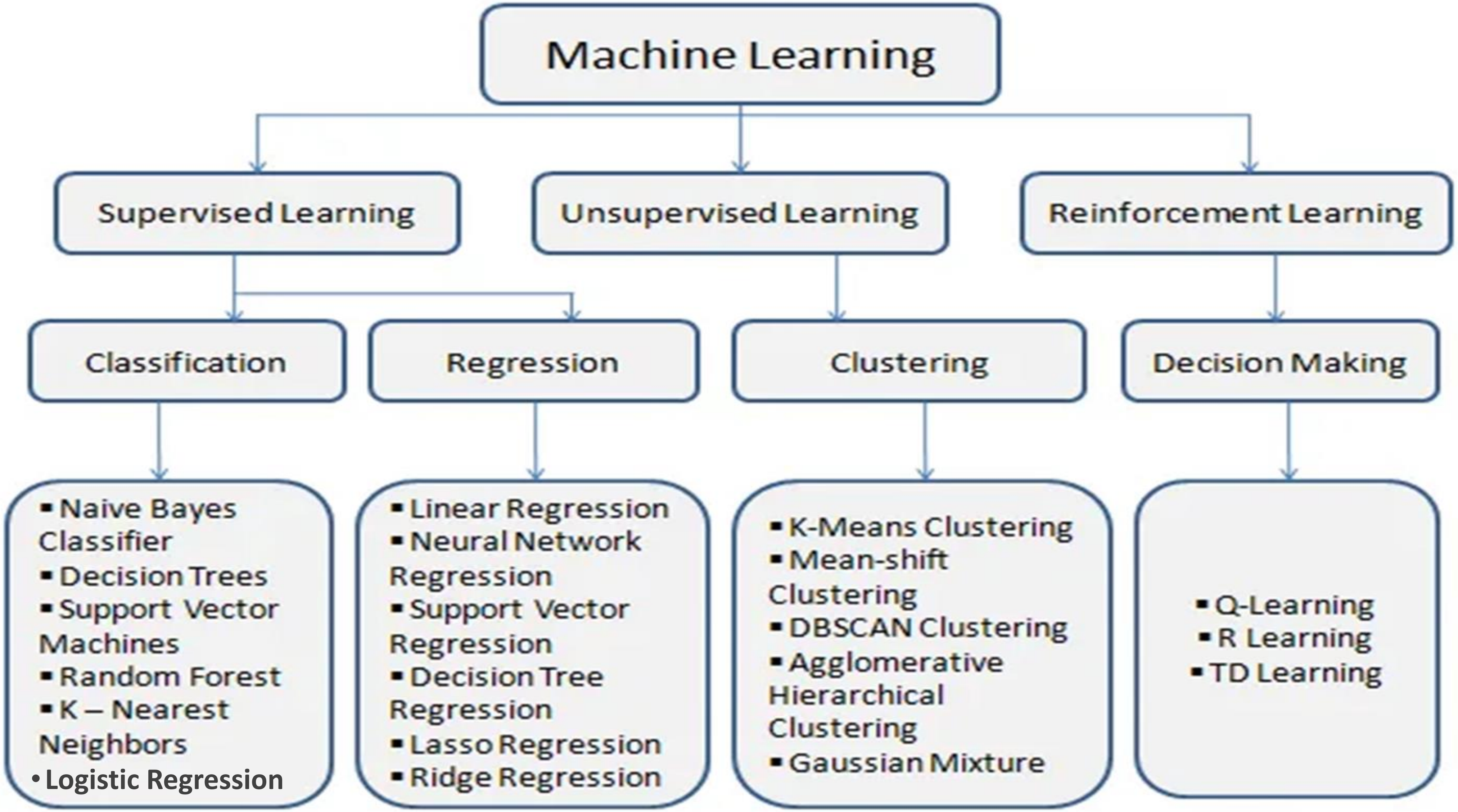
Logistic Regression


Machine Learning
Supervised Learning
Unsupervised Learning
Reinforcement Learning
Classification
Regression
Clustering
Decision Making
▪ Naive Bayes Classifier
▪ Decision Trees
▪ Support Vector Machines
▪ Random Forest
▪ K – Nearest Neighbors
• Logistic Regression
▪ Linear Regression
▪ Neural Network Regression
▪ Support Vector Regression
▪ Decision Tree Regression
▪ Lasso Regression
▪ Ridge Regression
▪ K-Means Clustering
▪ Mean-shift Clustering
▪ DBSCAN Clustering
▪ Agglomerative Hierarchical Clustering
▪ Gaussian Mixture
▪ Q-Learning
▪ R Learning
▪ TD Learning

Logistic regression

- **Logistic regression** is one of the most popular Machine Learning algorithms, which comes under the **Supervised Learning technique**.
- It is used for **predicting the categorical dependent variable** using a given **set of independent variables.**
- Therefore, the outcome must be a **categorical** or **discrete value**.

- It can be either Yes or No, 0 or 1, True or False, etc.
- But instead of giving the exact value as 0 and 1, it gives the probabilistic values which lie between 0 and 1.

Logistic Regression and Linear Regression

- **Logistic Regression** is much similar to **Linear Regression** except for how they are used.

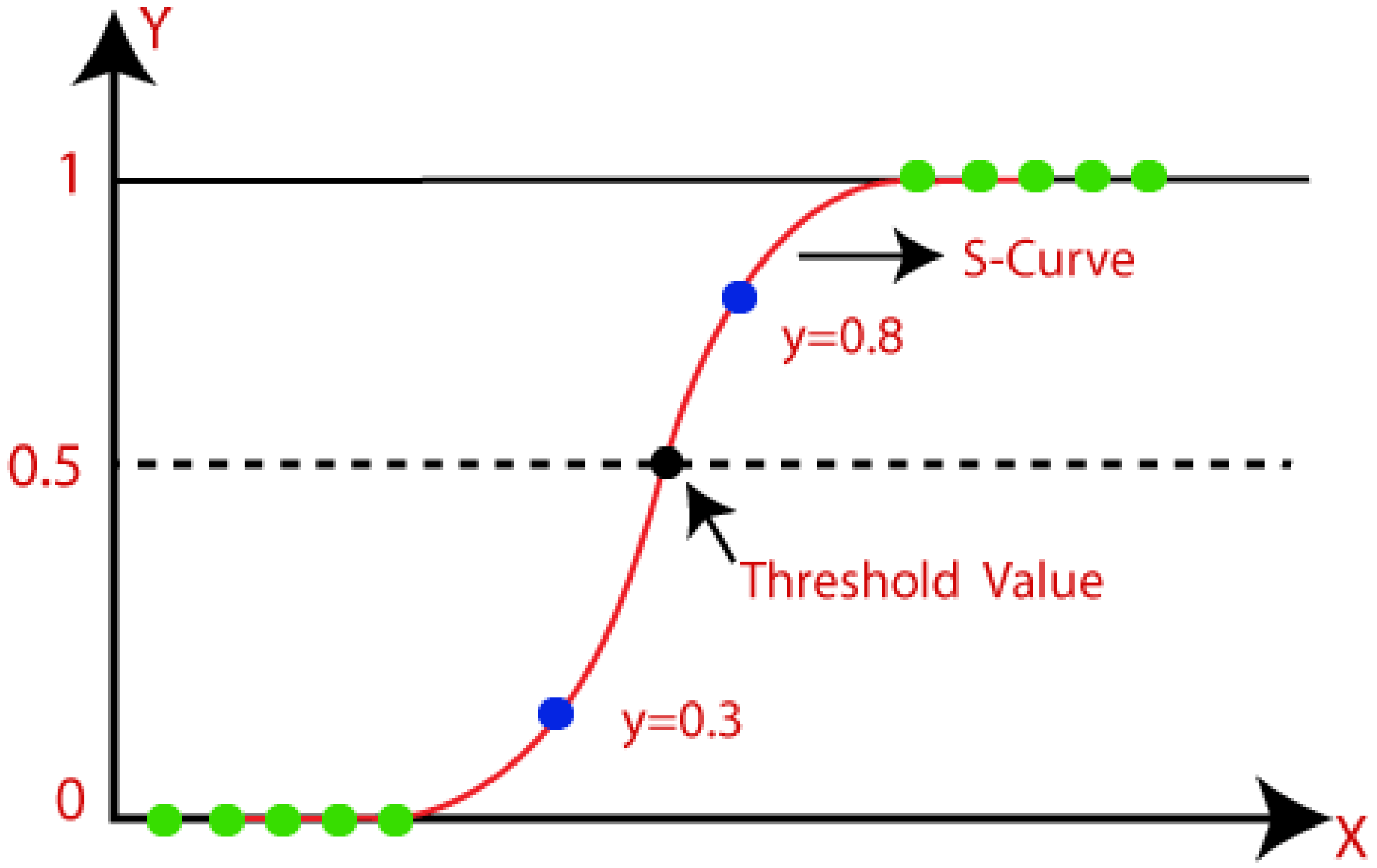
- **Linear Regression** is used for solving **Regression problems**, whereas **Logistic regression is used for solving the classification problems**.

- In **Logistic regression**, instead of fitting a regression line, we fit an **"S" shaped logistic function**, which predicts two maximum values **(0 or 1**).

- The curve from the logistic function indicates the likelihood of something such as whether the cells are cancerous or not, whether a mouse is obese or not based on its weight, etc.

- Logistic Regression can be used to classify the observations using different types of data and can easily determine the most effective variables used for the classification.

- Note: **Logistic regression uses the concept of predictive modeling as regression; therefore, it is called logistic regression, but is used to classify samples; therefore, it falls under the classification algorithm.**


Y
1
S-Curve
y=0.8
0.5
Threshold Value
y=0.3
0
X

Logistic Function (Sigmoid Function)

- The **sigmoid function** is a mathematical function used to map the predicted values to probabilities.

- It maps any real value into another value within a range of 0 and 1.

- The value of the logistic regression must be between **0 and 1**, which cannot go beyond this limit, so it forms a curve like the **"S"** form. The S-form curve is called the **Sigmoid function** or the **logistic function**.

- In logistic regression, we use the concept of the **threshold value**, which defines the **probability of either 0 or 1**.

- The values above the threshold value tend to be 1, and a value below the threshold value tends to be 0.

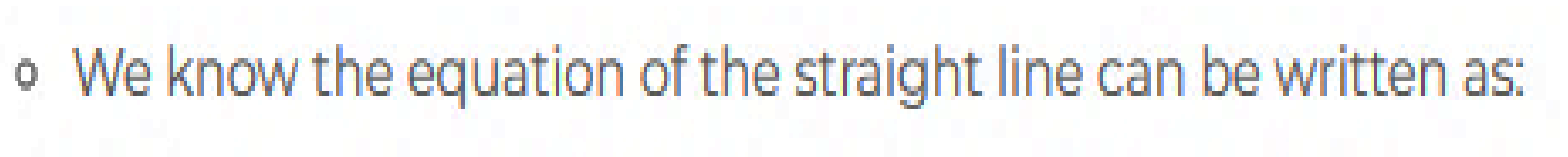
Assumptions for Logistic Regression

- The dependent variable must be categorical in nature.
- The independent variable should not have multi-collinearity.

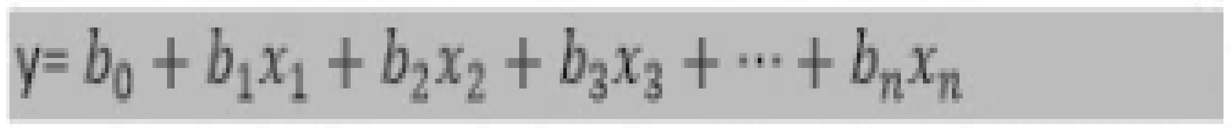
Logistic Regression Equation - 1

- The Logistic regression equation can be obtained from the Linear Regression equation.

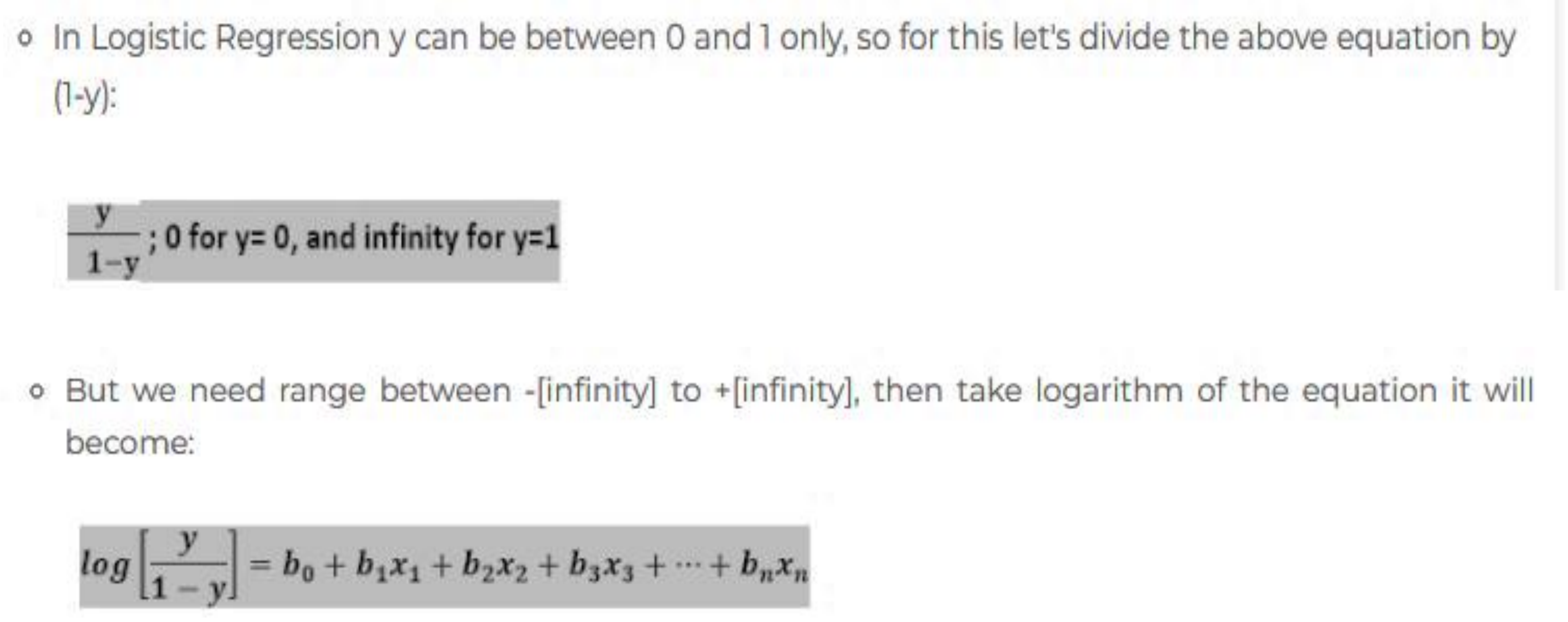
- The mathematical steps to get Logistic Regression equations are given below:



- We know the equation of the straight line can be written as:



$$y = b_0 + b_1x_1 + b_2x_2 + b_3x_3 + \cdots + b_nx_n$$



- In Logistic Regression y can be between 0 and 1 only, so for this let's divide the above equation by (1-y):

$$\frac{y}{1-y}\text{; 0 for y= 0, and infinity for y=1}$$

- But we need range between -[infinity] to +[infinity], then take logarithm of the equation it will become:

$$log\left[\frac{y}{1-y}\right] = b_0 + b_1x_1 + b_2x_2 + b_3x_3 + \cdots + b_nx_n$$

- The above equation is the final **equation for Logistic Regression.**

Logistic Regression Equation - 2

$$\boxed{s(x) = \frac{1}{1+e^{-x}}}$$

OR

$$\boxed{y = \frac{1}{1+e^{-x}}}$$

Where,

$x = a_0 + a_1x$

Here,

a_0 = Intercept

a_1 is the coefficient of Slope

OR

$$Y = \text{Sigmoid}\,(b_0 + b_1X_1 + b_2X_2 + \ldots + b_nX_n)$$

where, $\text{Sigmoid} = f(x) = \dfrac{1}{1 + e^{-Y}}$

Therefore, $Y = \dfrac{1}{1 + e^{-(b0 + b1X1 + b2X2 + \ldots + bnXn)}}$

Where,

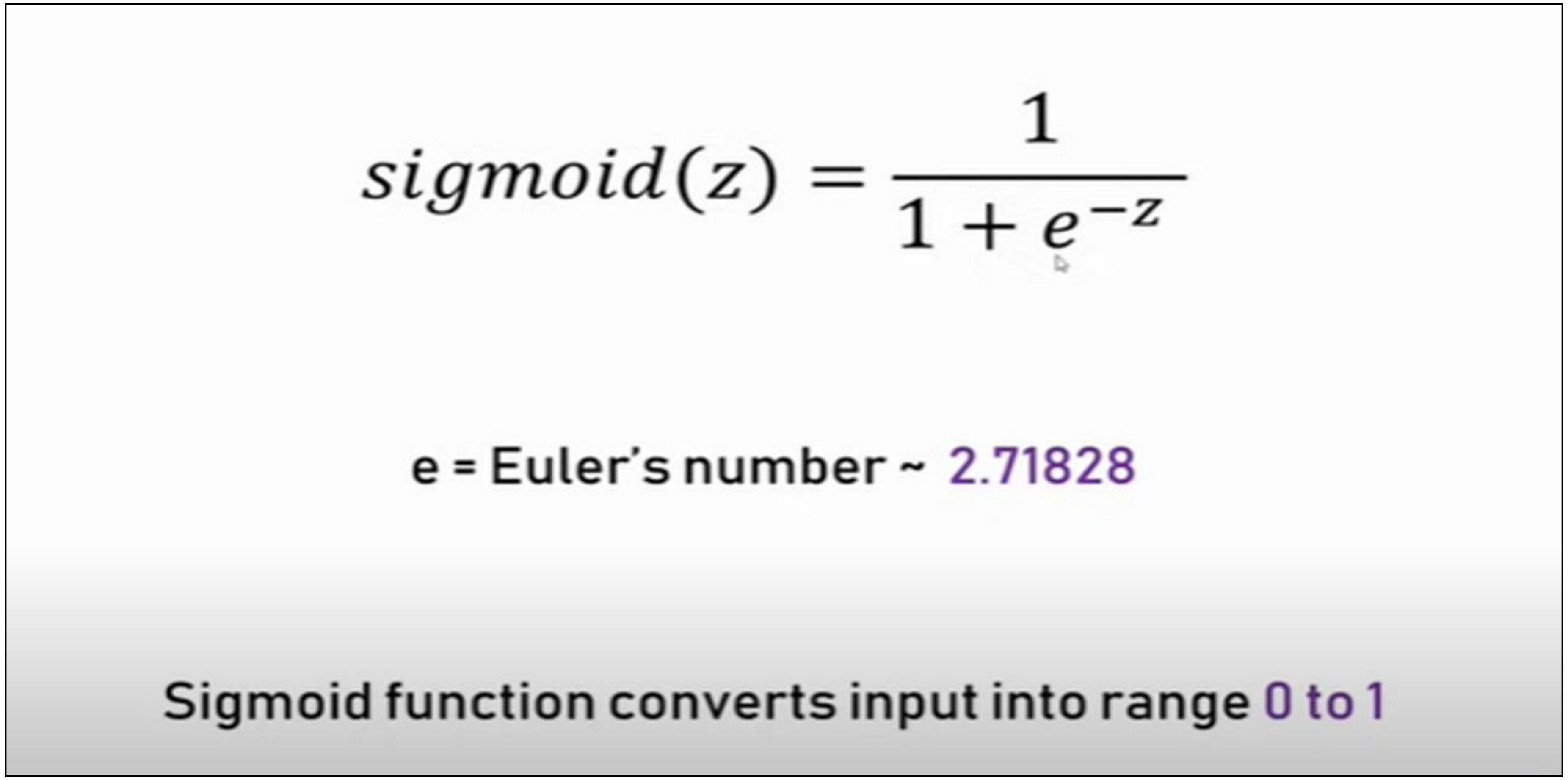
Y = Dependent Variable (DV)

X_1, X_2, X_n = Independent Variable (IV)

$b0$ = Y intercept

b_1, b_2, b_n = Coefficient of slope

Another way to write the equation is :



$$sigmoid(z) = \frac{1}{1 + e^{-z}}$$

e = Euler's number ~ 2.71828

Sigmoid function converts input into range 0 to 1

Type of Logistic Regression

- Based on the categories, Logistic Regression can be classified into three types:

- **Binomial:** In binomial Logistic regression, there can be only **two possible types** of dependent variables, such as **0 or 1, Pass or Fail, etc.**

- **Multinomial:** In multinomial Logistic regression, there can be **3 or more possible unordered types** of the dependent variable, such as **"cat", "dogs", or "sheep“**

- **Ordinal:** In ordinal Logistic regression, there can be **3 or more possible ordered types** of dependent variables, such as **"low", "Medium", or "High".**

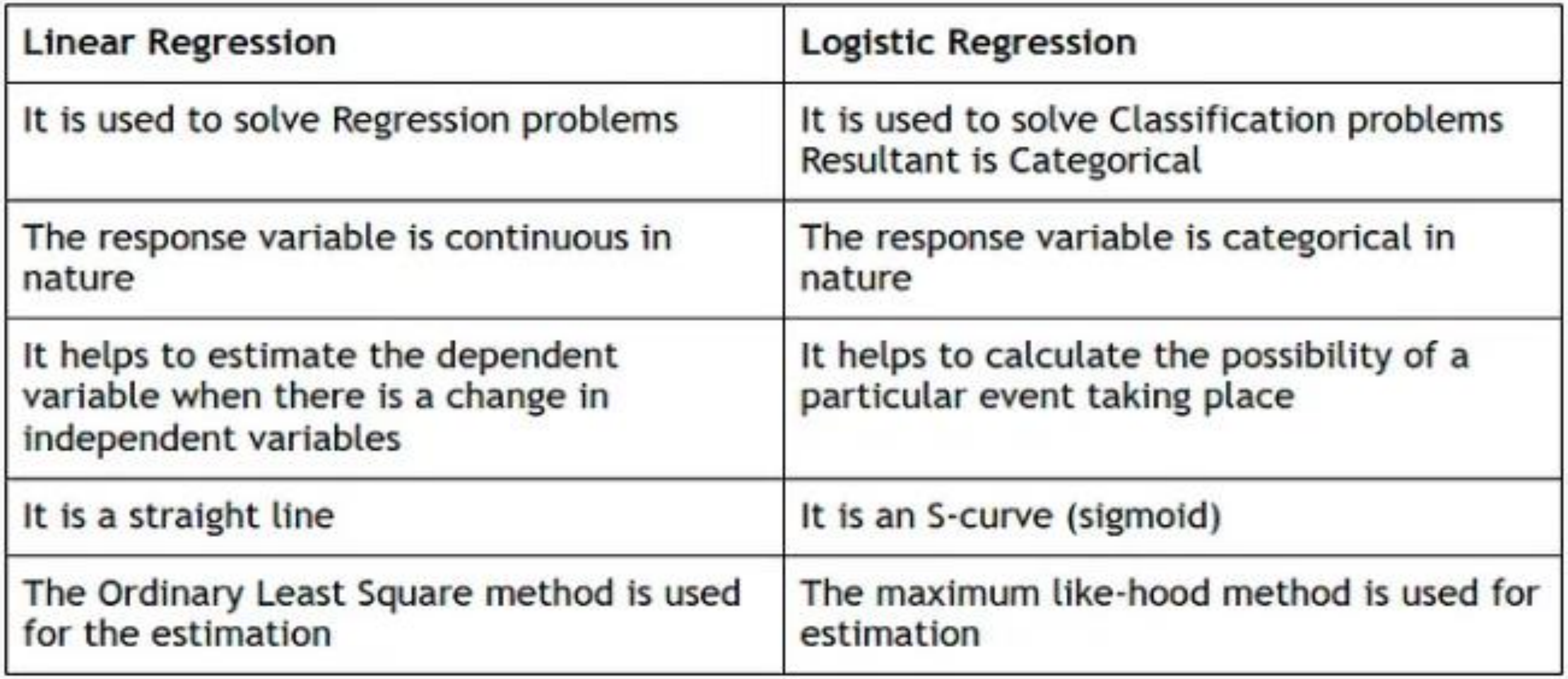
Advantages of the Logistic Regression Algorithm

- Logistic regression performs better when the data is linearly separable.

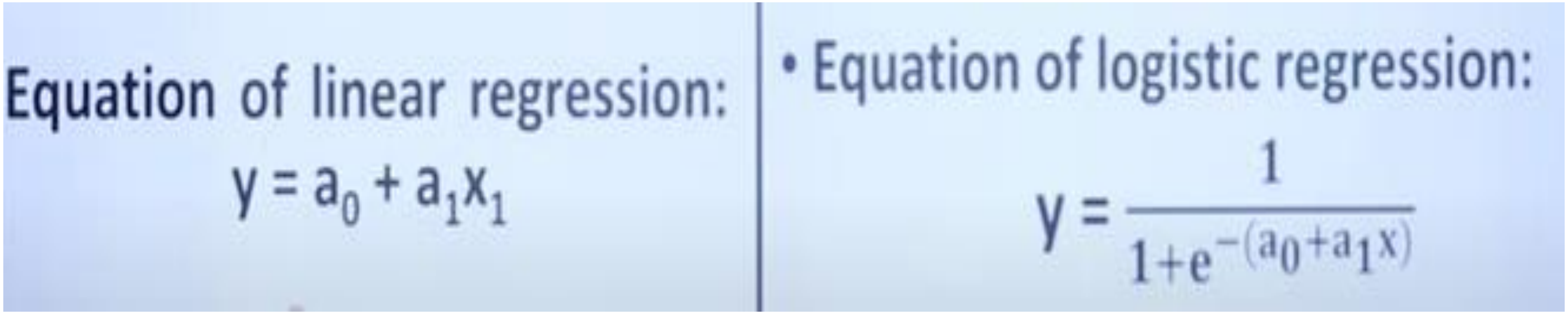
- It does not require too many computational resources as it’s highly interpretable.

- It is easy to implement and train a model using logistic regression

Linear Regression Vs Logistic Regression



Linear Regression	Logistic Regression
It is used to solve Regression problems	It is used to solve Classification problems Resultant is Categorical
The response variable is continuous in nature	The response variable is categorical in nature
It helps to estimate the dependent variable when there is a change in independent variables	It helps to calculate the possibility of a particular event taking place
It is a straight line	It is an S-curve (sigmoid)
The Ordinary Least Square method is used for the estimation	The maximum like-hood method is used for estimation

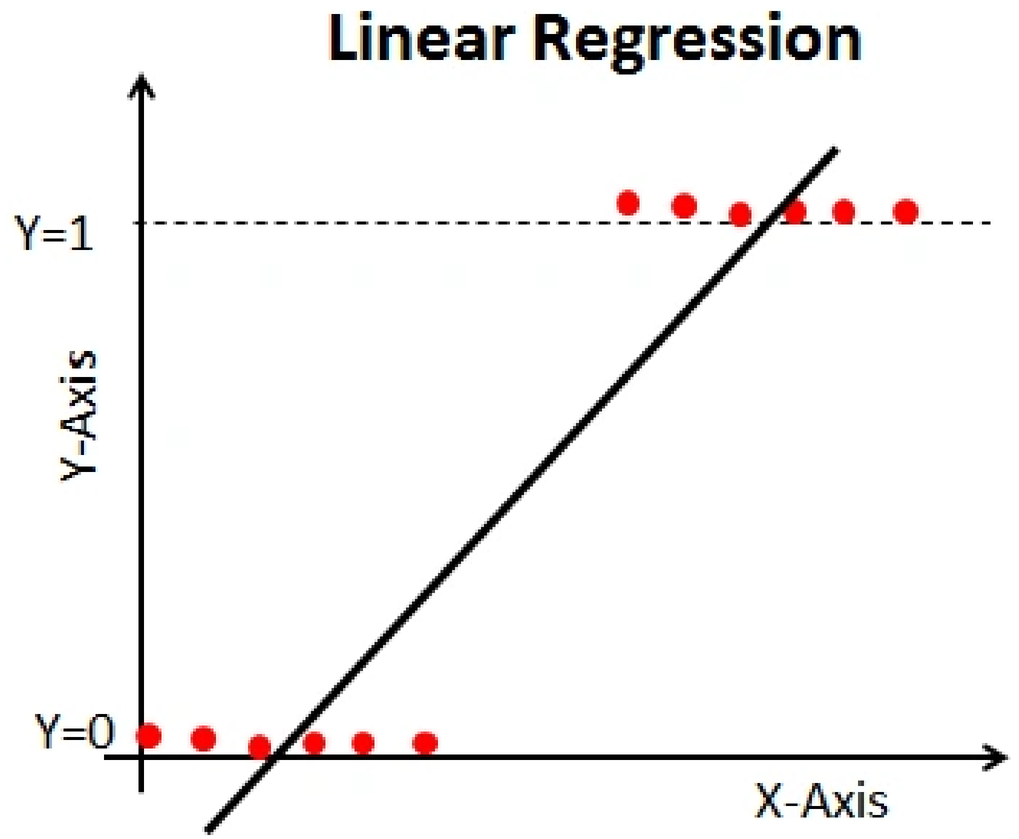


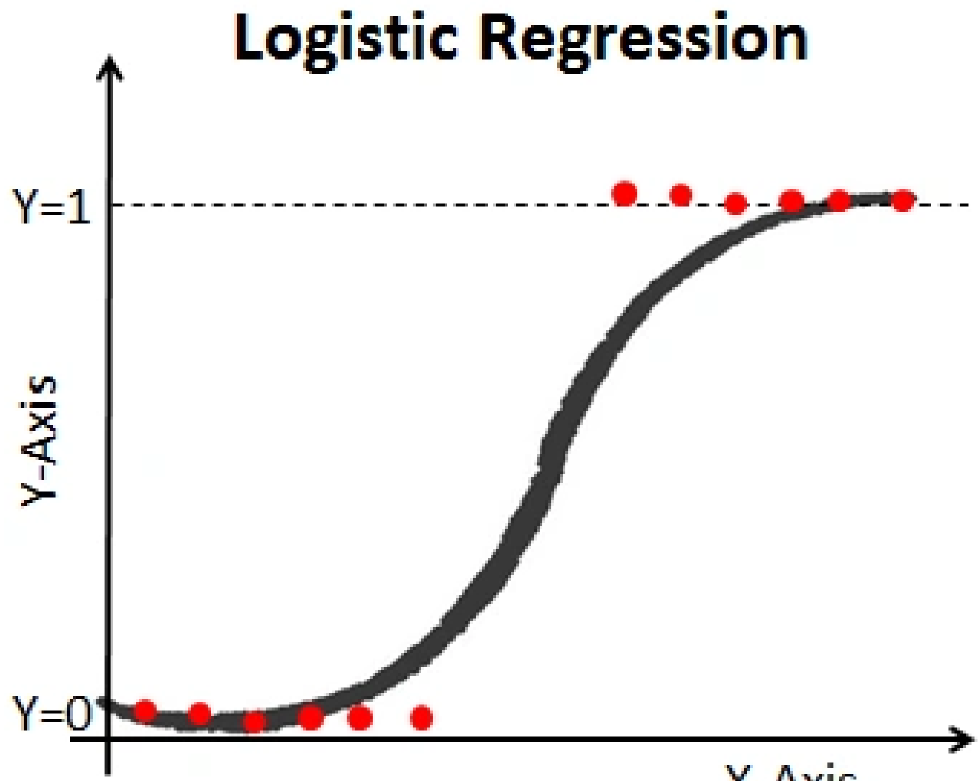
Equation of linear regression:

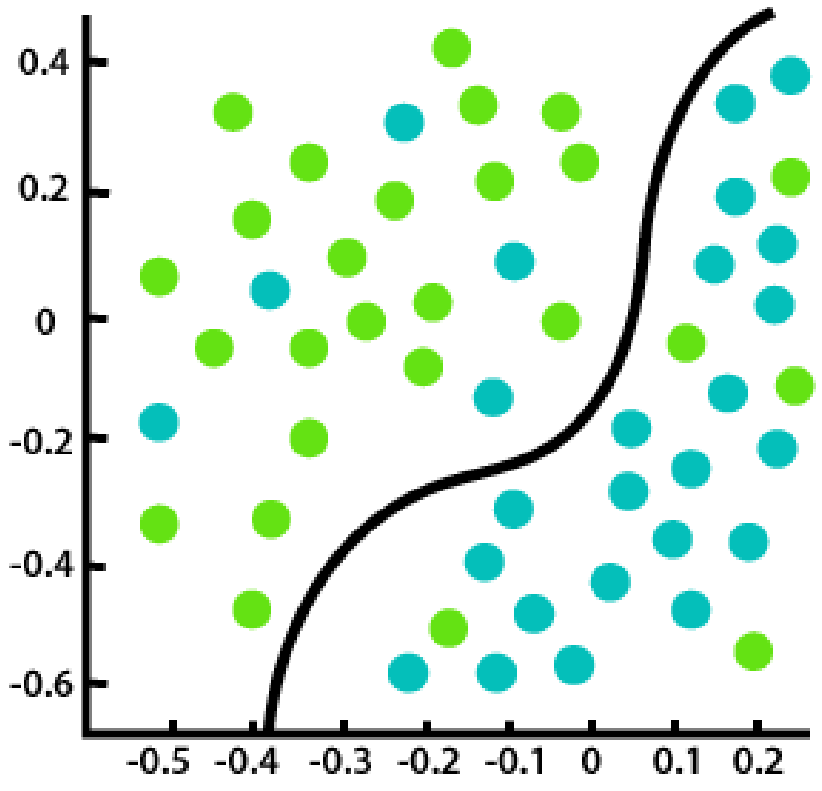
$$y = a_0 + a_1 x_1$$

- Equation of logistic regression:

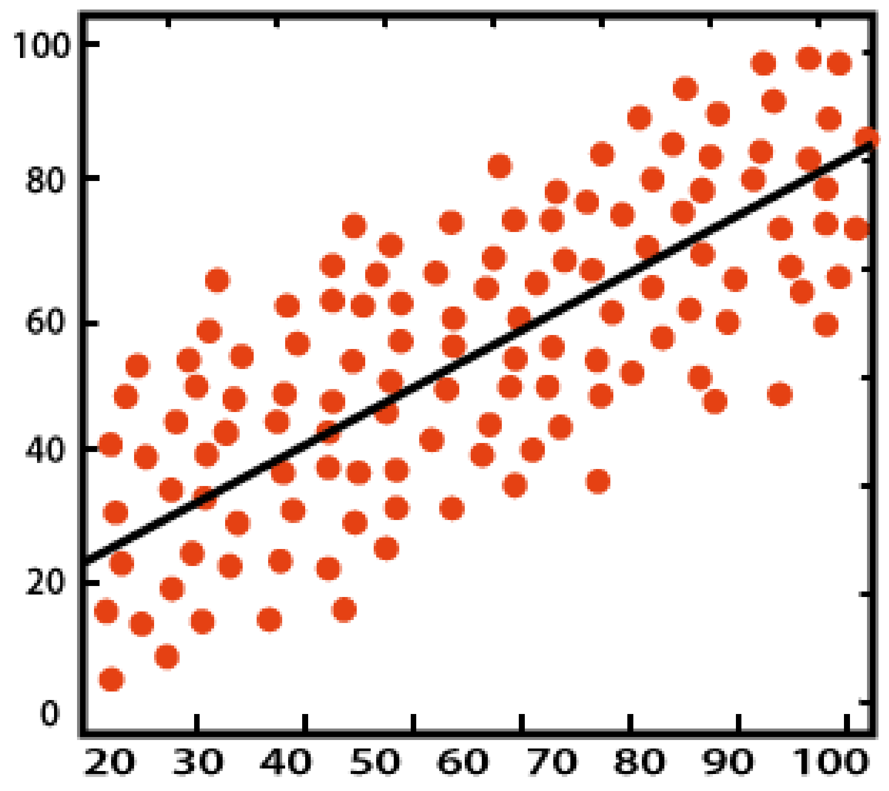
$$y = \frac{1}{1+e^{-(a_0+a_1 x)}}$$


Linear Regression
Y=1
Y-Axis
Y=0
X-Axis


Logistic Regression
Y=1
Y-Axis
Y=0
X-Axis



Classification



Regression

Numerical Problems on Logistic Regression

Q1) A logistic regression model is given by: $z = -3 + 0.5X1 + 2X2$

For $X1 = 4$ and $X2 = 1$, compute the probability that $Y = 1$.

Steps:

➔Compute z .

➔sigmoid function

Answer:(z=1, P(y=1) = 73.1%)

Given a logistic regression equation:

$$P(Y=1) = \frac{1}{1+e^{-(1.5X_1 - 2X_2 + 1)}}$$

Find the **log-odds** when $X_1 = 2$ and $X_2 = 3$.

Question - 1

- The dataset of pass or fail in an exam of 5 students is given in the table.
- Use logistic regression as classifier to answer the following questions.

1. Calculate the probability of pass for the student who studied 33 hours.
2. At least how many hours student should study that makes he will pass the course with the probability of more than 95%.

Hours Study	Pass (1) / Fail (0)
29	0
15	0
33	1
28	1
39	1

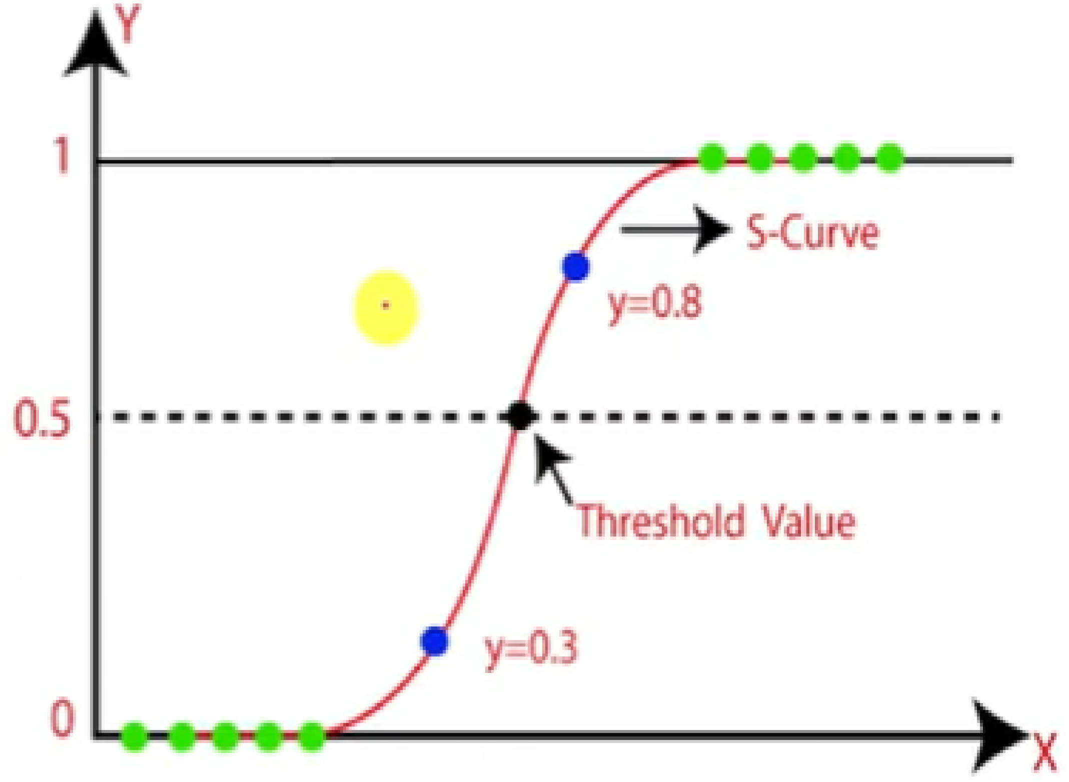
Assume the model suggested by the optimizer for odds of passing the course is,

$$\log(odds) = -64 + 2 * hours$$

Solution

- We use Sigmoid Function in logistic regression

- $s(x) = \frac{1}{1+e^{-x}}$



1. Calculate the probability of pass for the student who studied 33 hours.

$$s(x) = \frac{1}{1 + e^{-x}}$$

Hours Study	Pass (1) / Fail (0)
29	0
15	0
33	1
28	1
39	1

- $p = \frac{1}{1+e^{-z}}$
- $z = -64 + 2 * 33 = -64 + 66 =$ 2
- $p = \frac{1}{1+e^{-2}} = 0.88$
- That is, if student studies 33 hours, then there is **88% chance** that the student will pass the exam

$$\log(odds) = z = -64 + 2 * hours$$

2. At least how many hours student should study that makes he will pass the course with the probability of more than 95%.

Hours Study	Pass (1) / Fail (0)
29	0
15	0
33	1
28	1
39	1

$$\boldsymbol{p = \frac{1}{1+e^{-z}} = 0.95}$$

$$0.95 * (1 + e^{-z}) = 1$$

$$0.95 * e^{-z} = 1 - 0.95$$

$$e^{-z} = \frac{0.05}{0.95} = 0.0526$$

- Taking natural log **(ln)** on both sides, we get,

$$\ln(e^{-z}) = \ln(0.0526)$$

- But, we have a formula like $\ln(e^{x}) = x$

$$-z = ln(0.0526) = -2.94$$

$$z = 2.94$$

We know,

- $z = 2.94$

Given, →

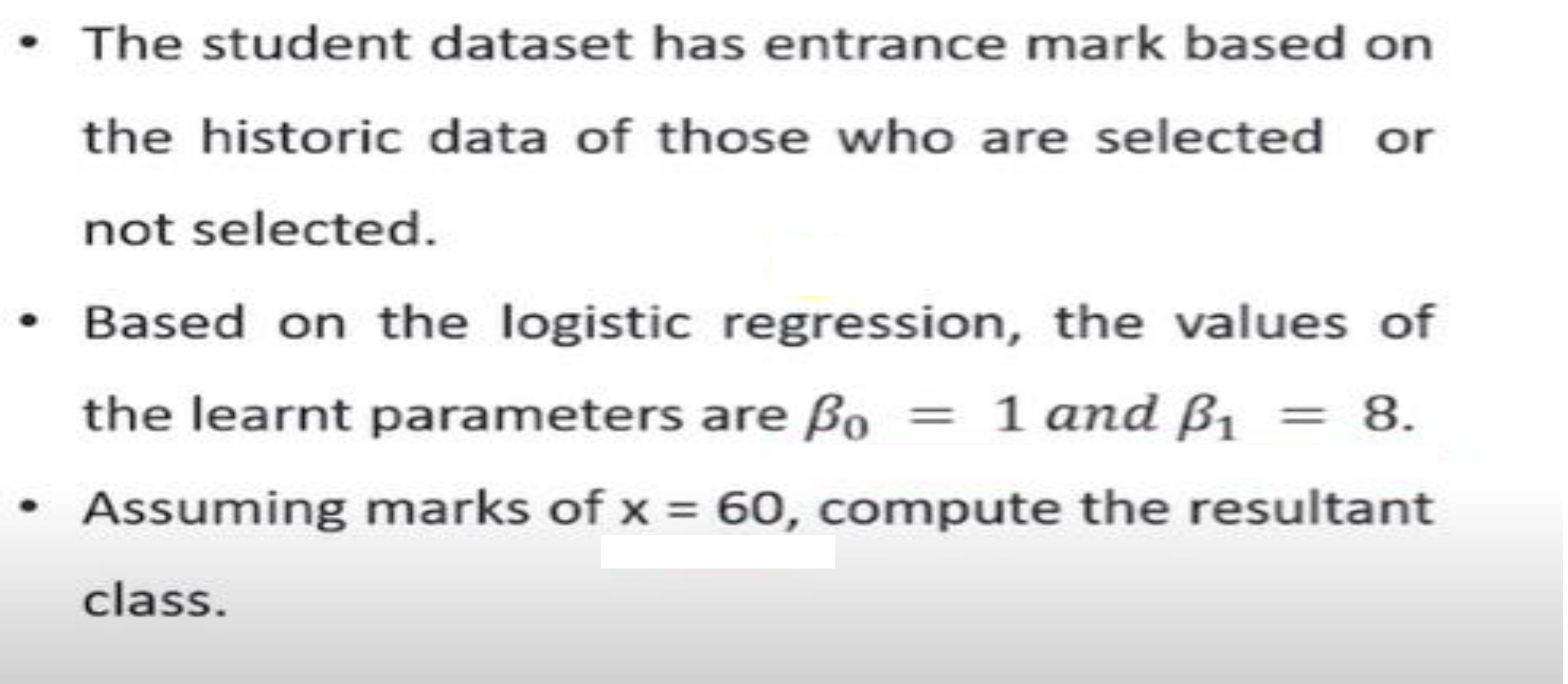
- $\log(odds) = z = -64 + 2 * hours$

Substituting Z in the above equation →

- $2.94 = -64 + 2 * hours$
- $2 * hours = 2.94 + 64$
- $2 * hours = 66.94$
- $hours = \frac{66.94}{2}$
- $\boldsymbol{hours = 33.47\ Hours}$

- The student should study **at least 33.47 hours**, so that he will pass the exam with more than 95% probability

Question - 2



- The student dataset has entrance mark based on the historic data of those who are selected or not selected.
- Based on the logistic regression, the values of the learnt parameters are $\beta_0 = 1 \, and \, \beta_1 = 8$.
- Assuming marks of x = 60, compute the resultant class.

Solution

$$p(x) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x)}}$$

$$\beta_0 + \beta_1 x = 481$$

$$p(x) = \frac{1}{1 + e^{-481}} = 0.44$$

If we assume the threshold value as 0.5, then it is observed that 0.44 < 0.5, therefore, the candidate with marks 60 is not selected.

Some Points to Remember

- Logistic Regression Equation is given as :

$$y = \frac{1}{1 + e^{-z}}$$

$$Or, \qquad y = \frac{1}{1 + e^{-(a.x+b)}}$$

- Y in the equation is the probability that given example will fall in certain class.

- Its value ranges from 0 to 1 as the value of sigmoid function ranges from 0 to 1.

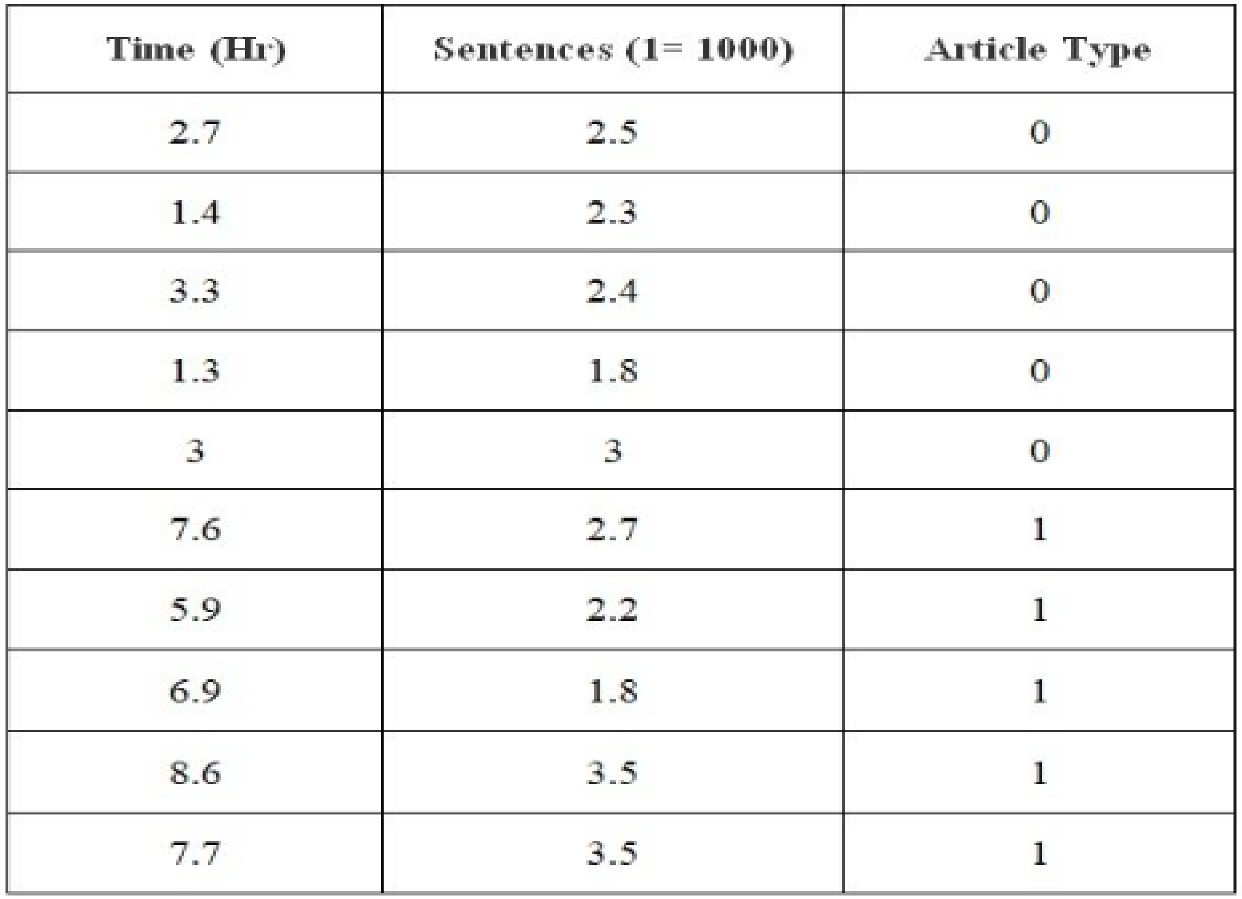
- If the result is near **0**, we can say that the example falls to **negative class**.

- If it is closer to **1**, we can say it falls to **positive class**.

$$\text{prediction} = + \text{ IF } y > 0.5$$
$$\text{prediction} = - \text{ IF } y \leq 0.5$$

Question -3

- Given a sample as shown below,



Time (Hr)	Sentences (1= 1000)	Article Type
2.7	2.5	0
1.4	2.3	0
3.3	2.4	0
1.3	1.8	0
3	3	0
7.6	2.7	1
5.9	2.2	1
6.9	1.8	1
8.6	3.5	1
7.7	3.5	1

Some samples of two classes **Technical (1)** and **Non-technical(0)**

- Given two classes labeled **0** and **1** representing **non-technical** and **technical article**(*class 0 is the negative class which means if we get a probability less than 0.5 from the sigmoid function, it is classified as 0. Similarly, class 1 is a positive class and if we get a probability greater than 0.5, it is classified as 1*).

- Each class has two features **Time**, which represents the average time required to read an article in an hour, and **Sentences**, representing the number of sentences in a book (here 2.2 means 2.2k or 2200 sentences).
- Now we need to train our logistic regression model.

- Training involves finding optimal values of coefficients which are B0, B1, and B2. While training, we find some value of coefficients **i** in the first step and use those coefficients in another step to optimize their value. We continue to do it until we get consistent accuracy from the model. In this example, we have iterated for 20 times, but we can iterate more to get higher accuracy.

- From **Logistic Regression** implementation, after 20 iterations, the following data values are given:
  - B0 = **-0.1068913**
  - B1 = **0.41444855**
  - B3 = **-0.2486209**

The decision boundary is given as:

Z = B0+B1*X1+B2*X2

Also given, **X1 = 1.9 and X2 = 3.1,**

Using the sigmoid function from **Logistic Regression,** find the probability and predict the class of given variables.

Solution

Given the following data:-

Z = B0+B1*X1+B2*X2

- B0 = **-0.1068913**
- B1 = **0.41444855**
- B3 = **-0.2486209**

- **X1 = 1.9 and X2 = 3.1**

Let’s now calculate Z using the given decision equation,

Z = - 0.1068913 +0.41444855*Time-0.2486209*Sentences

Z = -0.101818+0.41444855*1.9 -0.2486209*3.1

Z = **-0.085090545**

- Now, we use the sigmoid function from **Logistic Regression** to find the probability and thus predict the class of given variables as:-

$$y = \frac{1}{1 + e^{-Z}}$$

$$Or, \quad y = \frac{1}{1 + e^{-(-0.085090545)}}$$

$$Or, \quad y = 0.47747429$$

- *As y is less than 0.5 (y< 0.5), we can safely classify given a sample to class Non-technical.*

Hypothesis Representation in Logistic Regression

- In **logistic regression**, the **hypothesis function** is used to predict the probability that a given input X belongs to a particular class.
- Hypothesis function maps input features (x) to an output value between 0 and 1, representing a probability.
- Unlike linear regression, logistic regression uses the **sigmoid function** to ensure the output remains in the range [0,1].
- Hypothesis Function $h\theta(X)$in logistic regression is given by:

$$h_\theta(X) = g(\theta^T X) = \frac{1}{1 + e^{-(\theta_0 + \theta_1 X_1 + \theta_2 X_2 + \cdots + \theta_n X_n)}}$$

- X is the input feature vector, θ is the parameter vector (weights).

Q) Compute Hypothesis Function in Logistic Regression for given feature values and parameters.

- **Input features:**

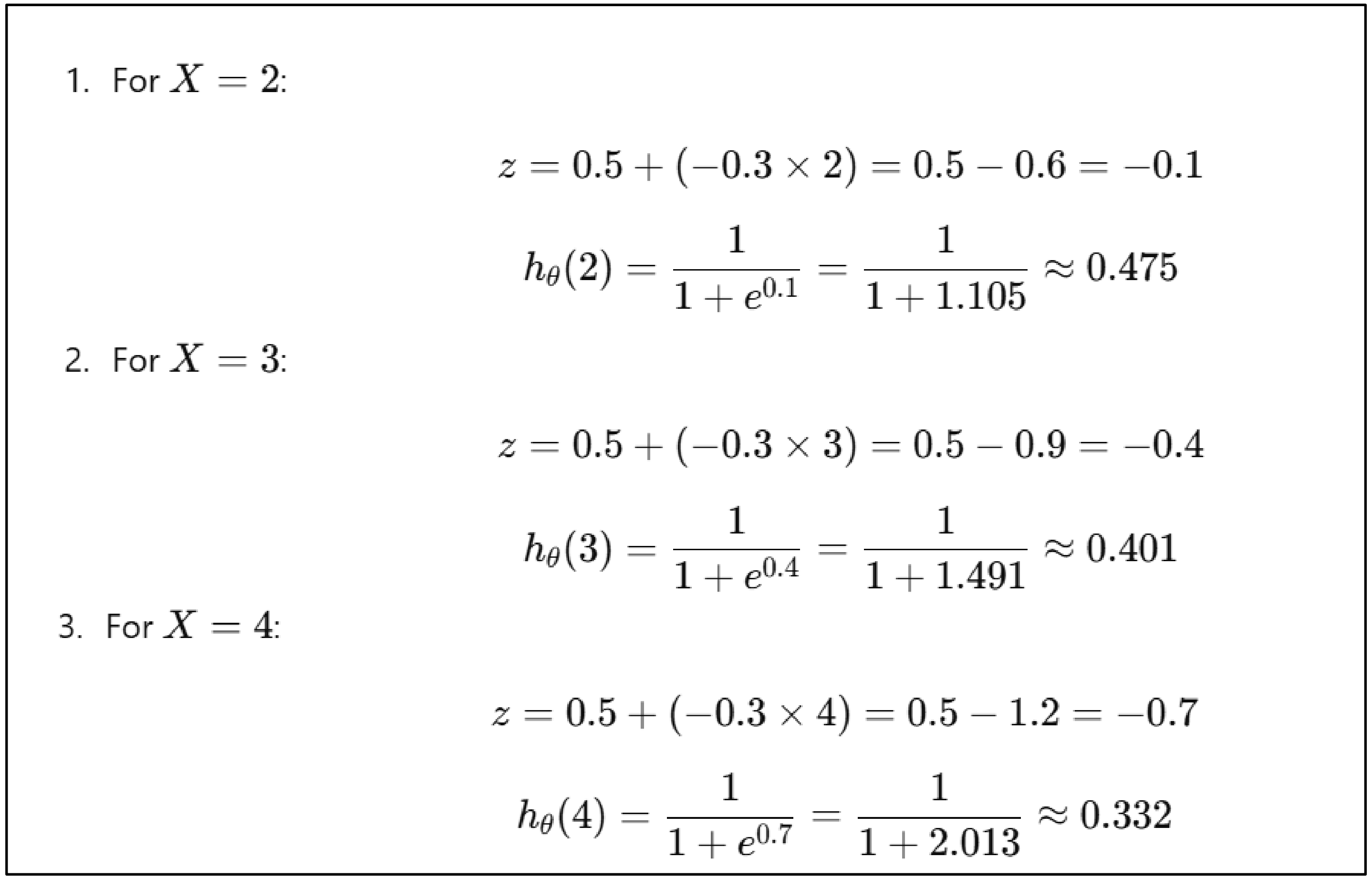
  X=2, 3, 4

- **Parameter values (weights):**

  $\theta_0=0.5, \theta_1=-0.3$

Answer:

$$h_\theta(X) = \frac{1}{1+e^{-(\theta_0+\theta_1 X)}}$$



1. For $X = 2$:

$$z = 0.5 + (-0.3 \times 2) = 0.5 - 0.6 = -0.1$$

$$h_\theta(2) = \frac{1}{1+e^{0.1}} = \frac{1}{1+1.105} \approx 0.475$$

2. For $X = 3$:

$$z = 0.5 + (-0.3 \times 3) = 0.5 - 0.9 = -0.4$$

$$h_\theta(3) = \frac{1}{1+e^{0.4}} = \frac{1}{1+1.491} \approx 0.401$$

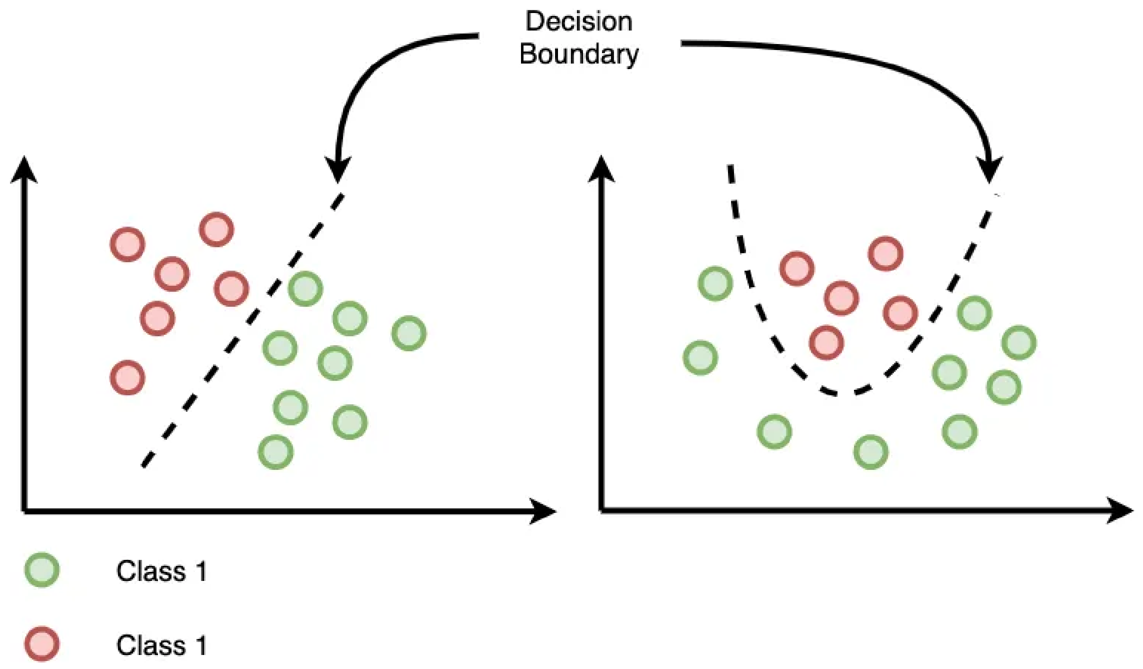
3. For $X = 4$:

$$z = 0.5 + (-0.3 \times 4) = 0.5 - 1.2 = -0.7$$

$$h_\theta(4) = \frac{1}{1+e^{0.7}} = \frac{1}{1+2.013} \approx 0.332$$

Decision Boundary in Logistic Regression

- **Decision boundary** is the surface that separates different classes based on the hypothesis function. It defines the point where the model switches from predicting one class to another.
-



- In logistic regression decide a proper fit to the decision boundary so that it will be able to predict which class a new feature set might correspond to.
- The interesting fact about logistic regression is the utilization of the sigmoid function as the target class estimator.

Q2)A logistic regression model is: $P(Y=1) = \frac{1}{1+e^{-(2X_1+3X_2-4)}}$

Find the decision boundary equation where P(Y=1)=0.5.

Answer:

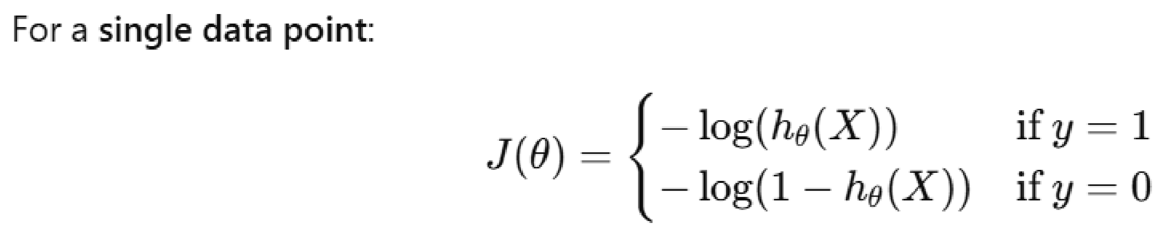
The decision boundary is $2X1+3X2=4$

Cost Function

- A cost function is a mathematical function that calculates the difference between the target actual values (ground truth) and the values predicted by the model.
- A function that assesses a machine learning model's performance also referred to as a loss function or objective function.
- Usually, the objective of a machine learning algorithm is to reduce the error or output of cost function.
- When it comes to Linear Regression, the conventional Cost Function employed is the Mean Squared Error.

- Mean Squared Error (MSE) as a cost function, but it is not suitable for logistic regression due to its nonlinearity introduced by the sigmoid function.
- Instead, use a logarithmic loss function (log loss), which is always convex and easier for gradient descent to minimize.
- If the actual value y=1, we want the predicted probability $h\theta(X)$to be close to 1.
- If $h\theta(X)$ is close to 0 (wrong prediction), the cost should be very high.
- Similarly, if y=0, we want $h\theta(X)$ to be close to 0, and we penalize it if it's closer to 1

Cost function



For a **single data point**:

$$J(\theta) = \begin{cases} -\log(h_\theta(X)) & \text{if } y = 1 \\ -\log(1 - h_\theta(X)) & \text{if } y = 0 \end{cases}$$

- If y=1 and the model predicts 0.99, cost is small.
- If y=1 and the model predicts 0.01, cost is large.
- If y=0 and the model predicts 0.99, cost is large.
- If y=0 and the model predicts 0.01, cost is small.

Q) Compute Cost for a Single Example

θ=[0.5,−0.4]

X=[1,2] (including bias term)

- Step 1: Compute z

$$z = \theta_0 + \theta_1 X_1$$

$$z = (0.5) + (-0.4 \times 2) = 0.5 - 0.8 = -0.3$$

- Step 2: Apply Sigmoid Function

$$h_\theta(X) = \frac{1}{1 + e^{-(-0.3)}}$$

$$= \frac{1}{1 + e^{0.3}} = \frac{1}{1 + 1.3499} = \frac{1}{2.3499} = 0.4256$$

Step 3: Compute Cost

$$J(\theta) = -\left[y\log(h_\theta(X)) + (1-y)\log(1-h_\theta(X))\right]$$

Case 1: When $y = 1$

$$J(\theta) = -[1 \cdot \log(0.4256) + 0 \cdot \log(1 - 0.4256)]$$

$$= -\log(0.4256) = -(-0.3711) = 0.857$$

Case 2: When $y = 0$

$$J(\theta) = -[0 \cdot \log(0.4256) + 1 \cdot \log(1 - 0.4256)]$$

$$= -\log(0.5744) = -(-0.2405) = 0.556$$

Multiclass Classification in Logistic Regression

- Logistic regression is naturally a **binary classifier**, meaning it predicts one of two classes (e.g., 0 or 1).
- However, it can be extended for **multiclass classification** using strategies like **One-vs-All (OvA)** and **Softmax Regression (Multinomial Logistic Regression).**

One-vs-All (OvA) / One-vs-Rest (OvR)

- Train **k** separate **binary logistic regression models**, one for each class.
- Each model predicts whether a sample belongs to that particular class or not.
- The final class is chosen as the one with the **highest probability**

Steps:

1. For a dataset with k classes, train k binary logistic regression models.
2. For a new input x, compute predictions $h_\theta^{(1)}(x), h_\theta^{(2)}(x), ..., h_\theta^{(k)}(x)$.
3. Assign the class with the **highest probability**.

Softmax Regression (Multinomial Logistic Regression)

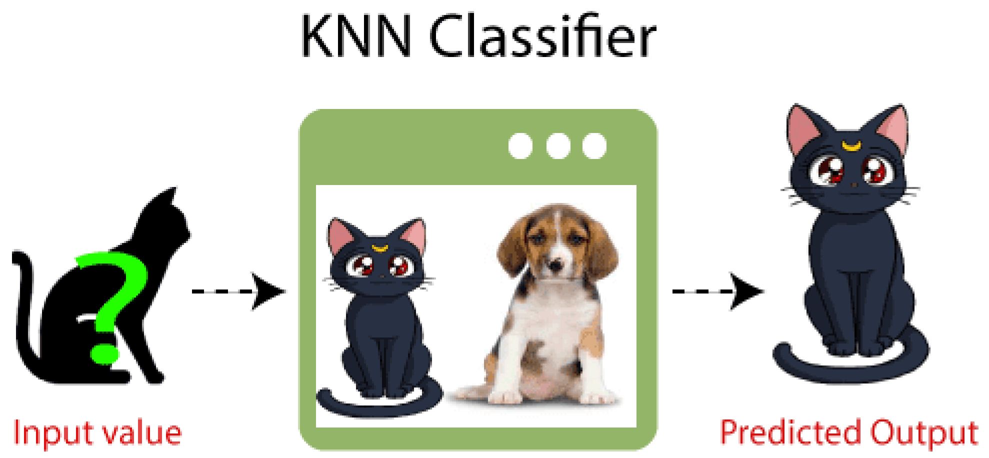
- Instead of training multiple binary classifiers, **Softmax** directly predicts the probability of each class.

K-Nearest Neighbors (KNN) Algorithm

- KNN algorithm is a type of supervised ML algorithm which can be used for both classification as well as regression predictive problems.
- it is mainly used for classification predictive problems in industry.
- The main idea behind KNN is to find the k-nearest data points to a given test data point and use these nearest neighbors to make a prediction.
- The value of k is a hyperparameter that needs to be tuned, and it represents the number of neighbors to consider.

- For classification problems, the KNN algorithm assigns the test data point to the class that appears most frequently among the k-nearest neighbors. In other words, the class with the highest number of neighbors is the predicted class.
- For regression problems, the KNN algorithm assigns the test data point the average of the k-nearest neighbors' values.
- The distance metric used to measure the similarity between two data points is an essential factor that affects the KNN algorithm's performance. The most commonly used distance metrics are Euclidean distance, Manhattan distance, and Minkowski distance.

- **Example:** Suppose, we have an image of a creature that looks similar to cat and dog, but we want to know either it is a cat or dog. So for this identification, we can use the KNN algorithm, as it works on a similarity measure. Our KNN model will find the similar features of the new data set to the cats and dogs images and based on the most similar features it will put it in either cat or dog category.



K-Nearest Neighbors Algorithm Working

- Step-1: Select the number K of the neighbors
- Step-2: Calculate the Euclidean distance of K number of neighbors
- Step-3: Take the K nearest neighbors as per the calculated Euclidean distance.
- Step-4: Among these k neighbors, count the number of the data points in each category.
- Step-5: Assign the new data points to that category for which the number of the neighbor is maximum.
- Step-6: Our model is ready.

KNN: Example

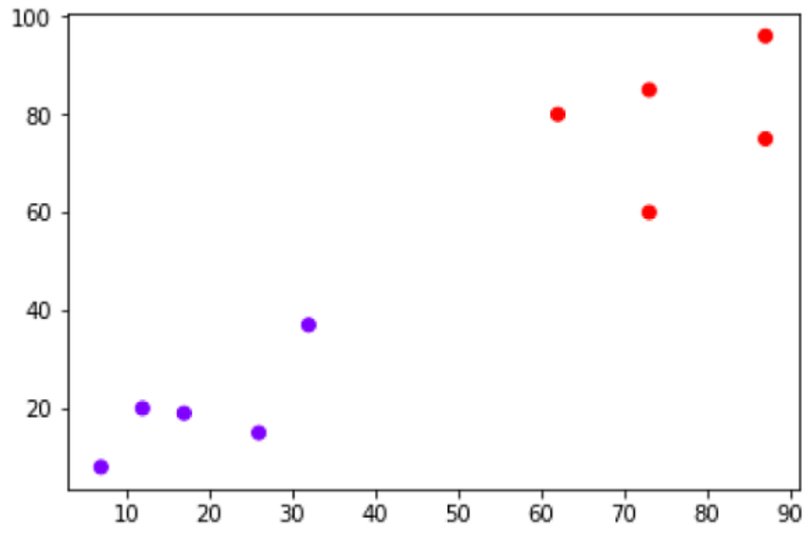


Fig 1. a plotted dataset

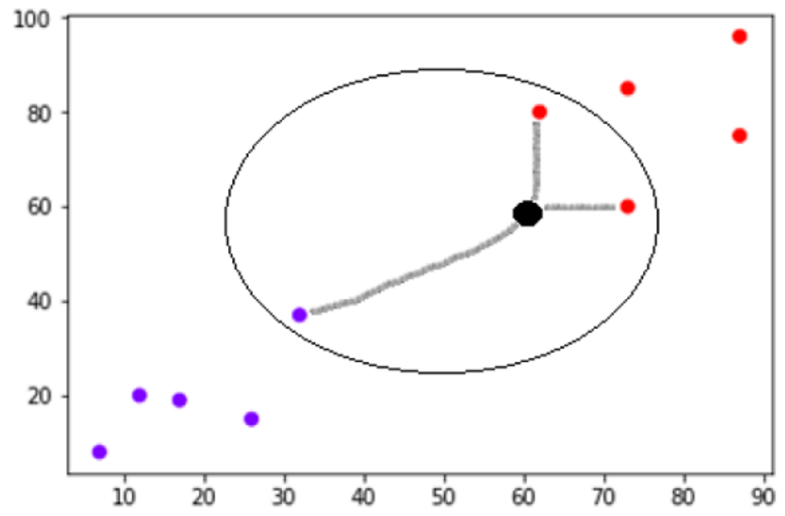


Fig 2: classify new data point with black dot into blue or red class. We are assuming K = 3 i.e. it would find three nearest data points.

Here the three nearest neighbors of the data point with black dot. Among those three, two of them lies in Red class hence the black dot will also be assigned in red class.

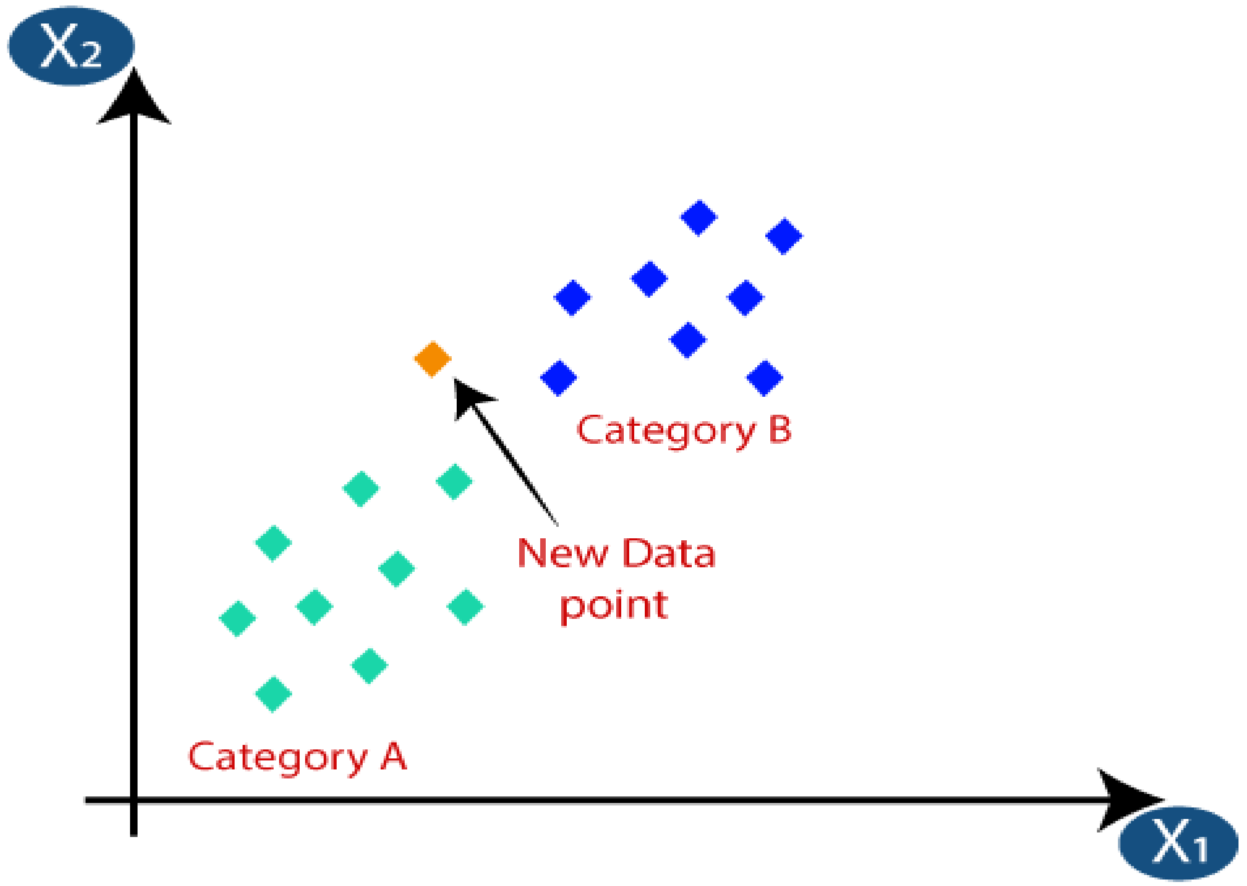
KNN Model working

- **Load the data** − This can be done using various libraries such as pandas or numpy.
- **Split the data** − split the data into training and test sets. The training set is used to train the KNN algorithm, while the test set is used to evaluate its performance.
- **Normalize the data** − Before training the KNN algorithm, it is essential to normalize the data to ensure that each feature contributes equally to the distance metric calculation.
- **Calculate distances** − Once the data is normalized, the KNN algorithm calculates the distances between the test data point and each data point in the training set.
- **Select k-nearest neighbors** − The KNN algorithm selects the k-nearest neighbors based on the distances calculated in the previous step.

- **Make a prediction** − For classification problems,assigns the test data point to the class that appears most frequently among the k-nearest neighbors. For regression problems,assigns the test data point the average of the k-nearest neighbors' values.
- **Evaluate performance** − evaluating using various metrics such as accuracy, precision, recall, and F1-score.

Example

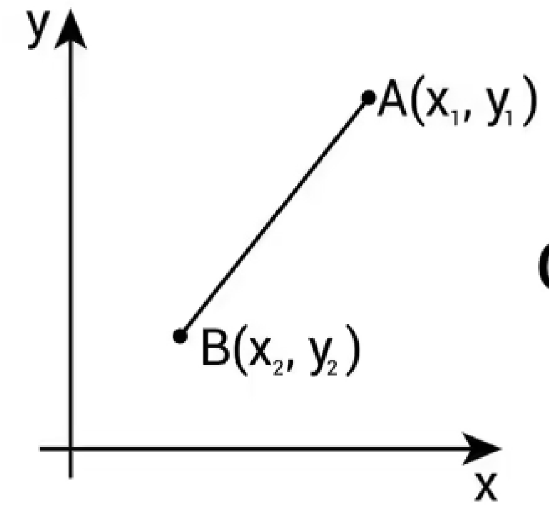
- Suppose we have a new data point, and we need to put it in the required category. Consider the below image:



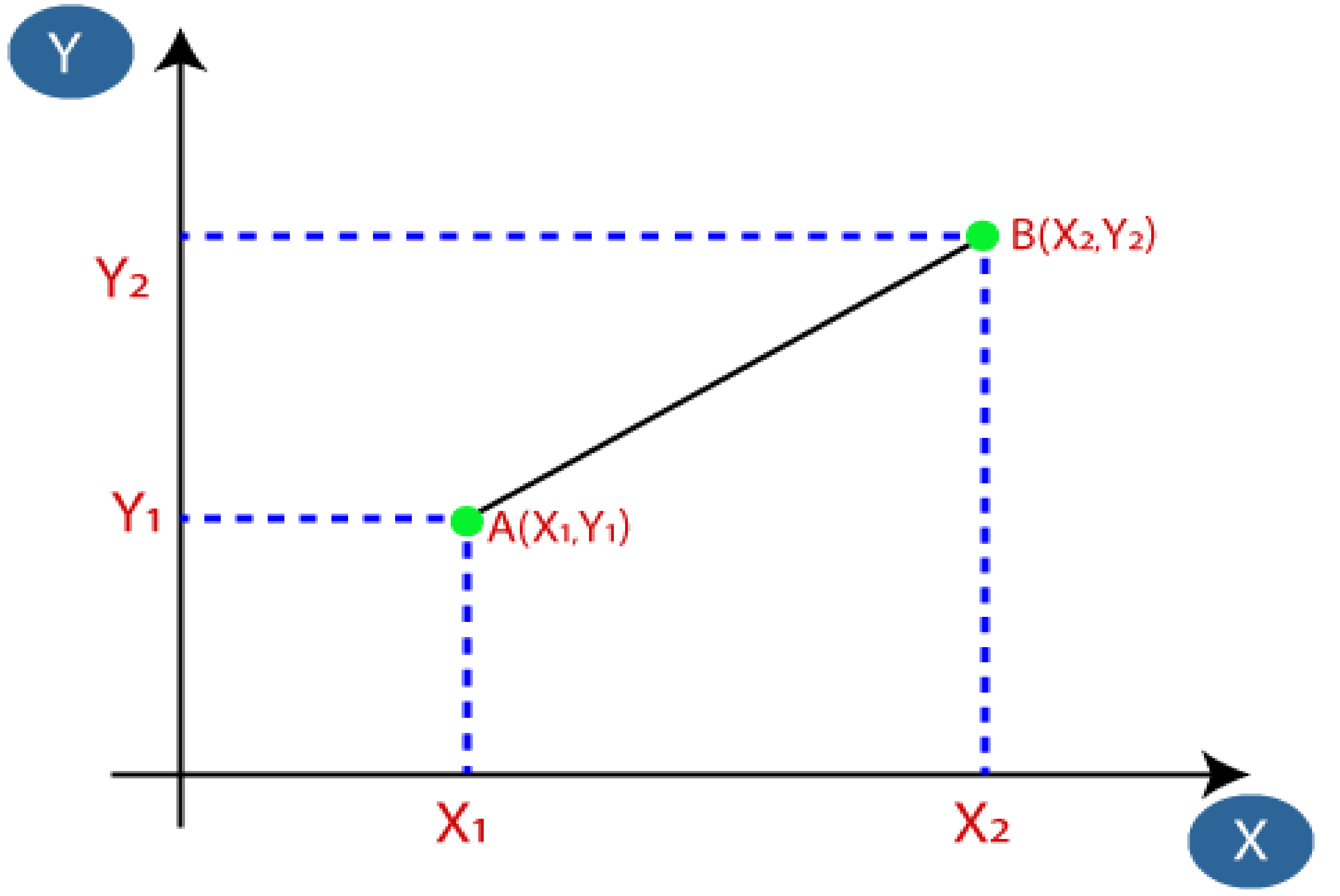
- Firstly, we will choose the number of neighbors, so we will choose the **k=5.**

- Next, we will calculate the **Euclidean distance** between the data points.

- The **Euclidean distance** is the distance between two points, which we have already studied in geometry. It can be calculated as:

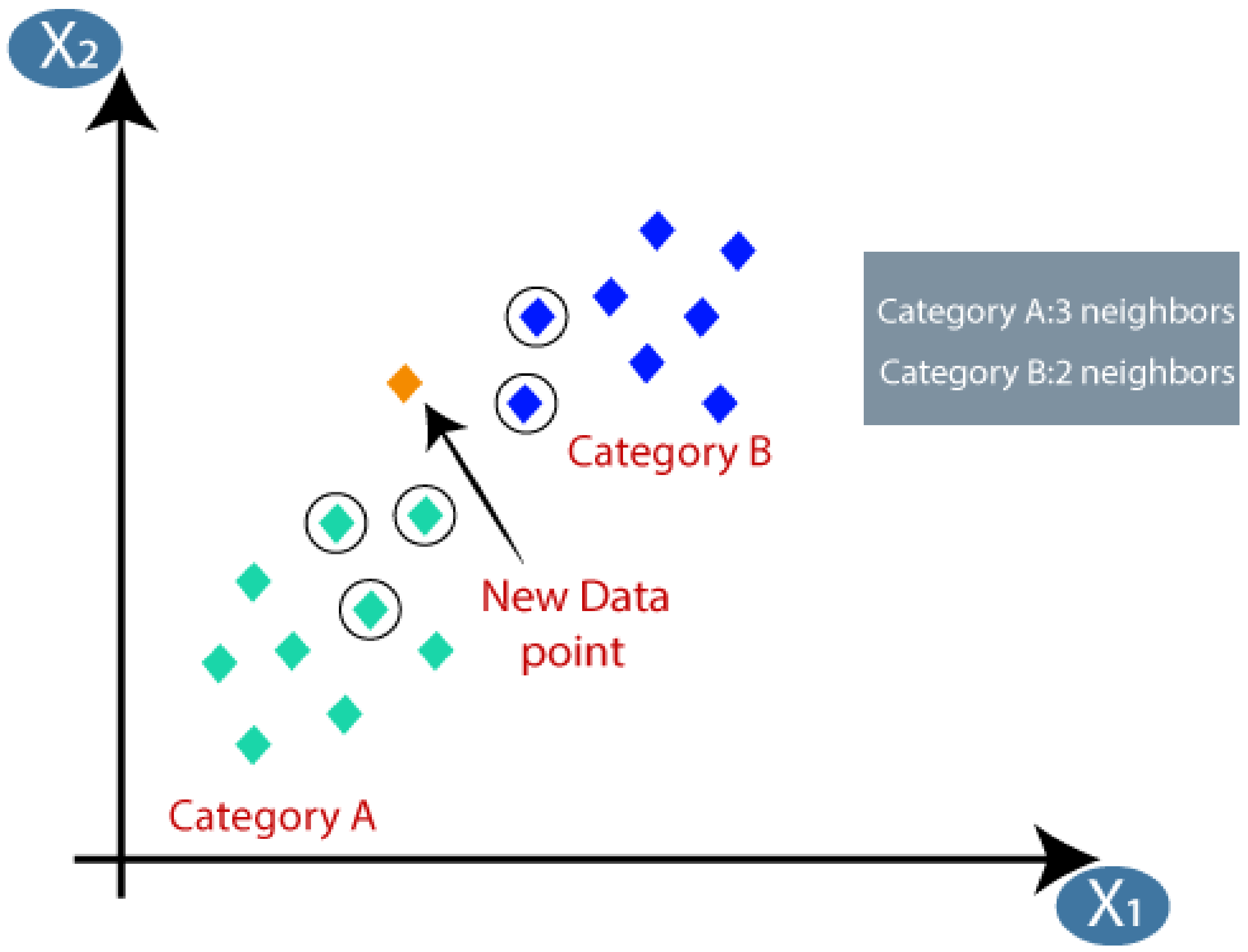


$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$



Euclidean Distance between A_1 and B_2 = $\sqrt{(X_2-X_1)^2+(Y_2-Y_1)^2}$

- By calculating the Euclidean distance we will get the nearest neighbors, as three nearest neighbors in category A and two nearest neighbors in category B. Consider the below image:



- As we can see the 3 nearest neighbors are from Category A, the new data point must belong to Category A.

How to select the value of K in the K-NN Algorithm?

- Below are some points to remember while selecting the value of K in the K-NN algorithm:

- There is no particular way to determine the best value for "K", so we need to try some values to find the best out of them. The most preferred value for K is 5.

- A very low value for K such as K=1 or K=2, can be noisy and lead to the effects of outliers in the model.

- Large values for K are good, but it may find some difficulties.

Advantages of KNN Algorithm

- It is simple to implement.
- It is robust to noisy training data
- It can be more effective if the training data is large.

Disadvantages of KNN Algorithm

- Always needs to determine the value of K which may be complex sometimes.

- The computation cost is high because of calculating the distance between the data points for all the training samples.

Distance Metrics Used in KNN Algorithm

- Distance metrics are used in supervised and unsupervised learning to calculate similarity in data points.

- The three main types of distance metrics are

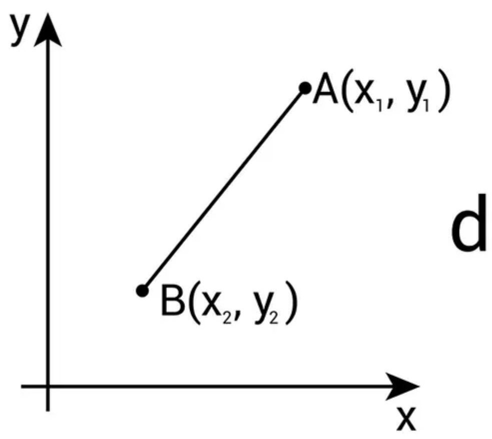
  ✓ **Euclidean Distance**

  ✓ **Manhattan Distance**

1. Euclidean Distance

- **Euclidean Distance** represents the shortest distance between two vectors.

- It is the square root of the sum of squares of differences between corresponding elements.



$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

OR

$$d((x,y),(a,b)) = \sqrt{(x-a)^2 + (y-b)^2}$$

2. Manhattan Distance

- Manhattan Distance is the sum of absolute differences between points across all the dimensions.

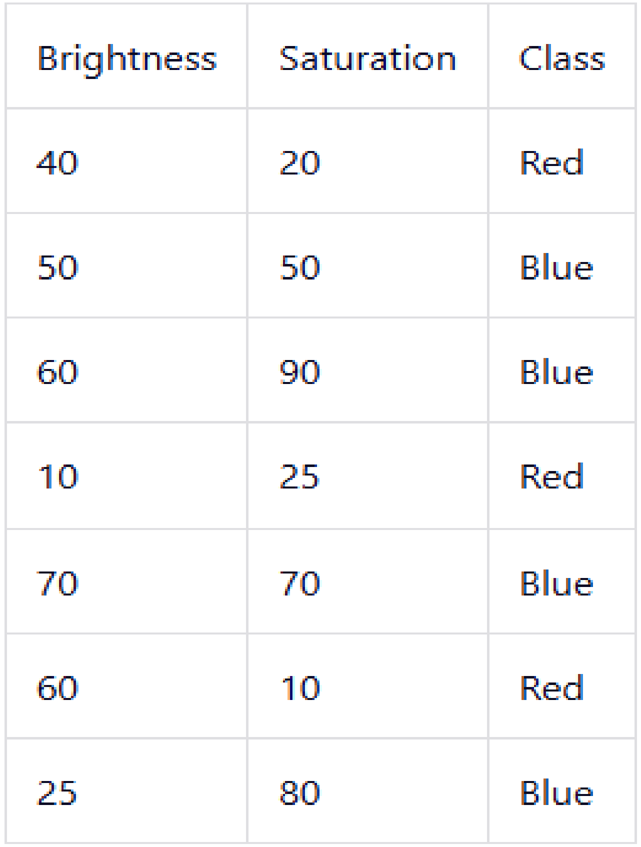
$$\text{Manhattan Distance} = |X1 - X2| + |Y1 - Y2|$$

- The generalized formula for an n-dimensional space is given as:

$$D_m = \sum_{i=1}^{n} |p_i - q_i|$$

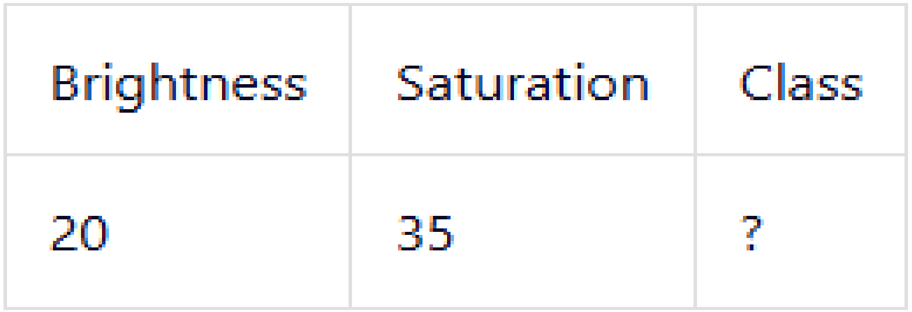
Example - 1

- Perform KNN classification algorithm on the following dataset and predict the class for the new data point when k=5



Brightness	Saturation	Class
40	20	Red
50	50	Blue
60	90	Blue
10	25	Red
70	70	Blue
60	10	Red
25	80	Blue

The New Data Point is:



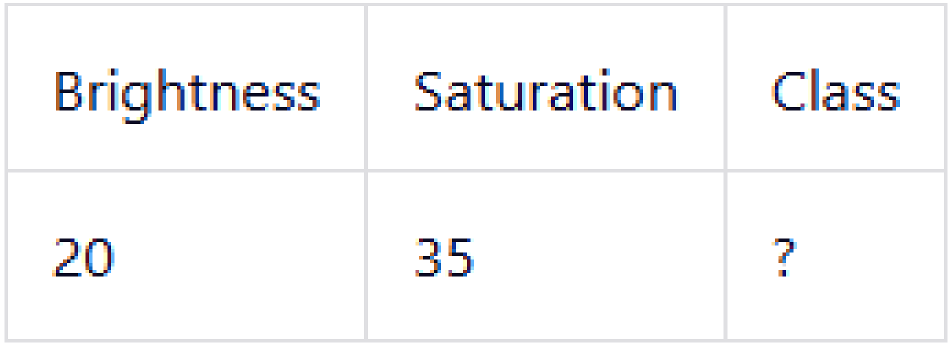
Brightness	Saturation	Class
20	35	?

Solution

- **Step 1** – Select the number of K
- Here, K = 5

- **Step 2** – Calculate the Euclidean Distance

- Here's the new data entry is given as :



Brightness	Saturation	Class
20	35	?

We have a new entry but it doesn't have a class yet. To know its class, we have to calculate the distance from the new entry to other entries in the data set using the Euclidean distance formula.

Here's the formula: $\sqrt{(X_2-X_1)^2+(Y_2-Y_1)^2}$

Where:

- X_2 = New entry's brightness (20).
- X_1= Existing entry's brightness.
- Y_2 = New entry's saturation (35).
- Y_1 = Existing entry's saturation.

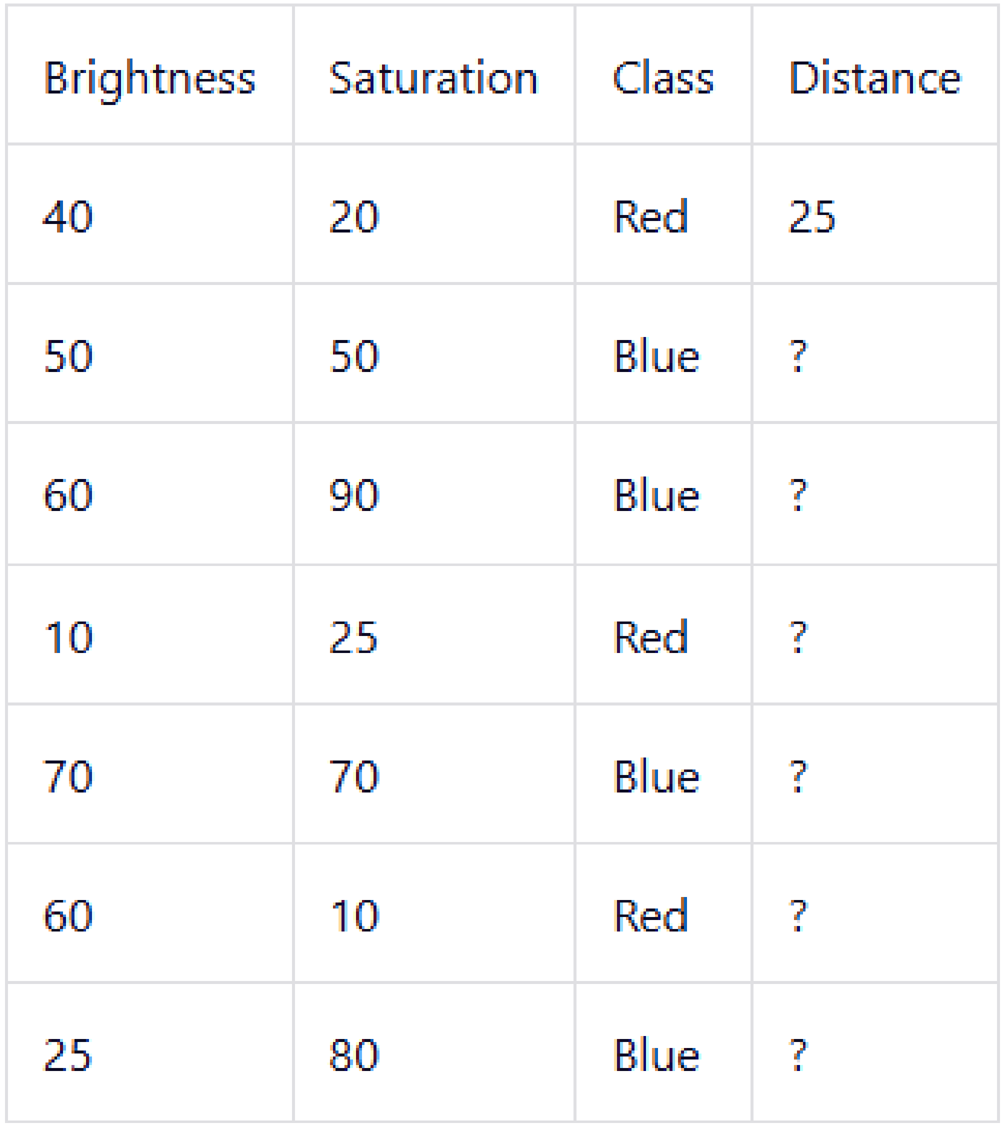
Distance #1

For the first row, d1:

Brightness	Saturation	Class
40	20	Red

$$d1 = \sqrt{(20 - 40)^2 + (35 - 20)^2}$$
$$= \sqrt{400 + 225}$$
$$= \sqrt{625}$$
$$= 25$$

- We now know the distance from the new data entry to the first entry in the table. Let's update the table.



Brightness	Saturation	Class	Distance
40	20	Red	25
50	50	Blue	?
60	90	Blue	?
10	25	Red	?
70	70	Blue	?
60	10	Red	?
25	80	Blue	?

Distance #2

For the second row, d2:

Brightness	Saturation	Class	Distance
50	50	Blue	?

$d2 = \sqrt{(20 - 50)^2 + (35 - 50)^2}$

$= \sqrt{900 + 225}$

$= \sqrt{1125}$

$= 33.54$

Distance #3

For the third row, d3:

Brightness	Saturation	Class	Distance
60	90	Blue	?

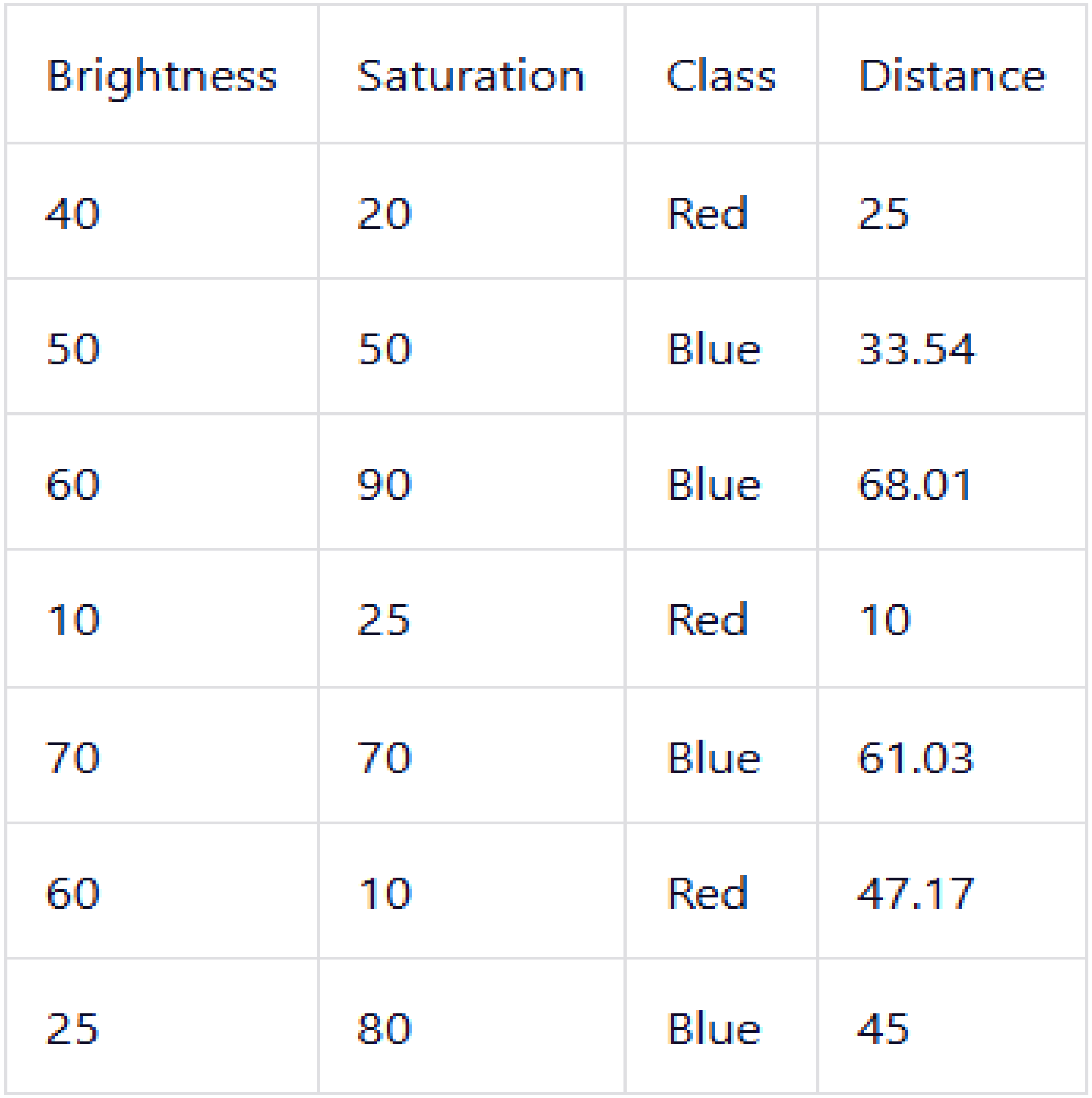
d2 = $\sqrt{(20 - 60)^2 + (35 - 90)^2}$

= $\sqrt{1600 + 3025}$

= $\sqrt{4625}$

= 68.01

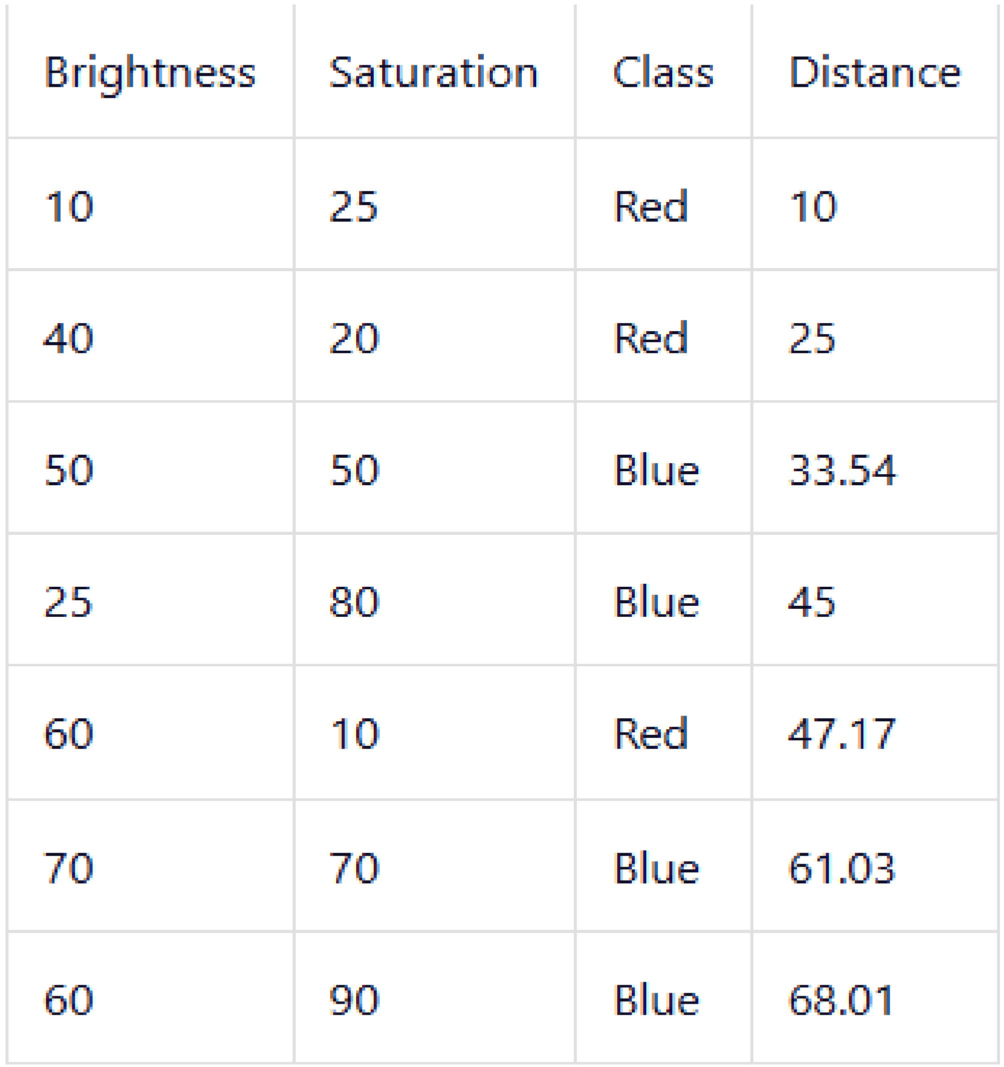
- Here's what the table will look like after all the distances have been calculated:



Brightness	Saturation	Class	Distance
40	20	Red	25
50	50	Blue	33.54
60	90	Blue	68.01
10	25	Red	10
70	70	Blue	61.03
60	10	Red	47.17
25	80	Blue	45

- **Step 3** – Rank the data points based on distance value

- Let's rearrange the distances in ascending order:



Brightness	Saturation	Class	Distance
10	25	Red	10
40	20	Red	25
50	50	Blue	33.54
25	80	Blue	45
60	10	Red	47.17
70	70	Blue	61.03
60	90	Blue	68.01

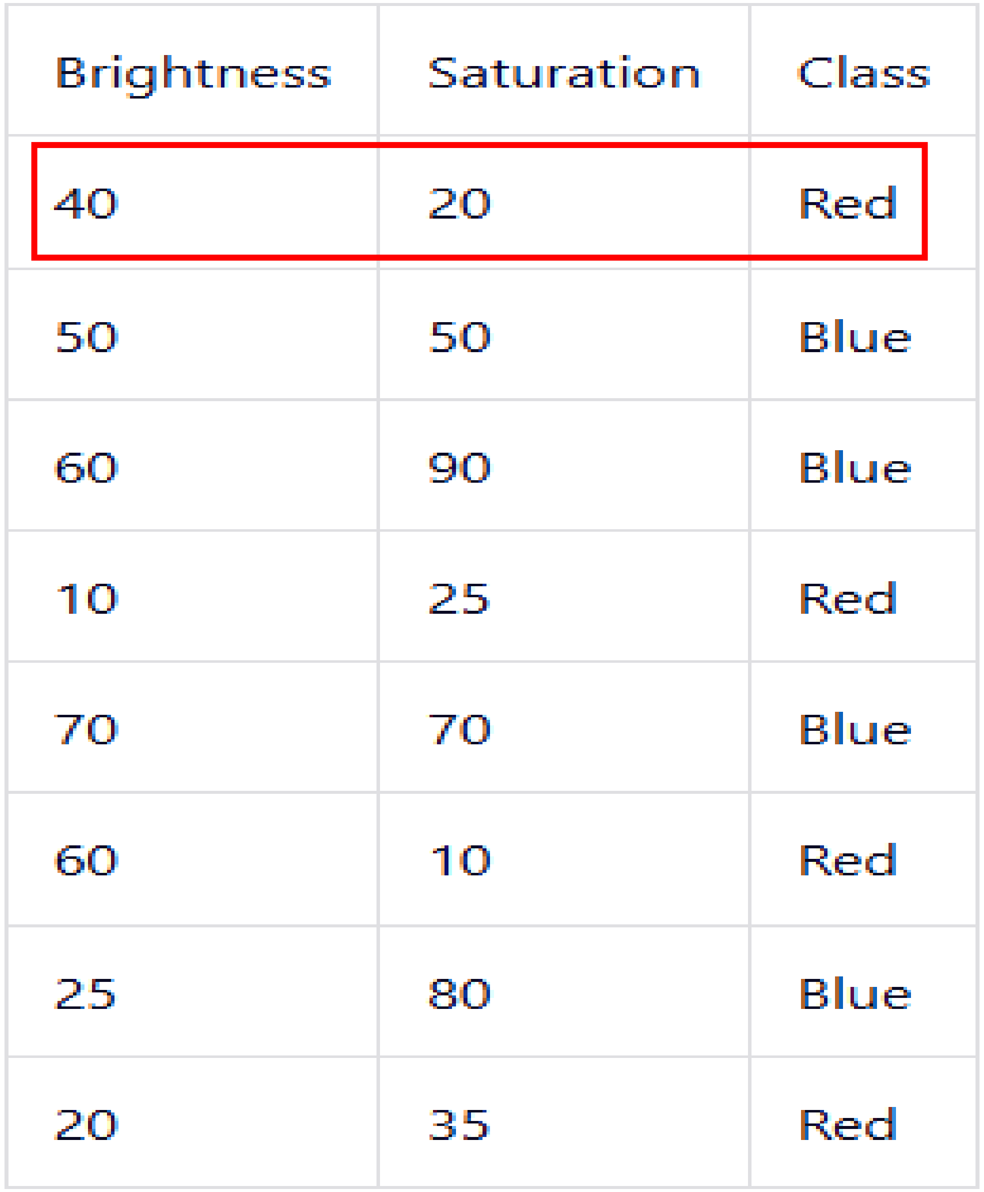
- **Step 4 –** Selecting the data points based on K Value.

- Here, K = 5,
- So, we'll only consider the first five rows.

Brightness	Saturation	Class	Distance
10	25	Red	10
40	20	Red	25
50	50	Blue	33.54
25	80	Blue	45
60	10	Red	47.17

As you can see, the majority class within the 5 nearest neighbors to the new entry is **Red**. Therefore, we'll classify the new entry as **Red**.

- Thus, the result will be:-



Brightness	Saturation	Class
40	20	Red
50	50	Blue
60	90	Blue
10	25	Red
70	70	Blue
60	10	Red
25	80	Blue
20	35	Red

Example – 2

- Apply KNN algorithm and predict the class for X(P1 = 3 and P2 = 7)for K=3.

P1	**P2**	**Class**
7	7	False
7	4	False
3	4	True
1	4	True

Solution

- **Step 1** = Here, K = 3

- **Step 2** – Calculate the Euclidean Distance for each data points from the new data point

P1	**P2**	**Class**	**Distance**
7	7	False	4
7	4	False	5
3	4	True	3
1	4	True	3.6

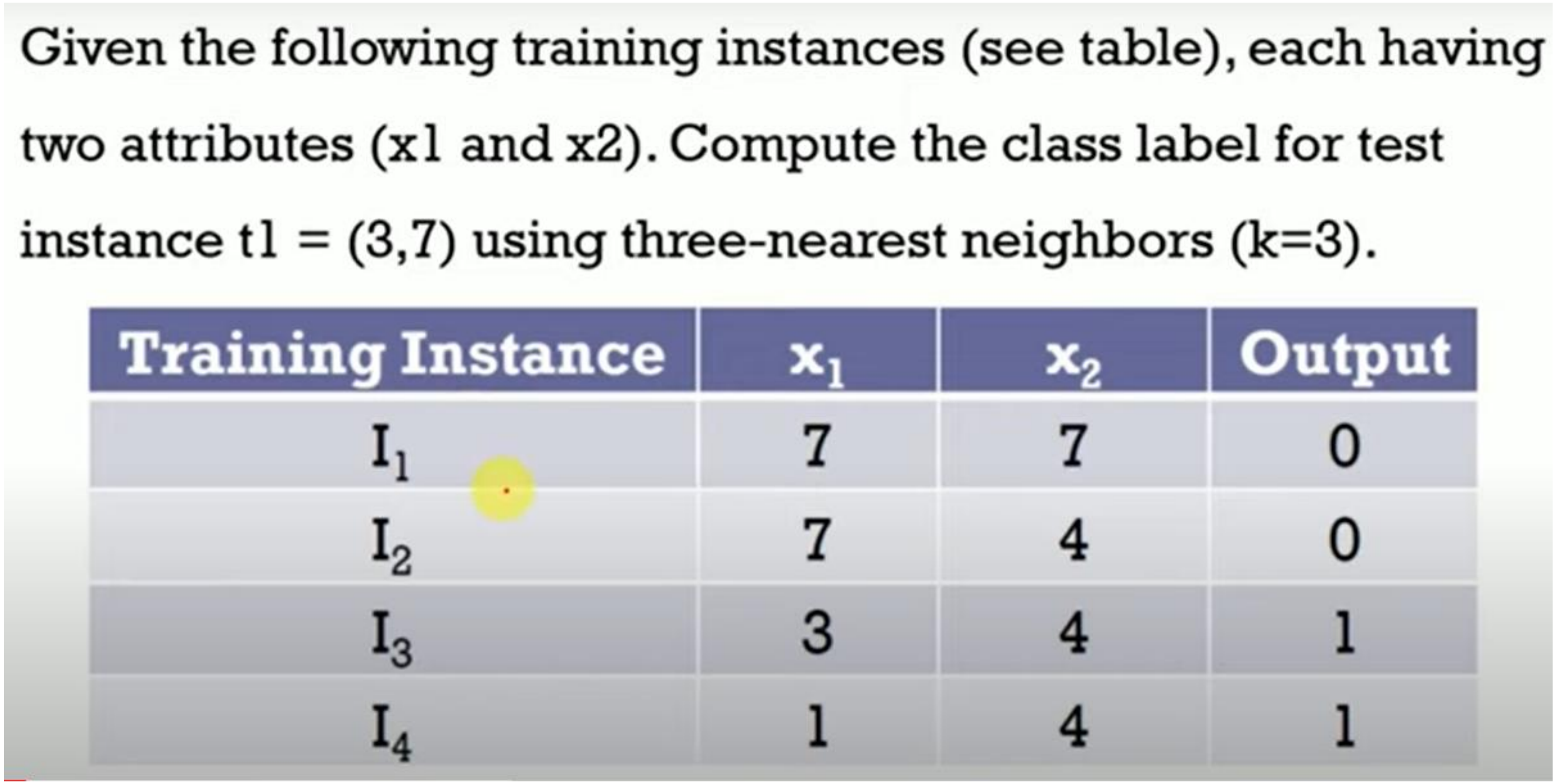
- **Step -3** – Rank based on the minimum distance and K value.

P1	P2	Class	Distance	
7	7	False	4	3
7	4	False	5	
3	4	True	3	1
1	4	True	3.6	2

- Here, since **K = 3**, we will select the first 3 ranks to decide the class for the new data point.

- We get 2 TRUE values and 1 FALSE Value; Thus, we can decide that the new data point belongs to the class **TRUE**.

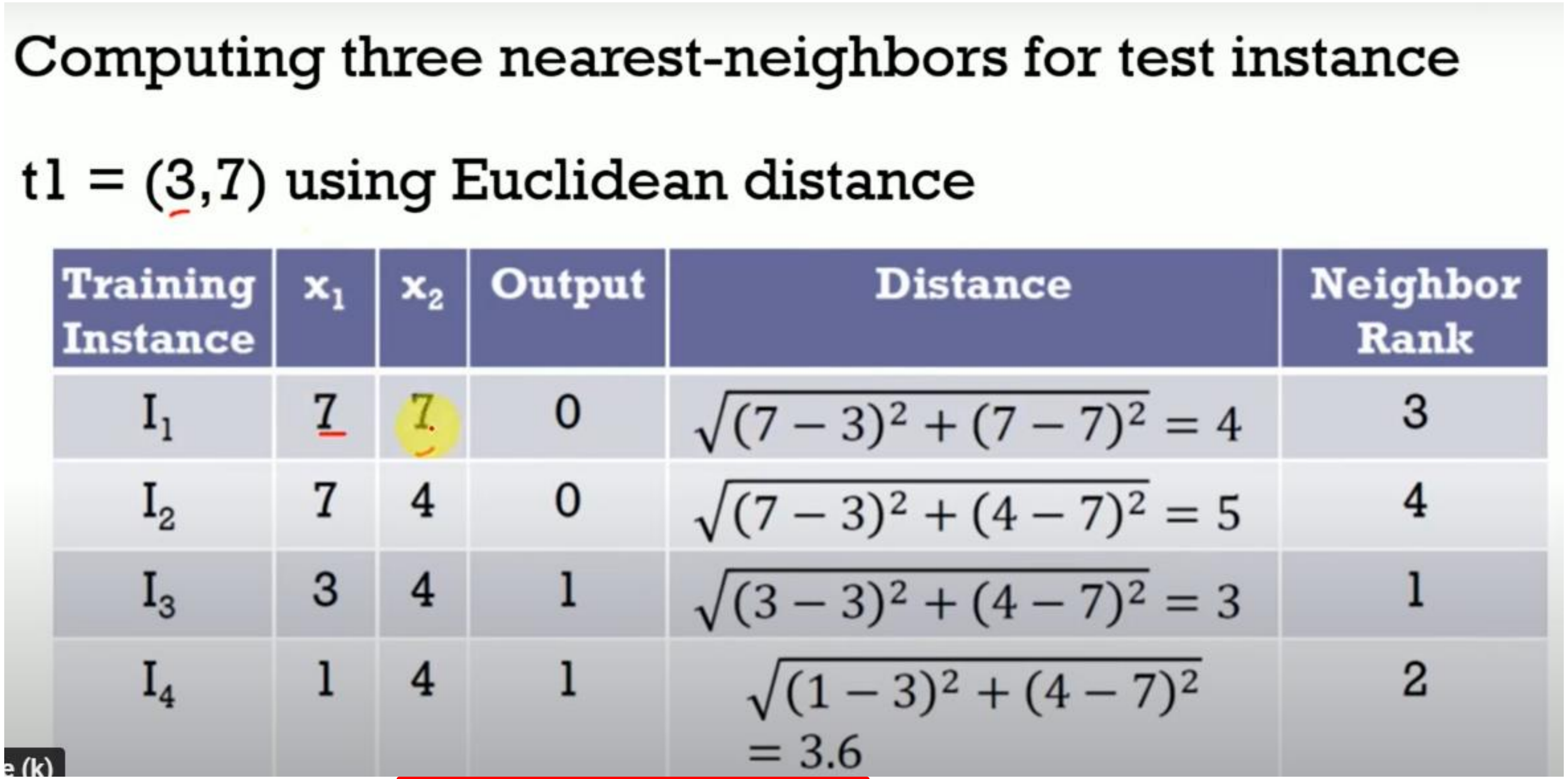
Example - 3



Given the following training instances (see table), each having two attributes (x1 and x2). Compute the class label for test instance t1 = (3,7) using three-nearest neighbors (k=3).

Training Instance	x_1	x_2	Output
I_1	7	7	0
I_2	7	4	0
I_3	3	4	1
I_4	1	4	1

Solution



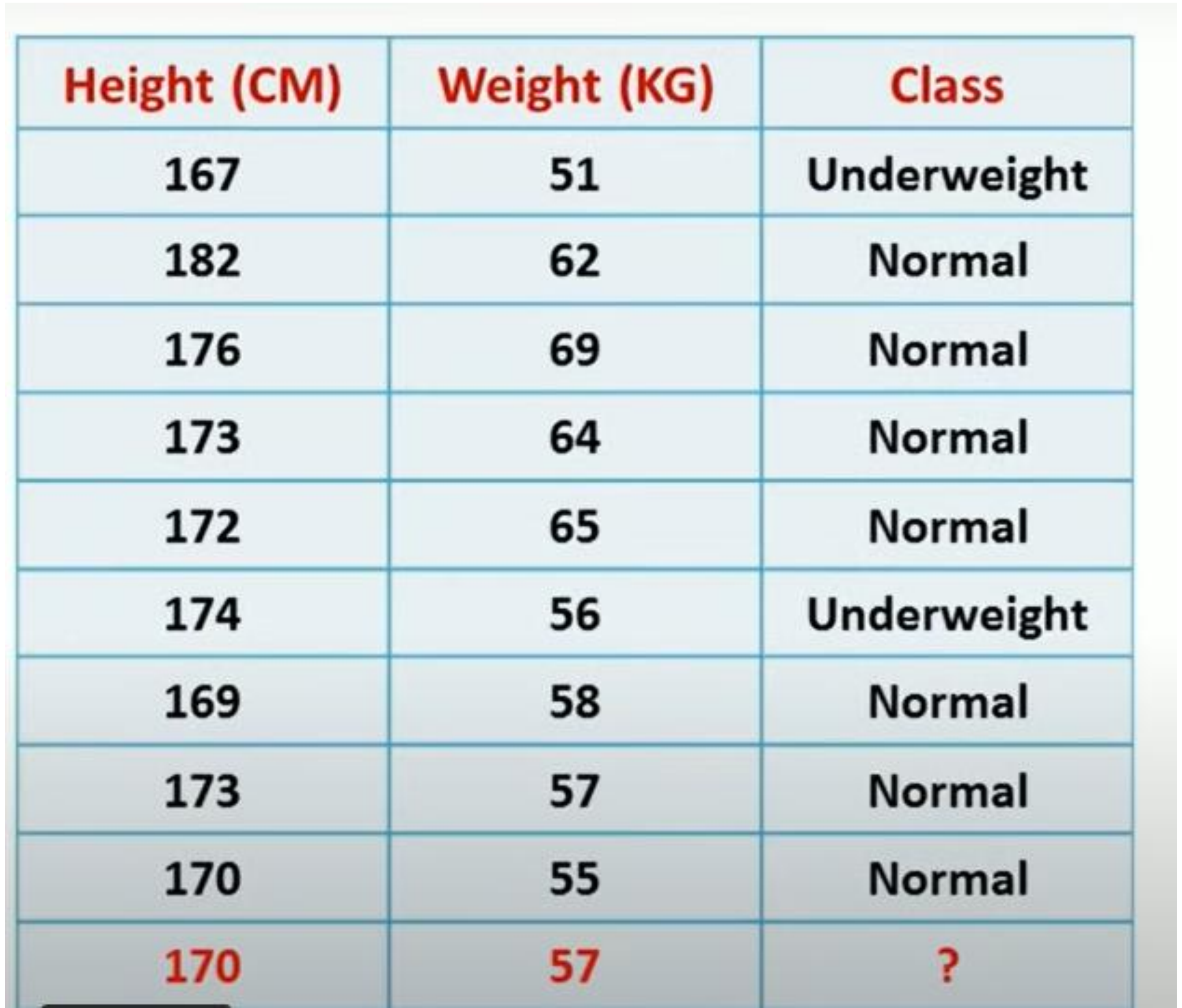
Computing three nearest-neighbors for test instance

t1 = (3,7) using Euclidean distance

Training Instance	x_1	x_2	Output	Distance	Neighbor Rank
I_1	7	7	0	$\sqrt{(7-3)^2+(7-7)^2}=4$	3
I_2	7	4	0	$\sqrt{(7-3)^2+(4-7)^2}=5$	4
I_3	3	4	1	$\sqrt{(3-3)^2+(4-7)^2}=3$	1
I_4	1	4	1	$\sqrt{(1-3)^2+(4-7)^2}$ $=3.6$	2

$\boxed{t1 = (3,7) \rightarrow 1}$

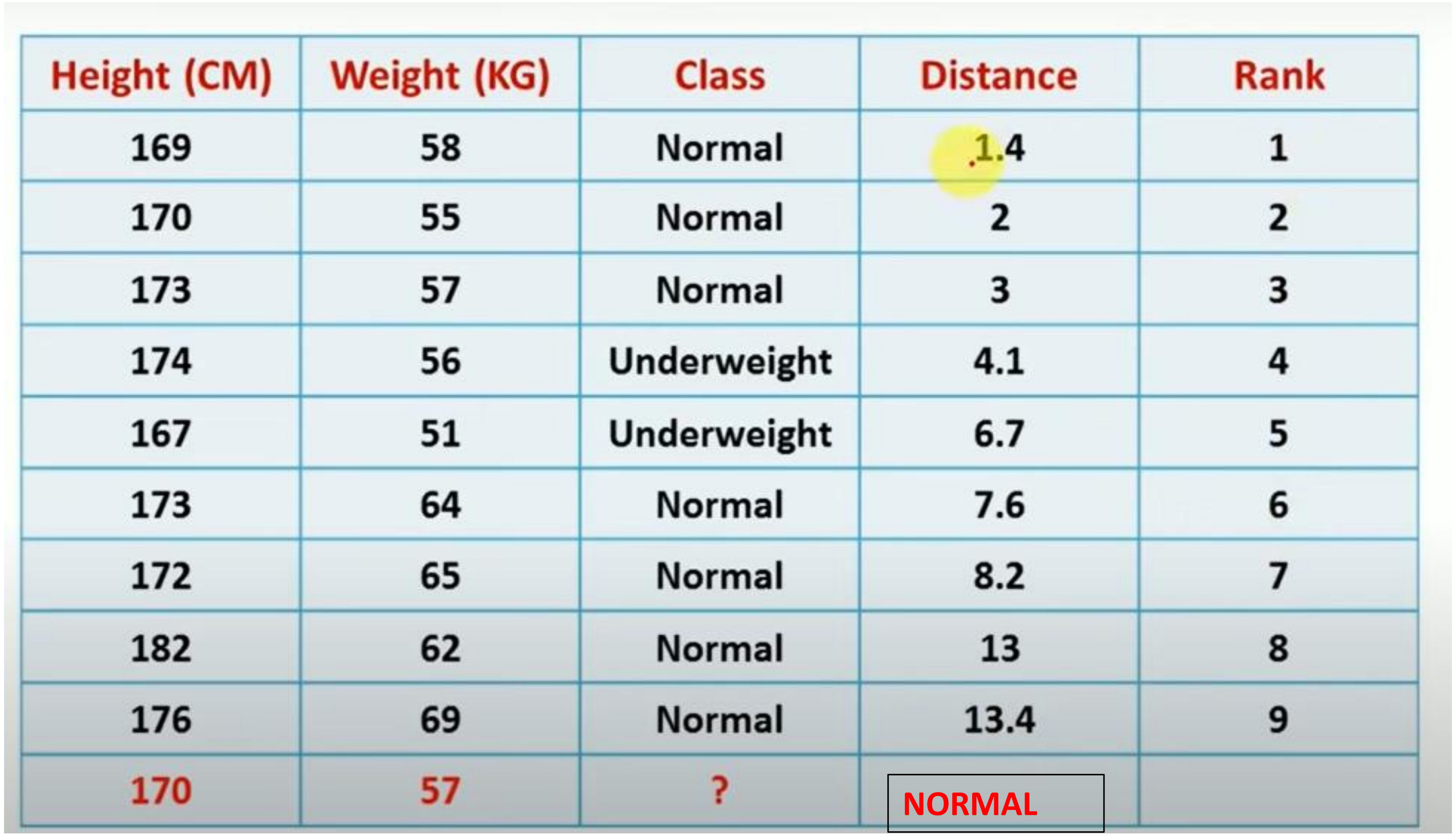
Example - 4



Height (CM)	Weight (KG)	Class
167	51	Underweight
182	62	Normal
176	69	Normal
173	64	Normal
172	65	Normal
174	56	Underweight
169	58	Normal
173	57	Normal
170	55	Normal
170	57	?

Apply KNN for K = 5

Solution



Height (CM)	Weight (KG)	Class	Distance	Rank
169	58	Normal	1.4	1
170	55	Normal	2	2
173	57	Normal	3	3
174	56	Underweight	4.1	4
167	51	Underweight	6.7	5
173	64	Normal	7.6	6
172	65	Normal	8.2	7
182	62	Normal	13	8
176	69	Normal	13.4	9
170	57	?	NORMAL	

Example-5

- Using KNN, predict the class label of record 15 and k is set to 3. The first 10 are training data and the other 10 are testing data.

Record ID	**Age**	**Spectacle Description**	**Astigmatic**	**Tear Production Rate**	**Class Label Lenses**
1	Young	Myope	No	Reduced	No-Contact
2	Young	Myope	No	Normal	Soft-Contact
3	Young	Myope	Yes	Reduced	No-Contact
4	Young	Myope	Yes	Normal	Hard-Contact
5	Young	Hypermetropic	No	Reduced	No-Contact
6	Young	Hypermetropic	No	Normal	Soft-Contact
7	Young	Hypermetropic	Yes	Reduced	No-Contact
8	Young	Hypermetropic	Yes	Normal	Hard-Contact
9	Pre-Presbyopic	Myope	No	Reduced	No-Contact
10	Pre-Presbyopic	Myope	No	Normal	Soft-Contact
15	**Pre-Presbyopic**	**Hypermetropic**	**Yes**	**Reduced**	**?**

Solution

- Since the data given is categorical Variable and non-numerical data, We need to assign values to the given variable as follows: -

- **AGE →**

Young	**→ 0**
Pre-presbyopic	**→ 1**

- Spectacle Description →

Myope	→ 0
Hypermetropic	→ 1

- Astigmatic →

No	→ 0
Yes	→ 1

- Tear Production Rate →

Reduced	→ 0
Normal	→ 1

- Now, let’s formulate the table dataset using these categorical variables and their corresponding values as follows:-

Record ID	Age	Spectacle Description	Astigmatic	Tear Prediction Rate	Class Label Lenses
1	0	0	0	0	No-Contact
2	0	0	0	1	Soft-Contact
3	0	0	1	0	No-Contact
4	0	0	1	1	Hard-Contact
5	0	1	0	0	No-Contact
6	0	1	0	1	Soft-Contact
7	0	1	1	0	No-Contact
8	0	1	1	1	Hard-Contact
9	1	0	0	0	No-Contact
10	1	0	0	1	Soft-Contact
15	**1**	**1**	**1**	**0**	**?**

- Next. Let’s calculate the distance between these datapoints using Euclidian's formula:

$$D1 = \sqrt{(1-0)^2 + (1-0)^2 + (1-0)^2 + (0-0)^2}$$

$$= \sqrt{3} = 1.732$$

$$D2 = \sqrt{(1-0)^2 + (1-0)^2 + (1-0)^2 + (0-1)^2}$$

$$= \sqrt{4} = 2$$

$$D3 = \sqrt{(1-0)^2 + (1-0)^2 + (1-1)^2 + (0-0)^2}$$

$$= \sqrt{2} = 1.414$$

Record ID	Age	Spectacle Description	Astigmatic	Tear Prediction Rate	Class Label Lenses	Distance
1	0	0	0	0	Non-Contact	1.732
2	0	0	0	1	Soft-Contact	2
3	0	0	1	0	Non-Contact	1.414
4	0	0	1	1	Hard-Contact	1.732
5	0	1	0	0	Non-Contact	1.732
6	0	1	0	1	Soft-Contact	1.732
7	0	1	1	0	Non-Contact	1
8	0	1	1	1	Hard-Contact	1.414
9	1	0	0	0	Non-Contact	1.732
10	1	0	0	1	Soft-Contact	1.732
15	**1**	**1**	**1**	**0**	**?**	

- For K=3, by ranking the distance, we get

Record ID	Age	Spectacle Description	Astigmatic	Tear Prediction Rate	Class Label Lenses	Distance	
1	0	0	0	0	No-Contact	1.732	
2	0	0	0	1	Soft-Contact	2	
3	0	0	1	0	No-Contact	1.414	**2**
4	0	0	1	1	Hard-Contact	1.732	
5	0	1	0	0	No-Contact	1.732	
6	0	1	0	1	Soft-Contact	1.732	
7	0	1	1	0	No-Contact	1	**1**
8	0	1	1	1	Hard-Contact	1.414	**3**
9	1	0	0	0	No-Contact	1.732	
10	1	0	0	1	Soft-Contact	1.732	
15	**1**	**1**	**1**	**0**	**No-Contact**		

- Thus, we can conclude that R15 also belongs to the class → **No-Contact**