Parul® University

NAAC A++

Chapter-4: Regression, Classification and Clustering

Dr. Vinod Patidar
Associate Professor
Department of Computer Science and Engineering

Parul® University

NAAC A++

INDEX

Content

1. Linear Regression

- Linear regression is a data analysis technique that predicts the value of unknown data by using another related and known data value.

- It mathematically models the unknown or dependent variable and the known or independent variable as a linear equation.

Linear Regression…

- For instance, suppose that you have data about your expenses and income for last year. Linear regression techniques analyze this data and determine that your expenses are half your income.
- They then calculate an unknown future expense by halving a future known income.

Why is Linear regression Important ?

- Linear regression models are relatively simple and provide an easy-to-interpret mathematical formula to generate predictions.
- Linear regression is an established statistical technique and applies easily to software and computing. Businesses use it to reliably and predictably convert raw data into business intelligence and actionable insights.
- Scientists in many fields, including biology and the behavioral, environmental, and social sciences, use linear regression to conduct preliminary data analysis and predict future trends.

Types of Linear Regression

1. Simple Linear Regression

2. Multiple Linear regression

Types of Linear Regression…

Simple Linear Regression:

If a single independent variable is used to predict the value of a numerical dependent variable, then such a Linear Regression algorithm is called Simple Linear Regression.

Multiple Linear regression:

If more than one independent variable is used to predict the value of a numerical dependent variable, then such a Linear Regression algorithm is called Multiple Linear Regression.

Simple Linear Regression

- Simple linear regression is used to estimate the relationship between two quantitative variables. You can use simple linear regression when you want to know:
    - How strong the relationship is between two variables (e.g., the relationship between rainfall and soil erosion).
    - The value of the dependent variable at a certain value of the independent variable (e.g., the amount of soil erosion at a certain level of rainfall).

Assumptions of Linear Regression

- Regression analysis is commonly used for modeling the relationship between a single dependent variable Y and one or more predictors. When we have one predictor, we call this "simple" linear regression:

$$E[Y] = \beta 0 + \beta 1X$$

- That is, the expected value of Y is a straight-line function of X. The betas are selected by choosing the line that minimizing the squared distance between each Y value and the line of best fit. The betas are chose such that they minimize this expression:

Assumptions of Linear Regression…

- **Linearity**: The relationship between X and Y must be linear. Check this assumption by examining a scatterplot of x and y.
- **Independence of errors**: There is not a relationship between the residuals and the Y variable;
  - In other words, Y is independent of errors. Check this assumption by examining a scatterplot of “residuals versus fits”; the correlation should be approximately 0.
  - In other words, there should not look like there is a relationship.

Assumptions of Linear Regression…

- Simple linear regression is used to find out the best relationship between a single input variable (predictor, independent variable, input feature, input parameter) & output variable (predicted, dependent variable, output feature, output parameter) provided that both variables are continuous in nature.

- This relationship represents how an input variable is related to the output variable and how it is represented by a straight line.

Assumptions of Linear Regression…

$$y= a0+a1x+ \varepsilon$$

Where,

a0 = It is the intercept of the Regression line (can be obtained putting x=0)

a1= It is the slope of the regression line, which tells whether the line is increasing or decreasing.

ε = The error term. (For a good model it will be negligible)

Assumptions of Linear Regression…

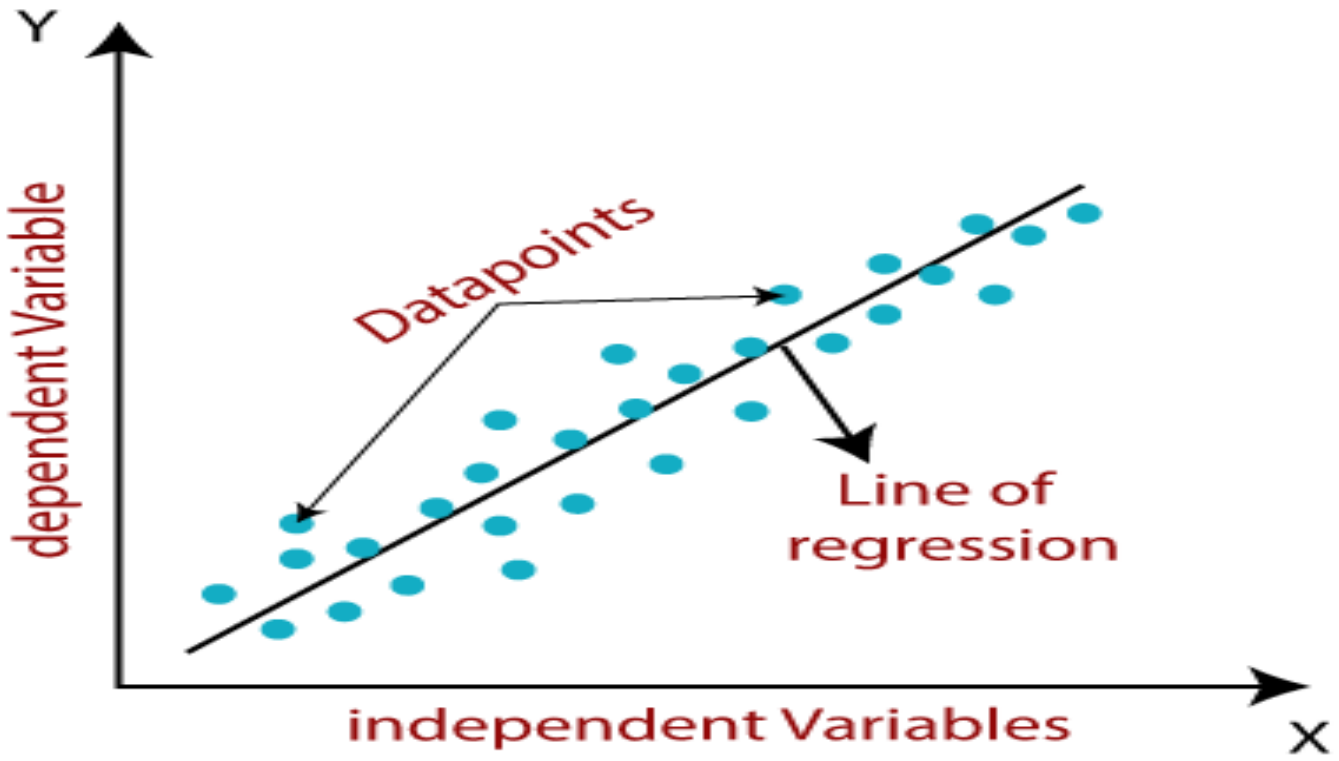


Fig. 1.1: Linear Regression

Assumptions of Linear Regression…

Y= Dependent Variable (Target Variable).

X= Independent Variable (predictor Variable).

a0= intercept of the line (Gives an additional degree of freedom).

a1 = Linear regression coefficient (scale factor to each input value).

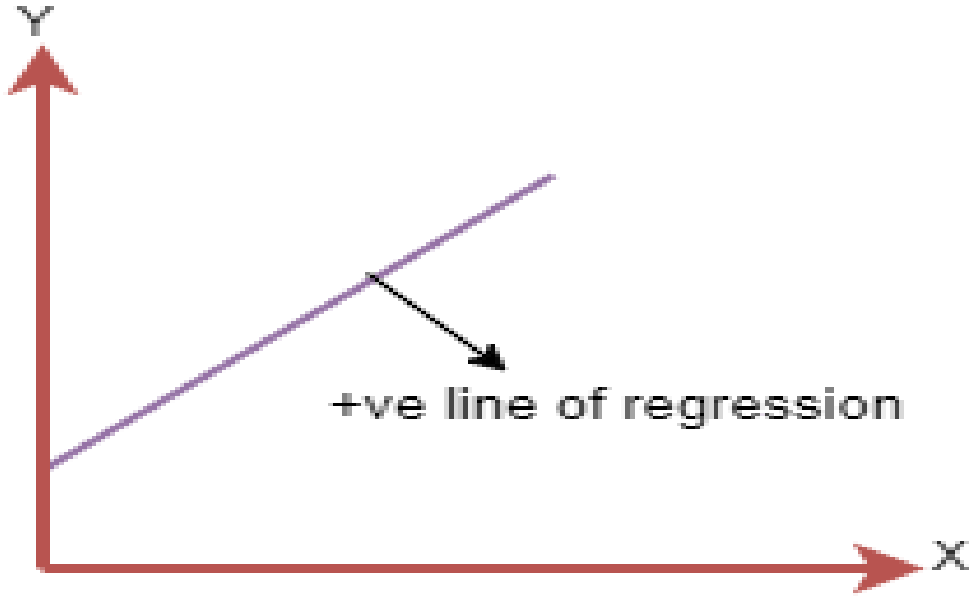
ε = random error.

Assumptions of Linear Regression…

A linear line showing the relationship between the dependent and independent variables is called a **regression line**.
A regression line can show two types of relationship:

Positive Linear Relationship:
If the dependent variable increases on the Y-axis and independent variable increases on X-axis, then such a relationship is termed as a Positive linear relationship.

Linear Regression Lines-



The line equation will be: $Y = a_0 + a_1 x$

Fig. 1.2: Positive Linear Regression Line

Linear regression Lines…

Negative Linear Relationship:

If the dependent variable decreases on the Y-axis and independent variable increases on the X-axis, then such a relationship is called a negative linear relationship

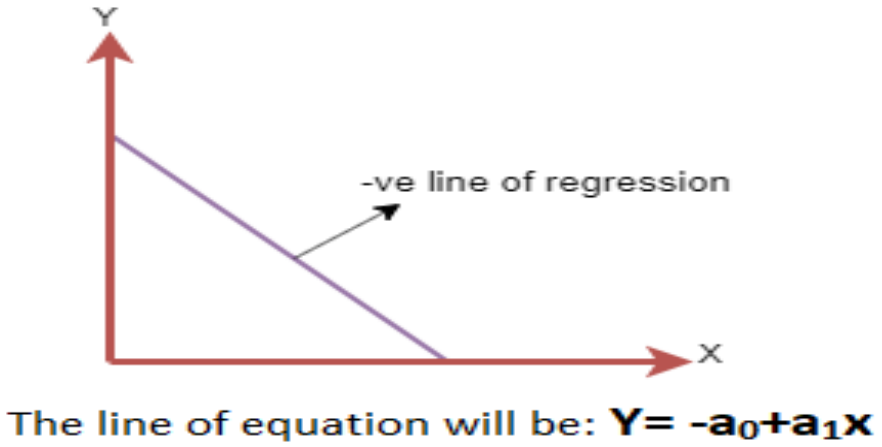


Fig. 1.3: Negative Linear Regression Line

Implementation

Here we are taking a dataset that has two variables: salary (dependent variable) and experience (Independent variable).
The goals of this problem is:

We want to find out if there is any correlation between these two variables.
We will find the best fit line for the dataset.
How the dependent variable is changing by changing the independent variable.
In this section, we will create a Simple Linear Regression model to find out the best fitting line for representing the relationship between these two variables.

Implementation…

Step-1: Data Pre-processing

The first step for creating the Simple Linear Regression model is data pre-processing. We have already done it earlier in this tutorial. But there will be some changes, which are given in the below steps:

First, we will import the three important libraries, which will help us for loading the dataset, plotting the graphs, and creating the Simple Linear Regression model.

Implementation…

```
import numpy as nm

import matplotlib.pyplot as mtp

import pandas as pd
```

Next, we will load the dataset into our code:

```
data_set= pd.read_csv('Salary_Data.csv')
```

Implementation…

Index	YearsExperience	Salary
0	1	32383
1	1.1	45207
2	1.3	39751
3	2	43525
4	2.2	39891
5	2.7	56642
6	3	60150
7	3.2	54445
8	3.2	64445
9	3.7	57189
10	3.9	63218
11	4	55794
12	4	56957
13	4.1	57081

Implementation…

Here above output shows the dataset, which has two variables: Salary and Experience.

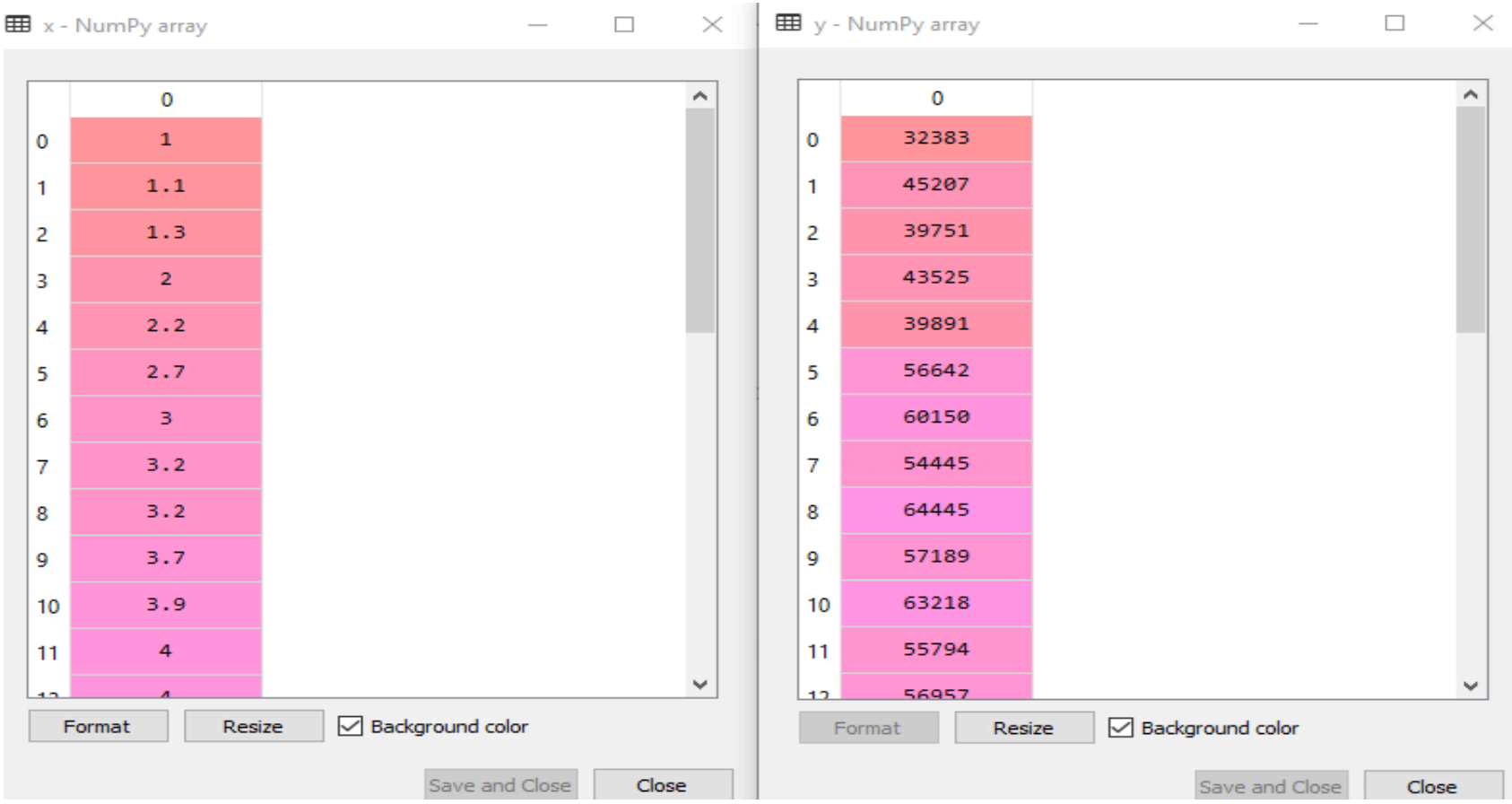
- he independent variable After that, we need to extract the dependent and independent variables from the given dataset.

- T is years of experience, and the dependent variable is salary.

- Below is code for it:

```
x= data_set.iloc[:, :-1].values
y= data_set.iloc[:, 1].values
```

Implementation…



Implementation…

- In the above lines of code, for x variable, we have taken -1 value since we want to remove the last column from the dataset.
- For y variable, we have taken 1 value as a parameter, since we want to extract the second column and indexing starts from the zero

Implementation…

- In the above output image, we can see the X (independent) variable and Y (dependent) variable has been extracted from the given dataset.
- Next, we will split both variables into the test set and training set. We have 30 observations, so we will take 20 observations for the training set and 10 observations for the test set.
- We are splitting our dataset so that we can train our model using a training dataset and then test the model using a test dataset.

Implementation…

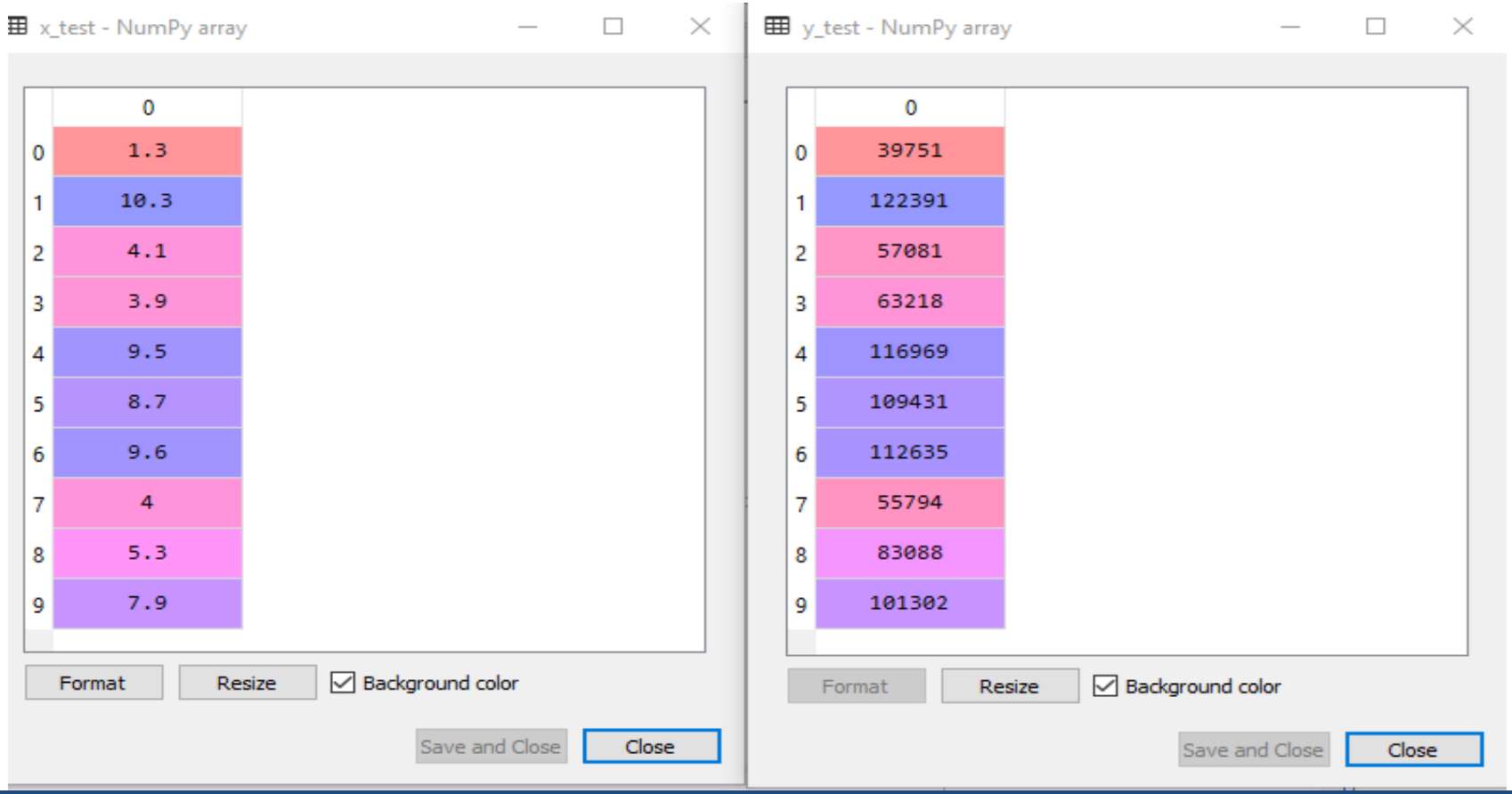
- Splitting the dataset into training and test set.

- from sklearn.model_selection import train_test_split
- x_train, x_test, y_train, y_test= train_test_split(x, y, test_size= 1/3, random_state=0)

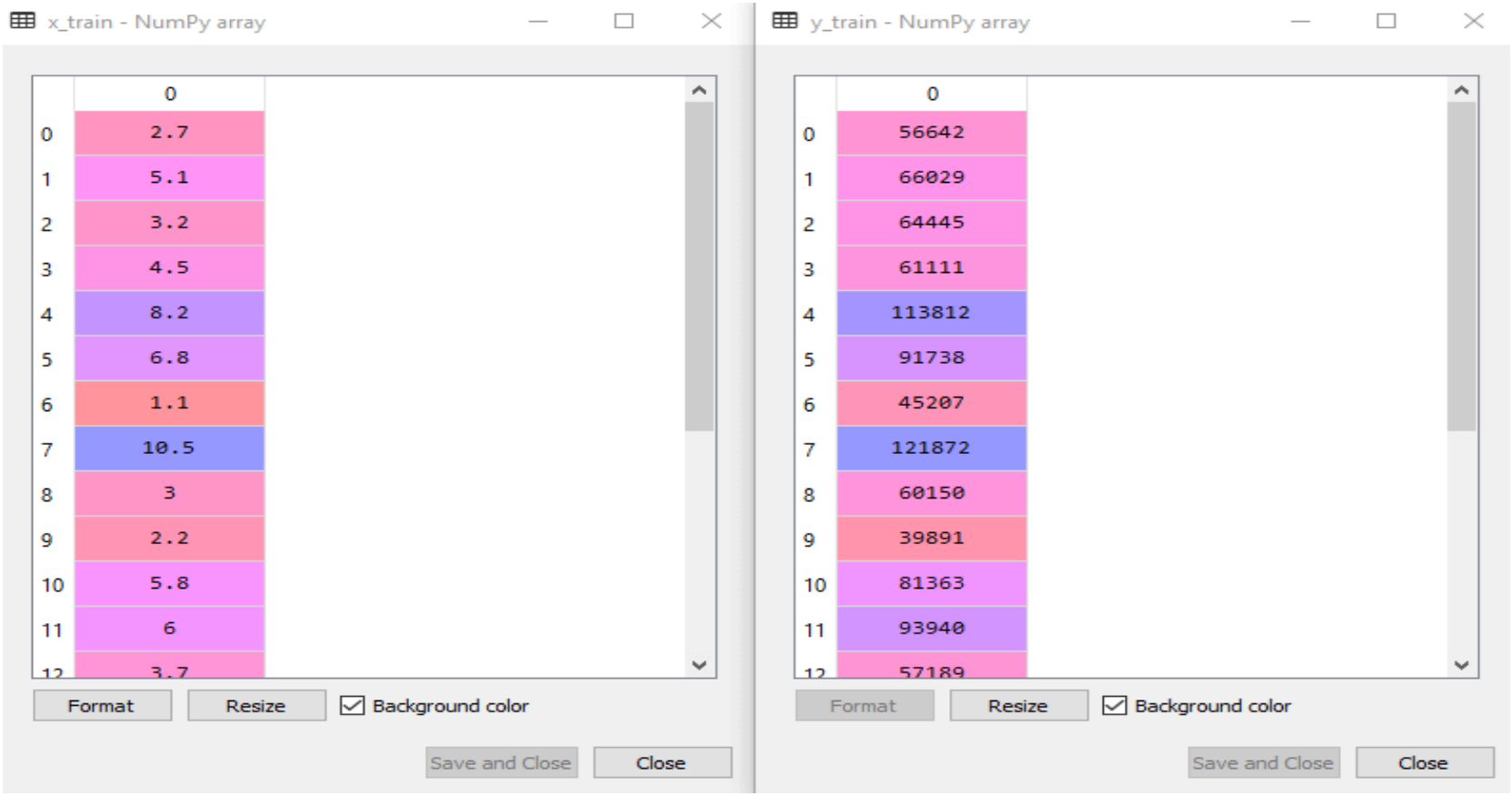
- By executing the above code, we will get x-test, x-train and y-test, y-train dataset.

Consider the below images:

Implementation…



Implementation…



Implementation…

- For simple linear Regression, we will not use Feature Scaling. Because Python libraries take care of it for some cases, so we don't need to perform it here.
- Now, our dataset is well prepared to work on it and we are going to start building a Simple Linear Regression model for the given problem.

Step-2: Fitting the Simple Linear Regression to the Training Set:

Now the second step is to fit our model to the training dataset. To do so, we will import the **Linear Regression** class of the **linear_model** library from the **scikit learn**. After importing the class, we are going to create an object of the class named as a **regressor**.

Implementation…

The code for this is given below:

- Fitting the Simple Linear Regression model to the training dataset
- from sklearn.linear_model import LinearRegression

- regressor= LinearRegression()

- regressor.fit(x_train, y_train)

Implementation…

- In the above code, we have used a fit() method to fit our Simple Linear Regression object to the training set.
- In the fit() function, we have passed the x_train and y_train, which is our training dataset for the dependent and an independent variable.
- We have fitted our regressor object to the training set so that the model can easily learn the correlations between the predictor and target variables. After executing the above lines of code, we will get the below output.

Implementation…

Prediction of test set result:

- Dependent (salary) and an independent variable (Experience). So, now, our model is ready to predict the output for the new observations.

- In this step, we will provide the test dataset (new observations) to the model to check whether it can predict the correct output or not.

- We will create a prediction vector y_pred, and x_pred, which will contain predictions of test dataset, and prediction of training set respectively.

Implementation…

Prediction of Test and Training set result

```
y_pred= regressor.predict(x_test)
x_pred= regressor.predict(x_train)
```

- On executing the above lines of code, two variables named y_pred and x_pred will generate in the variable explorer options that contain salary predictions for the training set and test set.

Implementation…

Step: 4. visualizing the Training set results:

- Now in this step, we will visualize the training set result. To do so, we will use the scatter() function of the pyplot library, which we have already imported in the pre-processing step.
- The scatter () function will create a scatter plot of observations.
- In the x-axis, we will plot the years of Experience of employees and on the y-axis, salary of employees.
- In the function, we will pass the real values of training set, which means a year of experience x_train, training set of Salaries y_train, and color of the observations. Here we are taking a green color for the observation, but it can be any color as per the choice

Implementation…

- Now, we need to plot the regression line, so for this, we will use the plot() function of the pyplot library.
- In this function, we will pass the years of experience for training set, predicted salary for training set x_pred, and color of the line.
- Next, we will give the title for the plot. So here, we will use the title() function of the pyplot library and pass the name ("Salary vs Experience (Training Dataset)".

Implementation…

- After that, we will assign labels for x-axis and y-axis using xlabel() and ylabel() function.

- Finally, we will represent all above things in a graph using show(). The code is given below:

- mtp.scatter(x_train, y_train, color="green")
- mtp.plot(x_train, x_pred, color="red")
- mtp.title("Salary vs Experience (Training Dataset)")
- mtp.xlabel("Years of Experience")
- mtp.ylabel("Salary(In Rupees)")
- mtp.show()

Implementation…

Fig. 1.4: Linear Regression Data's

Implementation…

In the above plot, we can see the real values observations in green dots and predicted values are covered by the red regression line. The regression line shows a correlation between the dependent and independent variable.

The good fit of the line can be observed by calculating the difference between actual values and predicted values. But as we can see in the above plot, most of the observations are close to the regression line, hence our model is good for the training set.

Multiple Linear Regression

- In the previous topic, we have learned about Simple Linear Regression, where a single Independent/Predictor(X) variable is used to model the response variable (Y).

- But there may be various cases in which the response variable is affected by more than one predictor variable; for such cases, the Multiple Linear Regression algorithm is used.

- Moreover, Multiple Linear Regression is an extension of Simple Linear regression as it takes more than one predictor variable to predict the response variable.

- We can define it as:

Multiple Linear Regression…

- Multiple Linear Regression is one of the important regression algorithms which models the linear relationship between a single dependent continuous variable and more than one independent variable.

- In Multiple Linear Regression, the target variable(Y) is a linear combination of multiple predictor variables x1, x2, x3, ...,xn.

- Since it is an enhancement of Simple Linear Regression, so the same is applied for the multiple linear regression equation, the equation becomes:
- Y=b₀+b₁x₁b₂x₂+b₃x₃+.....bnxn...(a)

Multiple Linear Regression…

Where,

Y= Output/Response variable

b0, b1, b2, b3 , bn....= Coefficients of the model

x1, x2, x3, x4,...= Various Independent/feature variable

Assumptions For Multiple linear Regression

- A linear relationship should exist between the Target and predictor variables.
- The regression residuals must be normally distributed.
- MLR assumes little or no multicollinearity (correlation between the independent variable) in data.

Implementation

Step-1: Data Pre-processing Step:

- The very first step is data pre-processing, which we have already discussed in this tutorial. This process contains the below steps:

- Importing libraries:

Firstly, we will import the library which will help in building the model.

- Below is the code for it:

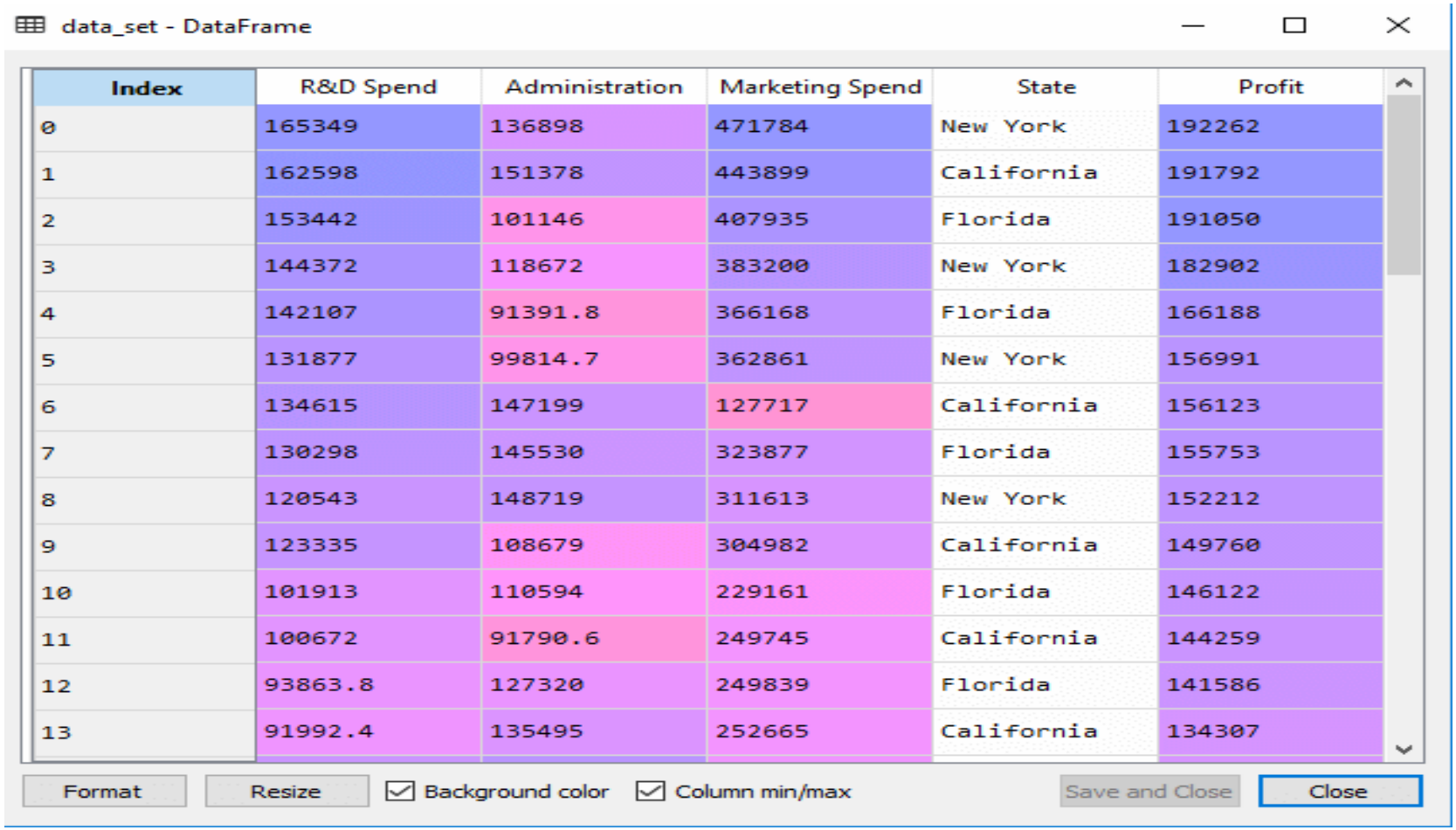
```
importing libraries
import numpy as nm
import matplotlib.pyplot as mtp
import pandas as pd
```

Implementation…

- Importing dataset: Now we will import the dataset(50_CompList), which contains all the variables. Below is the code for it:

```
#importing datasets
data_set= pd.read_csv('50_CompList.csv')
```

Implementation…



Index	R&D Spend	Administration	Marketing Spend	State	Profit
0	165349	136898	471784	New York	192262
1	162598	151378	443899	California	191792
2	153442	101146	407935	Florida	191050
3	144372	118672	383200	New York	182902
4	142107	91391.8	366168	Florida	166188
5	131877	99814.7	362861	New York	156991
6	134615	147199	127717	California	156123
7	130298	145530	323877	Florida	155753
8	120543	148719	311613	New York	152212
9	123335	108679	304982	California	149760
10	101913	110594	229161	Florida	146122
11	100672	91790.6	249745	California	144259
12	93863.8	127320	249839	Florida	141586
13	91992.4	135495	252665	California	134307

Implementation…

- In above output, we can clearly see that there are five variables, in which four variables are continuous and one is categorical variable.

- Extracting dependent and independent Variables:

```
#Extracting Independent and dependent Variable
x= data_set.iloc[:, :-1].values
y= data_set.iloc[:, 4].values
```

Logistic Regression

- Logistic regression aims to solve classification problems.
- It does this by predicting categorical outcomes, unlike linear regression that predicts a continuous outcome.
- In the simplest case there are two outcomes, which is called binomial, an example of which is predicting if a tumor is malignant or benign.
- Other cases have more than two outcomes to classify, in this case it is called multinomial.
- A common example for multinomial logistic regression would be predicting the class of an iris flower between 3 different species.

How does it Work?

- import numpy
- Store the independent variables in X.
- Store the dependent variable in y.
- Below is a sample dataset:

X represents the size of a tumor in centimeters.

X = numpy.array([3.78, 2.44, 2.09, 0.14, 1.72, 1.65, 4.92, 4.37, 4.96, 4.52, 3.69, 5.88]).reshape(-1,1)

y = numpy.array([0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1])

How does it Work?...

From the sklearn module we will use the LogisticRegression() method to create a logistic regression object

This object has a method called fit() that takes the independent and dependent values as parameters and fills the regression object with data that describes the relationship:

```
logr = linear_model.LogisticRegression()
logr.fit(X,y)
```

Now we have a logistic regression object that is ready to whether a tumor is cancerous based on the tumor size:

2. Linear Regression v/s Logistic Regression

- Linear Regression is one of the simplest Machine learning algorithm that comes under Supervised Learning technique and used for solving regression problems.
- It is used for predicting the continuous dependent variable with the help of independent variables.
- The goal of the Linear regression is to find the best fit line that can accurately predict the output for the continuous dependent variable.
- If single independent variable is used for prediction, then it is called Simple Linear Regression and if there are more than two independent variables then such regression is called as Multiple Linear Regression.

Linear Regression v/s Logistic Regression…

Linear Regression	Logistic Regression
Linear Regression is a supervised regression model.	Logistic Regression is a supervised classification model.
Equation of linear regression: y = a0 + a1x1 + a2x2 + ….+ aixi Here, y = response variable xi = ith predictor variable ai = average effect on y as xi increases by 1	Equation of logistic regression y(x) = e(a0 + a1x1 + a2x2 + … + aixi) / (1 + e(a0 + a1x1 + a2x2 + … + aixi)) Here, y = response variable xi = ith predictor variable ai = average effect on y as xi increases by 1

Linear Regression v/s Logistic Regression…

In Linear Regression, we predict the value by an integer number.	In Logistic Regression, we predict the value by 1 or 0.
Here no activation function is used.	Here activation function is used to convert a linear regression equation to the logistic regression equation
Here no threshold value is needed.	Here a threshold value is added.

Linear Regression v/s Logistic Regression…

Here we calculate Root Mean Square Error(RMSE) to predict the next weight value.	Here we use precision to predict the next weight value.
Here dependent variable should be numeric and the response variable is continuous to value.	Here the dependent variable consists of only two categories. Logistic regression estimates the odds outcome of the dependent variable given a set of quantitative or categorical independent variables.

Linear Regression v/s Logistic Regression…

It is based on the least square estimation.	It is based on maximum likelihood estimation.
Here when we plot the training datasets, a straight line can be drawn that touches maximum plots.	Any change in the coefficient leads to a change in both the direction and the steepness of the logistic function. It means positive slopes result in an S-shaped curve and negative slopes result in a Z-shaped curve.

Linear Regression v/s Logistic Regression…

Linear regression is used to estimate the dependent variable in case of a change in independent variables. For example, predict the price of houses.	Whereas logistic regression is used to calculate the probability of an event. For example, classify if tissue is benign or malignant.
Linear regression assumes the normal or gaussian distribution of the dependent variable.	Logistic regression assumes the binomial distribution of the dependent variable.

Linear Regression v/s Logistic Regression…

Applications of linear regression: • Financial risk assessment • Business insights • Market analysis	Applications of logistic regression: • Medicine • Credit scoring • Hotel Booking • Gaming • Text editing

Linear Regression Vs Logistic Regression…

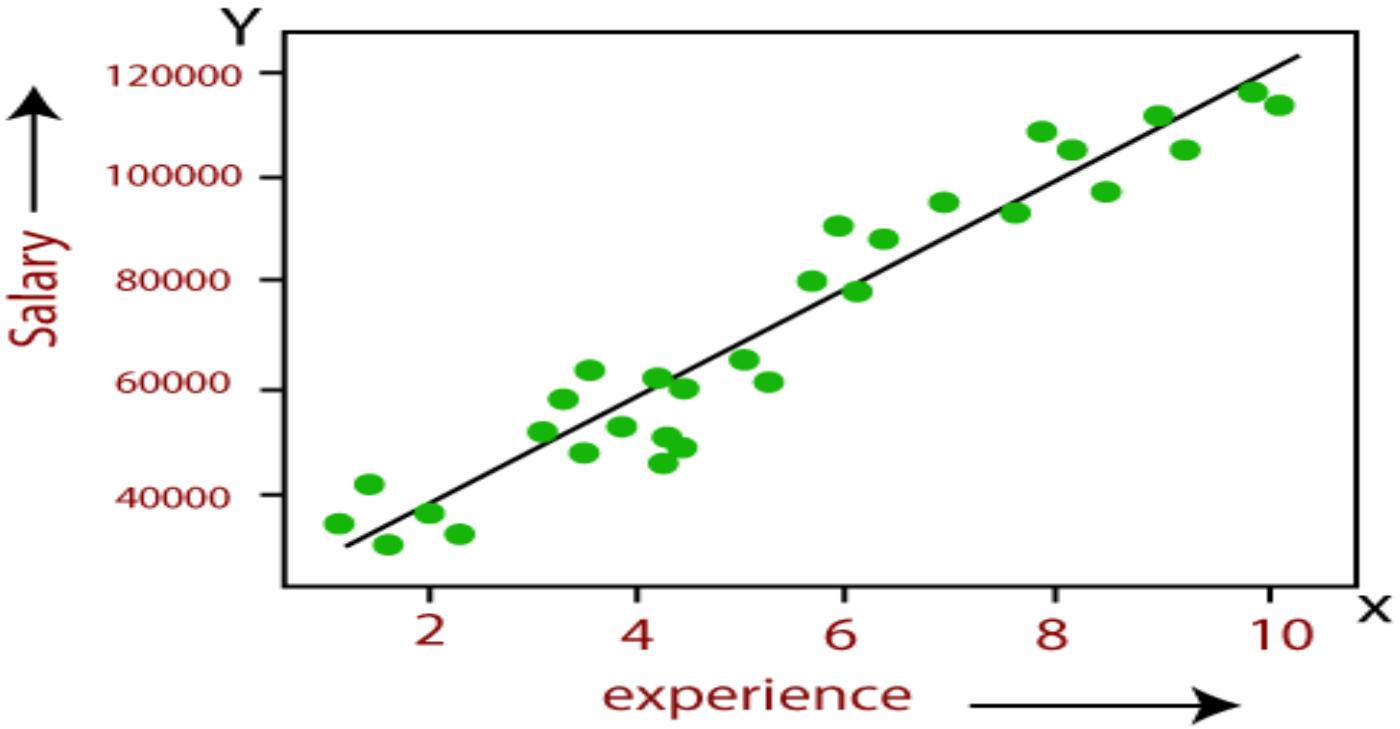


Fig. 2.1: Linear Regression Vs Logistic Regression

Linear Regression v/s Logistic Regression…

$$y = m * x + b$$

$$y = \frac{1}{1 + e^{-(m*x+b)}}$$

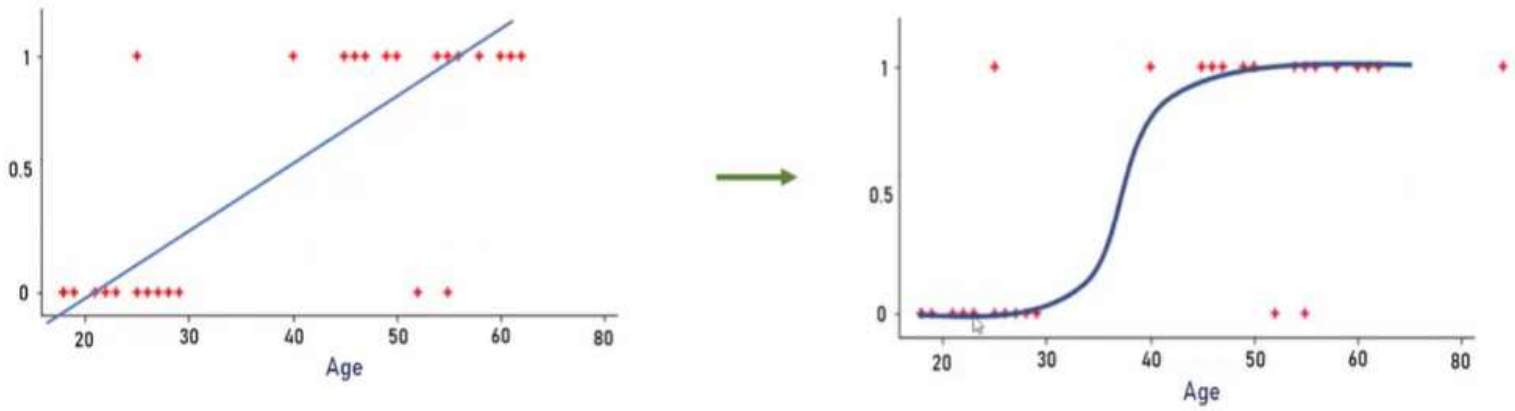


Fig. 2.2: Linear Regression Vs Logistic Regression Equation Graph

3. Clustering

- A way of grouping the data points into different clusters, consisting of similar data points.
- The objects with the possible similarities remain in a group that has less or no similarities with another group.

Clustering…

- Example: Let's understand the clustering technique with the real-world example of Mall:
- When we visit any shopping mall, we can observe that the things with similar usage are grouped together. Such as the t-shirts are grouped in one section, and trousers are at other sections,
- similarly, at vegetable sections, apples, bananas, Mangoes, etc., are grouped in separate sections, so that we can easily find out the things.
- The clustering technique also works in the same way.

Clustering…

Other examples of clustering are grouping documents according to the topic-

(i). Market Segmentation
(ii). Statistical data analysis
(iii). Social network analysis
(iv). Image segmentation
(v). Anomaly detection, etc.

- Apart from these general usages, it is used by the Amazon in its recommendation system to provide the recommendations as per the past search of products.
- Netflix also uses this technique to recommend the movies and web-series to its users as per the watch history.

4. Types of Clustering

1. Partition-Based Clustering
2. Hierarchical Clustering
3. Density-Based Clustering
4. Model-Based Clustering
5. Grid-Based Clustering
6. Fuzzy Clustering (Soft Clustering)

Types of Clustering…

1. Partition-Based Clustering-

- Divides data into distinct groups.
- Each point belongs to exactly one cluster.
- **Goal**: Minimize intra-cluster distance, maximize inter-cluster distance.
  - **Popular algorithms**:
    - K-Means Clustering
    - K-Medoids Clustering

Types of Clustering…

2. Hierarchical Clustering-

- Builds nested clusters organized as a tree (called a **dendrogram**).
- Two approaches:
  - **Agglomerative** (bottom-up merging)
  - **Divisive** (top-down splitting)
- **Popular algorithms**:
  - **Agglomerative Hierarchical Clustering**
  - **BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies)**

Types of Clustering…

3. Density-Based Clustering-

- Clusters are formed based on regions of high density.
- Useful for complex cluster shapes and identifying noise (outliers).
- **Popular algorithms**:
  - **DBSCAN** (Density-Based Spatial Clustering of Applications with Noise)
  - **OPTICS** (Ordering Points To Identify the Clustering Structure)

Types of Clustering…

4. Model-Based Clustering-

- Assumes data is generated from a mixture of underlying probability distributions.
- Estimates model parameters to best fit the data.
- **Popular algorithms**:
  - **Gaussian Mixture Models (GMMs)**
  - **Expectation-Maximization (EM)**

Types of Clustering…

5. Grid-Based Clustering-

- The data space is divided into a finite number of grids, and clustering is performed on these grids.
- Fast and efficient for large datasets.
- **Popular algorithms**:
  - **STING**
  - **CLIQUE**

Types of Clustering…

6. Fuzzy Clustering (Soft Clustering)-

- Data points can belong to multiple clusters with varying degrees of membership.
- Useful when boundaries between clusters are not clear-cut.
- **Popular algorithm**:
  - **Fuzzy C-Means Clustering**

5. K Means Clustering

- K-Means Clustering is an unsupervised learning algorithm that is used to solve the clustering problems in machine learning or data science.

- In this topic, we will learn what is K-means clustering algorithm, how the algorithm works, along with the Python implementation of k-means clustering.

What is K–Means Algorithm

- K-Means Clustering is an Unsupervised Learning algorithm, which groups the unlabeled dataset into different clusters.
- Here K defines the number of pre-defined clusters that need to be created in the process, as if K=2, there will be two clusters, and for K=3, there will be three clusters, and so on.

K–Means Algorithm…

- It is an iterative algorithm that divides the unlabeled dataset into k different clusters in such a way that each dataset belongs only one group that has similar properties.
- It allows us to cluster the data into different groups and a convenient way to discover the categories of groups in the unlabeled dataset on its own without the need for any training.
- It is a centroid-based algorithm, where each cluster is associated with a centroid.
- The main aim of this algorithm is to minimize the sum of distances between the data point and their corresponding clusters.

What is k –Means Algorithm…

- Determines the best value for K center points or centroids by an iterative process.
- Assigns each data point to its closest k-center. Those data points which are near to the particular k-center, create a cluster.
- Hence each cluster has datapoints with some commonalities, and it is away from other clusters.

k –Means Algorithm…

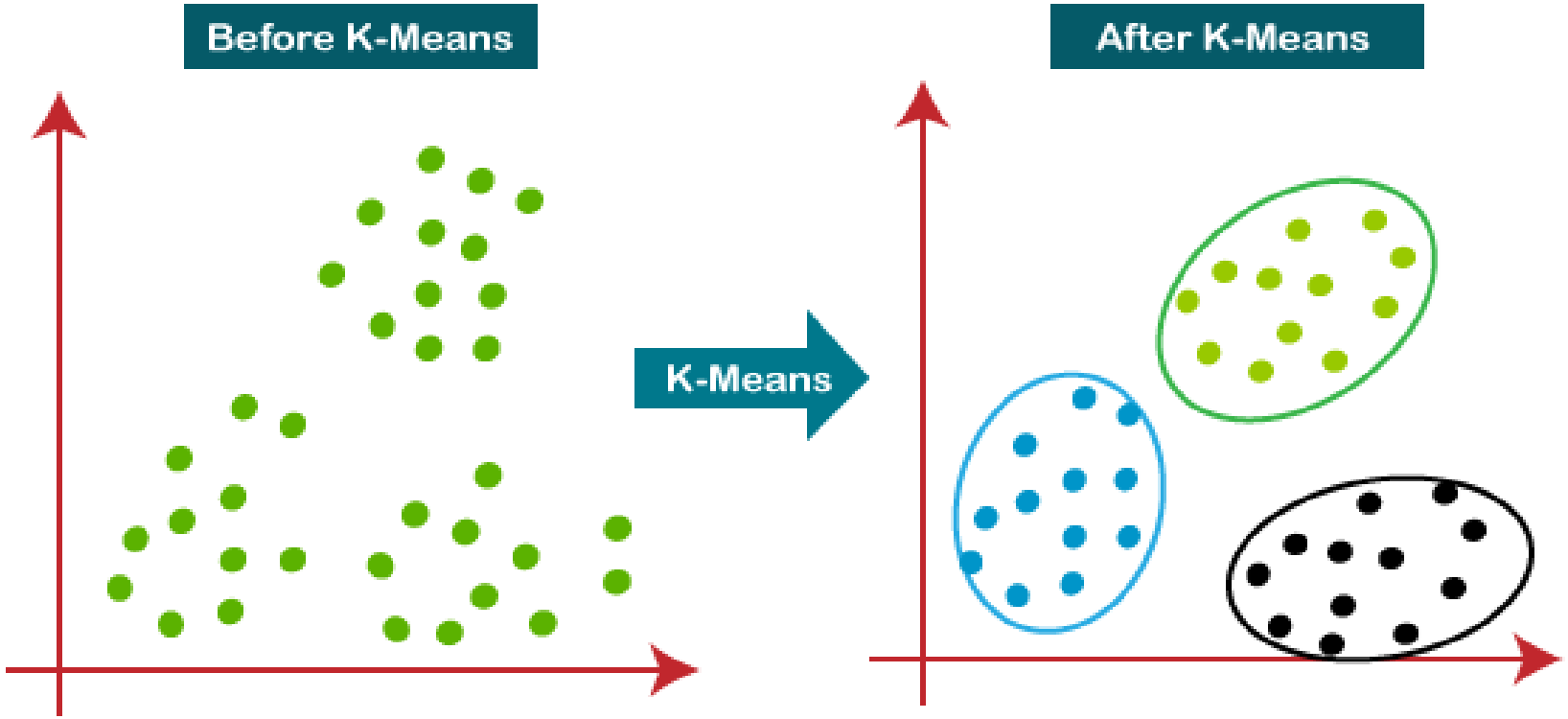


Fig. 5.1: K- Means Algorithm

k –Means Algorithm…

- **Step-1:** Select the number K to decide the number of clusters.
- **Step-2:** Select random K points or centroids. (It can be other from the input dataset).
- **Step-3:** Assign each data point to their closest centroid, which will form the predefined K clusters.
- **Step-4:** Calculate the variance and place a new centroid of each cluster.
- **Step-5:** Repeat the third steps, which means reassign each datapoint to the new closest centroid of each cluster.
- **Step-6:** If any reassignment occurs, then go to step-4 else go to FINISH.
- **Step-7:** The model is ready.

Implementation

- The given dataset, we have **Customer_Id, Gender, Age, Annual Income ($), and Spending Score** (which is the calculated value of how much a customer has spent in the mall, the more the value, the more he has spent).
- From this dataset, we need to calculate some patterns, as it is an unsupervised method, so we don't know what to calculate exactly.

The steps to be followed for the implementation are given below:

- **Data Pre-processing**
- **Finding the optimal number of clusters using the elbow method**
- **Training the K-means algorithm on the training dataset**
- **Visualizing the clusters**

Implementation…

- **Step-1: Data pre-processing Step**

The first step will be the data pre-processing, as we did in our earlier topics of Regression and Classification. But for the clustering problem, it will be different from other models. Let's discuss it:

- **Importing Libraries**

As we did in previous topics, firstly, we will import the libraries for our model, which is part of data pre-processing. The code is given below:

```
1. importing libraries
2.import numpy as nm
3.import matplotlib.pyplot as mtp
4.import pandas as pd
```

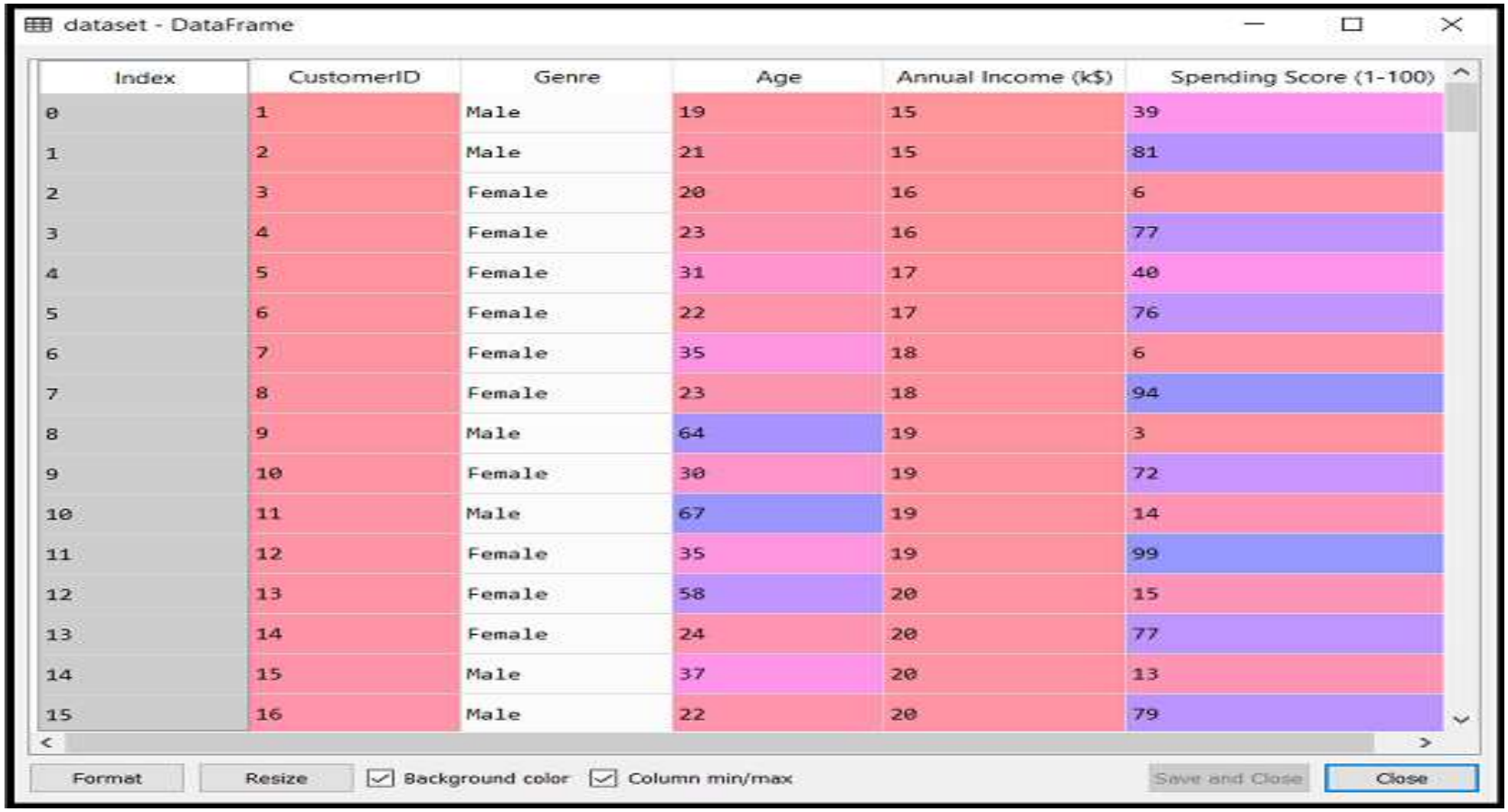
Implementation…

Importing the Dataset:

Next, we will import the dataset that we need to use. So here, we are using the Mall_Customer_data.csv dataset. It can be imported using the below code:

```
# Importing the dataset
dataset = pd.read_csv('Mall_Customers_data.csv')
```

Implementation…



Index	CustomerID	Genre	Age	Annual Income (k$)	Spending Score (1-100)
0	1	Male	19	15	39
1	2	Male	21	15	81
2	3	Female	20	16	6
3	4	Female	23	16	77
4	5	Female	31	17	40
5	6	Female	22	17	76
6	7	Female	35	18	6
7	8	Female	23	18	94
8	9	Male	64	19	3
9	10	Female	30	19	72
10	11	Male	67	19	14
11	12	Female	35	19	99
12	13	Female	58	20	15
13	14	Female	24	20	77
14	15	Male	37	20	13
15	16	Male	22	20	79

Implementation…

Extracting Independent Variables

- Here we don't need any dependent variable for data pre-processing step as it is a clustering problem, and we have no idea about what to determine.
- So we will just add a line of code for the matrix of features.

```
x = dataset.iloc[:, [3, 4]].values
```

Implementation…

- **Step-2:** Finding the optimal number of clusters using the elbow method.
- In the second step, we will try to find the optimal number of clusters for our clustering problem. So, as discussed above, here we are going to use the elbow method for this purpose.

- As we know, the elbow method uses the WCSS concept to draw the plot by plotting WCSS values on the Y-axis and the number of clusters on the X-axis.

- So we are going to calculate the value for WCSS for different k values ranging from 1 to 10. Below is the code for it:

Implementation…

Step- 3:
Training the K-means algorithm on the training dataset

- As we have got the number of clusters, so we can now train the model on the dataset.
- To train the model, we will use the same two lines of code as we have used in the above section, but here instead of using i, we will use 5, as we know there are 5 clusters that need to be formed.

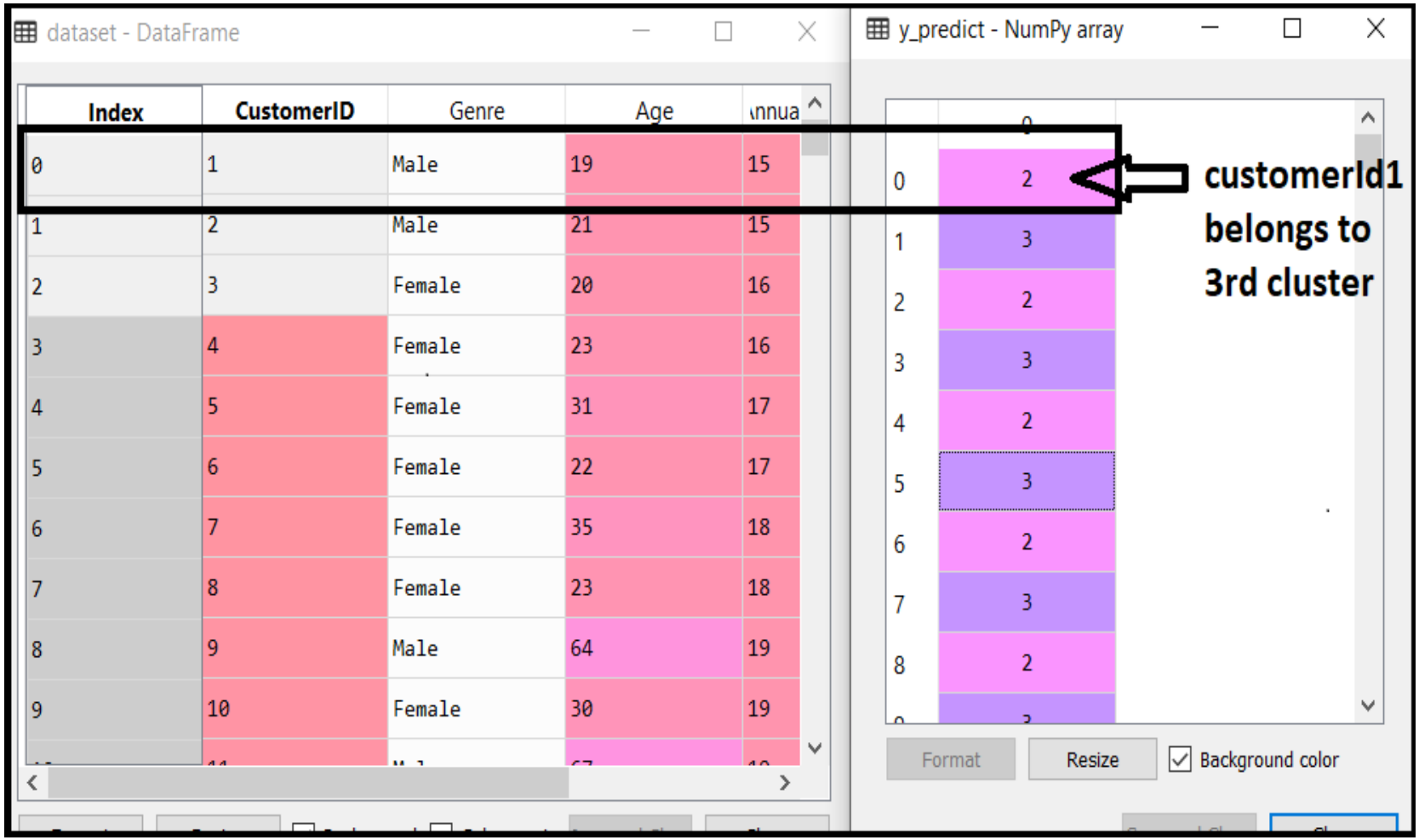
The code is given below:

```
#training the K-means model on a dataset
kmeans = KMeans(n_clusters=5, init='k-
means++', random_state= 42)
y_predict= kmeans.fit_predict(x)
```

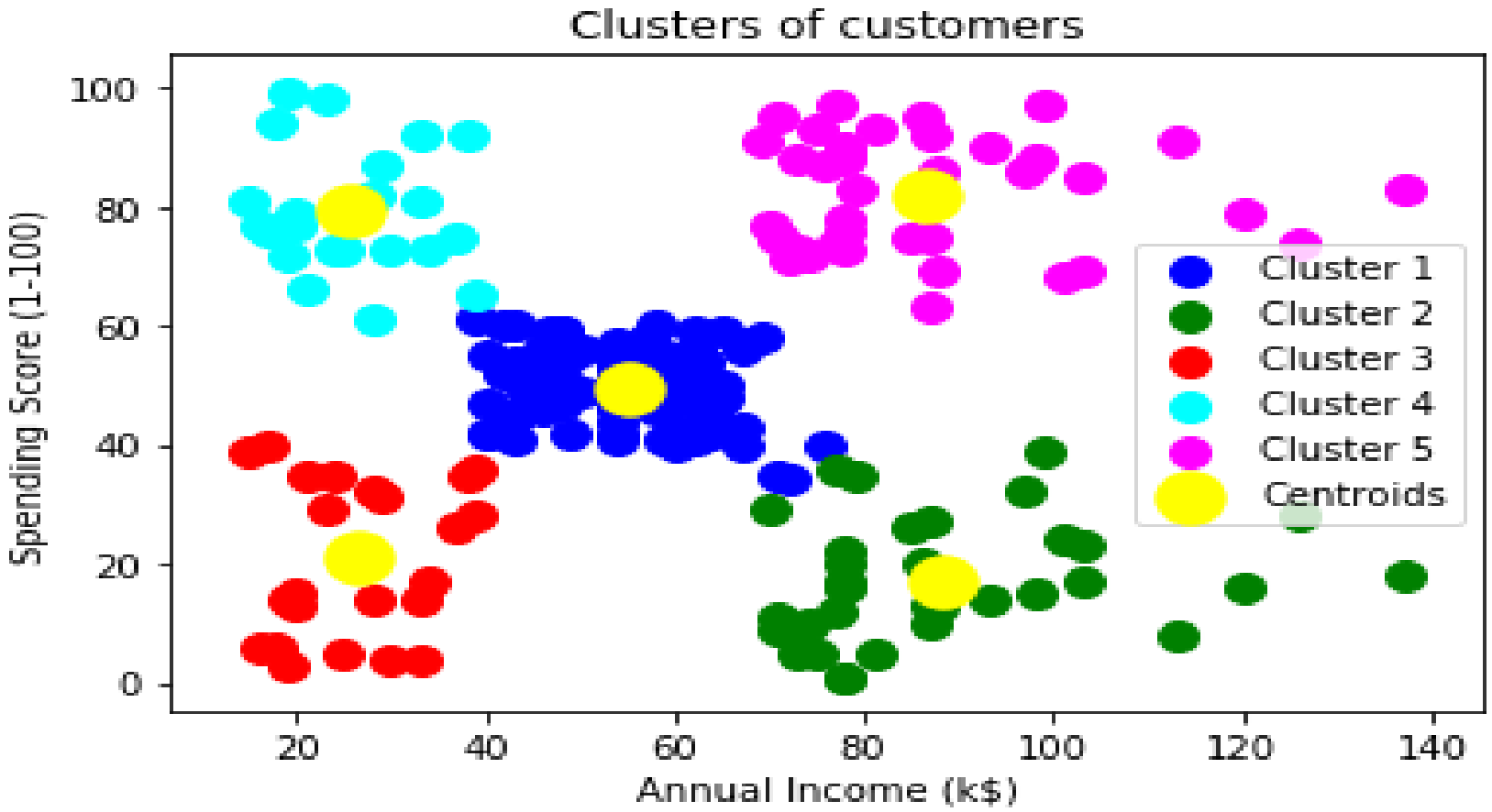
Implementation…

- The first line is the same as above for creating the object of K-Means class.
- In the second line of code, we have created the dependent variable **y_predict** to train the model.
- By executing the above lines of code, we will get the y_predict variable. We can check it under **the variable explorer** option in the Spyder IDE.
- We can now compare the values of y_predict with our original dataset. Consider the below image:

Implementation…



Implementation…



Implementation…

- The output image is clearly showing the five different clusters with different colors.
- The clusters are formed between two parameters of the dataset; Annual income of customer and Spending. We can change the colors and labels as per the requirement or choice.

- We can also observe some points from the above patterns, which are given below:

Cluster:1 shows the customers with average salary and average spending so we can categorize these customers as

Implementation…

- **Cluster:2** shows the customer has a high income but low spending, so we can categorize them as careful.
- **Cluster:3** shows the low income and also low spending so they can be categorized as sensible.
- **Cluster:4** shows the customers with low income with very high spending so they can be categorized as **careless**.
- **Cluster:5** shows the customers with high income and high spending so they can be categorized as target, and these customers can be the most profitable customers for the mall owner.

6. Hierarchical Clustering

- Hierarchical clustering is a popular method for grouping objects.
- It creates groups so that objects within a group are similar to each other and different from objects in other groups.
- Clusters are visually represented in a hierarchical tree called a *dendrogram*.

Hierarchical Clustering…

Hierarchical clustering has a couple of key benefits:

1. There is no need to pre-specify the number of clusters. Instead, the dendrogram can be cut at the appropriate level to obtain the desired number of clusters.

2. Data is easily summarized/organized into a hierarchy using dendrograms. Dendrograms make it easy to examine and interpret clusters.

Applications

There are many real-life applications of Hierarchical clustering. They include:

Bioinformatics: grouping animals according to their biological features to reconstruct phylogeny trees

Business: dividing customers into segments or forming a hierarchy of employees based on salary.

Image processing: grouping handwritten characters in text recognition based on the similarity of the character shapes.

Information Retrieval: categorizing search results based on the query.

7. Classification

- The Classification algorithm is a Supervised Learning technique that is used to identify the category of new observations on the basis of training data.
- In Classification, a program learns from the given dataset or observations and then classifies new observation into a number of classes or groups.
- Such as, Yes or No, 0 or 1, Spam or Not Spam, cat or dog, etc. Classes can be called as targets/labels or categories.

Classification…

- Unlike regression, the output variable of Classification is a category, not a value, such as "Green or Blue", "fruit or animal", etc.
- Since the Classification algorithm is a Supervised learning technique, hence it takes labeled input data, which means it contains input with the corresponding output.

Classification…

- In classification algorithm, a discrete output function(y) is mapped to input variable(x).

  y=f(x), where y = categorical output

- The best example of an ML classification algorithm is **Email Spam Detector**.

- The main goal of the Classification algorithm is to identify the category of a given dataset, and these algorithms are mainly used to predict the output for the categorical data.

Classification…

- Classification algorithms can be better understood using the below diagram. In the below diagram, there are two classes, class A and Class B. These classes have features that are similar to each other and dissimilar to other classes.

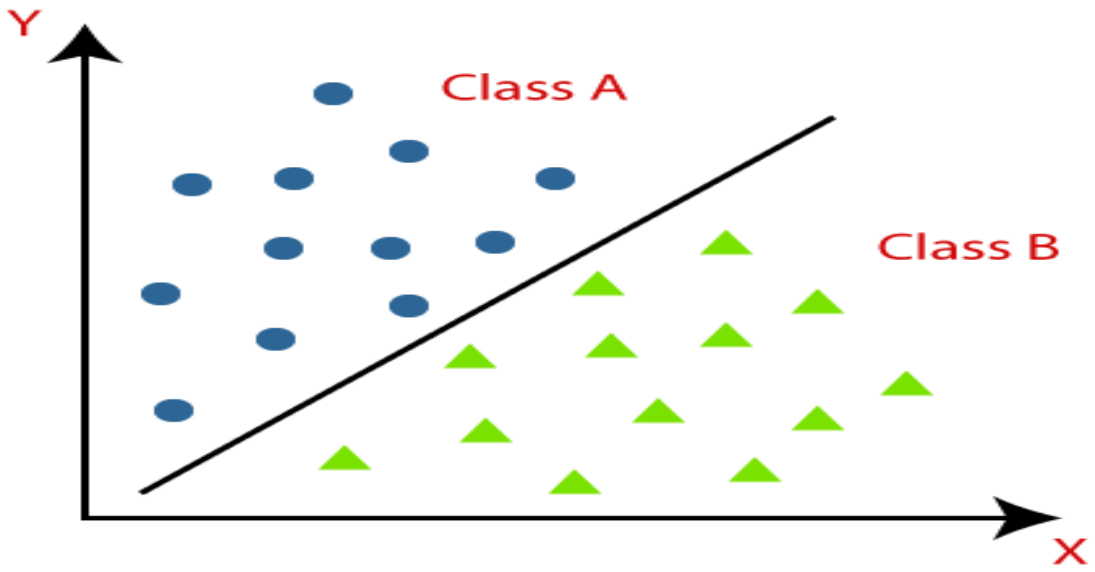


Fig. 7.1: Classification

Classification…

- **Binary Classifier:** If the classification problem has only two possible outcomes, then it is called as Binary Classifier.

  Examples: YES or NO, MALE or FEMALE, SPAM or NOT SPAM, CAT or DOG, etc.

- **Multi-class Classifier:** If a classification problem has more than two outcomes, then it is called as Multi-class Classifier.

  Example: Classifications of types of crops, Classification of types of music.

Regression v/s Classification

- Regression and Classification algorithms are Supervised Learning algorithms.
- Both the algorithms are used for prediction in Machine learning and work with the labeled datasets.
- But the difference between both is how they are used for different machine learning problems.

Regression v/s Classification…

- The main difference between Regression and Classification algorithms that Regression algorithms are used to predict the continuous values such as price, salary, age, etc. and Classification algorithms are used to predict/Classify the discrete values such as Male or Female, True or False, Spam or Not Spam, etc.

Regression v/s Classification…

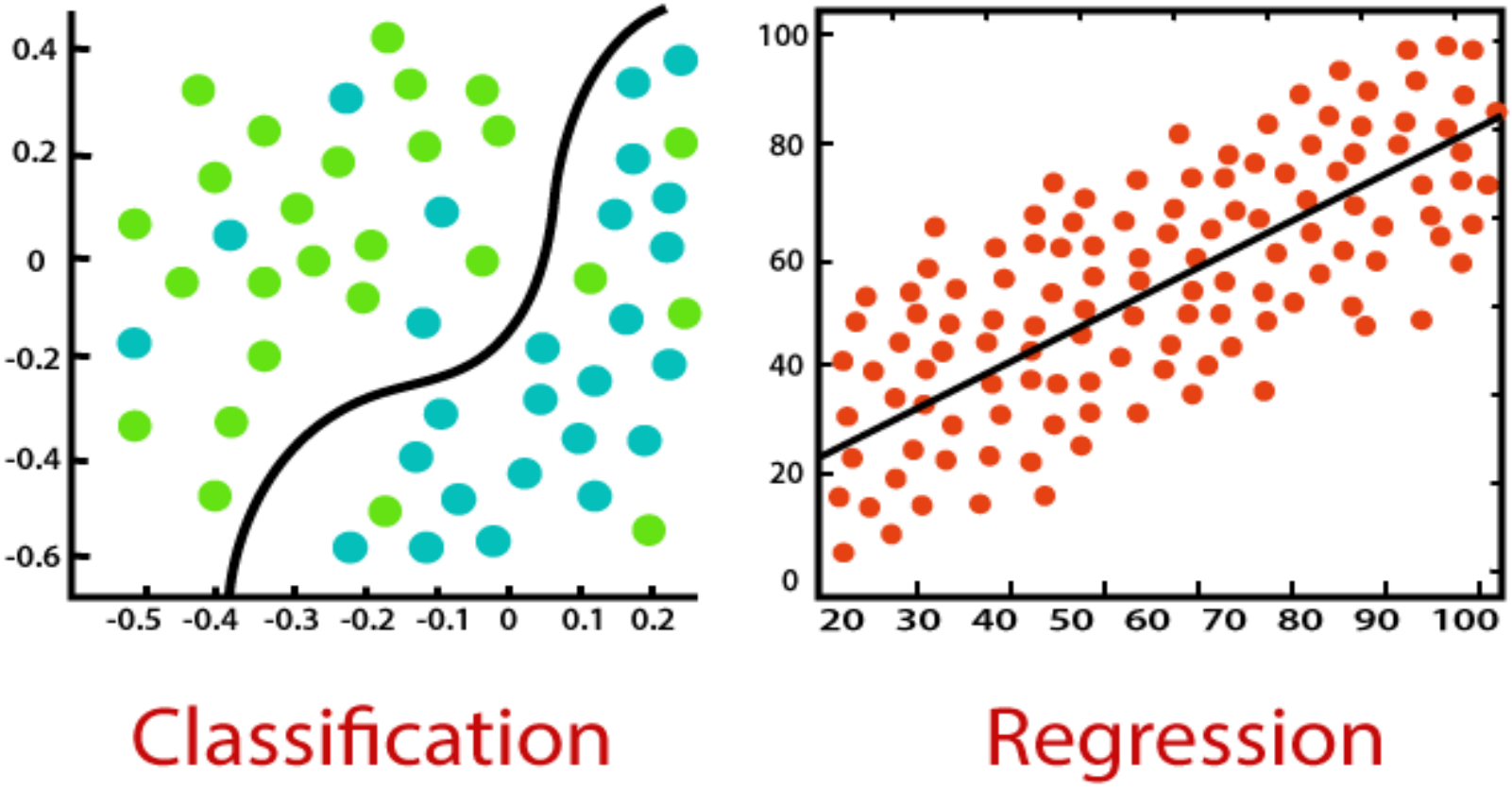


Fig. 7.2: Regression & Classification

Regression

- Regression is a process of finding the correlations between dependent and independent variables.
- It helps in predicting the continuous variables such as prediction of **Market Trends**, prediction of House prices, etc.

Regression…

The task of the Regression algorithm is to find the mapping function to map the input variable(x) to the continuous output variable(y).

Difference In Classification And Regression

Regression Algorithm	Classification Algorithm
In Regression, the output variable must be of continuous nature or real value.	In Classification, the output variable must be a discrete value.
The task of the regression algorithm is to map the input value (x) with the continuous output variable(y).	The task of the classification algorithm is to map the input value(x) with the discrete output variable(y).
Regression Algorithms are used with continuous data.	Classification Algorithms are used with discrete data.

Difference In Classification And Regression…

Regression Algorithm	Classification Algorithm
In Regression, we try to find the best fit line, which can predict the output more accurately.	In Classification, we try to find the decision boundary, which can divide the dataset into different classes.
Regression algorithms can be used to solve the regression problems such as Weather Prediction, House price prediction, etc.	Classification Algorithms can be used to solve classification problems such as Identification of spam emails, Speech Recognition, Identification of cancer cells, etc.

Decision Tree Classification

- Decision Tree is a Supervised learning technique that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems.

- It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome.

- In a Decision tree, there are two nodes, which are the Decision Node and Leaf Node.

Decision Tree Classification…

- Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches.

- The decisions or the test are performed on the basis of features of the given dataset.
- It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.

Decision Tree Classification…

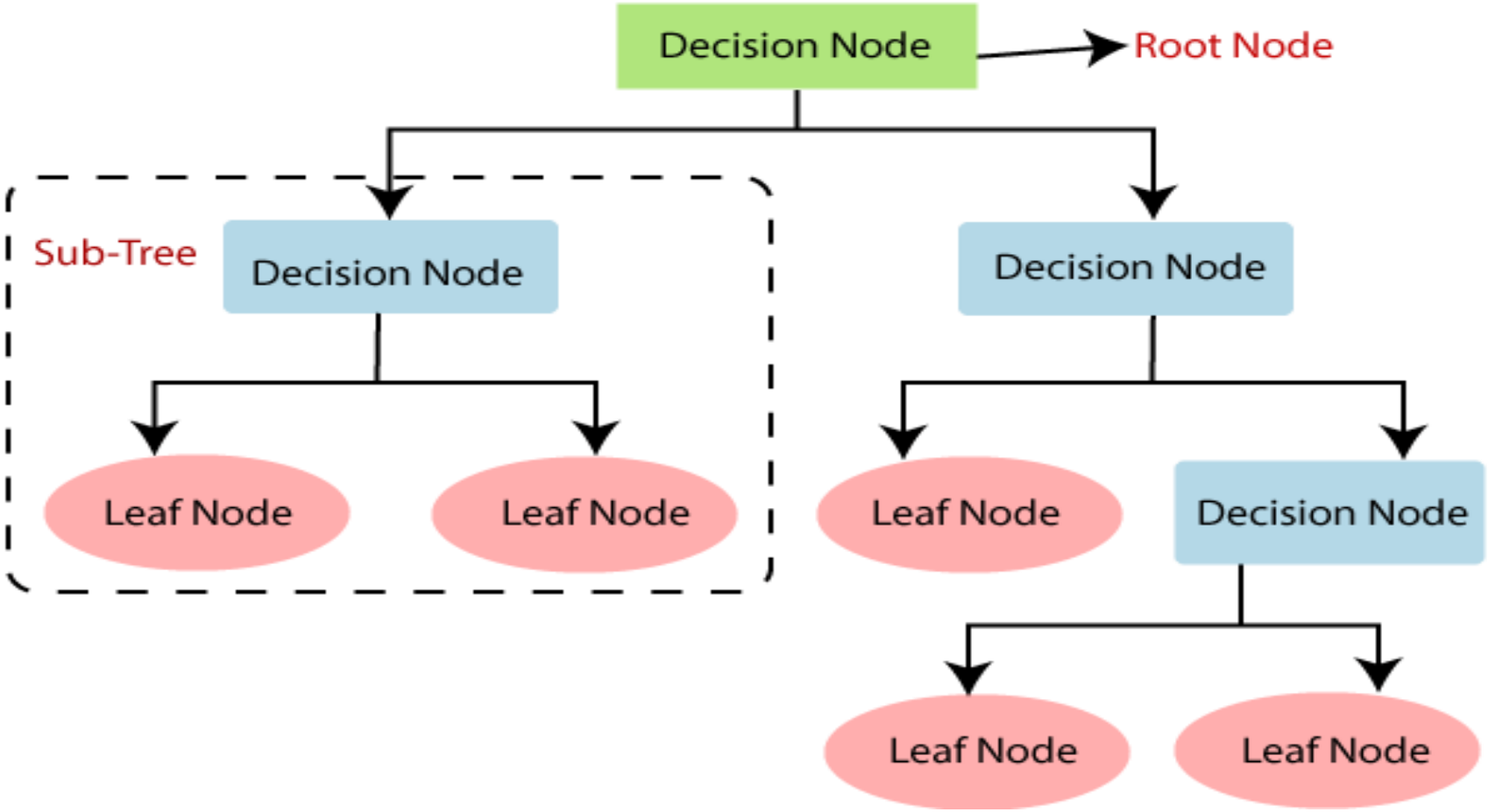


Fig. 7.3: Decision Tree Classification

Why Use Decision Tree Algorithm?

There are various algorithms in Machine learning, so choosing the best algorithm for the given dataset and problem is the main point to remember while creating a machine learning model.

Below are the two reasons for using the Decision tree:

- Decision Trees usually mimic human thinking ability while making a decision, so it is easy to understand.
- The logic behind the decision tree can be easily understood because it shows a tree-like structure.

How Does Decision Tree Algorithm Work?

- In a decision tree, for predicting the class of the given dataset, the algorithm starts from the root node of the tree.

- This algorithm compares the values of root attribute with the record (real dataset) attribute and, based on the comparison, follows the branch and jumps to the next node.

- For the next node, the algorithm again compares the attribute value with the other sub-nodes and move further.

- It continues the process until it reaches the leaf node of the tree. The complete process can be better understood using the below algorithm:

How Does Decision Tree Algorithm Work?...

Step-1: Begin the tree with the root node, says S, which contains the complete dataset.

Step-2: Find the best attribute in the dataset using **Attribute Selection Measure (ASM).**

Step-3: Divide the S into subsets that contains possible values for the best attributes.

Step-4: Generate the decision tree node, which contains the best attribute.

Step-5: Recursively make new decision trees using the subsets of the dataset created in step -3. Continue this process until a stage is reached where you cannot further classify the nodes and called the final node as a leaf node.

How Does Decision Tree Algorithm Work?...

- **Example:** Suppose there is a candidate who has a job offer and wants to decide whether he should accept the offer or Not.
- So, to solve this problem, the decision tree starts with the root node (Salary attribute by ASM).
- The root node splits further into the next decision node (distance from the office) and one leaf node based on the corresponding labels.
- The next decision node further gets split into one decision node (Cab facility) and one leaf node.
- Finally, the decision node splits into two leaf nodes (Accepted offers and Declined offer).
- Consider the below diagram:

How Does Decision Tree Algorithm Work?...

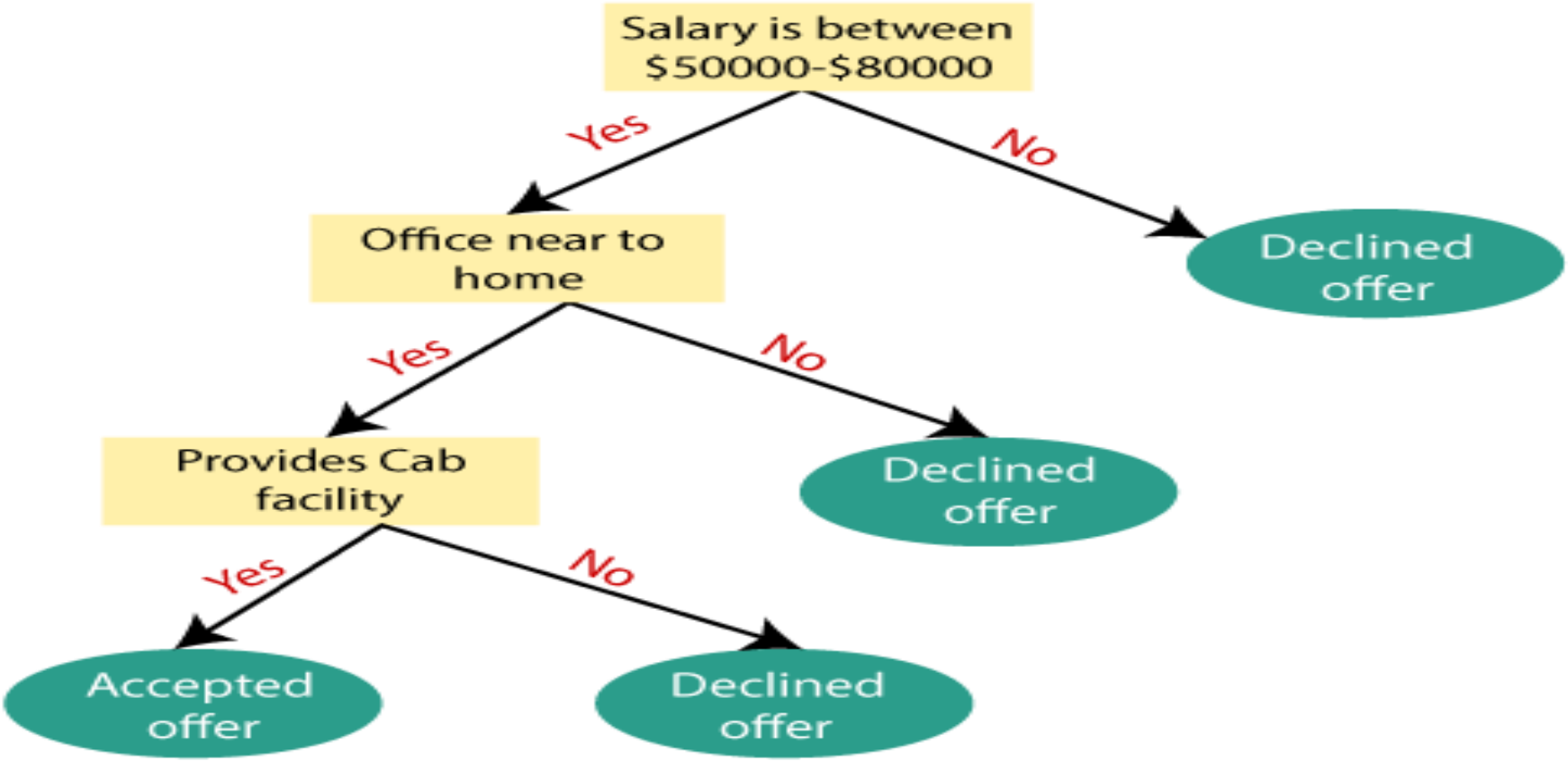


Fig. 7.4: Decision Tree Algorithm work

Information Gain And Gini Index

- While implementing a Decision tree, the main issue arises that how to select the best attribute for the root node and for sub-nodes.

- So, to solve such problems there is a technique which is called as Attribute selection measure or ASM.

- By this measurement, we can easily select the best attribute for the nodes of the tree. There are two popular techniques for ASM, which are:

- ❑ Information Gain
- ❑ Gini Index

Information Gain-

Information Gain:

- Information gain is the measurement of changes in entropy after the segmentation of a dataset based on an attribute.
- It calculates how much information a feature provides us about a class.
- According to the value of information gain, we split the node and build the decision tree.
- A decision tree algorithm always tries to maximize the value of information gain, and a node/attribute having the highest information gain is split first. It can be calculated using the below formula:

Entropy-

Information Gain= Entropy(S)-
[(Weighted Avg) *Entropy(each feature)

Entropy: Entropy is a metric to measure the impurity in a given attribute. It specifies randomness in data. Entropy can be calculated as:

Entropy(s)= -P(yes)log2 P(yes)- P(no) log2 P(no)

Where,

S= Total number of samples

P(yes)= probability of Success

Gini Index

Gini Index:

- Gini index is a measure of impurity or purity used while creating a decision tree in the CART(Classification and Regression Tree) algorithm.

- An attribute with the low Gini index should be preferred as compared to the high Gini index.

- It only creates binary splits, and the CART algorithm uses the Gini index to create binary splits.

- Gini index can be calculated using the below formula:
  Gini Index= 1- ∑jPj2

Advantages And Disadvantages

Advantages of the Decision Tree-

- It is simple to understand as it follows the same process which a human follow while making any decision in real-life.
- It can be very useful for solving decision-related problems.
- It helps to think about all the possible outcomes for a problem.
- There is less requirement of data cleaning compared to other algorithms.

Advantages And Disadvantages…

Disadvantages of the Decision Tree

- The decision tree contains lots of layers, which makes it complex.
- It may have an overfitting issue, which can be resolved using the Random Forest algorithm.
- For more class labels, the computational complexity of the decision tree may increase.

Confusion Matrix

- The confusion matrix is a matrix used to determine the performance of the classification models for a given set of test data.
- It can only be determined if the true values for test data are known.

Confusion Matrix…

- The matrix itself can be easily understood, but the related terminologies may be confusing. Since it shows the errors in the model performance in the form of a matrix, hence also known as an error matrix.
- Some features of Confusion matrix are given below:

Confusion Matrix Methods

For the 2 prediction classes of classifiers, the matrix is of 2*2 table, for 3 classes, it is 3*3 table, and so on.

- The matrix is divided into two dimensions, that are predicted values and actual values along with the total number of predictions.
- Predicted values are those values, which are predicted by the model, and actual values are the true values for the given observations.

Confusion Matrix Methods…

True Negative: Model has given prediction No, and the real or actual value was also No.

True Positive: The model has predicted yes, and the actual value was also true.

False Negative: The model has predicted no, but the actual value was Yes, it is also called as Type-II error.

False Positive: The model has predicted Yes, but the actual value was No. It is also called a Type-I error.

Need of Confusion Matrix

Need for Confusion Matrix in Machine learning

- It evaluates the performance of the classification models, when they make predictions on test data, and tells how good our classification model is.
- It not only tells the error made by the classifiers but also the type of errors such as it is either type-I or type-II error.
- With the help of the confusion matrix, we can calculate the different parameters for the model, such as accuracy, precision, etc.

Need of Confusion Matrix…

n = 100	Actual: No	Actual: Yes	
Predicted: No	TN: 65	FP: 3	68
Predicted: Yes	FN: 8	TP: 24	32
	73	27	

Need of Confusion Matrix…

From the above example, we can conclude that:

- The table is given for the two-class classifier, which has two predictions "Yes" and "NO." Here, Yes defines that patient has the disease, and No defines that patient does not has that disease.

- The classifier has made a total of **100 predictions**. Out of 100 predictions, **89 are true predictions**, and **11 are incorrect predictions**.

- The model has given prediction "yes" for 32 times, and "No" for 68 times. Whereas the actual "Yes" was 27, and actual "No" was 73 times.

Parul® University | NAAC GRADE A++

https://paruluniversity.ac.in/