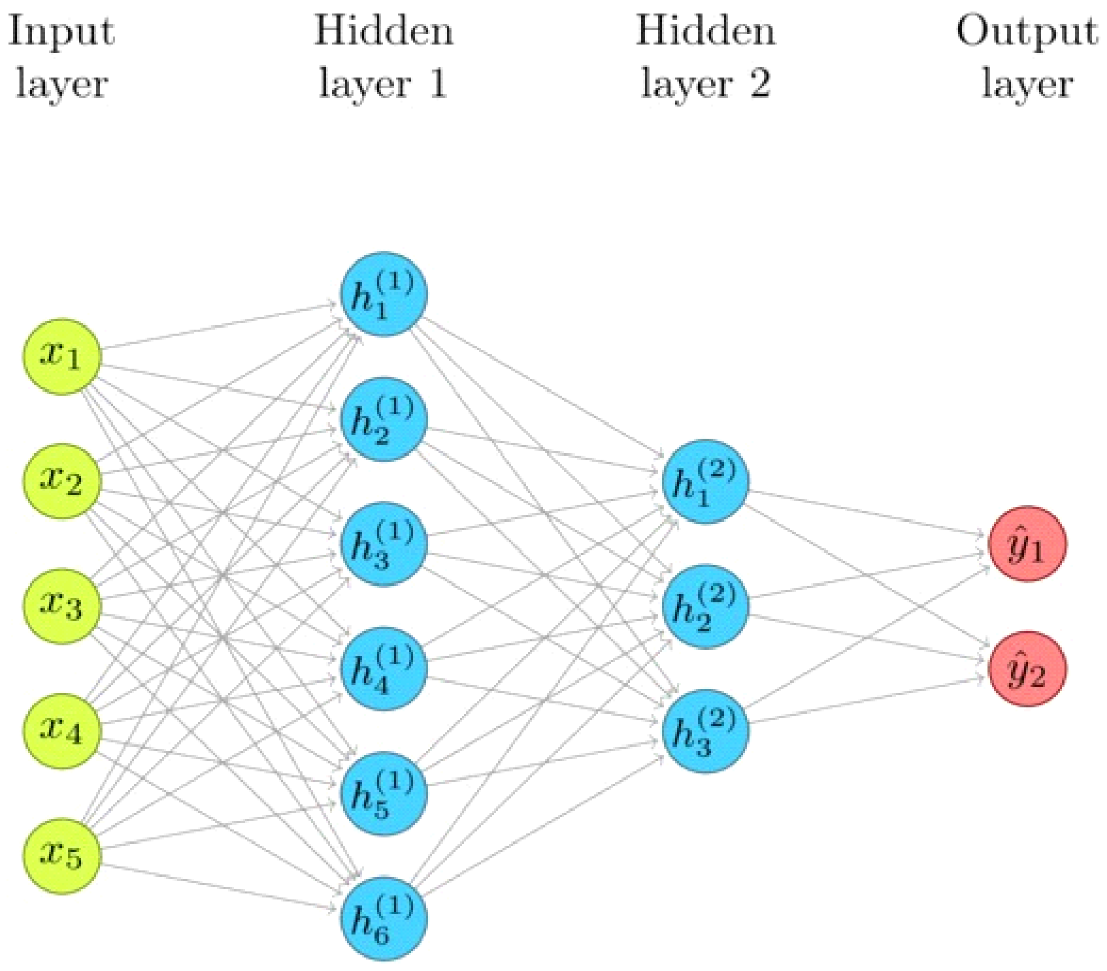

Input layer
Hidden layer 1
Hidden layer 2
Output layer
x_1
x_2
x_3
x_4
x_5
$h_1^{(1)}$
$h_2^{(1)}$
$h_3^{(1)}$
$h_4^{(1)}$
$h_5^{(1)}$
$h_6^{(1)}$
$h_1^{(2)}$
$h_2^{(2)}$
$h_3^{(2)}$
$\hat{y}_1$
$\hat{y}_2$

Fig. 16 ANN

Chapter 2

Project Report

Let us consider a two way relay (TWR) network as shown in Fig. 1, where bidirectional information exchange takes place between two source nodes S_a and S_b via a relay node R. We also consider that S_a and S_b can exchange information via direct link available between them. The transceiver used at the S_a, S_b, and R have hardware impairments. An amplify-and-forward protocol used at the relay R for information processing. The relay node suffers from non-linearity which caused due to high power amplifier. The relay node R harvests energy from the radio- frequency signals, which is received from both the source nodes S_a and S_b. A hybrid receiver is used at the relay node for energy harvesting (EH) and information processing (IP). The hybrid receiver combines both the basic receivers time-switching (TS) and power splitting (PS). The time-switching factor α is an important parameter which decides the fraction of the time period dedicated for EH and IT at the relay node to achieve the optimum performance. Outage probability (OP) is an important performance metric which determines the link failure probability. For the considered network OP at S_i is given as

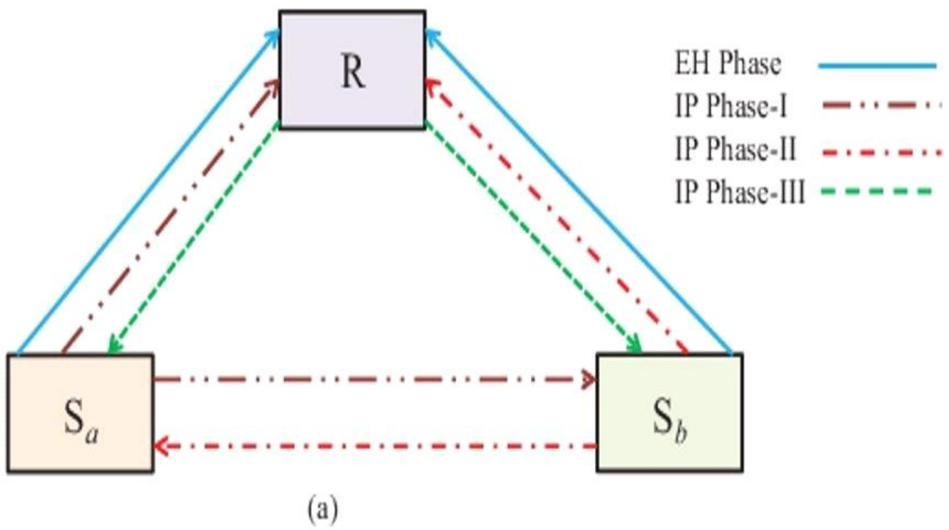


Fig. 17 Two Way Relay Network

Problem Statement:

Find the optimal value of the time-switching factor to design an energyefficient system.

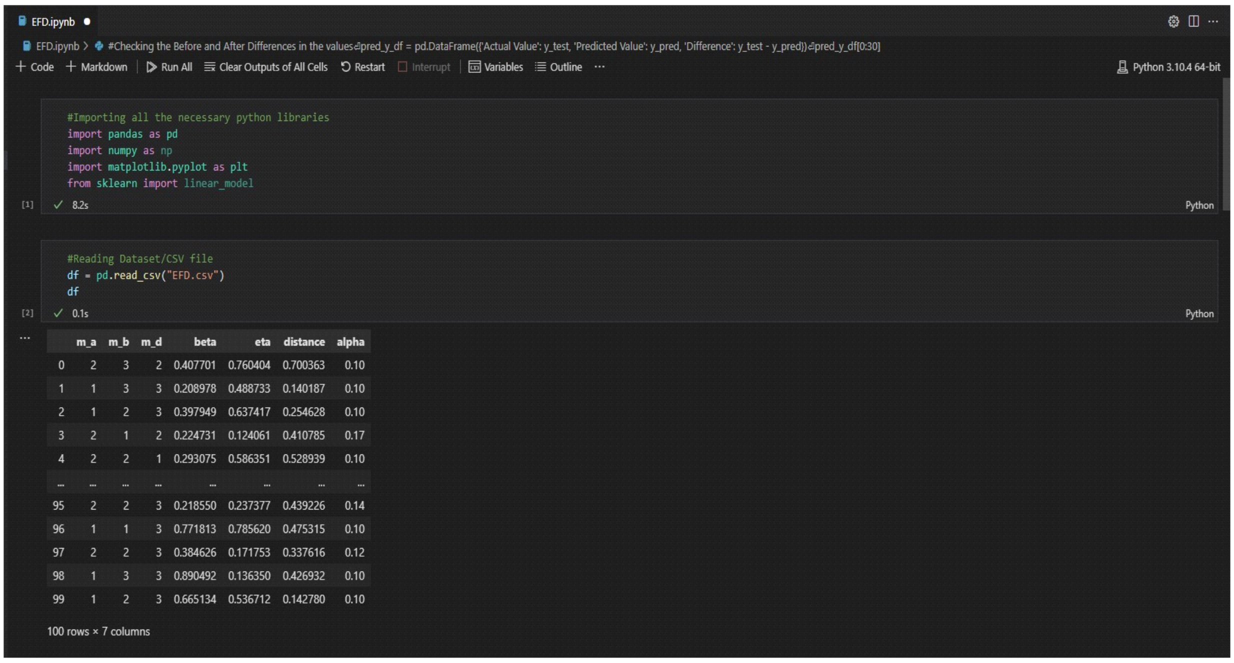
Problem Solution:

- Import python libraries like sklearn, numpy, pandas and linear regressionmodel from sklearn.
- Load the dataset.
- Perform ML model (Linear Regression) on the dataset.
- Predict the value.
- Then test and train the model

.

Chapter 3

Code


EFD.ipynb
EFD.ipynb > #Checking the Before and After Differences in the values⏎pred_y_df = pd.DataFrame({'Actual Value': y_test, 'Predicted Value': y_pred, 'Difference': y_test - y_pred})⏎pred_y_df[0:30]
+ Code + Markdown Run All Clear Outputs of All Cells Restart Interrupt Variables Outline
Python 3.10.4 64-bit

```
#Importing all the necessary python libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn import linear_model
```

[1] ✓ 8.2s Python

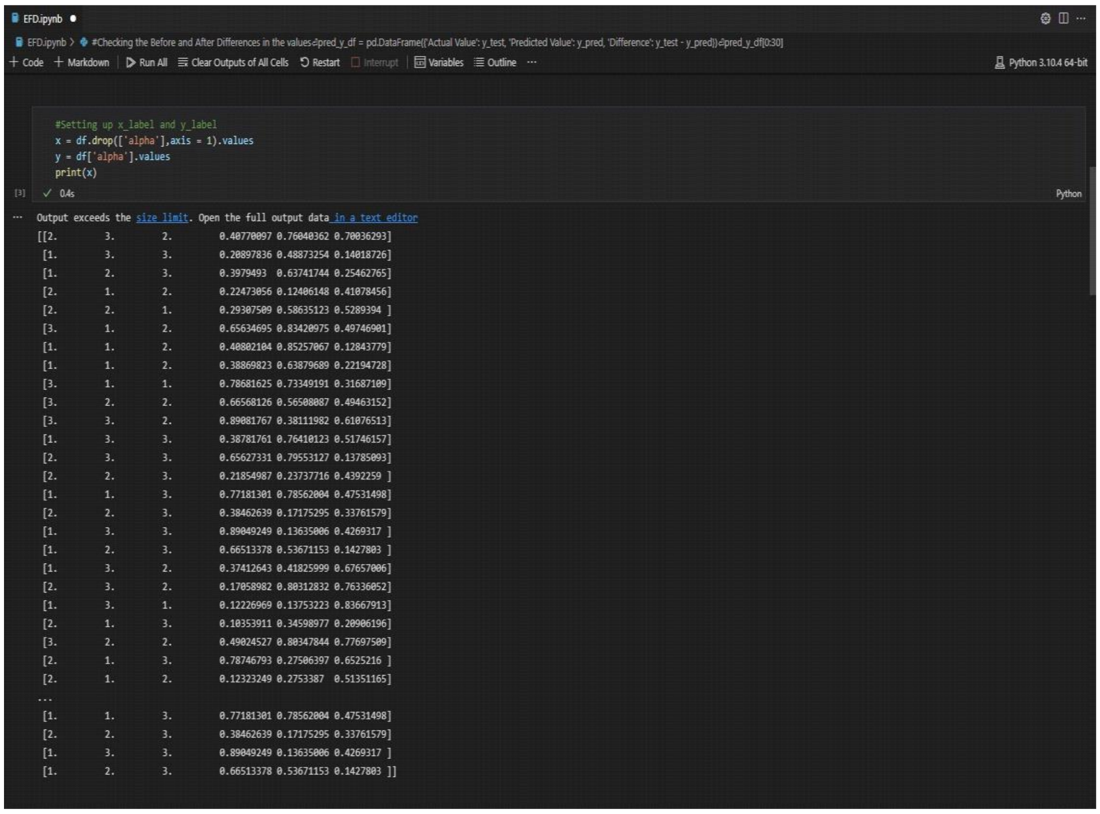
```
#Reading Dataset/CSV file
df = pd.read_csv("EFD.csv")
df
```

[2] ✓ 0.1s Python

```
    m_a  m_b  m_d      beta       eta  distance  alpha
0     2    3    2  0.407701  0.760404  0.700363   0.10
1     1    3    3  0.208978  0.488733  0.140187   0.10
2     1    2    3  0.397949  0.637417  0.254628   0.10
3     2    1    2  0.224731  0.124061  0.410785   0.17
4     2    2    1  0.293075  0.586351  0.528939   0.10
...  ...  ...  ...       ...       ...       ...    ...
95    2    2    3  0.218550  0.237377  0.439226   0.14
96    1    1    3  0.771813  0.785620  0.475315   0.10
97    2    2    3  0.384626  0.171753  0.337616   0.12
98    1    3    3  0.890492  0.136350  0.426932   0.10
99    1    2    3  0.665134  0.536712  0.142780   0.10

100 rows × 7 columns
```

Importing the python libraries and read dataset

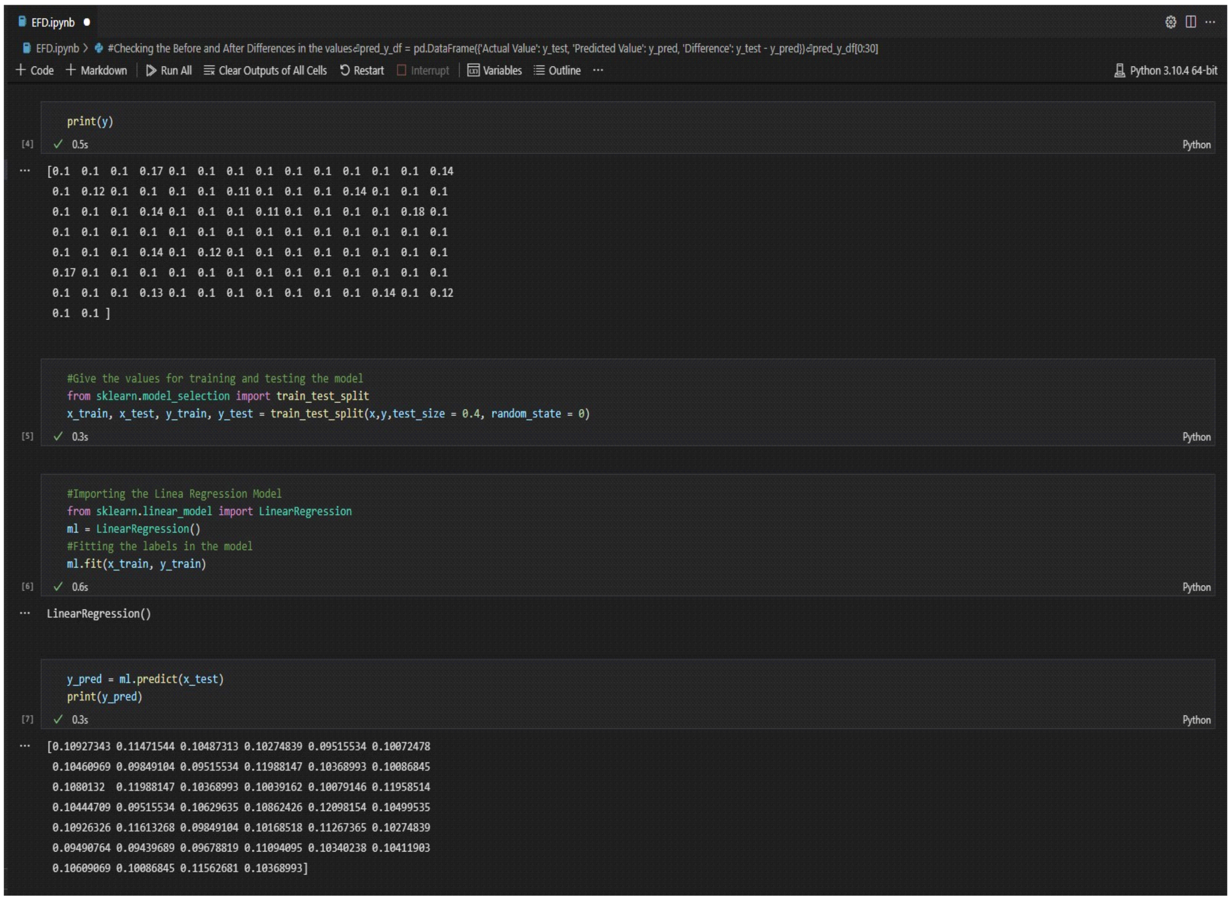
Define x and y labels


EFD.ipynb
EFD.ipynb > #Checking the Before and After Differences in the values⏎pred_y_df = pd.DataFrame({'Actual Value': y_test, 'Predicted Value': y_pred, 'Difference': y_test - y_pred})⏎pred_y_df[0:30]
+ Code + Markdown | Run All Clear Outputs of All Cells Restart Interrupt | Variables Outline ···
Python 3.10.4 64-bit
[3] ✓ 0.4s
Python

```
#Setting up x_label and y_label
x = df.drop(['alpha'],axis = 1).values
y = df['alpha'].values
print(x)
```

```
Output exceeds the size limit. Open the full output data in a text editor
[[2.         3.         2.         0.40770097 0.76040362 0.70036293]
 [1.         3.         3.         0.20897836 0.48873254 0.14018726]
 [1.         2.         3.         0.3979493  0.63741744 0.25462765]
 [2.         1.         2.         0.22473056 0.12406148 0.41078456]
 [2.         2.         1.         0.29307509 0.58635123 0.5289394 ]
 [3.         1.         2.         0.65634695 0.83420975 0.49746901]
 [1.         1.         2.         0.40802104 0.85257067 0.12843779]
 [1.         1.         2.         0.38869823 0.63879689 0.22194728]
 [3.         1.         1.         0.78681625 0.73349191 0.31687109]
 [3.         2.         2.         0.66568126 0.56508087 0.49463152]
 [3.         3.         2.         0.89081767 0.38111982 0.61076513]
 [1.         3.         3.         0.38781761 0.76410123 0.51746157]
 [2.         3.         3.         0.65627331 0.79553127 0.13785093]
 [2.         2.         3.         0.21854987 0.23737716 0.4392259 ]
 [1.         1.         3.         0.77181301 0.78562004 0.47531498]
 [2.         2.         3.         0.38462639 0.17175295 0.33761579]
 [1.         3.         3.         0.89049249 0.13635006 0.4269317 ]
 [1.         2.         3.         0.66513378 0.53671153 0.1427803 ]
 [1.         3.         2.         0.37412643 0.41825999 0.67657006]
 [2.         3.         2.         0.17058982 0.80312832 0.76336052]
 [1.         3.         1.         0.12226969 0.13753223 0.83667913]
 [2.         1.         3.         0.10353911 0.34598977 0.20906196]
 [3.         2.         2.         0.49024527 0.80347844 0.77697509]
 [2.         1.         3.         0.78746793 0.27506397 0.6525216 ]
 [2.         1.         2.         0.12323249 0.2753387  0.51351165]
...
 [1.         1.         3.         0.77181301 0.78562004 0.47531498]
 [2.         2.         3.         0.38462639 0.17175295 0.33761579]
 [1.         3.         3.         0.89049249 0.13635006 0.4269317 ]
 [1.         2.         3.         0.66513378 0.53671153 0.1427803 ]]
```

Training and fitting labels in model


EFD.ipynb
EFD.ipynb > #Checking the Before and After Differences in the values⏎pred_y_df = pd.DataFrame({'Actual Value': y_test, 'Predicted Value': y_pred, 'Difference': y_test - y_pred})⏎pred_y_df[0:30]
+ Code + Markdown | Run All Clear Outputs of All Cells Restart Interrupt | Variables Outline ···
Python 3.10.4 64-bit
[4] ✓ 0.5s
Python
[5] ✓ 0.3s
Python
[6] ✓ 0.6s
Python
[7] ✓ 0.3s
Python

```
print(y)
```

```
[0.1  0.1  0.1  0.17 0.1  0.1  0.1  0.1  0.1  0.1  0.1  0.1  0.1  0.14
 0.1  0.12 0.1  0.1  0.1  0.1  0.11 0.1  0.1  0.1  0.14 0.1  0.1  0.1
 0.1  0.1  0.1  0.14 0.1  0.1  0.1  0.11 0.1  0.1  0.1  0.1  0.18 0.1
 0.1  0.1  0.1  0.1  0.1  0.1  0.1  0.1  0.1  0.1  0.1  0.1  0.1  0.1
 0.1  0.1  0.1  0.14 0.1  0.12 0.1  0.1  0.1  0.1  0.1  0.1  0.1  0.1
 0.17 0.1  0.1  0.1  0.1  0.1  0.1  0.1  0.1  0.1  0.1  0.1  0.1  0.1
 0.1  0.1  0.1  0.13 0.1  0.1  0.1  0.1  0.1  0.1  0.1  0.14 0.1  0.12
 0.1  0.1 ]
```

```
#Give the values for training and testing the model
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x,y,test_size = 0.4, random_state = 0)
```

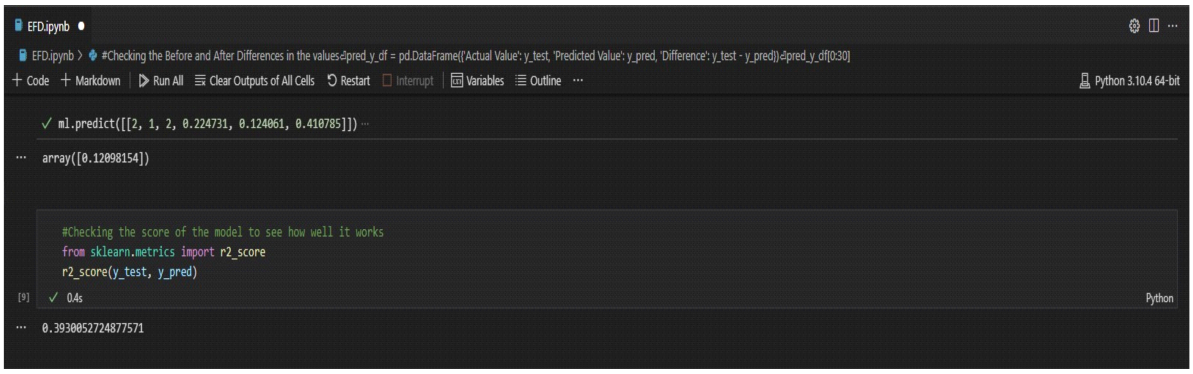
```
#Importing the Linea Regression Model
from sklearn.linear_model import LinearRegression
ml = LinearRegression()
#Fitting the labels in the model
ml.fit(x_train, y_train)
```

```
LinearRegression()
```

```
y_pred = ml.predict(x_test)
print(y_pred)
```

```
[0.10927343 0.11471544 0.10487313 0.10274839 0.09515534 0.10072478
 0.10460969 0.09849104 0.09515534 0.11988147 0.10368993 0.10086845
 0.1080132  0.11988147 0.10368993 0.10039162 0.10079146 0.11958514
 0.10444709 0.09515534 0.10629635 0.10862426 0.12098154 0.10499535
 0.10926326 0.11613268 0.09849104 0.10168518 0.11267365 0.10274839
 0.09490764 0.09439689 0.09678819 0.11094095 0.10340238 0.10411903
 0.10609069 0.10086845 0.11562681 0.10368993]
```

Checking score



EFD.ipynb > #Checking the Before and After Differences in the values⏎pred_y_df = pd.DataFrame({'Actual Value': y_test, 'Predicted Value': y_pred, 'Difference': y_test - y_pred})⏎pred_y_df[0:30]

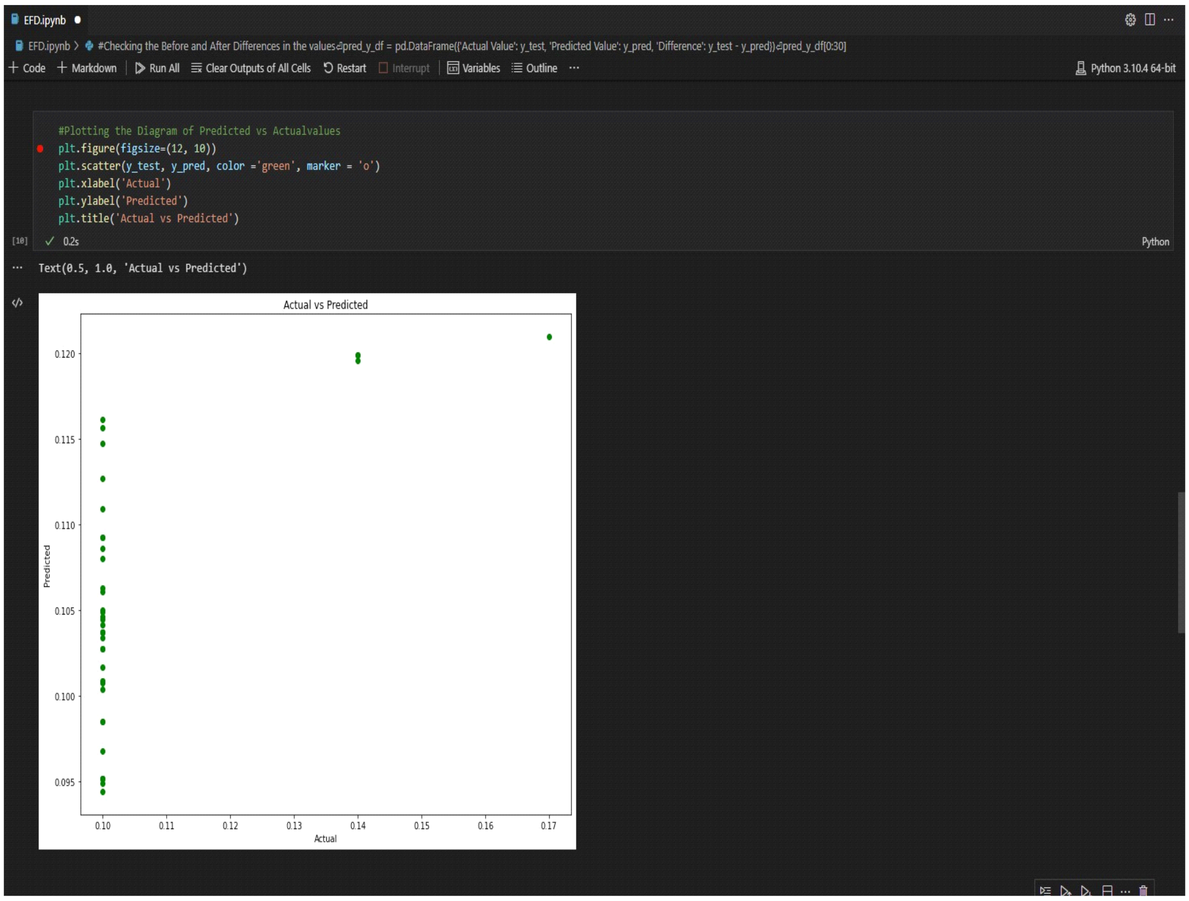
```
ml.predict([[2, 1, 2, 0.224731, 0.124061, 0.410785]])

array([0.12098154])
```

```
#Checking the score of the model to see how well it works
from sklearn.metrics import r2_score
r2_score(y_test, y_pred)
```

[9] ✓ 0.4s

```
0.3930052724877571
```

Plotting the values



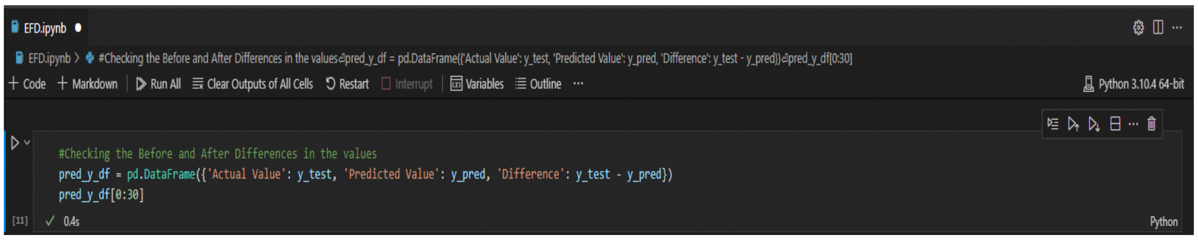
EFD.ipynb > #Checking the Before and After Differences in the values⏎pred_y_df = pd.DataFrame({'Actual Value': y_test, 'Predicted Value': y_pred, 'Difference': y_test - y_pred})⏎pred_y_df[0:30]

```
#Plotting the Diagram of Predicted vs Actualvalues
plt.figure(figsize=(12, 10))
plt.scatter(y_test, y_pred, color ='green', marker = 'o')
plt.xlabel('Actual')
plt.ylabel('Predicted')
plt.title('Actual vs Predicted')
```

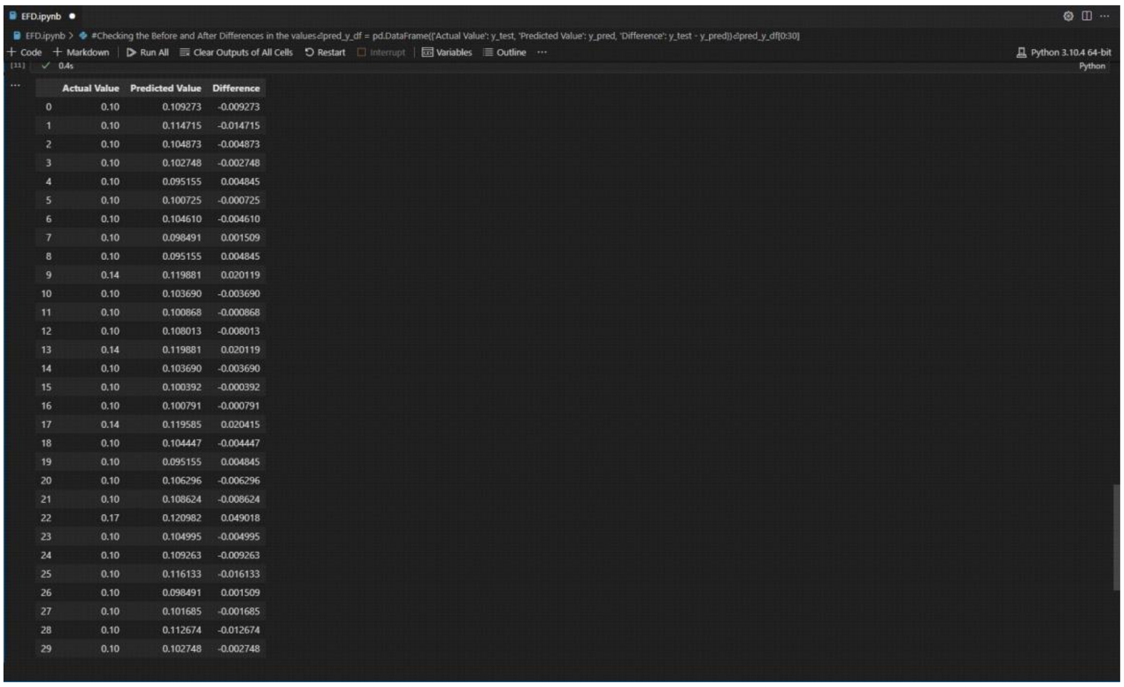
[10] ✓ 0.2s

```
Text(0.5, 1.0, 'Actual vs Predicted')
```

Differencing the actual and predicted values


EFD.ipynb
EFD.ipynb > #Checking the Before and After Differences in the values⏎pred_y_df = pd.DataFrame({'Actual Value': y_test, 'Predicted Value': y_pred, 'Difference': y_test - y_pred})⏎pred_y_df[0:30]
+ Code + Markdown | Run All Clear Outputs of All Cells Restart Interrupt | Variables Outline ...
Python 3.10.4 64-bit
[11]
✓ 0.4s
Python

```
#Checking the Before and After Differences in the values
pred_y_df = pd.DataFrame({'Actual Value': y_test, 'Predicted Value': y_pred, 'Difference': y_test - y_pred})
pred_y_df[0:30]
```


EFD.ipynb
EFD.ipynb > #Checking the Before and After Differences in the values⏎pred_y_df = pd.DataFrame({'Actual Value': y_test, 'Predicted Value': y_pred, 'Difference': y_test - y_pred})⏎pred_y_df[0:30]
+ Code + Markdown | Run All Clear Outputs of All Cells Restart Interrupt | Variables Outline ...
Python 3.10.4 64-bit
[11] ✓ 0.4s
Python

	Actual Value	Predicted Value	Difference
0	0.10	0.109273	-0.009273
1	0.10	0.114715	-0.014715
2	0.10	0.104873	-0.004873
3	0.10	0.102748	-0.002748
4	0.10	0.095155	0.004845
5	0.10	0.100725	-0.000725
6	0.10	0.104610	-0.004610
7	0.10	0.098491	0.001509
8	0.10	0.095155	0.004845
9	0.14	0.119881	0.020119
10	0.10	0.103690	-0.003690
11	0.10	0.100868	-0.000868
12	0.10	0.108013	-0.008013
13	0.14	0.119881	0.020119
14	0.10	0.103690	-0.003690
15	0.10	0.100392	-0.000392
16	0.10	0.100791	-0.000791
17	0.14	0.119585	0.020415
18	0.10	0.104447	-0.004447
19	0.10	0.095155	0.004845
20	0.10	0.106296	-0.006296
21	0.10	0.108624	-0.008624
22	0.17	0.120982	0.049018
23	0.10	0.104995	-0.004995
24	0.10	0.109263	-0.009263
25	0.10	0.116133	-0.016133
26	0.10	0.098491	0.001509
27	0.10	0.101685	-0.001685
28	0.10	0.112674	-0.012674
29	0.10	0.102748	-0.002748

Chapter 4

Conclusion

By using the ML models we can build the system where you can train and test yourdata set and get the machine predicted output which is better and less time consuming than humans. In the two-way relay network we faced the time- switching factor problem (α) so in this code we used linear regression model to solve the problem and predict the value of time-switching factor (α). By solving this problem we try to predict the almost accurate result to achieve the optimum performance of Relay which suffers from non-linearity.

The predicted value of time switching factor alpha is showing in difference images output.

The r2_score is: 0.3988 which is 39% accurate

Chapter 5

References

- https://www.geeksforgeeks.org/ml-linear-regression/
- https://youtu.be/WngoqVB6cXw
- https://www.researchgate.net/figure/A-two-way-relay-network_fig1_220734122
- https://www.sciencedirect.com/science/article/abs/pii/S1434841116308925
- https://www.hindawi.com/journals/ijap/2015/563737/
- https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Linear
- https://www.javatpoint.com/linear-regression-in-machine-learning