Definition of machine learning

- **Arthur Samuel**, an early American leader in the field of computer gaming and artificial intelligence, **coined the term "Machine Learning" in 1959 while at IBM**.

- He defined machine learning as "**the field of study that gives computers the ability to learn without being explicitly programmed.**"

- However, there is **no universally accepted definition for machine learning**. Different authors define the term differently. We give below two more definitions.

- 1. Machine learning is programming computers to optimize a performance criterion using example data or past experience. We have a model defined up to some parameters, and learning is the execution of a computer program to optimize the parameters of the model using the training data or past experience. The model may be predictive to make predictions in the future, or descriptive to gain knowledge from data, or both.

- 2. The field of study known as machine learning is concerned with the question of how to construct computer programs that automatically improve with experience.

Definition of learning

- **A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks T, as measured by P, improves with experience E.**

- Examples
- **i) Handwriting recognition learning problem**
- Task T: Recognising and classifying handwritten words within images
- Performance P: Percent of words correctly classified
- Training experience E: A dataset of handwritten words with given classifications

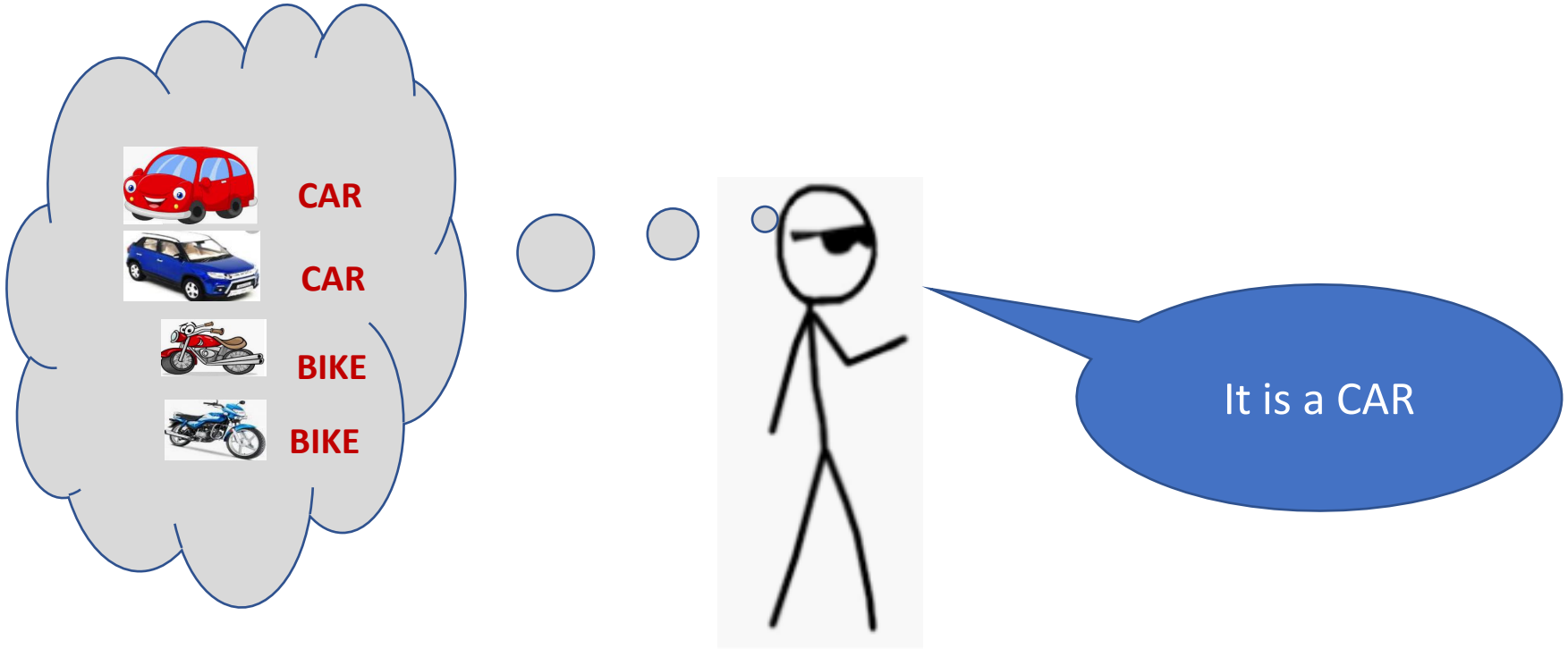
- **ii) A robot driving learning problem**
- Task T: Driving on highways using vision sensors
- Performance measure P: Average distance traveled before an error
- Training experience E: A sequence of images and steering commands recorded while observing a human driver

- A **computer program which learns from experience is called a machine learning program or simply a learning program.** Such a program is sometimes also referred to as a **learner.**

Human can learn from past experience
and make decision of its own

What is this object?

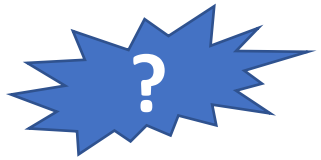
What is this object?



Let us ask the same question to him

What is this object?

Let us ask the same question to him



What is this object?

[But, he is a human being. He can observe and learn]

Let us make him learn

show him

Let us make him learn

show him

CAR

CAR

BIKE

BIKE

Let us ask the same question now

What is this object?

CAR

CAR

BIKE

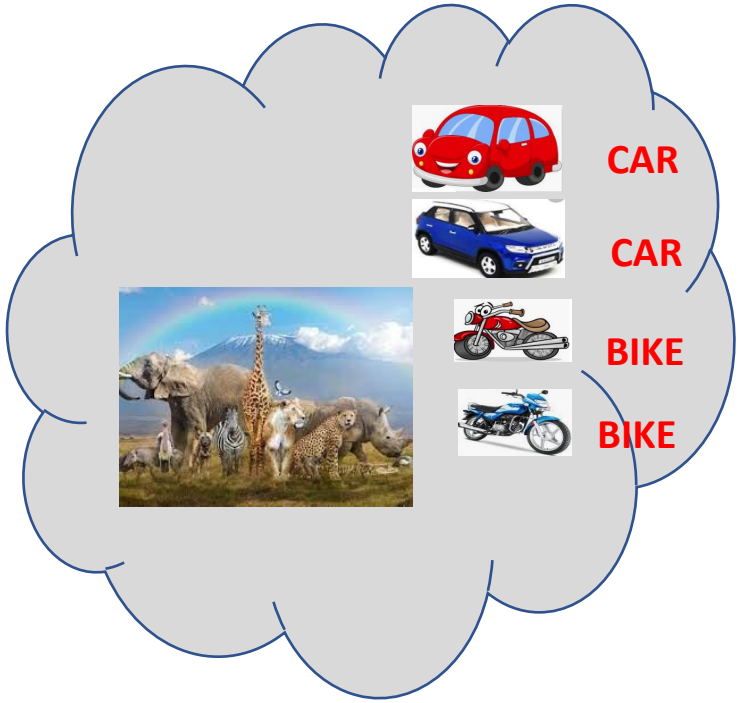
BIKE

Past experience

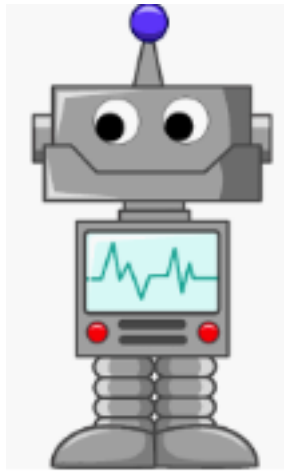
Let us ask the same question now

CAR

What is this object?



What about a Machine ?



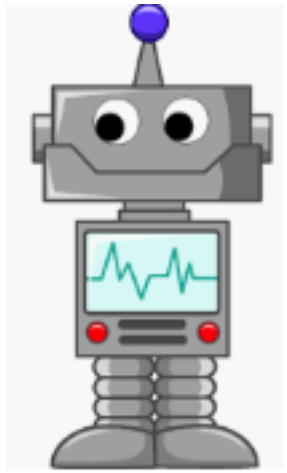
Machines follow instructions

[It can not take decision of its own]

What about a Machine ?

We can ask a machine

- To perform an arithmetic operations such as
  - Addition
  - Multiplication
  - Division



Machines follow instructions

What about a Machine ?

- Comparison
- Print
- Plotting a chart

Machines follow instructions

What is Machine Learning?

[We want a machine to act like a human]

What is Machine Learning?

[to identify this object.]

What is Machine Learning?

Price in 2025?

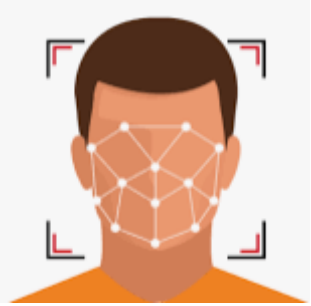
[predict the price in future]

What is Machine Learning?

I ~~made~~ met him yesterday

[Natural Language understand, and correct grammar]

What is Machine Learning?



recognize face

[Recognize Faces]

What is Machine Learning?

[What do we do?

Just like, what we did to human,

we need to provide experience to the machine.

]

What is Machine Learning?

+

Dataset

[
This what we called as Data or Training dataset

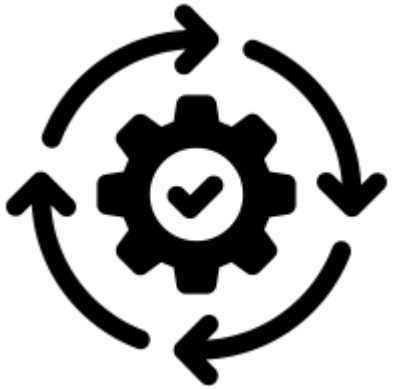
So, we first need to provide training dataset to the machine
]

What is Machine Learning?

+

Dataset

+



[Then, devise algorithms and execute programs on the data

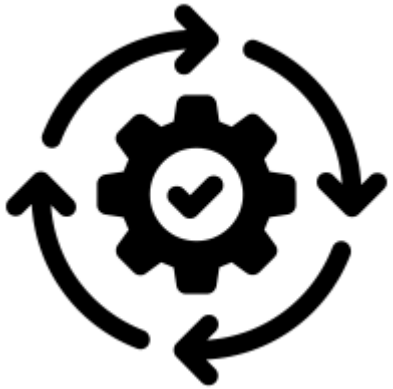
With respect to the underlying target tasks]

What is Machine Learning?

+

Dataset

+



+

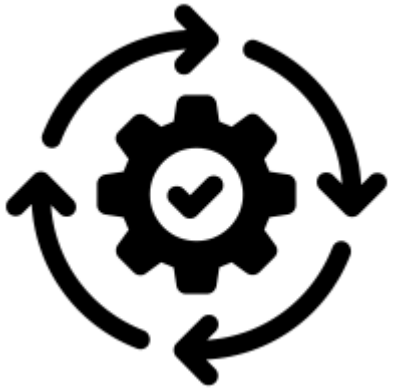
[Then, using the programs, Identify required rules]

What is Machine Learning?

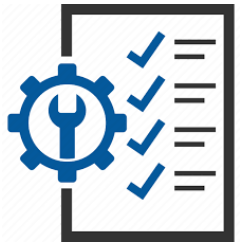
+

Dataset

+



+



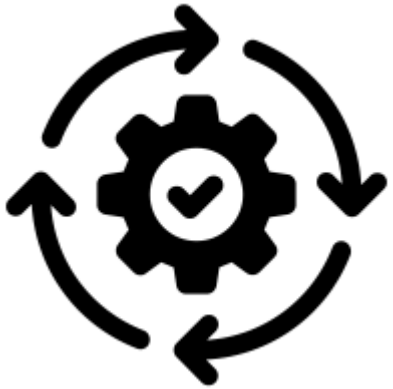
[extract required patterns]

What is Machine Learning?

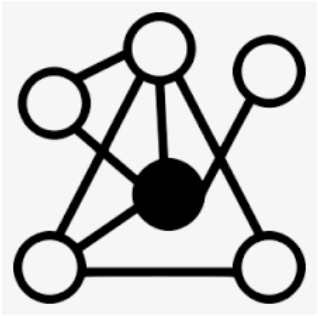
+

Dataset

+

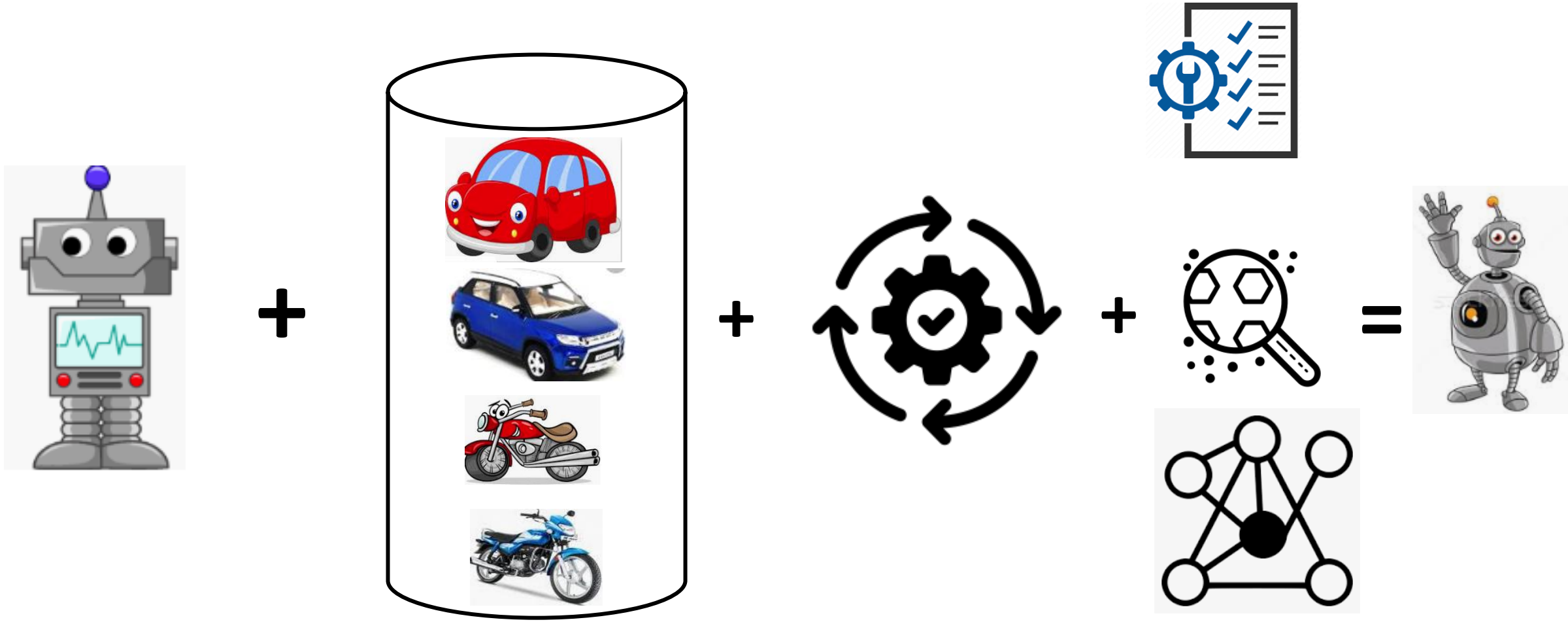


+



[Identify relations]

What is Machine Learning?



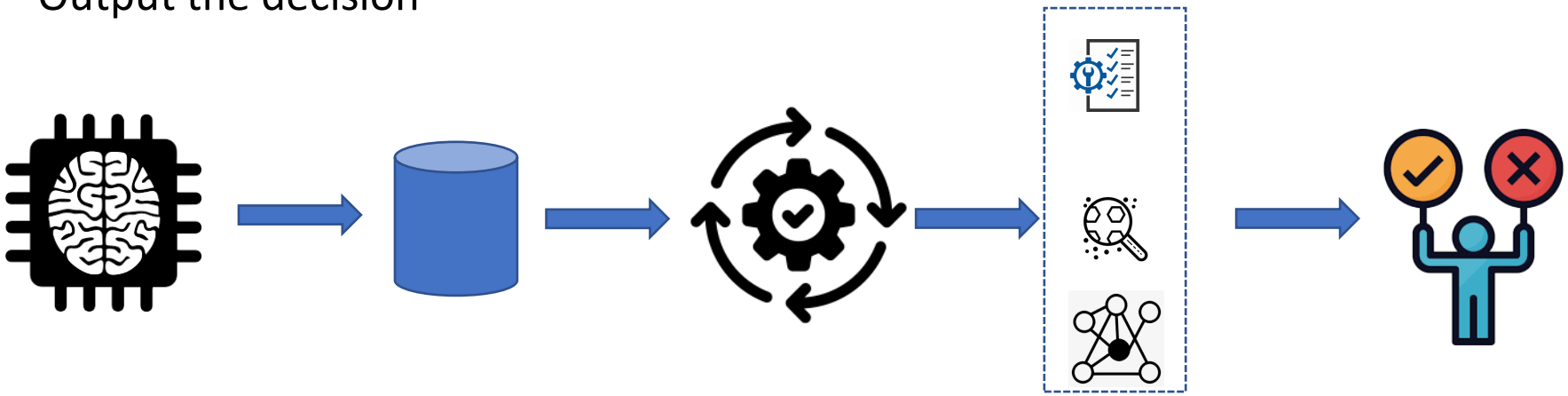
Dataset

[So that machine can derive inferences from the data]

In summary, what is machine learning?

Given a machine learning problem

- Identify and create the appropriate dataset
- Perform computation to learn
  - Required rules, pattern and relations
- Output the decision



Applications of machine learning

- 1. In **retail business**, machine learning is used to **study consumer behaviour.**
- 2. In **finance**, banks analyze their past data to build models to use in **credit applications, fraud detection, and the stock market.**
- 3. In **manufacturing**, learning models are used for **optimization, control, and troubleshooting.**
- 4. In **medicine**, learning programs are used for **medical diagnosis.**
- 5. In **telecommunications**, call patterns are analyzed for **network optimization and maximizing the quality of service.**

Applications of machine learning

- 6. In **science**, large amounts of **data in physics, astronomy, and biology can only be analyzed fast enough by computers**. The **World Wide Web is huge; it is constantly growing and searching for relevant information cannot be done manually.**

- 7. In **artificial intelligence**, it is used to teach a system to **learn and adapt to changes so that the system designer need not foresee and provide solutions for all possible situations.**

- 8. It is used to **find solutions to many problems in vision, speech recognition, and robotics.**

- 9. Machine learning methods are applied in the **design of computer-controlled vehicles to steer correctly when driving on a variety of roads.**

- 10. Machine learning methods have been used to **develop programmes for playing games such as chess, backgammon and Go.**

Types of machine learning :- (i) Supervised Learning

- Supervised learning is the **machine learning task of learning a function that maps an input to an output based on example input-output pairs.**
- In supervised learning, **each example in the training set is a pair consisting of an input object (typically a vector) and an output value.**
- A supervised learning algorithm **analyzes the training data and produces a function, which can be used for mapping new examples.**
- In the **optimal case, the function will correctly determine the class labels for unseen instances. Both classification and regression problems are supervised learning problems.**
- A wide range of supervised learning algorithms are available, each with its strengths and weaknesses. **There is no single learning algorithm that works best on all supervised learning problems.**

(i) Supervised Learning

- **Remarks:**

- A “supervised learning” is so called because the **process of an algorithm learning from the training dataset can be thought of as a teacher supervising the learning process.**

- We know the correct answers (that is, the correct outputs), the algorithm iteratively makes predictions on the training data and is corrected by the teacher.

- **Learning stops when the algorithm achieves an acceptable level of performance.**

What is Supervised Learning?

Samples

CAR

CAR

BIKE

BIKE

Labels

= Training Dataset

[In supervised learning, we need some thing called a Labelled Training Dataset]

What is Supervised Learning?




CAR

CAR

BIKE

BIKE

Samples

Labels

Training Dataset

$f($  $,\quad\quad)=$

[Given a labelled dataset, the task is to devise a function which takes the dataset, and a new sample, and produces an output value.]

What is Supervised Learning?




CAR

CAR

BIKE

BIKE

Samples

Labels

= Training Dataset

$f($  ,  $)=$

[Given a labelled dataset, the task is to devise a function which takes the dataset, and a new sample, and produces an output value.]

What is Supervised Learning?


CAR


CAR

\+


BIKE


BIKE

Samples Labels

= **Training Dataset**

$f($  ,  $)$ = CAR

[Given a labelled dataset, the task is to devise a function which takes the dataset, and a new sample, and produces an output value.]

What is Supervised Learning?




CAR

CAR

BIKE

BIKE

Samples

Labels

Training Dataset

Classification

$f($  ,  $)=$ CAR

[If the possible output values of the function are predefined and discrete/categorical, it is called Classification

What is Supervised Learning?




CAR

CAR

BIKE

BIKE

Samples

Labels

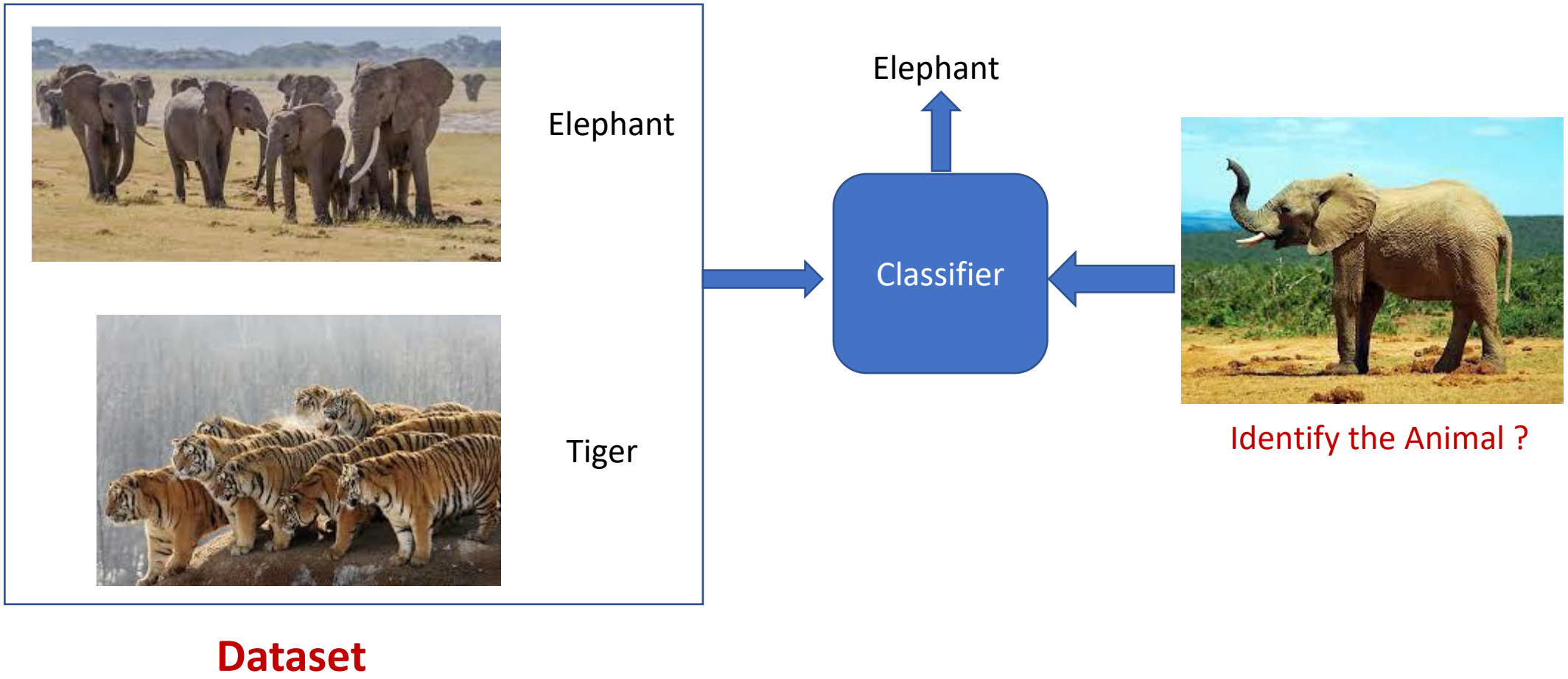
Training Dataset

Classification

$f($  ,  $)=$ CAR

[Predefined classes means, it will produce output only from the labels defined in the dataset. For example, even if we input a bus, it will produce either CAR or BIKE]

Classifier


Elephant
Tiger
Dataset
Classifier
Elephant
Identify the Animal ?

Regression

Dataset

Regression

$f($  ,  $) = 20500.50$

[If the possible output values of the function are continuous real values, then it is called Regression

[
The classification and Regression problems are supervised, because the decision depends on the characteristics of the ground truth labels or values present in the dataset, which we define as experience
]

(ii) Unsupervised Learning

- Unsupervised learning is a type of machine learning algorithm **used to draw inferences from datasets consisting of input data without labeled responses.**
- In unsupervised learning algorithms, **a classification or categorization is not included in the observations.**
- There are **no output values and so there is no estimation of functions.**
- Since the **examples given to the learner are unlabeled, the accuracy of the structure that is output by the algorithm cannot be evaluated.**
- The most common unsupervised learning method is **cluster analysis, which is used for exploratory data analysis to find hidden patterns or grouping in data.**

What is Unsupervised Learning


CAR


CAR


BIKE


BIKE

Dataset

[In the unsupervised learning, we do not need to know the labels or Ground truth values]

What is Unsupervised Learning

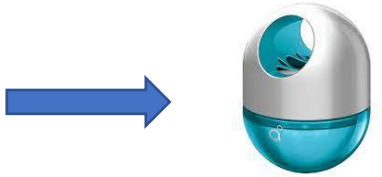
Dataset

Clustering

[The task is to identify the patterns like group the similar objects together]

What is Unsupervised Learning

Dataset



Association Rules Mining

[Association rules like]

More Example Unsupervised Learning

Dataset

More Example Unsupervised Learning

Dataset

More Example Unsupervised Learning

(iii) Reinforcement learning

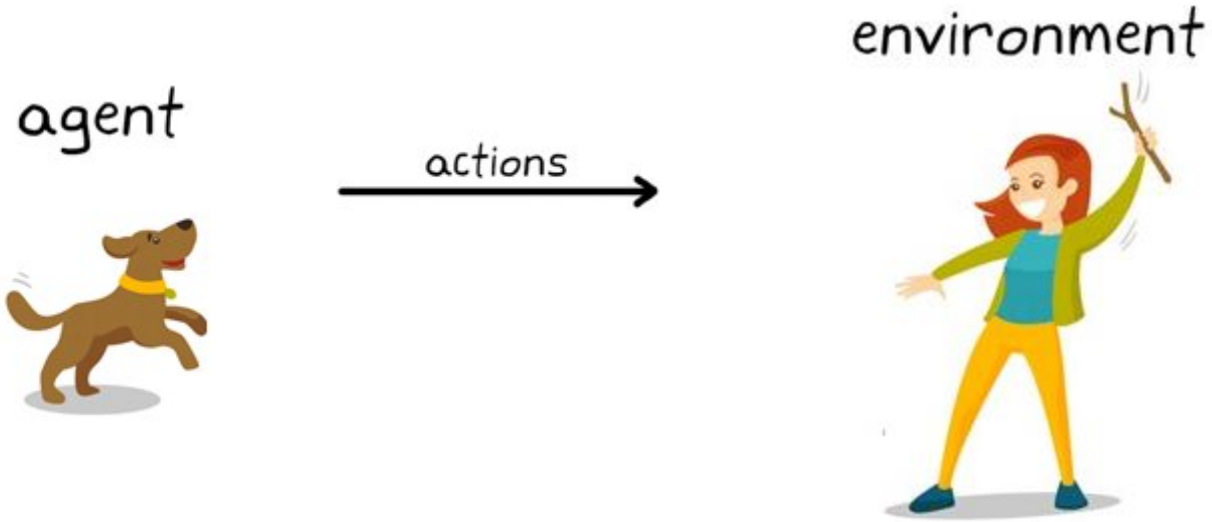
- Reinforcement learning is the **problem of getting an agent to act in the world so as to maximize its rewards.**
- A learner (the program) is not told what actions to take as in most forms of machine learning, but instead must **discover which actions yield the most reward by trying them.**
- In the most interesting and challenging cases, **actions may affect not only the immediate reward but also the next situations and, through that, all subsequent rewards.**
- For example, **consider teaching a dog a new trick: we cannot tell it what to do, but we can reward/punish it if it does the right/wrong thing.**
- **It has to find out what it did that made it get the reward/punishment.**
- We can use a similar method to train computers to do many tasks, such as playing backgammon or chess, scheduling jobs, and controlling robot limbs.

What is Reinforcement Learning

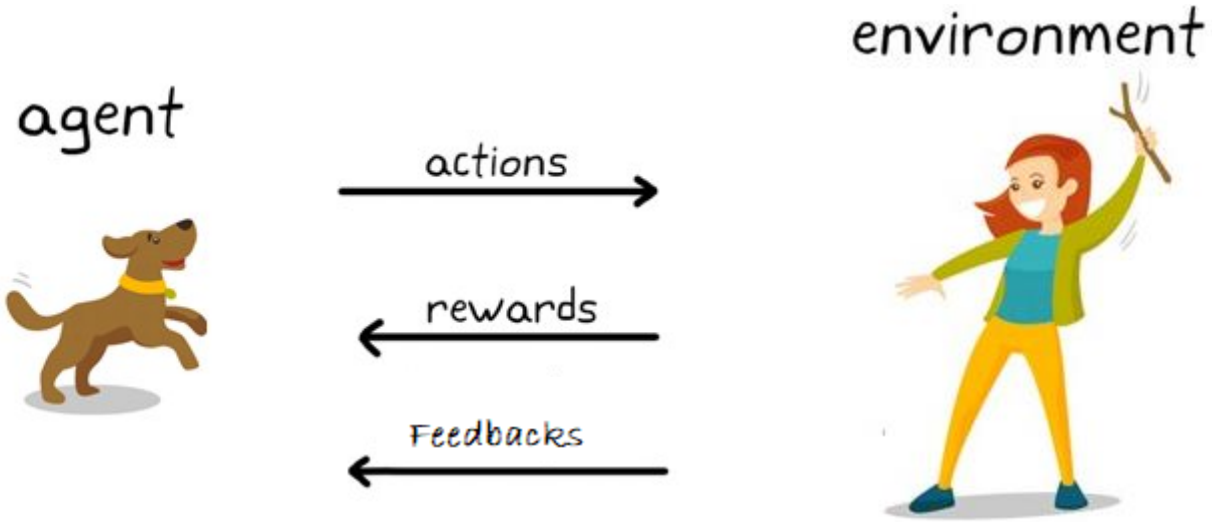
[It is also known as learning from trials and errors]

What is Reinforcement Learning

What is Reinforcement Learning



What is Reinforcement Learning



Another Example

Agent

Task

Environment

Reinforcement Learning

Punishment

Reinforcement Learning

Reward

Reinforcement Learning

Reward

Baby Learn from the Trials and Errors

Reinforcement Learning

Regression

- **Definition**
- In machine learning, a regression problem is the problem of **predicting the value of a numeric variable based on observed values of the variable.** The **value of the output variable may be a number, such as an integer or a floating point value.**
- These are often quantities, such as amounts and sizes. **The input variables may be discrete or real-valued.**
- **Example**
- Consider the data on car prices given in Table 1.2.

Price (US$)	Age (years)	Distance (KM)	Weight (pounds)
13500	23	46986	1165
13750	23	72937	1165
13950	24	41711	1165
14950	26	48000	1165
13750	30	38500	1170
12950	32	61000	1170
16900	27	94612	1245
18600	30	75889	1245
21500	27	19700	1185
12950	23	71138	1105

Table 1.2: Prices of used cars: example data for regression

- Suppose we are required to estimate the price of a car aged 25 years with distance 53240 KM and weight 1200 pounds. It is an ex. of a regression problem beause we have to predict the value of the numeric variable “Price”.

Regression

- **General approach**
- Let x denote the set of input variables and y the output variable.
- In machine learning, the general approach to regression is to assume a model, that is, some mathematical relation between x and y, involving some parameters say, θ, in the following form:

$$y = f(x, \theta)$$

- The function $f(x, \theta)$ is called the **regression function**.
- The **machine learning algorithm optimizes the parameters in the set θ such that the approximation error is minimized;** that is, the estimates of the values of the dependent variable y are as close as possible to the correct values given in the training set.

- **Example**
- For example, if the input variables are “Age”, “Distance” and “Weight” and the output variable is “Price”, the model may be

$$y = f(x, \theta)$$

- Price = $a_0 + a_1 \times (\text{Age}) + a_2 \times (\text{Distance}) + a_3 \times (\text{Weight})$
- where x = (Age, Distance, Weight) denotes the the set of input variables and $\theta = (a_0, a_1, a_2, a_3)$ denotes the set of parameters of the model.

Regression

- **Different regression models**
- There are various types of regression techniques available to make predictions. These techniques mostly differ in **three aspects, namely, the number and type of independent variables, the type of dependent variables and the shape of regression line.** Some of these are listed below.
- **Simple linear regression: There is only one continuous independent variable x** and the assumed relation between the independent variable and the dependent variable y is

$$y = a + bx.$$

- **Multivariate (Multiple) linear regression: There are more than one independent variable, say $x_1, \ldots, x_n$**, and the assumed relation between the independent variables and the dependent variable is

$$y = a_0 + a_1x_1 + \cdots + a_nx_n.$$

- **Polynomial regression: There is only one continuous independent variable x** and the assumed model is

$$y = a_0 + a_1x + a_2x^{2\cdots} + a_nx^n, \text{ (for some positive integer } n > 1)$$

- **Logistic regression:** The **dependent variable is binary (0/1, True/False, Yes/No)** in nature. Even though the output is a binary variable, what is being sought is a **probability function which may take any value from 0 to 1.**

Criterion for minimization of error

- In regression, we would like to write the numeric output y, called the dependent variable, as a function of the input x, called the independent variable.
- We **assume that the output is the sum of a function f(x) of the input and some random error denoted by ϵ :**

$$y = f(x) + \epsilon.$$

- Here the **function f(x) is unknown and we would like to approximate it by some estimator g(x, θ) containing a set of parameters θ.**
- We **assume that the random error ϵ follows normal distribution with mean 0.**
- Let $x_1, \ldots, x_n$ **be a random sample of observations of the input variable x and** $y_1, \ldots, y_n$ **the corresponding observed values of the output variable y.**
- Using the assumption that the error ϵ follows normal distribution, we can **apply the method of maximum likelihood estimation to estimate the values of the parameter θ.**
- It can be shown that the values of θ which maximizes the likelihood function are the values of θ that minimizes the following sum of squares:

$$E(\theta) = (y_1 - g(x_1, \theta))^2 + \cdots + (y_n - g(x_n, \theta))^2$$

- The **method of finding the value of θ as that value of θ that minimizes E(θ) is known as the ordinary least squares method.**

Criterion for minimization of error

x	x_1	x_2	$\cdots$	x_n
y	y_1	y_2	$\cdots$	y_n

Table 7.1: Data set for simple linear regression

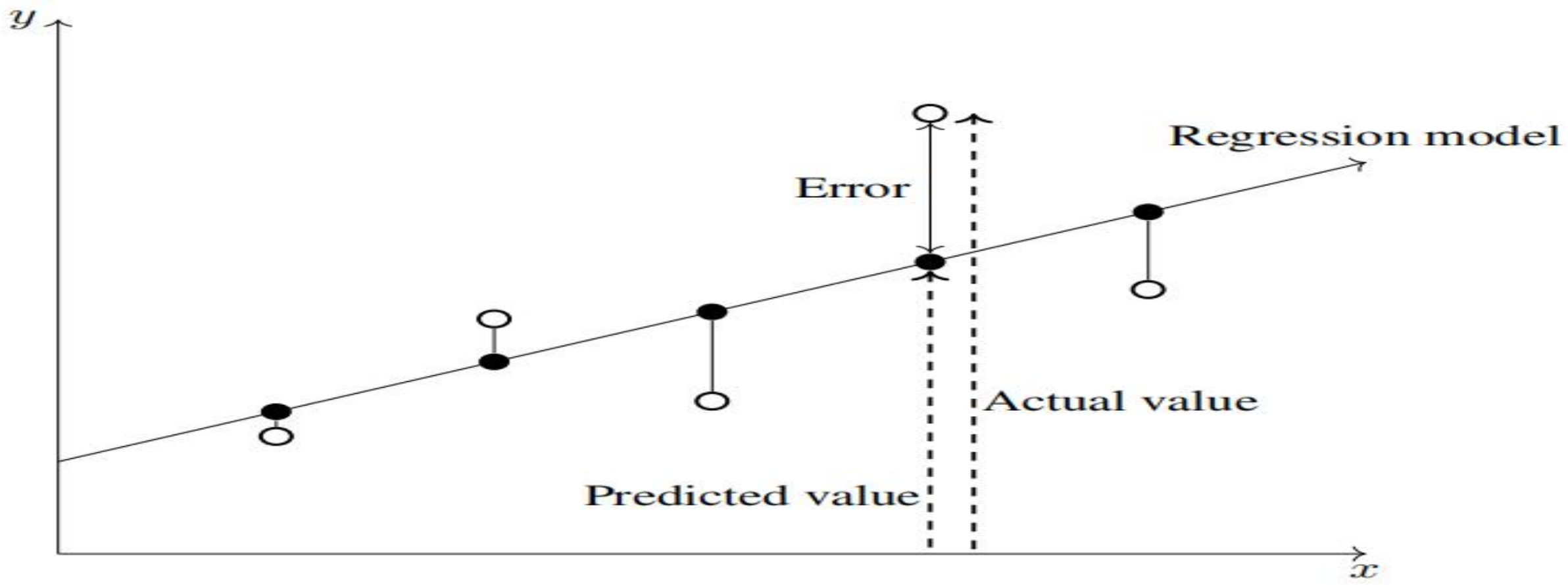


Figure 7.1: Errors in observed values

Simple linear regression

- Let x be the independent predictor variable and y the dependent variable.
- Assume that we have a set of observed values of x and y. A simple linear regression model defines the relationship between x and y using a line defined by an equation in the following form:

$$y = a + bx$$

- To determine the optimal estimates of α and β, an estimation method known as Ordinary Least Squares (OLS).

- **The OLS method**
- In the OLS method, the values of y-intercept and slope are chosen such that they minimize the sum of the squared errors; that is, the sum of the squares of the vertical distance between the predicted y-value and the actual y-value (see Figure 7.1). Let $\widehat{y_i}$ be the predicted value of y_i

- Then the sum of squares of errors is given by
- $\mathrm{E} = \sum_{i=1}^{n} (y_i - \widehat{y_i})^2$

  $= \sum_{i=1}^{n} [y_i - (a + bx_i)]^2$

- So we are required to find the values of **a** and **b** such that E is minimum.

Solution of Simple linear regression using OLS

- $E = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$

  $= \sum_{i=1}^{n} [y_i - (a + bx_i)]^2$

- To solve the above equation we have to take two partial derivations as below:

- $\frac{\partial E}{\partial a} = 0$ ------(i) and $\frac{\partial E}{\partial b} = 0$ ------(ii)

- By solving eq(i)

=> $2 \sum_{i=1}^{n} [y_i - a - bx_i](-1) = 0$

=> $-2 \sum_{i=1}^{n} y_i + 2a \sum_{i=1}^{n} 1 + 2b \sum_{i=1}^{n} x_i = 0$

=> $- \sum_{i=1}^{n} y_i + an + b \sum_{i=1}^{n} x_i = 0$

=> $an = \sum_{i=1}^{n} y_i - b \sum_{i=1}^{n} x_i$

=> $a = \frac{1}{n} \sum_{i=1}^{n} y_i - b \frac{1}{n} \sum_{i=1}^{n} x_i$

=> $\boldsymbol{a = \bar{y} - b\bar{x}}$

where $\bar{y} = \frac{1}{n} \sum_{i=1}^{n} y_i$ (mean of values of y), $\bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i$ (mean of values of x)

Solution of Simple linear regression using OLS

- $E = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$

  $= \sum_{i=1}^{n} [y_i - (a + bx_i)]^2$

- To solve the above equation we have to take two partial derivations as below:

- $\frac{\partial E}{\partial a} = 0$ ------(i) and $\frac{\partial E}{\partial b} = 0$ ------(ii)

- By solving eq(ii)

$=> 2\sum_{i=1}^{n} [y_i - a - bx_i](-x_i) = 0$

$=> -2\sum_{i=1}^{n} x_i y_i + 2a\sum_{i=1}^{n} x_i + 2b\sum_{i=1}^{n} x_i{}^2 = 0$

$=> -\sum_{i=1}^{n} x_i y_i + a\sum_{i=1}^{n} x_i + b\sum_{i=1}^{n} x_i{}^2 = 0$

$=> -\sum_{i=1}^{n} x_i y_i + (\bar{y} - b\bar{x})\sum_{i=1}^{n} x_i + b\sum_{i=1}^{n} x_i{}^2 = 0$

$=> b(\sum_{i=1}^{n} x_i{}^2 - \bar{x}\sum_{i=1}^{n} x_i) = \sum_{i=1}^{n} x_i y_i - \bar{y}\sum_{i=1}^{n} x_i$

$=> b[\sum_{i=1}^{n} x_i\{\sum_{i=1}^{n} (x_i - \bar{x})\}] = \sum_{i=1}^{n} x_i\{\sum_{i=1}^{n} (y_i - \bar{y})\}$

$=> \boldsymbol{b = \frac{\sum_{i=1}^{n} (y_i - \bar{y})}{\sum_{i=1}^{n} (x_i - \bar{x})}}$

By multiplying $\frac{1}{n-1}\{\sum_{i=1}^{n} (x_i - \bar{x})\}$ in the numerator and denominator of RHS

$=> b = \frac{\frac{1}{n-1}\{\sum_{i=1}^{n} (x_i - \bar{x})\}\sum_{i=1}^{n} (y_i - \bar{y})}{\frac{1}{n-1}\{\sum_{i=1}^{n} (x_i - \bar{x})\}\sum_{i=1}^{n} (x_i - \bar{x})} = \boldsymbol{\frac{Cov(x,y)}{Var(x)}}$

Solution of Simple linear regression using OLS

Formulas to find a and b

- Recall that the means of x and y are given by

$$\bar{x} = \frac{1}{n}\sum x_i$$
$$\bar{y} = \frac{1}{n}\sum y_i$$

- and also that the variance of x is given by

$$\text{Var}(x) = \frac{1}{n-1}\sum(x_i - \bar{x})^2$$

- The covariance of x and y, denoted by Cov(x, y) is defined as

$$\text{Cov}(x, y) = \frac{1}{n-1}\sum(x_i - \bar{x})(y_i - \bar{y})$$

- It can be shown that the values of a and b can be computed using the following formulas:

$$b = \frac{\text{Cov}(x, y)}{\text{Var}(x)}$$
$$a = \bar{y} - b\bar{x}$$

Variations of Least Square for Solving Simple linear regression

- **Remarks:**
- It is interesting to note why the least squares method discussed above is christened as “ordinary” least squares method.

- Several different variants of the least squares method have been developed over the years. For example, in the weighted least squares method, the coefficients **a** and **b** are estimated such that the weighted sum of squares of errors.

$$E = \sum_{i=1}^{n} w_i [y_i - (a + bx_i)]^2$$

- for some positive constants $w_1, \ldots, w_n$, is minimum. There are also methods known by the names **generalised least squares method, partial least squares method, total least squares method**, etc.

- The OLS method has a long history. The method is usually credited to **Carl Friedrich Gauss (1795),** but it was first published by **Adrien-Marie Legendre (1805).**

Simple linear regression Example

- **Example:**
- Obtain a linear regression for the data in Table 7.2 assuming that y is the independent variable.

x	1.0	2.0	3.0	4.0	5.0
y	1.00	2.00	1.30	3.75	2.25

Table 7.2: Example data for simple linear regression

- **Solution:**
- In the usual notations of simple linear regression, we have

$$
\begin{aligned}
n &= 5 \\
\bar{x} &= \frac{1}{5}(1.0 + 2.0 + 3.0 + 4.0 + 5.0) \\
&= 3.0 \\
\bar{y} &= \frac{1}{5}(1.00 + 2.00 + 1.30 + 3.75 + 2.25) \\
&= 2.06 \\
\mathrm{Cov}(x, y) &= \frac{1}{4}[(1.0 - 3.0)(1.00 - 2.06) + \cdots + (5.0 - 3.0)(2.25 - 2.06)] \\
&= 1.0625 \\
\mathrm{Var}(x) &= \frac{1}{4}[(1.0 - 3.0)^2 + \cdots + (5.0 - 3.0)^2] \\
&= 2.5 \\
b &= \frac{1.0625}{2.5} \\
&= 0.425 \\
a &= 2.06 - 0.425 \times 3.0 \\
&= 0.785
\end{aligned}
$$

Simple linear regression Example

- Therefore, the linear regression model for the data is

$$y = 0.785 + 0.425x.$$

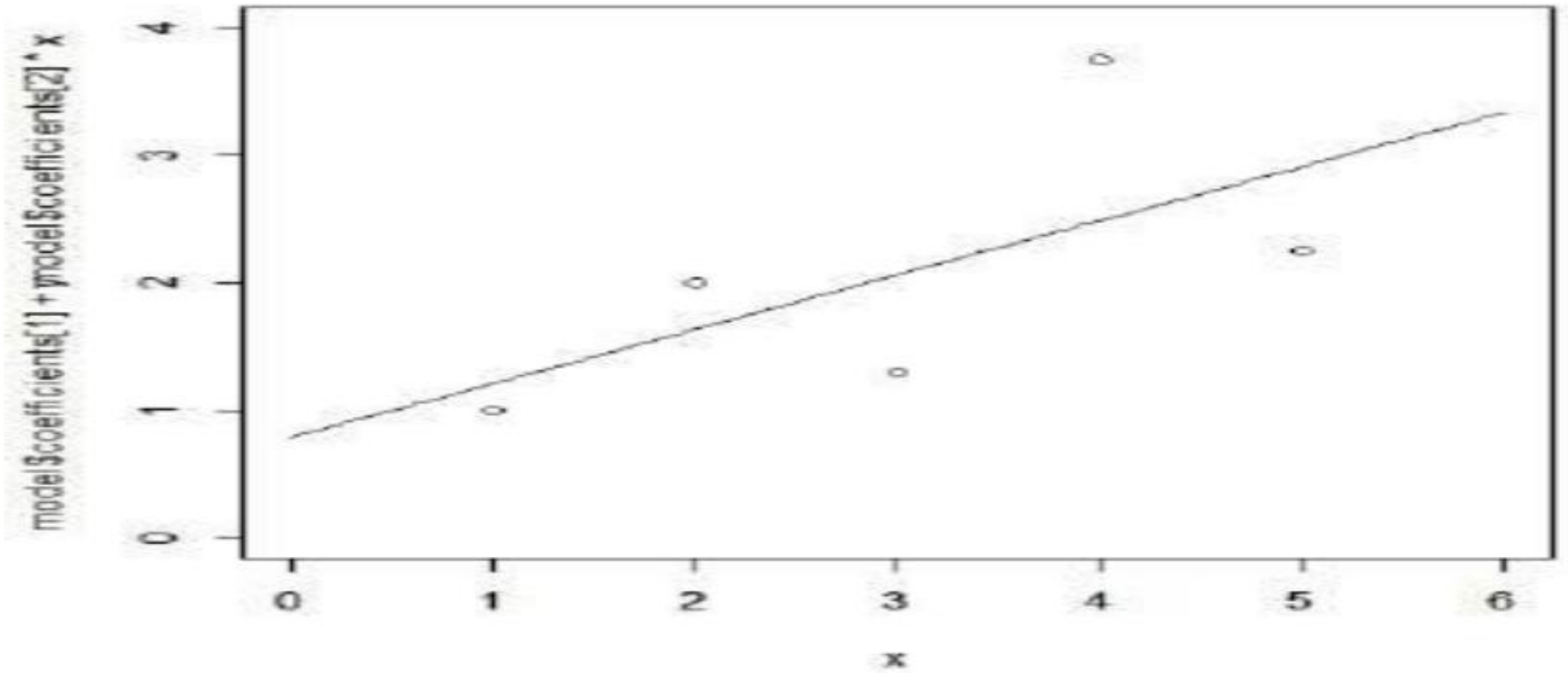
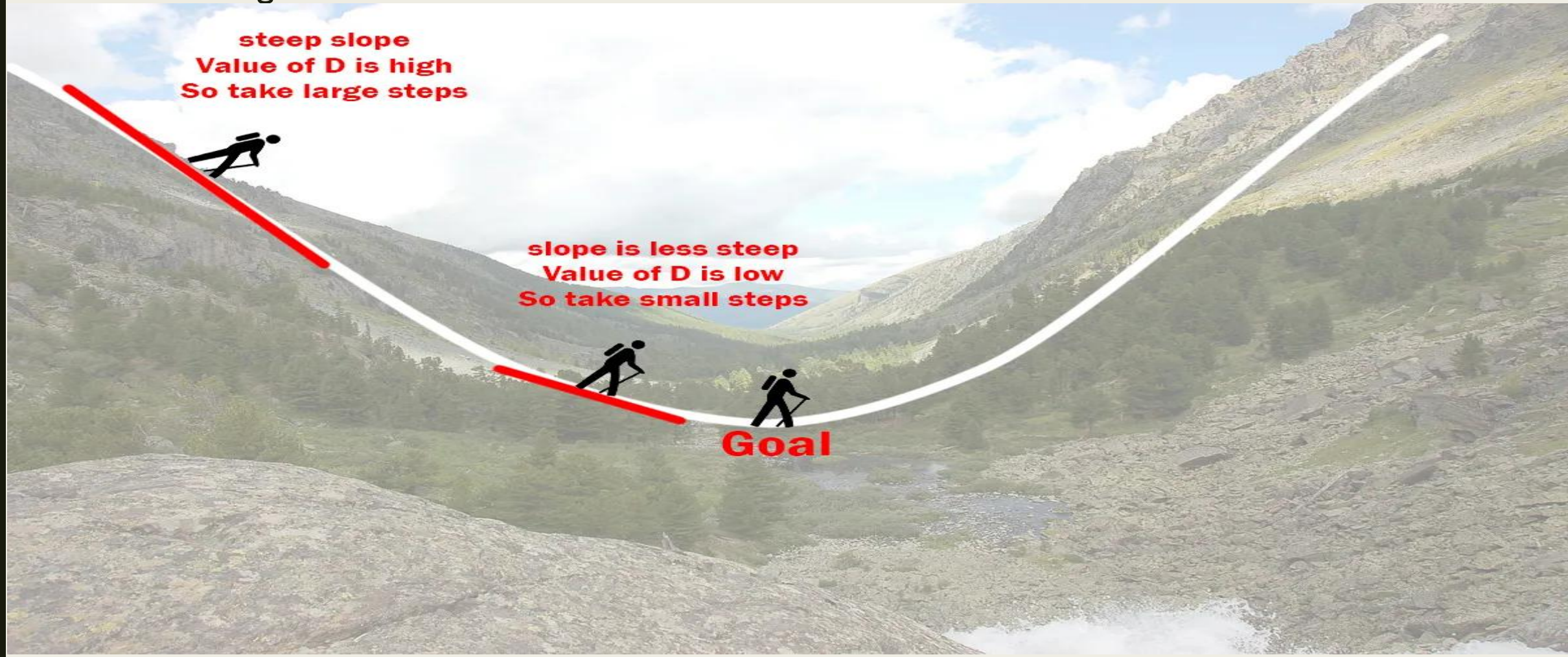


Figure 7.2: Regression model for Table 7.2

Linear Regression using Gradient Descent

- **Gradient descent is an iterative optimization algorithm to find the minimum of a function.** Here that function is our Loss Function. We will use **Mean Square Error (MSE)** as Loss Function in this topic which is shown below:
- $\mathrm{E} = \frac{1}{n}\sum_{i=1}^{n}[y_i - (a + bx_i)]^2 = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$
- Understanding Gradient Descent



Linear Regression using Gradient Descent

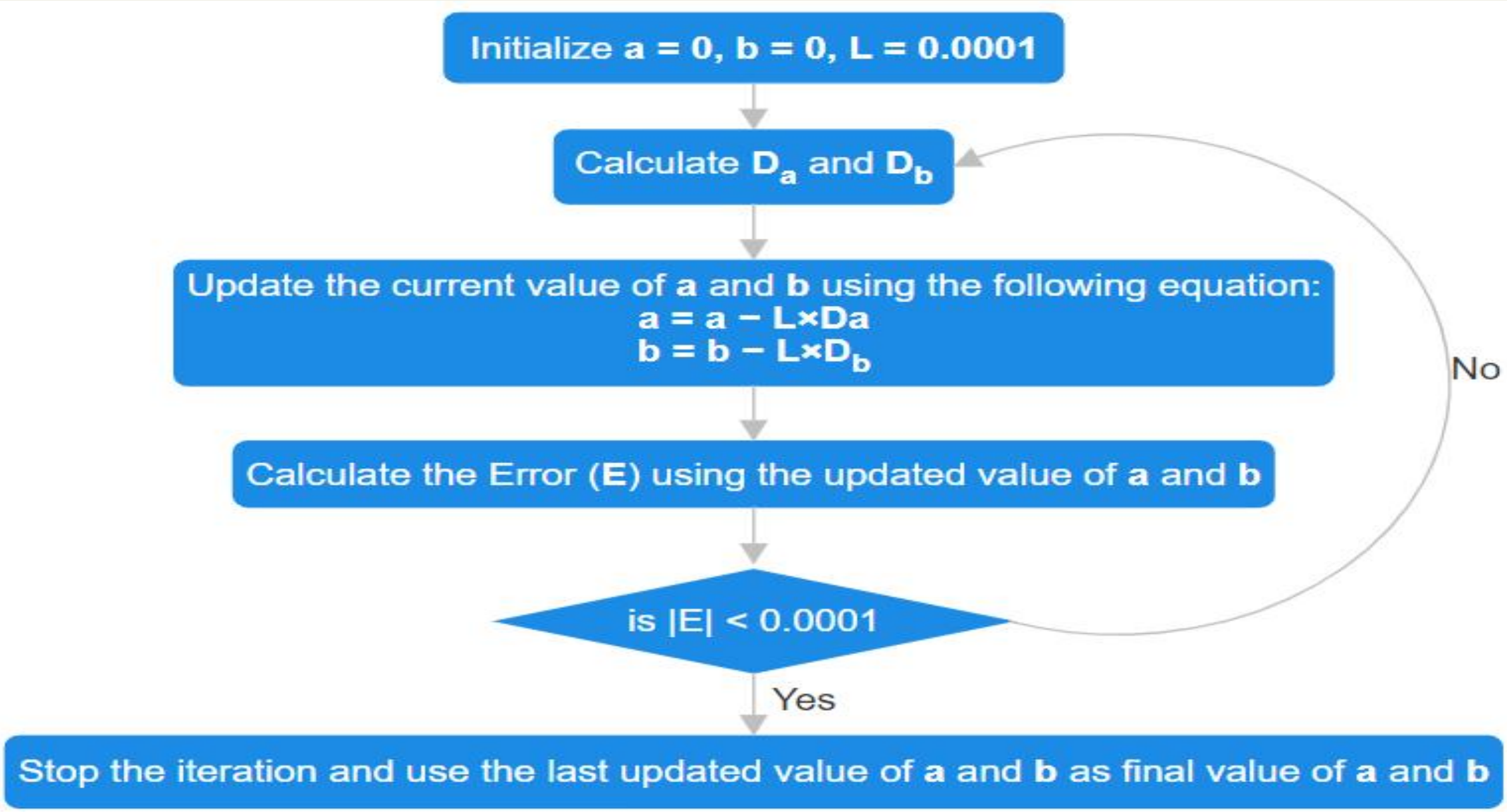
- **Mathematical derivation of Gradient Descent in simple Linear Regression :**
- 1. Initially let **a = 0** and **b = 0**. Let **L** be our learning rate. This controls how much the value of **b** changes with each step. **L** could be a small value like **0.0001 for good accuracy.**
- 2. Calculate the partial derivative of the loss function with respect to **a** and **b**, and plug in the current values of **x, y, b** and **a** in it to obtain the derivative value D.
- $D_b = 2\frac{1}{n}\sum_{i=1}^{n}[y_i - (a + bx_i)](-x_i)$
- $\mathbf{D_b = \frac{-2}{n}\sum_{i=1}^{n} x_i[y_i - (a + bx_i)]}$
- $\mathbf{D_b = \frac{-2}{n}\sum_{i=1}^{n} x_i(y_i - \hat{y}_i)}$
- $\mathbf{D_b}$ is the value of the partial derivative with respect to **b**.

- Similarly, the partial derivative with respect to **a** is $\mathbf{D_a}$**:**
- $D_a = 2\frac{1}{n}\sum_{i=1}^{n}[y_i - (a + bx_i)](-1)$
- $\mathbf{D_a = \frac{-2}{n}\sum_{i=1}^{n}[y_i - (a + bx_i)]}$
- $\mathbf{D_a = \frac{-2}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)}$

Linear Regression using Gradient Descent

- **Mathematical derivation of Gradient Descent in simple Linear Regression :**
- 3. Now we update the current value of **b** and **a** using the following equation:
- $b = b - L \times D_b$
- $a = a - L \times D_a$
- 4. We repeat this process until our loss function is a very small value or ideally **0** (which means **0 error or 100%** accuracy). The value of **b** and **a** that we are left with now will be the optimum values.
- Now going back to our analogy, **b** can be considered the current position of the person. **D is equivalent to the steepness of the slope and L can be the speed with which he moves.** Now the **new value of b that we calculate using the above equation will be his next position, and L×D will be the size of the steps he will take.**
- When the slope is more steep (D is more) he takes longer steps and when it is less steep (D is less), he takes smaller steps.
- Finally he arrives at the bottom of the valley which corresponds to our loss = 0.
- Now with the optimum value of **b** and **a** our model is ready to make predictions !

Flowchart of Linear Regression using Gradient Descent



Multiple linear regression

- We assume that there are N independent variables $x_1, x_2, \cdots, x_N$. Let the dependent variable be y.
- Let there also be n observed values of these variables:

Variables (features)	Values (examples)			
	Example 1	Example 2	⋯	Example n
x_1	x_{11}	x_{12}	⋯	x_{1n}
x_2	x_{21}	x_{22}	⋯	x_{2n}
⋯				
x_N	x_{N1}	x_{N2}	⋯	x_{Nn}
y (outcomes)	y_1	y_2	⋯	y_n

Table 7.3: Data for multiple linear regression

- The multiple linear regression model defines the relationship between the N independent variables and the dependent variable by an equation of the following form:

$$y = \beta_0 + \beta_1 x_1 + \cdots + \beta_N x_N$$

Multiple linear regression

- As in simple linear regression, here also **we use the ordinary least squares (OLS) method to obtain the optimal estimates of $\beta_0, \beta_1, \cdots, \beta_N$.** The method yields the following procedure for the computation of these optimal estimates. Let

$$X = \begin{bmatrix} 1 & x_{11} & x_{21} & \cdots & x_{N1} \\ 1 & x_{12} & x_{22} & \cdots & x_{N2} \\ \vdots & & & & \\ 1 & x_{1n} & x_{2n} & \cdots & x_{Nn} \end{bmatrix}, \quad Y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}, \quad B = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_N \end{bmatrix}$$

- Then it can be shown that the regression coefficients are given by

$$\mathbf{B = (X^T X)^{-1} X^T Y}$$

Multiple linear regression Example

- **Example:**
- Fit a multiple linear regression model to the following data:

x_1	1	1	2	0
x_2	1	2	2	1
y	3.25	6.5	3.5	5.0

Table 7.4: Example data for multi-linear regression

- **Solution:**
- In this problem, there are two independent variables and four sets of values of the variables. Thus, in the notations used above, we have n = 2 and N = 4. The multiple linear regression model for this problem has the form

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2.$$

- The computations are shown below.

$$X = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 2 \\ 1 & 2 & 2 \\ 1 & 0 & 1 \end{bmatrix}, \quad Y = \begin{bmatrix} 3.25 \\ 6.5 \\ 3.5 \\ 5.0 \end{bmatrix}, \quad B = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \end{bmatrix}$$

Multiple linear regression Example

$$X^T X = \begin{bmatrix} 4 & 4 & 6 \\ 4 & 6 & 7 \\ 6 & 7 & 10 \end{bmatrix}$$

$$(X^T X)^{-1} = \begin{bmatrix} \frac{11}{4} & \frac{1}{2} & -2 \\ \frac{1}{2} & 1 & -1 \\ -2 & -1 & 2 \end{bmatrix}$$

$$B = (X^T X)^{-1} X^T Y$$

$$= \begin{bmatrix} 2.0625 \\ -2.3750 \\ 3.2500 \end{bmatrix}$$

- The required model is

$$y = 2.0625 - 2.3750x_1 + 3.2500x_2$$

Multiple linear regression Example

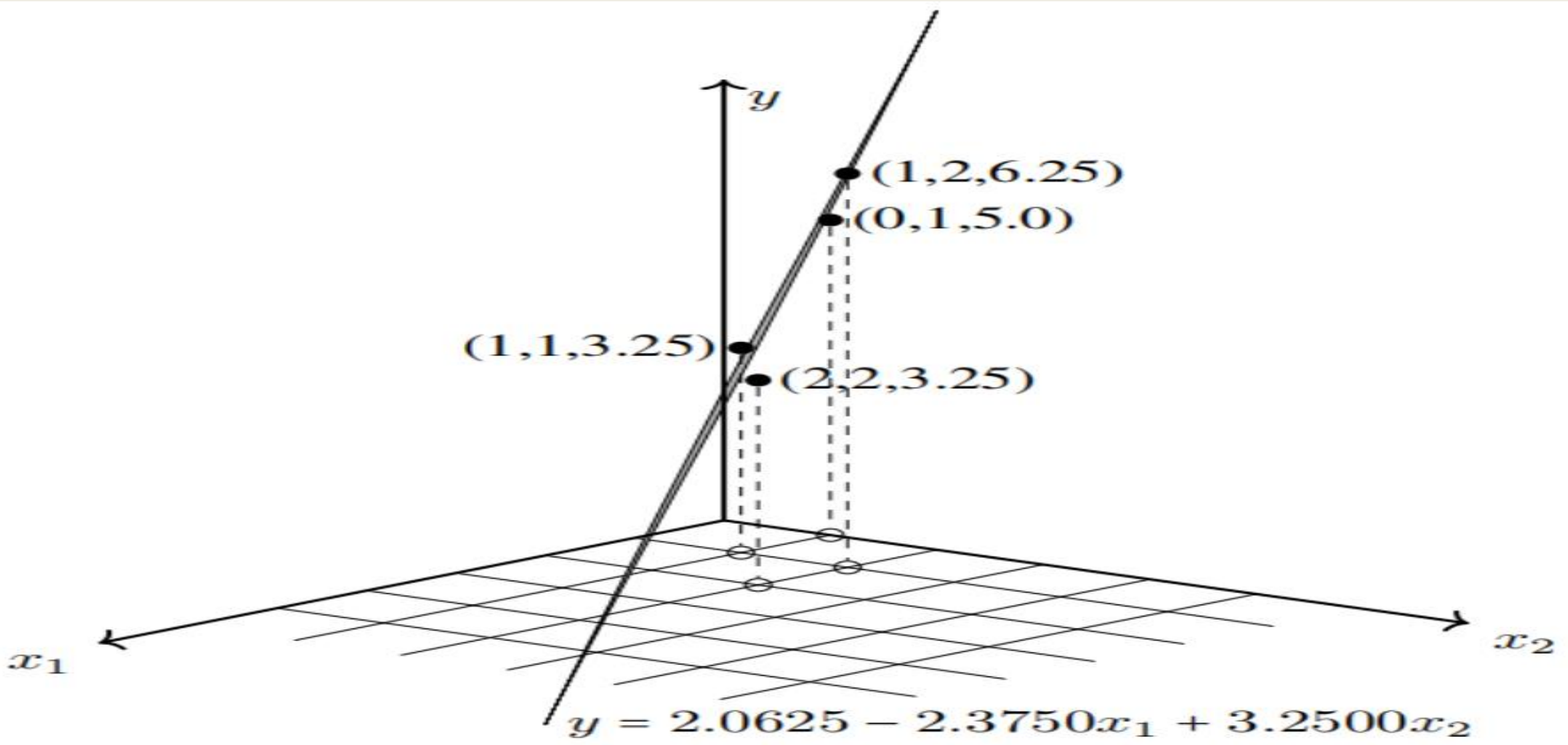
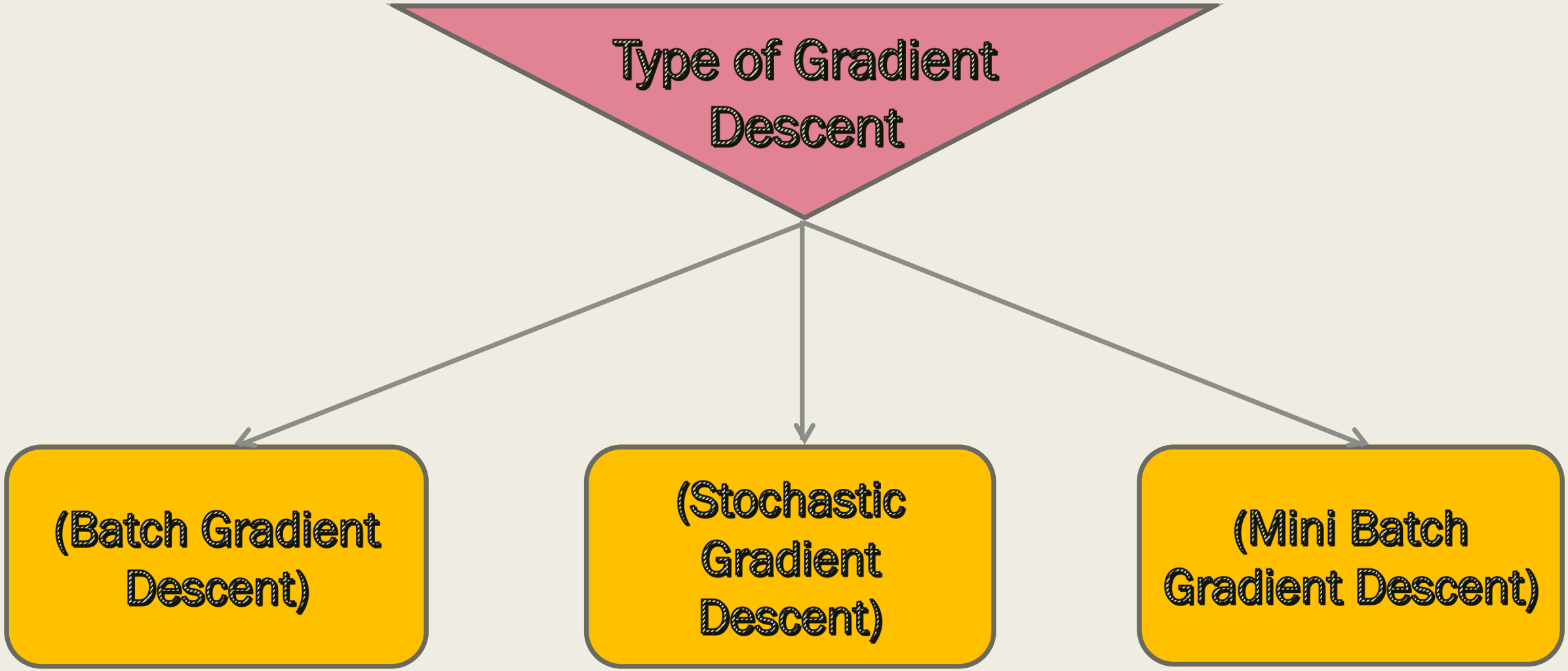
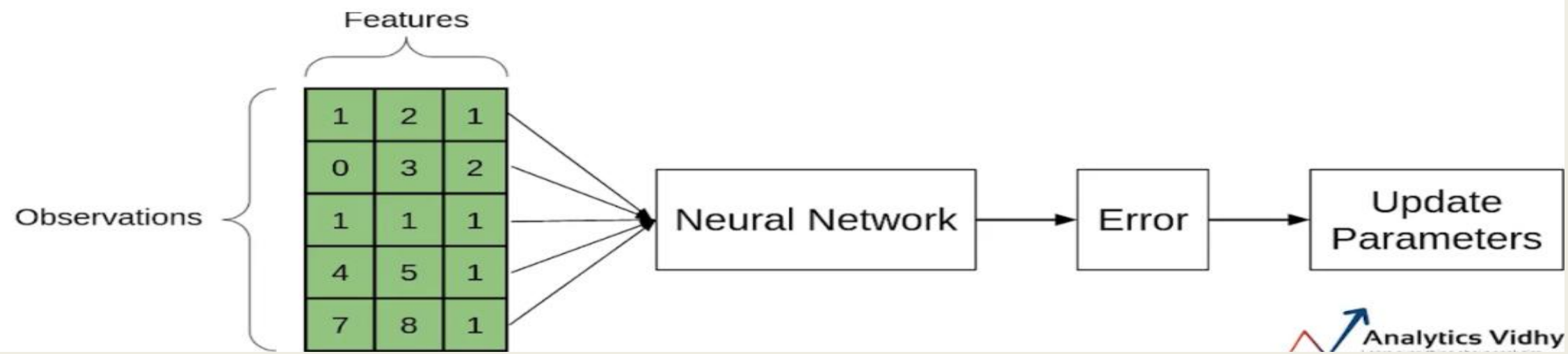


Figure 7.4: The regression plane for the data in Table 7.4


Type of Gradient Descent
(Batch Gradient Descent)
(Stochastic Gradient Descent)
(Mini Batch Gradient Descent)

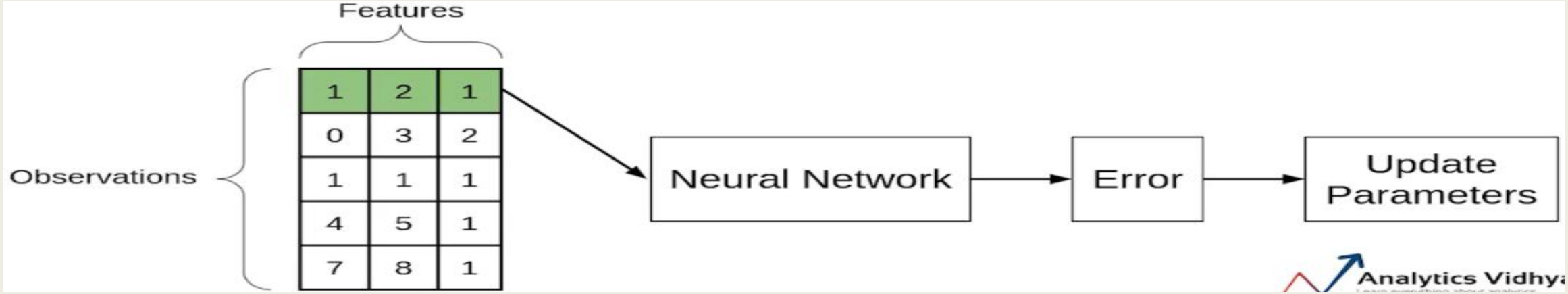
Batch Gradient Descent

- In Batch Gradient Descent, all the training data is taken into consideration to take a single step.
- We take the average of the gradients of all the training examples and then use that mean gradient to update our parameters. So that's just one step of gradient descent in one epoch.
- In batch Gradient Descent since we are using the entire training set, the parameters will be updated only once per epoch.
- Batch Gradient Descent is great for convex or relatively smooth error manifolds.
- In this case, we move somewhat directly towards an optimum solution.
- The graph of cost vs epochs is also quite smooth because we are averaging over all the gradients of training data for a single step. The cost keeps on decreasing over the epochs.



Stochastic Gradient Descent

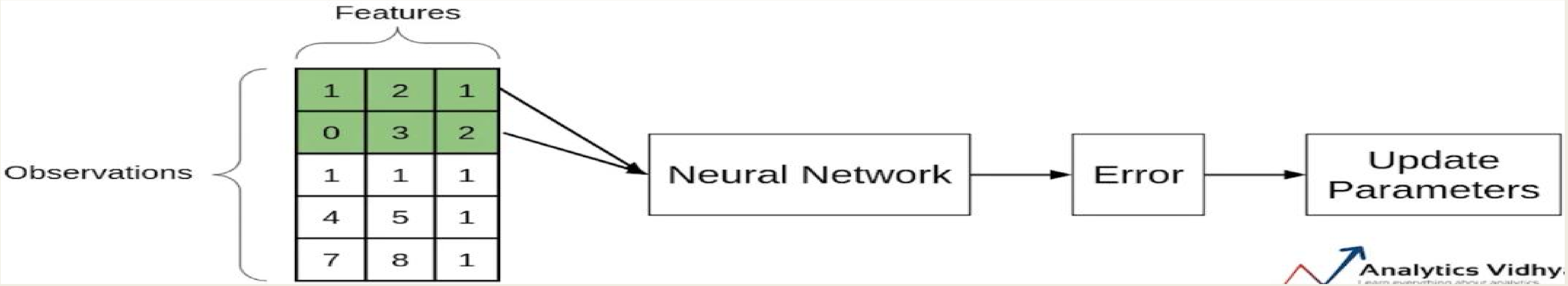
- In Batch Gradient Descent we were considering all the examples for every step of Gradient Descent. But what if our dataset is very huge.
- Suppose our dataset has 5 million examples, then just to take one step the model will have to calculate the gradients of all the 5 million examples.
- This does not seem an efficient way. To tackle this problem we have Stochastic Gradient Descent.
- In Stochastic Gradient Descent (SGD), we consider just one example at a time to take a single step.
- Since we are considering just one example at a time the cost will fluctuate over the training examples and it will not necessarily decrease. But in the long run, you will see the cost decreasing with fluctuations.
- Because the cost is so fluctuating, it will never reach the minima but it will keep dancing around it.
- SGD can be used for larger datasets. It converges faster when the dataset is large as it causes updates to the parameters more frequently.

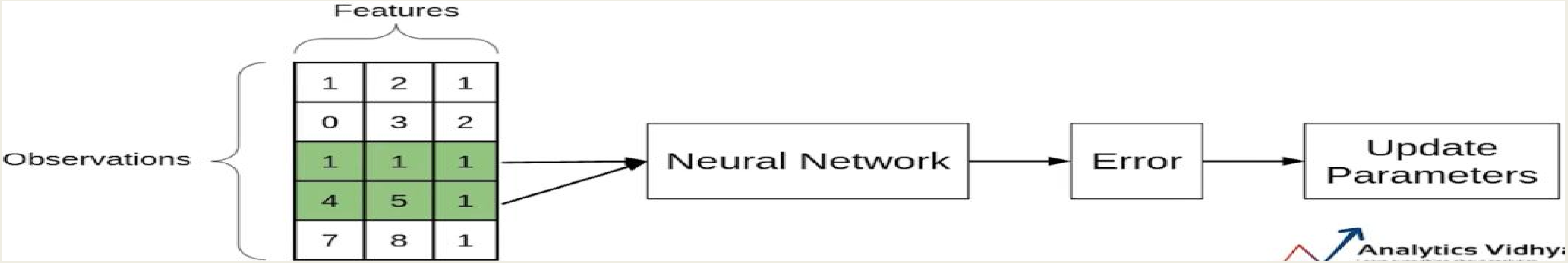


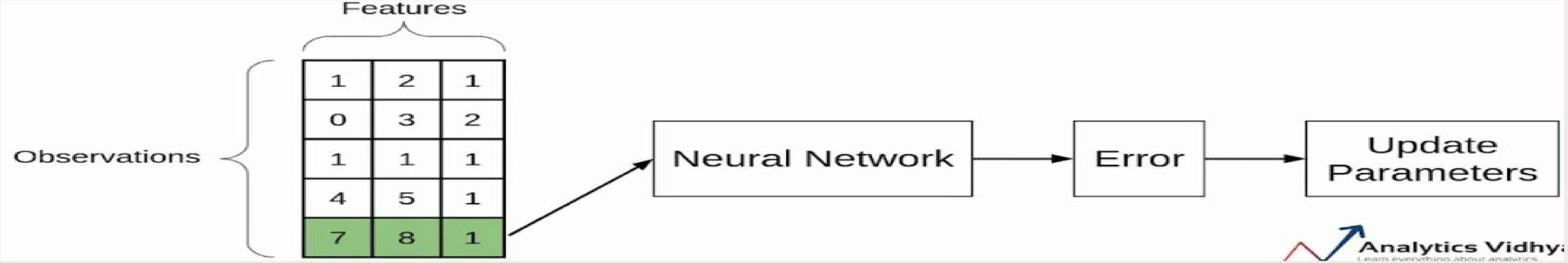
Mini Batch Gradient Descent

- Batch Gradient Descent converges directly to minima. SGD converges faster for larger datasets. But, since in SGD we use only one example at a time, we cannot implement the vectorized implementation on it.
- This can slow down the computations. To tackle this problem, a mixture of Batch Gradient Descent and SGD is used.
- Neither we use all the dataset all at once nor we use the single example at a time.
- We use a batch of a fixed number of training examples which is less than the actual dataset and call it a mini-batch. Doing this helps us achieve the advantages of both the former variants (GD and SGD).
- Just like SGD, the average cost over the epochs in mini-batch gradient descent fluctuates because we are averaging a small number of examples at a time.
- When we are using the mini-batch gradient descent we are updating our parameters frequently as well as we can use vectorized implementation for faster computations.

Mini Batch Gradient Descent

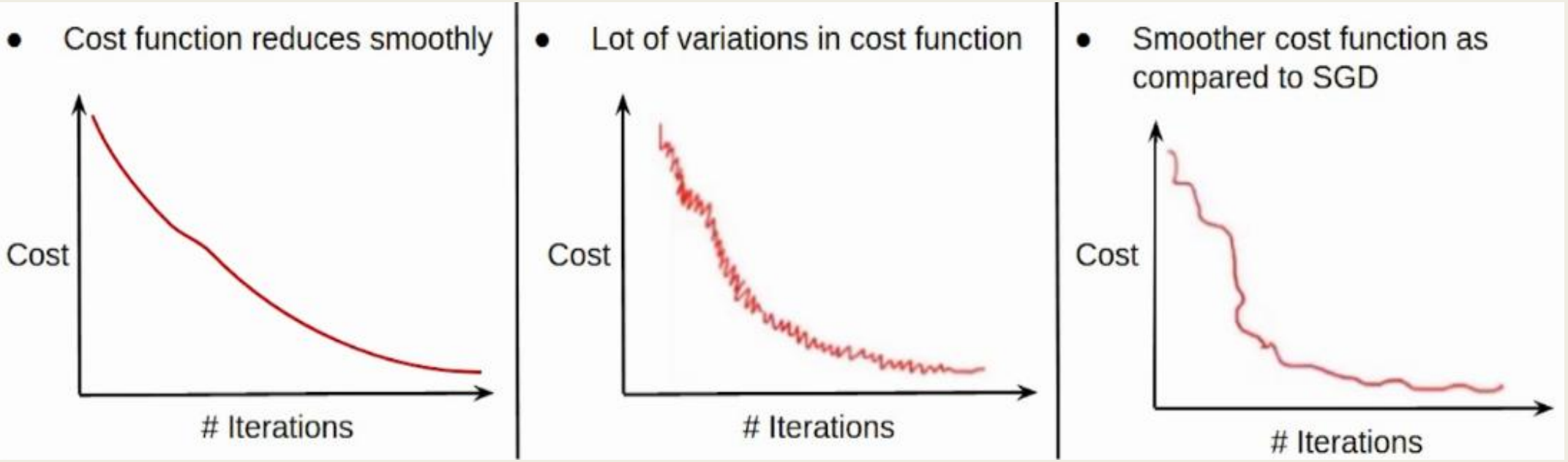






Comparison between different type of Gradient Descent

Batch Gradient Descent	Stochastic Gradient Descent (SGD)	Mini-Batch Gradient Descent
• Entire dataset for updation	• Single observation for updation	• Subset of data for updation
• Cost function reduces smoothly	• Lot of variations in cost function	• Smoother cost function as compared to SGD
• Computation cost is very high	• Computation time is more	• Computation time is lesser than SGD
		• Computation cost is lesser than Batch Gradient Descent



Feature Scaling (Need)

- In practice, we often encounter different types of variables in the same dataset.
- A significant issue is that the range of the variables may differ a lot.
- Using the original scale may put more weights on the variables with a large range.
-
- In order to deal with this problem, we need to apply the technique of features rescaling to independent variables or features of data in the step of data pre-processing.
- The terms normalisation and standardisation are sometimes used interchangeably, but they usually refer to different things.
- The goal of applying Feature Scaling is to make sure features are on almost the same scale so that each feature is equally important and make it easier to process by most ML algorithms

Feature Scaling (Example)

- This is a dataset that contains a dependent variable (Purchased) and 3 independent variables (Country, Age, and Salary). We can easily notice that the variables are not on the same scale because the range of Age is from 27 to 50, while the range of Salary going from 48 K to 83 K. The range of Salary is much wider than the range of Age. This will cause some issues in our models since a lot of machine learning models such as k-means clustering and nearest neighbour classification are based on the Euclidean Distance.

	Country	Age	Salary	Purchased
1	France	44	72000	No
2	Spain	27	48000	Yes
3	Germany	30	54000	No
4	Spain	38	61000	No
5	Germany	40		Yes
6	France	35	58000	Yes
7	Spain		52000	No
8	France	48	79000	Yes
9	Germany	50	83000	No
10	France	37	67000	Yes

```
dataset['Age'].min()
```

```
27.0
```

```
dataset['Salary'].min()
```

```
48000.0
```

```
dataset['Age'].max()
```

```
50.0
```

```
dataset['Salary'].max()
```

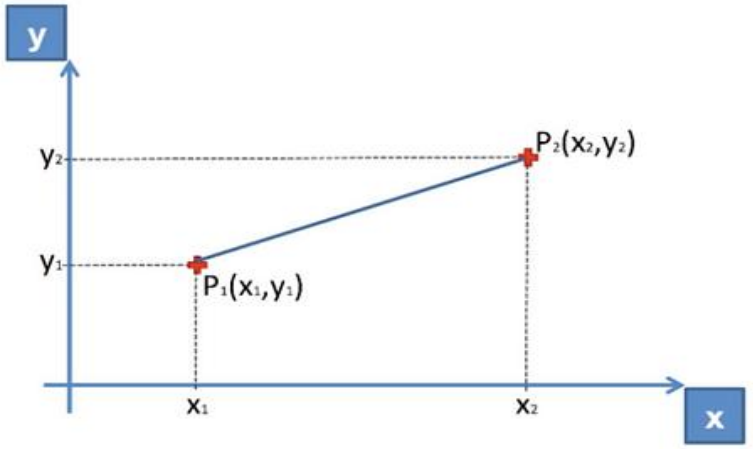
```
83000.0
```

The range of Age: 27 - 50

The range of Salary:48,000 - 83,000

Feature Scaling (Example)

- When we calculate the equation of Euclidean distance, the number of $(x2-x1)^2$ is much bigger than the number of $(y2-y1)^2$ which means the Euclidean distance will be dominated by the salary if we do not apply feature scaling.
- The difference in Age contributes less to the overall difference.
- Therefore, we should use Feature Scaling to bring all values to the same magnitudes and, thus, solve this issue.
- To do this, there are primarily two methods called **Standardisation and Normalisation.**



Euclidean Distance between P_1 and $P_2 = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$

Let x be the no. of Salary and y be the no. of Age

Example: x1&y1 are in row 2, x2&y2 are in row 9

(x2-x1)^2= (83000-48000)^2

=1225000000

(y2-y1)^2=(50-27)^2

=529

Standardisation

- The result of standardization (or **Z-score normalization**) is that the features will be rescaled to ensure the **mean and the standard deviation to be 0 and 1, respectively**. The equation is shown below:

$$x_{\text{stand}} = \frac{x - \text{mean}(x)}{\text{standard deviation } (x)}$$

- This technique is to re-scale features value is useful for the optimization algorithms, such as gradient descent, that are used within machine learning algorithms that weight inputs (e.g., regression and neural networks).
- Rescaling is also used for algorithms that use distance measurements, for example, K-Nearest-Neighbours (KNN).

Normalization

- Another common approach is the so-called **Max-Min Normalization (Min-Max scaling)**.
- This technique is to re-scales features with a distribution value between 0 and 1.
- For every feature, the minimum value of that feature gets transformed into 0, and the maximum value gets transformed into 1. The general equation is shown below:

$$x_{\text{norm}} = \frac{x - \min(x)}{\max(x) - \min(x)}$$

Standardisation vs Max-Min Normalization

- In contrast to standardisation, we will obtain smaller standard deviations through the process of Max-Min Normalisation. Let me illustrate more in this area using the above dataset.

Standardisation

	Age	Salary
0	0.758874	7.494733e-01
1	-1.711504	-1.438178e+00
2	-1.275555	-8.912655e-01
3	-0.113024	-2.532004e-01
4	0.177609	6.632192e-16
5	-0.548973	-5.266569e-01
6	0.000000	-1.073570e+00
7	1.340140	1.387538e+00
8	1.630773	1.752147e+00
9	-0.258340	2.937125e-01

Max-Min Normalization

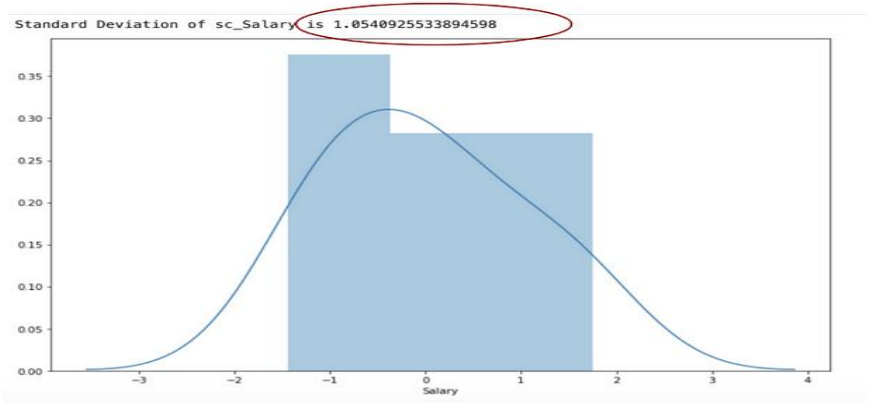
	Age	Salary
0	0.739130	0.685714
1	0.000000	0.000000
2	0.130435	0.171429
3	0.478261	0.371429
4	0.565217	0.450794
5	0.347826	0.285714
6	0.512077	0.114286
7	0.913043	0.885714
8	1.000000	1.000000
9	0.434783	0.542857

Standardisation vs Max-Min Normalization

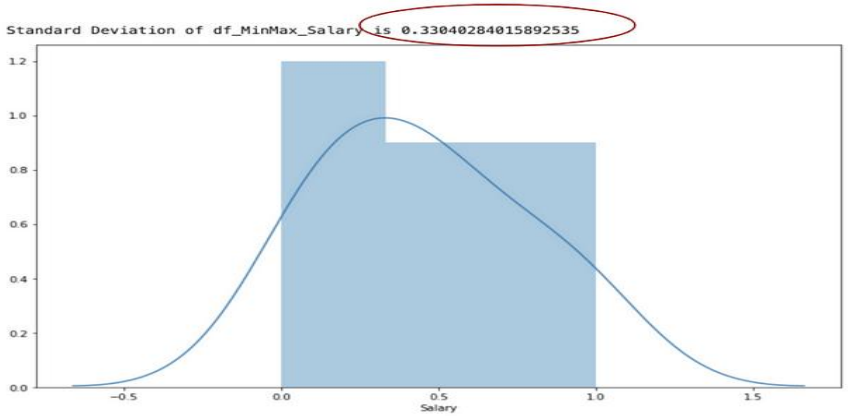
Column: Salary

Standard Deviation (Salary):
Max-Min Normalization (0.33) < Standardisation (1.05)

Standardisation



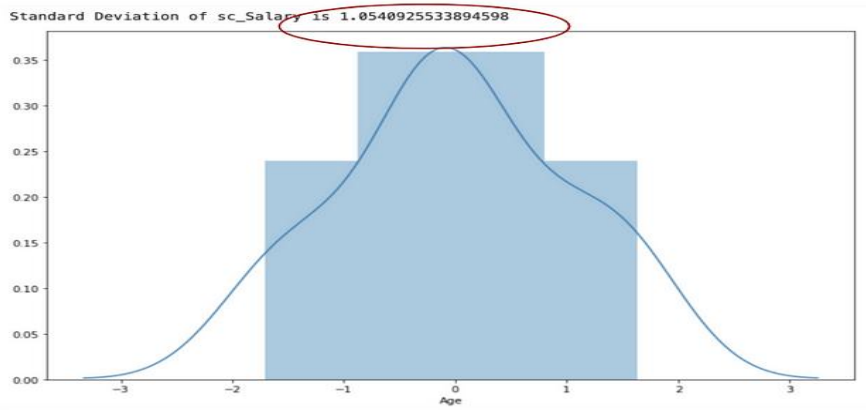
Max-Min Normalisation



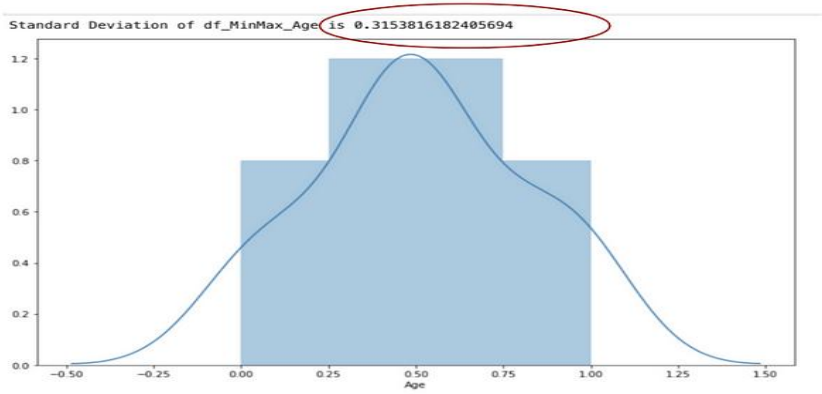
Column:Age

Standard Deviation (Age):
Max-Min Normalization (0.315) < Standardisation (1.05)

Standardisation



Max-Min Normalisation

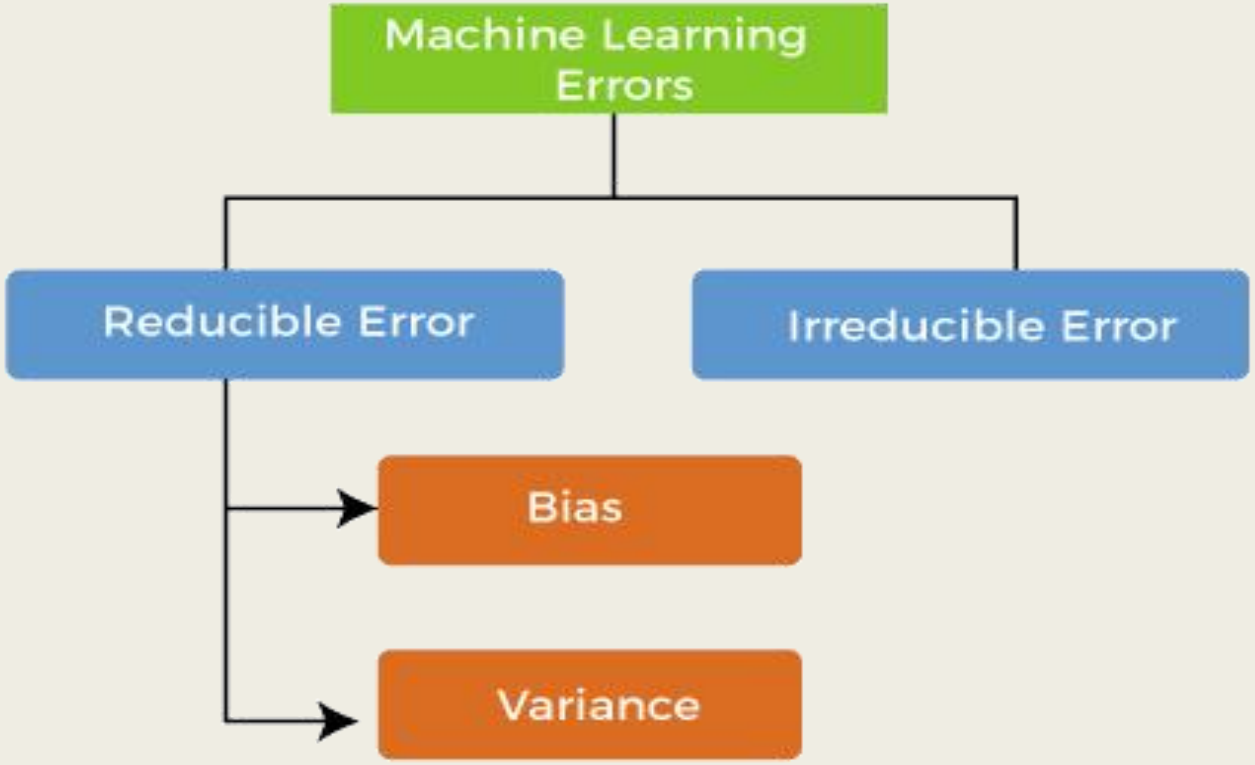


Standardisation vs Max-Min Normalization

- Max-Min Normalisation typically allows us to transform the data with varying scales so that no specific dimension will dominate the statistics, and it does not require making a very strong assumption about the distribution of the data, such as k-nearest neighbours and artificial neural networks.
- However, Normalisation does not treat outliners very well.
- On the contrary, standardisation allows users to better handle the outliers and facilitate convergence for some computational algorithms like gradient descent.
- Therefore, we usually prefer standardisation over Min-Max Normalisation.

Errors in Machine Learning

- In machine learning, an **error is a measure of how accurately an algorithm can make predictions for the previously unknown dataset**.
- On the basis of these errors, the machine learning model is selected that can perform best on the particular dataset. There are mainly two types of errors in machine learning, which are:
- **Reducible errors:** These errors **can be reduced to improve the model accuracy**. Such errors can further be classified into bias and Variance.
- **Irreducible errors:** These errors **will always be present in the model.**



What is Bias?

- While making predictions, **a difference occurs between prediction values made by the model and actual values, and this difference is known as bias errors** or Errors due to bias.
- It can be defined as **an inability of machine learning algorithms such as Linear Regression to capture the true relationship between the data points.**
- Each algorithm begins with some amount of bias because bias occurs from assumptions in the model, which makes the target function simple to learn. A model has either:
- **Low Bias:** A low bias model **will make fewer assumptions about the form of the target function.**
- **High Bias:** A model with a high bias **makes more assumptions, and the model becomes unable to capture the important features of our dataset.** A high bias model also cannot perform well on new data.
- Generally, a linear algorithm has a high bias, as it makes them learn fast. **The simpler the algorithm, the higher the bias it has likely to be introduced. Whereas a nonlinear algorithm often has low bias.**
- Some examples of machine learning algorithms with **low bias are Decision Trees, k-Nearest Neighbours and Support Vector Machines.**
- At the same time, **an algorithm with high bias is Linear Regression, Linear Discriminant Analysis and Logistic Regression.**

Ways to reduce High Bias

- High bias mainly **occurs due to a much simple model.**

- Below are some ways to reduce the high bias:

- **Increase the input features** as the model is underfitted.
- **Decrease the regularization term.**
- **Use more complex models, such as including some polynomial features**

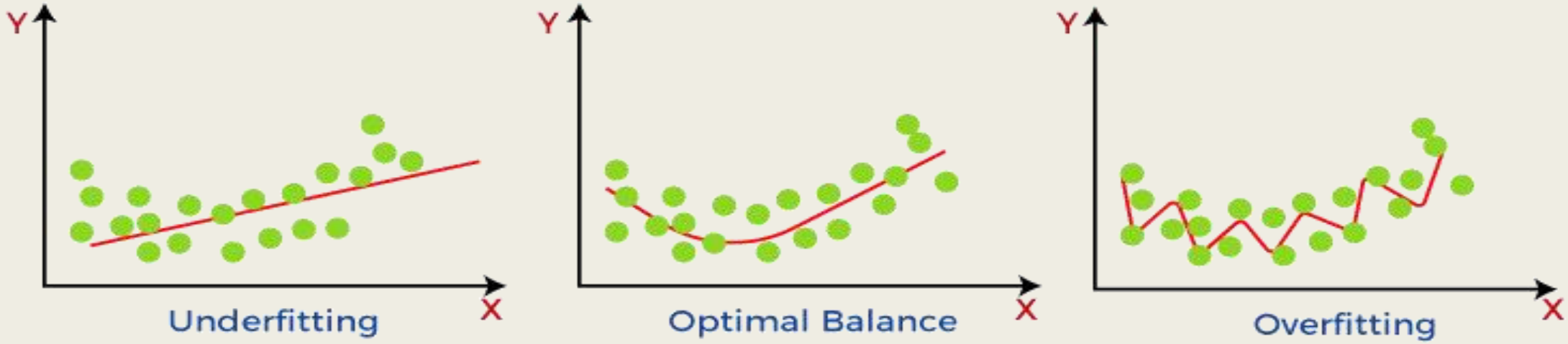
What is a Variance Error?

- The variance specify the **amount of variation in the prediction if the different training data was used.**
- **Ideally, a model should not vary too much from one training dataset to another, which means the algorithm should be good in understanding the hidden mapping between inputs and output variables.**
- Variance errors are either of low variance or high variance.
- **Low variance means there is a small variation in the prediction of the target function with changes in the training data set.**
- At the same time, **High variance shows a large variation in the prediction of the target function with changes in the training dataset.**
- A model that shows high variance learns a lot and perform well with the training dataset, and does not generalize well with the unseen dataset.
- As a result, such a model gives good results with the training dataset but shows high error rates on the test dataset.

What is a Variance Error?

- **With high variance, the model learns too much from the dataset, it leads to overfitting of the model.**
- A model with high variance has the below problems:

- A high variance model **leads to overfitting.**
- **Increase model complexities.**
- Usually, **nonlinear algorithms have a lot of flexibility to fit the model, have high variance.**



- Some examples of machine learning **algorithms with low variance are, Linear Regression, Logistic Regression, and Linear discriminant analysis.**
- At the same time, **algorithms with high variance are decision tree, Support Vector Machine, and K-nearest neighbours.**

Ways to Reduce High Variance

- **Reduce the input features or number of parameters** as a model is overfitted.
- **Do not use a much complex model.**
- **Increase the training data.**
- **Increase the Regularization term.**

Different Combinations of Bias-Variance

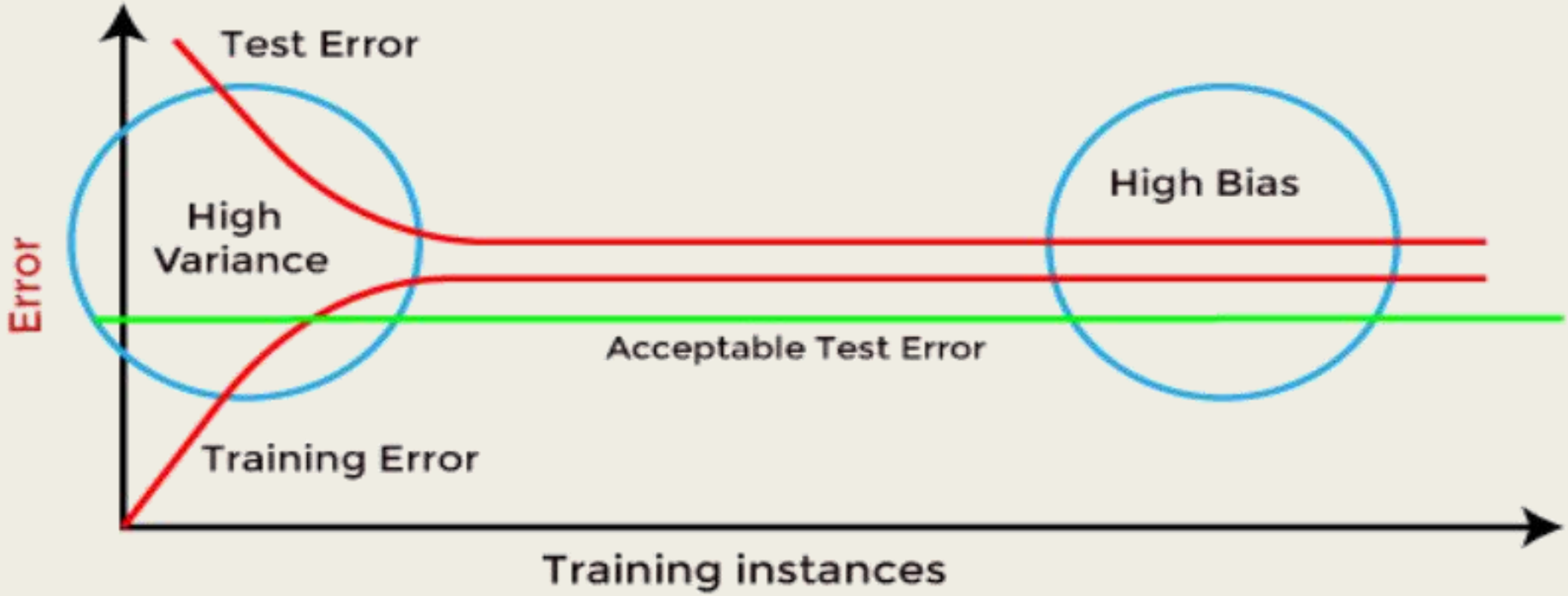
- **There are four possible combinations of bias and variances:-**

- **Low-Bias, Low-Variance:** The combination of low bias and low variance shows an **ideal machine learning model. However, it is not possible practically.**
- **Low-Bias, High-Variance:** With low bias and high variance, model **predictions are inconsistent and accurate on average.** This case occurs when the **model learns with a large number of parameters and hence leads to an overfitting**
- **High-Bias, Low-Variance:** With High bias and low variance, **predictions are consistent but inaccurate on average.** This case occurs when a **model does not learn well with the training dataset or uses few numbers of the parameter. It leads to underfitting problems in the model.**
- **High-Bias, High-Variance:** With high bias and high variance, **predictions are inconsistent and also inaccurate on average.**

How to identify High variance or High Bias?

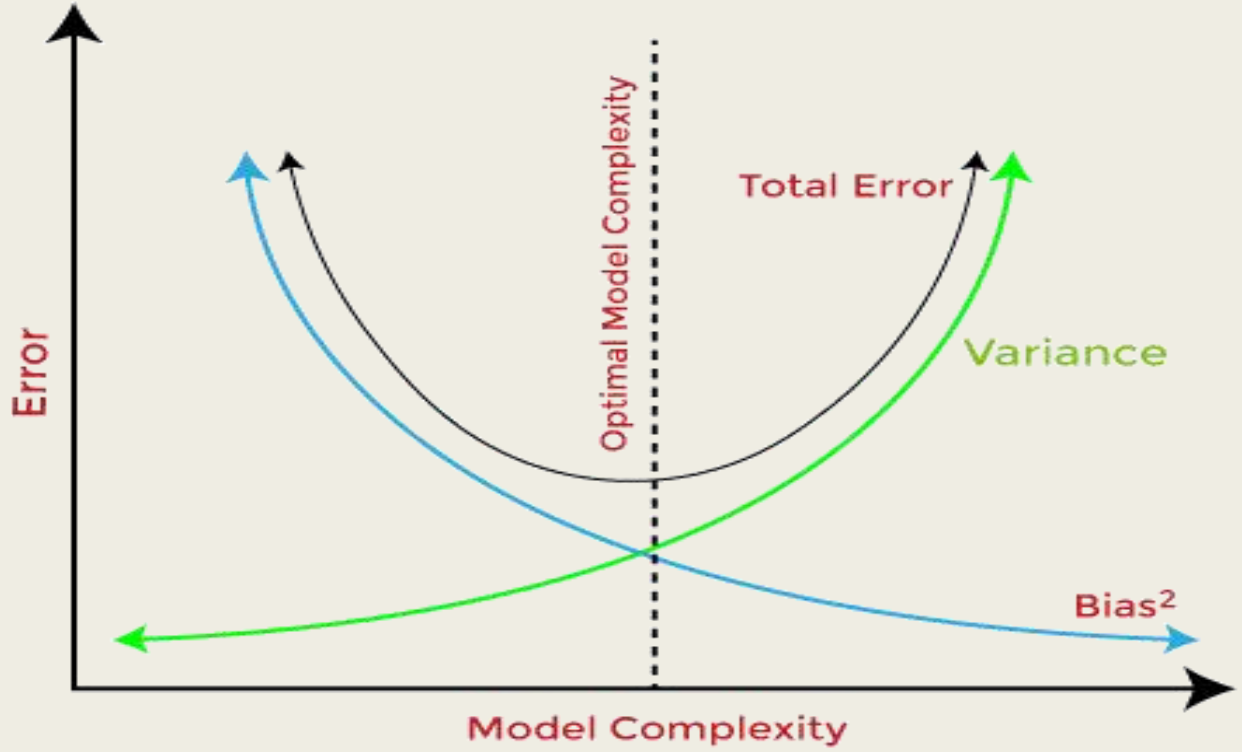
- High variance can be identified if the model has:
- **Low training error and high test error.**

- High Bias can be identified if the model has:
- **High training error and the test error is almost similar to training error.**



Bias-Variance Trade-Off

- While building the machine learning model, it is really important to take care of bias and variance in order to avoid overfitting and underfitting in the model.
- If the model is very simple with fewer parameters, it may have low variance and high bias.
- Whereas, if the model has a large number of parameters, it will have high variance and low bias.
- So, **it is required to make a balance between bias and variance errors, and this balance between the bias error and variance error is known as the Bias-Variance trade-off.**



Bias-Variance Trade-Off

- For an accurate prediction of the model, **algorithms need a low variance and low bias. But this is not possible because bias and variance are related to each other:**
  - **If we decrease the variance, it will increase the bias.**
  - **If we decrease the bias, it will increase the variance.**

- Bias-Variance trade-off is a central issue in supervised learning.
- Ideally, **we need a model that accurately captures the regularities in training data and simultaneously generalizes well with the unseen dataset.**
- Unfortunately, doing this is not possible simultaneously. Because a high variance algorithm may perform well with training data, but it may lead to overfitting to noisy data.
- Whereas, high bias algorithm generates a much simple model that may not even capture important regularities in the data. So, we need to find a sweet spot between bias and variance to make an optimal model.
- **Hence, the Bias-Variance trade-off is about finding the sweet spot to make a balance between bias and variance errors.**

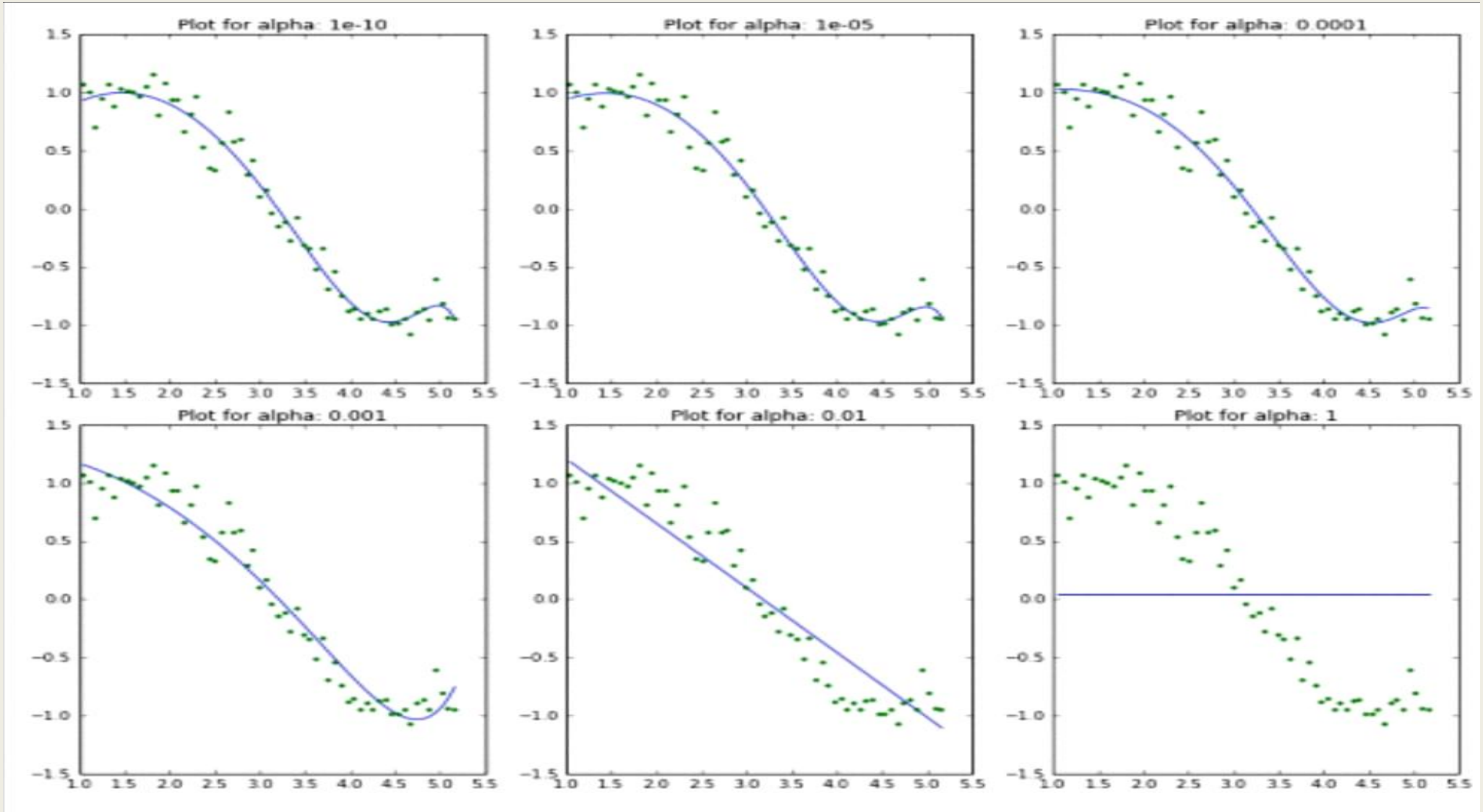
Regularization

- Regularization is a technique used in machine learning and statistical modelling to **prevent overfitting and improve the generalization ability of models.**
- When a model is **overfitting**, it has **learned the training data too well and may not perform well on new, unseen data.**
- Regularization introduces additional constraints or penalties to the model during the training process, aiming to control the complexity of the model and avoid over-reliance on specific features or patterns in the training data.
- By doing so, **regularization helps to strike a balance between fitting the training data well and generalizing it well to new data.**
- The most common regularization techniques used are **L1 regularization (Lasso), L2 regularization (Ridge), and Elastic Net regularization.**
- **L1 regularization adds the sum of the absolute values of the model's coefficients to the loss function,** encouraging sparsity and feature selection.
- **L2 regularization adds the sum of the squared values of the model's coefficients**, which enables smaller but non-zero coefficients.
- Finally, **Elastic Net regularization combines both L1 and L2 regularization.**

L1 - Regularization (Lasso)

- L1 regularization, also known as **Lasso (Least Absolute Shrinkage and Selection Operator)** regularization, **adds the sum of the absolute values of the model's coefficients to the loss function.**
- It encourages sparsity in the model by **shrinking some coefficients to precisely zero.**
- This has the effect of performing **feature selection, as the model can effectively ignore irrelevant or less important features.**
- L1 regularization is particularly useful when dealing with **high-dimensional datasets with desired feature selection.**
- Mathematically, the L1 regularization term can be written as:
- L1 regularization $= \frac{1}{n}\sum_{i=1}^{n} (y_i - \hat{y}_i)^2 + \lambda * \Sigma \mid \beta_i \mid$
- Here, λ **is the regularization parameter that controls the strength of regularization,** β_i **represents the individual model coefficients** and the sum is taken over all coefficients.

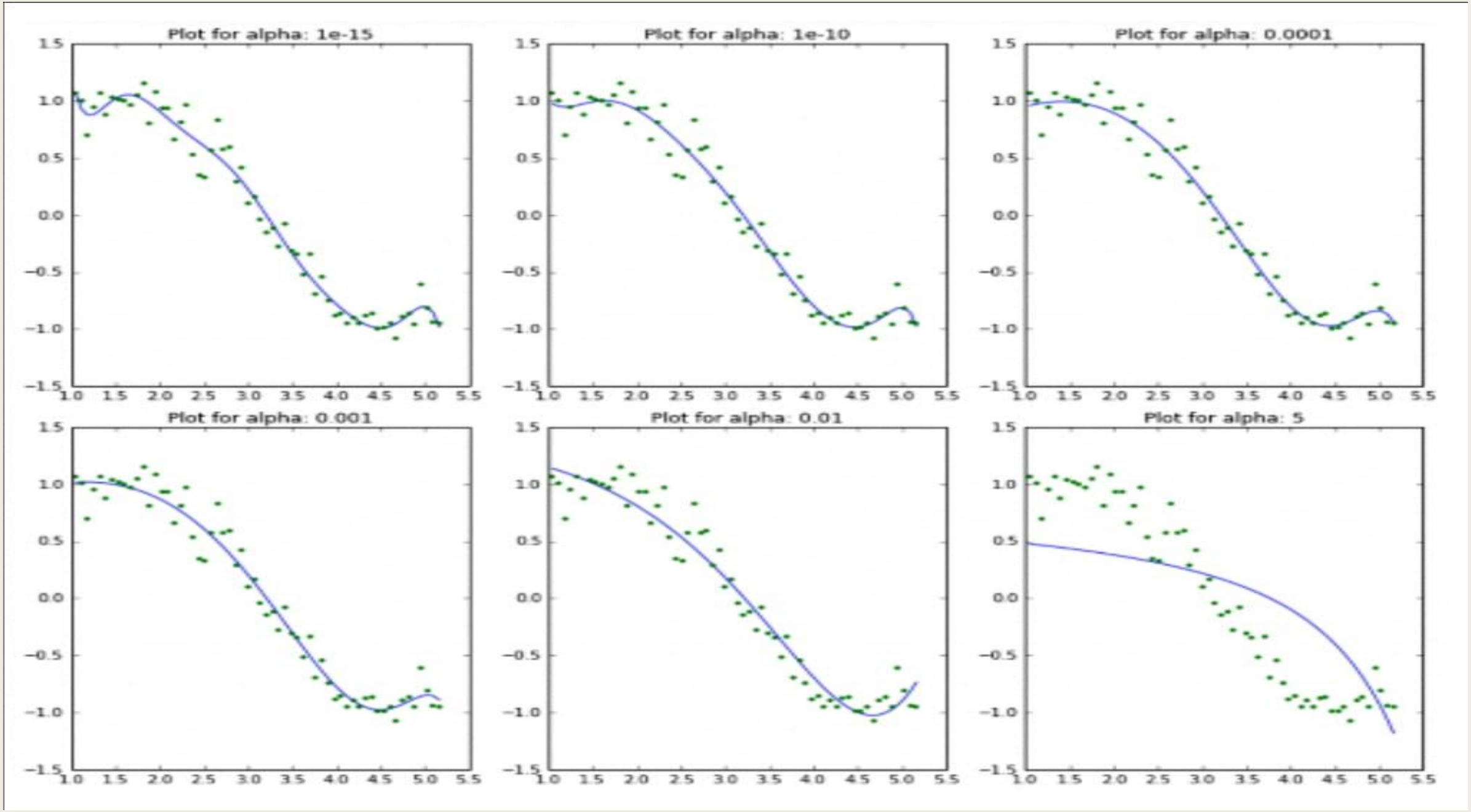
L1 - Regularization (Lasso)



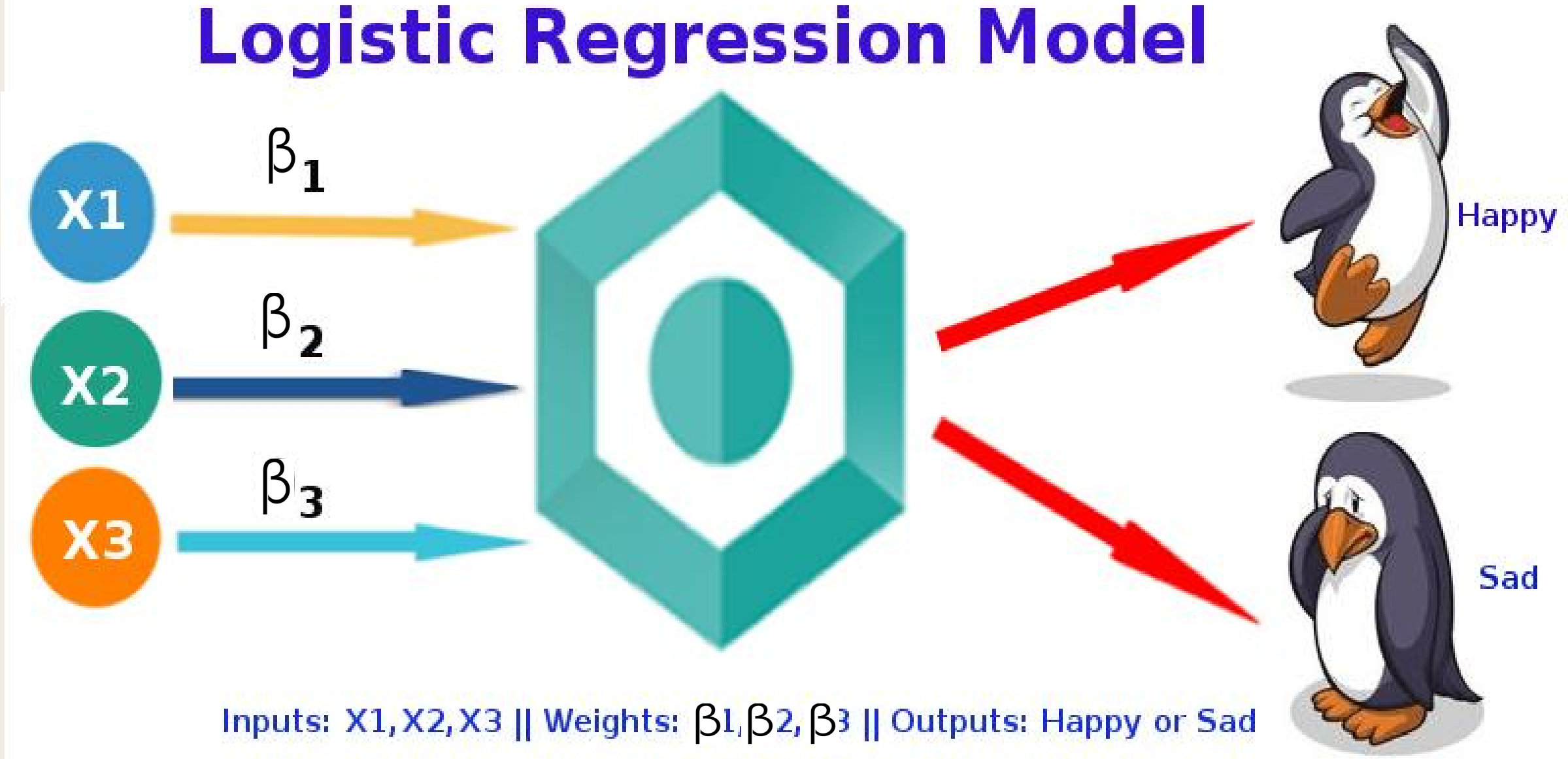
L2 - Regularization (Ridge)

- L2 regularization, also known as **Ridge regularization, adds the sum of the squared values of the model's coefficients to the loss function.**
- Unlike L1 regularization, L2 regularization does not force the coefficients to be exactly zero but instead **encourages them to be small.**
- L2 regularization can **prevent overfitting** by spreading the influence of a single feature across multiple features.
- It is **advantageous when there are correlations between the input features.**
- Mathematically, the L2 regularization term can be written as:
- L2 regularization $= \frac{1}{n}\sum_{i=1}^{n} (y_i - \hat{y}_i)^2 + \lambda * \Sigma (\beta_i{}^2)$
- Similar to L1 regularization, **λ is the regularization parameter, and β_i represents the model coefficients.**
- The sum is taken over all coefficients, and the squares of the coefficients are summed.

L2 - Regularization (Ridge)



Logistic Regression



What is Logistic Regression?

- Logistic regression is the appropriate regression analysis to conduct when the **dependent variable is dichotomous (binary).**

- Like all regression analyses, **logistic regression is a predictive analysis.**

- It is used to describe data and to explain the relationship between **one dependent binary variable and one or more nominal, ordinal, interval or ratio-level independent variables.**

- This regression technique is similar to linear regression and can be used to predict the **Probabilities for classification problems.**

Types of Logistic Regression

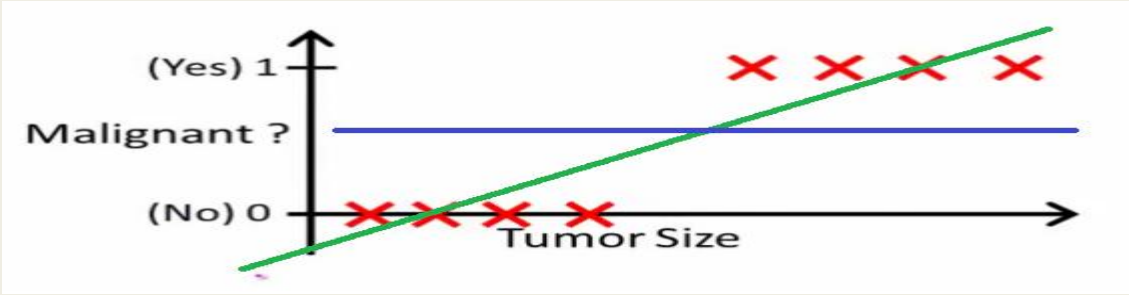
- **Binary logistic regression**
- It is used to **predict the probability of a binary outcome, such as yes or no, true or false, or 0 or 1.** For example, it could be used to predict whether a patient has a disease or not, or whether a loan will be repaid or not.

- **Multinomial logistic regression**
- It is used to **predict the probability of one of three or more possible outcomes,** such as the type of product a customer will buy, the rating a customer will give a product, or the political party a person will vote for.

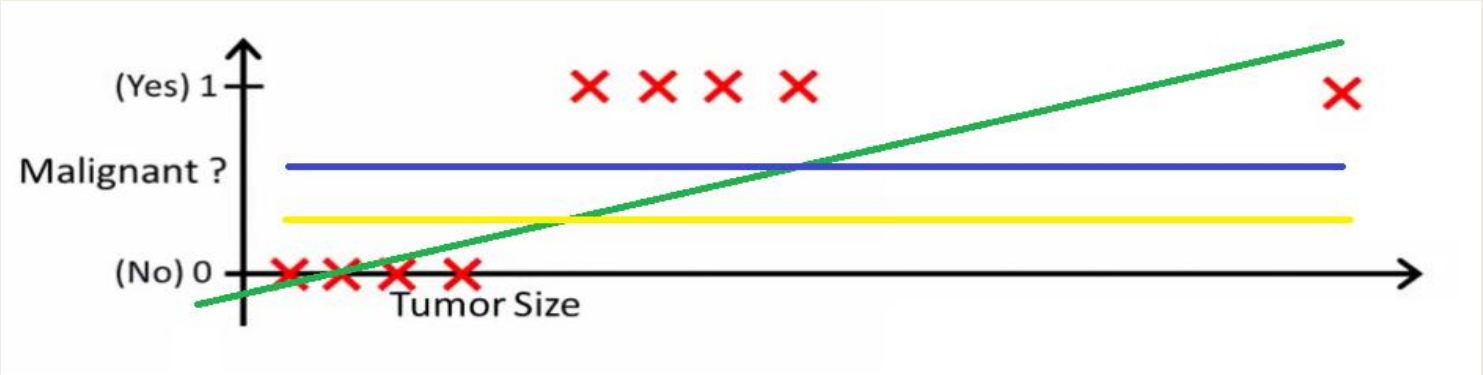
- **Ordinal logistic regression**
- It is used to **predict the probability of an outcome that falls into a predetermined order,** such as the level of customer satisfaction, the severity of a disease, or the stage of cancer.

Why do we use Logistic Regression rather than Linear Regression?

- **Reason 1: - Logistic Regression** is only used when our **dependent variable is binary** and in linear regression this **dependent variable is continuous.**
- **Reason 2: -** If we add an outlier in our dataset, the best fit line in linear regression shifts to fit that point.
- If we use linear regression to find the best fit line which aims at minimizing the distance between the predicted value and actual value, the line will be like this:



- Here the threshold value is 0.5, which means if the value of h(x) is greater than 0.5 then we predict malignant tumor (1) and if it is less than 0.5 then we predict benign tumor (0). Everything seems okay here but now let's change it a bit, we add some outliers in our dataset, now this best fit line will shift to that point. Hence the line will be somewhat like this:



Why do we use Logistic Regression rather than Linear Regression?

- The blue line represents the old threshold and the yellow line represents the new threshold which is maybe 0.2 here.
- To keep our predictions right we had to lower our threshold value.
- Hence we can say that linear regression is prone to outliers.
- Now here if h(x) is greater than 0.2 then only this regression will give correct outputs.
- Another problem with linear regression is that the predicted values may be out of range.
- We know that probability can be between 0 and 1, but if we use linear regression this probability may exceed 1 or go below 0.
- To overcome these problems we use Logistic Regression, **which converts this straight best fit line in linear regression to an S-curve using the sigmoid function, which will always give values between 0 and 1**.

Logistic Function

- Let's start by mentioning the formula of logistic function:

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

- We all know the equation of the best fit line in linear regression is:

$$y = \beta_0 + \beta_1 x$$

- Let's say instead of y we are taking probabilities (P). But there is an issue here, the value of (P) will exceed 1 or go below 0 and we know that range of Probability is (0-1). To overcome this issue we take "odds" of P:

$$\frac{P}{1 - P} = \beta_0 + \beta_1 x$$

- We know that odds can always be positive which means the range will always be $(0, +\infty)$. **Odds are nothing but the ratio of the probability of success and probability of failure.**

Logistic Function

- The problem here is that the range is restricted and we don't want a restricted range because if we do so then our correlation will decrease. By restricting the range we are actually decreasing the number of data points and of course, if we decrease our data points, our correlation will decrease. It is difficult to model a variable that has a restricted range. To control this we take the log of odds which has a range from (-∞,+∞).

$$\log\left(\frac{P}{1-P}\right) = \beta_0 + \beta_1 x$$

- Now we just want a function of P because we want to predict probability. To do so we will multiply by exponent on both sides and then solve for P.

$$\exp[\log(\frac{p}{1-p})] = \exp(\beta_0 + \beta_1 x)$$

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

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

$$p = e^{\left(\beta_0 + \beta_1 x\right)} - pe^{\left(\beta_0 + \beta_1 x\right)}$$

Logistic Function

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

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

$$p[1 + e^{(\beta_0 + \beta_1 x)}] = e^{(\beta_0 + \beta_1 x)}$$

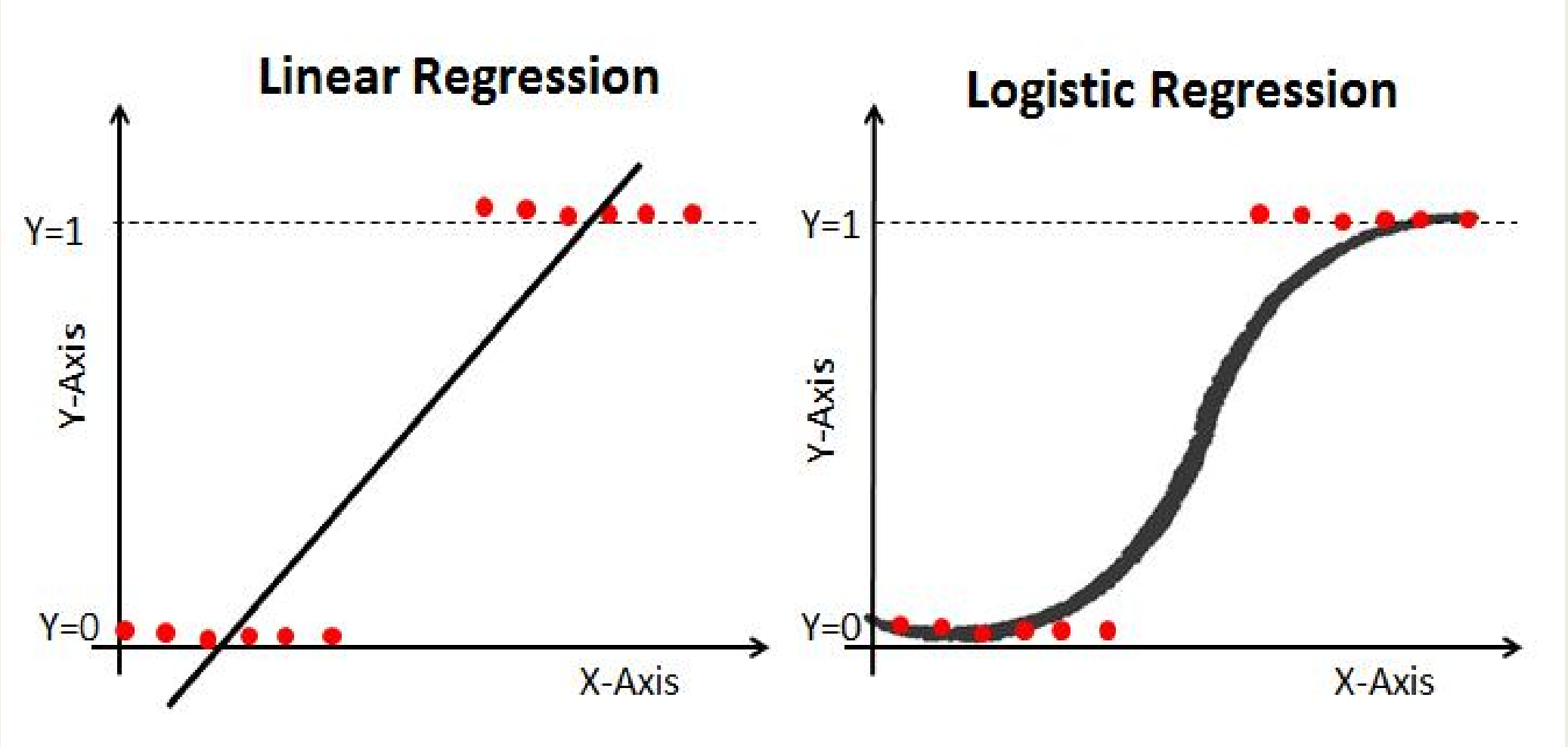
$$p = \frac{e^{(\beta_0 + \beta_1 x)}}{1 + e^{(\beta_0 + \beta_1 x)}}$$

Now dividing by $e^{(\beta_0 + \beta_1 x)}$, *we will get*

$$p = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x)}}$$ *This is our sigmoid function.*

Logistic Function

- Now we have our logistic function, also called a **sigmoid function.**
- The graph of a sigmoid function is as shown below. It squeezes a straight line into an S-curve.



Cost Function in Logistic Regression

- In linear regression, we use the Mean squared error as the cost function which can be shown as:

- In logistic regression Y_i is a non-linear function ($\hat{Y}=1/1+ e^{-z}$). If we use this in the above MSE equation then it will give a non-convex graph with many local minima as shown

Cost Function in Logistic Regression

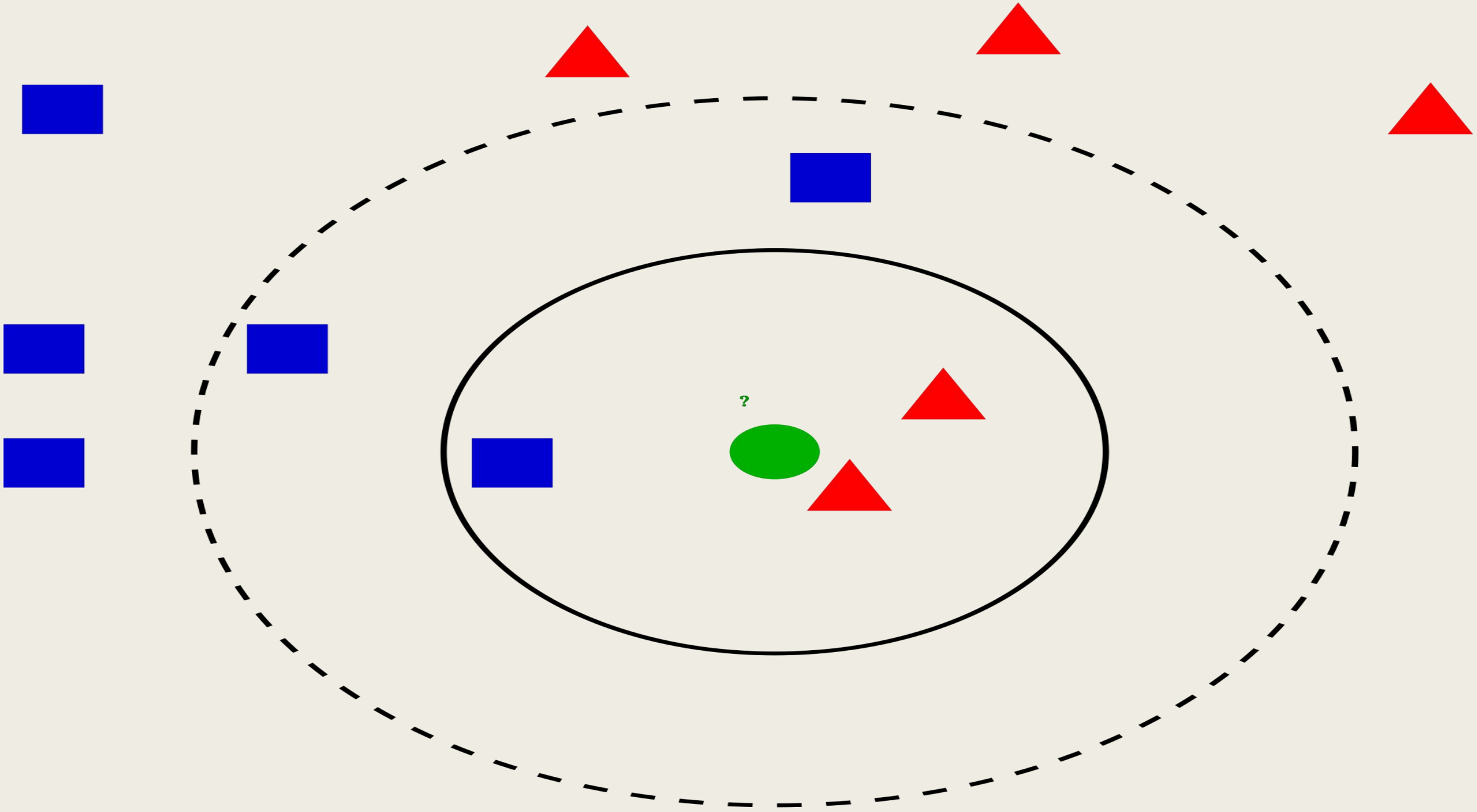
- The problem here is that this cost function will give results with local minima, which is a big problem because then we'll miss out on our global minima and our error will increase.
- In order to solve this problem, we derive a different cost function for logistic regression called **Log loss** which is derived from the maximum likelihood estimation method.

$$\text{Log loss} = \frac{1}{N} \sum_{i=1}^{N} -(y_i * \log(\hat{Y}_i) + (1 - y_i) * \log(1 - \hat{Y}_i))$$

- Above cost function can be solved using gradient descent as below:

$$\beta_{new} = \beta_{old} - \alpha\left[\sigma\left(\beta^T x\right) - y\right].x_j$$

K-Nearest Neighbors



What is KNN (K-Nearest Neighbor) Algorithm?

- The K-Nearest Neighbors (KNN) algorithm is a popular machine learning technique **used for classification and regression tasks.**
- It relies on the idea that **similar data points tend to have similar labels or values.**
- During the **training phase, the KNN algorithm stores the entire training dataset as a reference.**
- When making predictions, **it calculates the distance between the input data point and all the training examples, using a chosen distance metric such as Euclidean distance.**
- Next, the algorithm identifies the **K nearest neighbors to the input data point based on their distances.**
- In the case of **classification, the algorithm assigns the most common class label among the K neighbors as the predicted label for the input data point.**
- For **regression, it calculates the average or weighted average of the target values of the K neighbors to predict the value for the input data point.**
- The KNN algorithm is straightforward and easy to understand, making it a popular choice in various domains.
- However, **its performance can be affected by the choice of K and the distance metric,** so careful parameter tuning is necessary for optimal results.

When Do We Use the KNN Algorithm?

- KNN Algorithm can be used for both classification and regression predictive problems.
- However, it is more widely used in classification problems in the industry. To evaluate any technique, we generally look at 3 important aspects:
- 1. Ease of interpreting output
- 2. Calculation time
- 3. Predictive Power

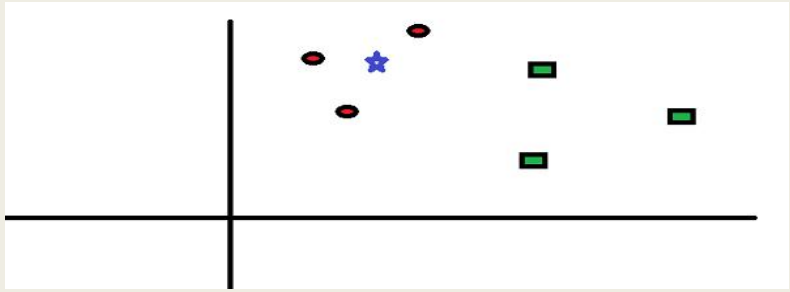
- Let us take a few examples to place KNN in the scale:

	Logistic Regression	CART	Random Forest	KNN
1. Ease to interpret output	2	3	1	3
2. Calculation time	3	2	1	3
3. Predictive Power	2	2	3	2

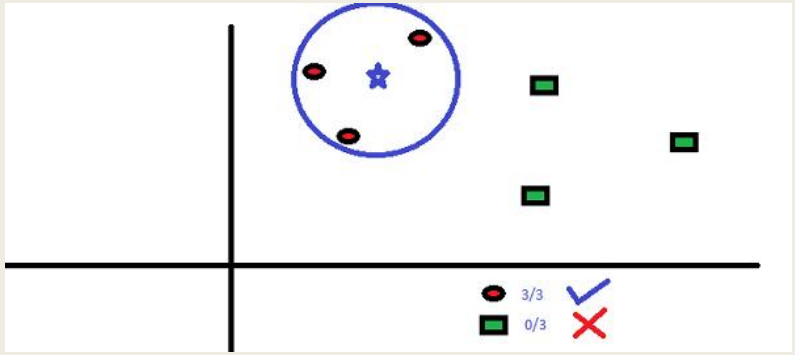
- KNN classifier fairs across all parameters of consideration.
- It is **commonly used for its ease of interpretation and low calculation time.**

How Does the KNN Algorithm Work?

- Let's take a simple case to understand this algorithm. Following is a spread of red circles (RC) and green squares (GS):



- We intend to find out the class of the blue star (BS). BS can either be RC or GS and nothing else. The "K" in KNN algorithm is the nearest neighbor we wish to take the vote from. Let's say K = 3. Hence, we will now make a circle with BS as the center just as big as to enclose only three data points on the plane as shown below:



- The three closest points to BS are all RC. Hence, with a good confidence level, we can say that the BS should belong to the class RC. Here, the choice became obvious as all three votes from the closest neighbor went to RC. The choice of the parameter K is very crucial in this algorithm.

How Does the KNN Algorithm Work?

- In the classification setting, the K-nearest neighbor algorithm essentially boils down to forming a majority vote between the K most similar instances to a given "unseen" observation. Similarity is defined according to a distance metric between two data points. A popular one is the Euclidean distance method:

Euclidean $\sqrt{\sum_{i=1}^{k}(x_i - y_i)^2}$

- Other methods are **Manhattan, Minkowski, and Hamming distance** methods.

- For **categorical variables, the Hamming Distance must be used.**

- K-NN is also a **lazy learner** because it **doesn't learn a discriminative function from the training data but "memorizes" the training dataset** instead.

Few ideas on picking a value for 'K'

- 1) There is **no structured method to find the best value for "K"**. We need to **find out with various values by trial and error** and assuming that training data is unknown.
- 2) **Smaller values for K can be noisy** and will have a higher influence on the result.
- 3) **Larger values of K will have smoother decision boundaries which mean lower variance but increased bias.** Also, **computationally expensive.**
- 4) Another way to choose K is though **cross-validation.** One way to select the cross-validation dataset from the training dataset. **Take the small portion from the training dataset and call it a validation dataset,** and then **use the same to evaluate different possible values of K.** This way we are going to **predict the label for every instance in the validation set using with K equals to 1, K equals to 2, K equals to 3.. and then we look at what value of K gives us the best performance on the validation set and then we can take that value** and use that as the final setting of our algorithm so we are minimizing the validation error .
- 5) In **general, practice**, choosing the value of k is **k = sqrt(N) where N stands for the number of samples in training dataset.**
- 6) Try and **keep the value of k odd in order to avoid confusion between two classes of data.**

Advantages and disadvantages of KNN

❖ **Advantages of KNN**

1. Simple to implement
2. Flexible to feature/distance choices
3. Naturally handles multi-class cases
4. Can do well in practice with enough representative data

❖ **Disadvantages of KNN**

1. Need to determine the value of parameter K (number of nearest neighbors)
2. Computation cost is quite high because we need to compute the distance of each query instance to all training samples.
3. Storage of data
4. Must know we have a meaningful distance function.

Importance of Data Splitting in Machine Learning

- Data splitting divides a dataset into three main subsets: **the training set, used to train the model; the validation set, used to track model parameters and avoid overfitting; and the testing set, used for checking the model's performance on new data.**
- Each subset serves a unique purpose in the iterative process of developing a machine-learning model. The importance of train-test-validation split are as follows:-

- **<u>Model Development and Tuning</u>**
- During the model development phase, the **training set is necessary for exposing the algorithm to various patterns within the data.** The model learns from this subset, adjusting its parameters to minimize errors. **The validation set is important during hyperparameter tracking, helping to optimize the model's configuration.**

- **<u>Overfitting Prevention</u>**
- Overfitting occurs when a model learns the training data well, capturing noise and irrelevant patterns. **The validation set acts as a checkpoint, allowing for the detection of overfitting.** By **evaluating the model's performance on a different dataset, you can adjust model complexity, techniques, or other hyperparameters to prevent overfitting and enhance generalization.**

Importance of Data Splitting in Machine Learning

- **Performance Evaluation**
- The testing set is essential to a machine learning model's performance. After training and validation, the model faces the testing set, which checks real-world scenarios. **A well-performing model on the testing set indicates that it has successfully adapted to new, unseen data.** This step is important for gaining **confidence in deploying the model for real-world applications.**

- **Bias and Variance Assessment**
- It helps in **understanding the bias trade-off. The training set provides information about the model's bias, capturing inherent patterns, while the validation and testing sets help assess variance, indicating the model's sensitivity to fluctuations in the dataset.** Striking the right balance between bias and variance is vital for achieving a model that generalizes well across different datasets.

- **Cross-Validation for Robustness**
- Beyond a simple train-validation-test split, techniques like k-fold cross-validation further enhance the robustness of models. **Cross-validation involves dividing the dataset into k subsets, training the model on k-1 subsets, and validating the remaining one. This process is repeated k times, and the results are averaged.** Cross-validation provides a more comprehensive understanding of a model's performance across different subsets of the data.

Training vs. Validation vs. Test Sets

- **Training Set**
- The training set is the portion of the dataset reserved to **fit the model.**
- In other words, the model sees and learns from the data in the training set to directly improve its parameters.
- To maximize model performance, the training set must be
- (i) large enough to yield meaningful results (but not too large that the model overfits) and
- (ii) representative of the dataset as a whole.
- This will allow the trained model to predict any unseen data that may appear in the future.
- Overfitting occurs when a machine learning model is too specialized and adapted to the training data that it is unable to generalize and make correct predictions on new data.
- As a result, an overfit model will overperform with the training set but underperform when presented with validation sets and test sets.
- **The rough standard for training set is 60-80% of total data.**

Training vs. Validation vs. Test Sets

- **Validation Set**
- The validation set is the set of data used to evaluate and fine-tune a machine learning model during training, helping to assess the model's performance and make adjustments.
- By evaluating a trained model on the validation set, we gain insights into its ability to generalize to unseen data.
- This assessment helps identify potential issues such as overfitting, which can have a significant impact on the model's performance in real-world scenarios.
- The validation set is also **essential for hyperparameter tuning.** Hyperparameters are settings that control the behavior of the model, such as learning rate or regularization strength.
- By experimenting with different hyperparameter values, training the model on the training set, and evaluating its performance with the validation set, we can identify the optimal combination of hyperparameters that yields the best results. This iterative process fine-tunes the model and maximizes its performance.
- **The rough standard for validation set is 10-20% of total data.**

Training vs. Validation vs. Test Sets

- **Test Set**
- The test set is the set of data used to **evaluate the final performance of a trained model.**
- It serves as an unbiased measure of how well the model generalizes to unseen data, assessing its generalization capabilities in real-world scenarios.
- By keeping the test set separate throughout the development process, we **obtain a reliable benchmark of the model's performance.**
- The test dataset also helps gauge the trained model's ability to handle new data. Since it represents unseen data that the model has never encountered before, evaluating the model fit on the test set provides an unbiased metric into its practical applicability.
- This assessment enables us to determine if the trained model has successfully learned relevant patterns and can make accurate predictions beyond the training and validation contexts.
- **The rough standard for test set is 10-20% of total data.**

3 Methods to Split Machine Learning Datasets

- **Random Sampling**
- The most common approach for dividing a dataset is random sampling. As the name suggests, the method involves **shuffling the dataset and randomly assigning samples to training, validation, or test sets according to predetermined ratios.** With class-balanced datasets, random sampling ensures the split is unbiased.
- While random sampling is the best approach for many ML problems, **it is not the correct approach with imbalanced datasets. When the data consists of skewed class proportions, random sampling will almost certainly create a bias in the model.**
- **Stratified Dataset Splitting**
- Stratified dataset splitting is a method **commonly used with imbalanced datasets, where certain classes or categories have significantly fewer instances than others.** In such cases, it is crucial to ensure that the **training, validation, and test sets adequately represent the class distribution to avoid bias in the final model.**
- In stratified splitting, the dataset is divided while **preserving the relative proportions of each class across the splits.** As a result, the training, validation, and test sets contain a representative subset from each class, maintaining the original class distribution. By doing so, **the model can learn to recognize patterns and make predictions for all classes, resulting in a more robust and reliable machine learning algorithm.**

3 Methods to Split Machine Learning Datasets

- **Cross-Validation Splitting**
- Cross-validation sampling is a technique used to split a dataset into training and validation sets for cross-validation purposes.
- It involves **creating multiple subsets of the data, each serving as a training set or validation set during different iterations of the cross-validation process.**
- **K-fold cross-validation and stratified k-fold cross-validation are common techniques.**
- By utilizing these cross-validation sampling techniques, researchers and machine learning practitioners can obtain more reliable and unbiased performance metrics for their machine learning models, enabling them to make better-informed decisions during model development and selection.

3 Mistakes to Avoid When Data Splitting

- **Inadequate Sample Size**
- Insufficient sample size in the training, validation, or test sets can lead to unreliable model performance metrics. **If the training set is too small, the model may not capture enough patterns or generalize well. Similarly, if the validation set or test set is too small, the performance evaluation may lack statistical significance.**
- **Data Leakage**
- Data leakage occurs **when information from the validation set or test set inadvertently leaks into the training set. This can lead to overly optimistic performance metrics and an inflated sense of the final model accuracy.** To prevent data leakage, it is crucial to ensure strict separation between the training set, validation set, and test set, making sure that no information from the evaluation sets are used during model training.
- **Improper Shuffle or Sorting**
- **Incorrectly shuffling or sorting the data before splitting can introduce bias and affect the generalization of the final model.** For example, if the dataset is not shuffled randomly before splitting into training set and validation set, it may introduce biases or patterns that the model can exploit during training. **As a result, the trained model may overfit to those specific patterns and fail to generalize well to new, unseen data.**

Error Analysis and Evaluation metrics

- Evaluation metrics are quantitative measures used to **assess the performance and effectiveness of a statistical or machine learning model.**
- These metrics provide insights into how well the model is performing and help in **comparing different models or algorithms.**
- When evaluating a machine learning model, it is crucial to assess its predictive ability, generalization capability, and overall quality. Evaluation metrics provide objective criteria to measure these aspects.
- The **choice of evaluation metrics depends on the specific problem domain, the type of data, and the desired outcome.**

- **Important definitions for calculating evaluation metrics:**
- **True Positive: You predicted positive, and it's true.**
- **True Negative: You predicted negative, and it's true.**
- **False Positive: (Type 1 Error): You predicted positive, and it's false.**
- **False Negative: (Type 2 Error): You predicted negative, and it's false.**

Accuracy, Precision and Recall (Sensitivity)

- Let a binary classifier classify a collection of test data. Let
- TP = Number of true positives
- TN = Number of true negatives
- FP = Number of false positives
- FN = Number of false negatives
- The **Precision (P)** is defined as

$$P = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

- The **Recall or Sensitivity (R)** is defined as

$$R = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

- The **Accuracy** is defined as

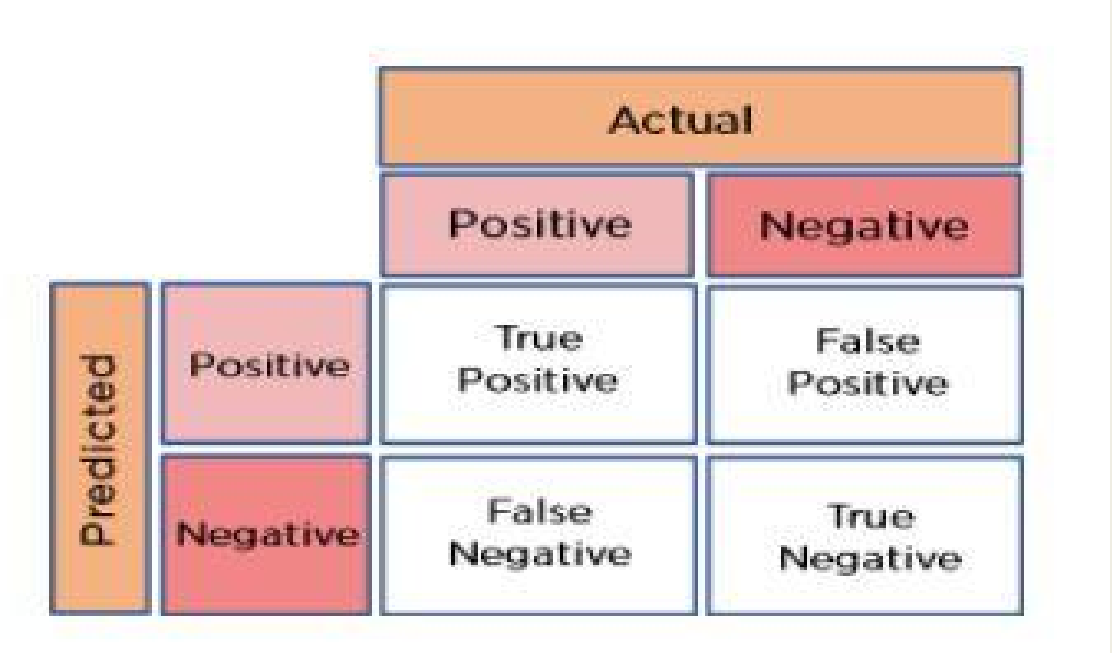
$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

F - measure (F1 Score) and Confusion Matrix

- **F1-Score: -** It is the **harmonic mean of precision and recall values for a classification problem**. The formula for F1-Score is as follows:

$$F_1 = \left(\frac{\text{recall}^{-1} + \text{precision}^{-1}}{2} \right)^{-1} = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}.$$

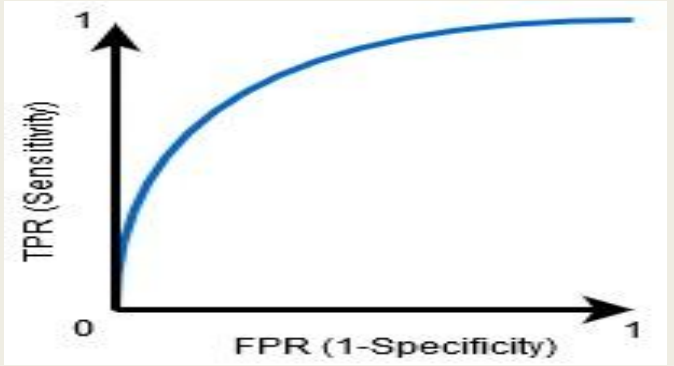
- **Confusion matrix: -** It is an **N x N matrix used for evaluating the performance of a classification model, where N is the total number of target classes.** The matrix compares the actual target values with those predicted by the machine learning model.



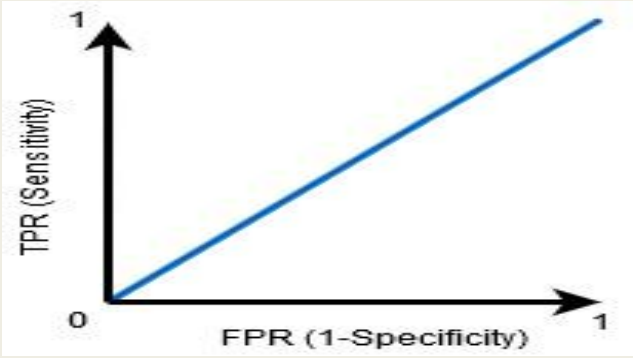
ROC curve

- **Receiver Operator Characteristic (ROC) curve: -** It is an **evaluation metric for binary classification problems.** It is a probability curve that **plots the TPR against FPR at various threshold values and essentially separates the 'signal' from the 'noise.'**
- In other words, it shows the performance of a classification model at all classification thresholds. The **Area Under the Curve (AUC) is the measure of the ability of a binary classifier to distinguish between classes and is used as a summary of the ROC curve.**
- **The higher the AUC, the better the model's performance** at distinguishing between the positive and negative classes.

When AUC = 1, the classifier can correctly distinguish between all the Positive and the Negative class points.



When 0.5<AUC<1, there is a high chance that the classifier will be able to distinguish the positive class values from the negative ones.



When AUC=0.5, then the classifier is not able to distinguish between Positive and Negative class points.

$$\text{TPR /Recall / Sensitivity} = \frac{TP}{TP + FN}$$

$$\text{Specificity} = \frac{TN}{TN + FP}$$

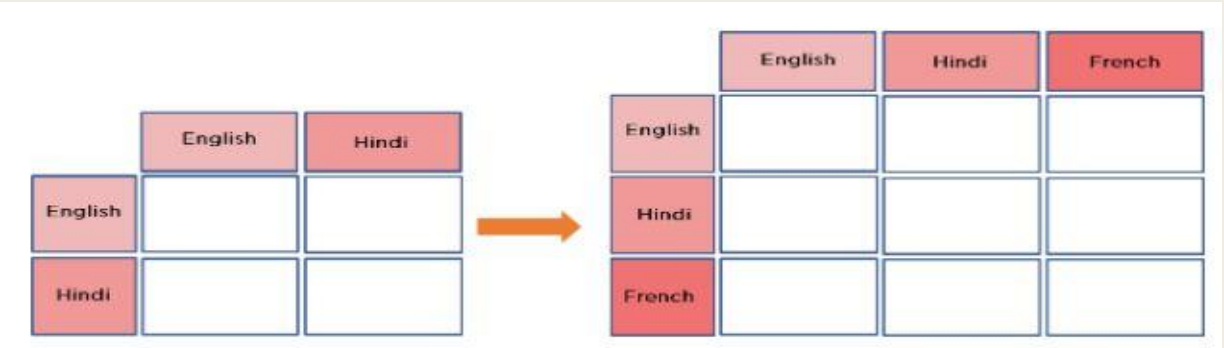
$$\text{FPR} = 1 - \text{Specificity} = \frac{FP}{TN + FP}$$

Solved problem on evaluation metrics

- Q. Consider a confusion matrix made for a classifier that classifies people based on whether they speak English or Spanish.

	English Speaker	Spanish Speaker
English Speaker	86	12
Spanish Speaker	10	79

- Ans:- From the above diagram, we can see that:
- True Positives (TP) = 86, True Negatives (TN) = 79, False Positives (FP) = 12, False Negatives (FN) = 10
- Accuracy = (86 +79) / (86 + 79 + 12 + 10) = 0.8823 = 88.23%
- Precision = 86 / (86 + 12) = 0.8775 = 87.75%
- Recall = 86 / (86 + 10) = 0.8983 = 89.83%
- F1-Score = (2* 0.8775 * 0.8983) / (0.8775 + 0.8983) = 0.8877 = 88.77%



To scale a confusion matrix, increase the number of rows and columns.
All the True Positives will be along the diagonal. The other values will be False Positives or False Negatives.

Different Metrics for comparing ML Algorithms

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall/ Sensitivity/True Positive Rate (TPR)} = \frac{TP}{TP + FN}$$

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

$$\text{Error Rate} = 1 - \text{Accuracy}$$

$$\text{Specificity} = \frac{TN}{TN + FP}$$

$$F - \text{measure/F1} - \text{Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{(\text{Precision} + \text{Recall})}$$

$$\text{False Positive Rate (FPR)} = \frac{FP}{FP + TN}$$

$$\text{True Negative Rate (TNR)} = \frac{TN}{TN + FP}$$

$$\text{AUC ROC} = \frac{1}{2} \times \left(\frac{TP}{TP + FN} + \frac{TN}{TN + FP} \right)$$

Naive Bayes Classifier

- Naïve Bayes algorithm is a supervised learning algorithm, which is **based on Bayes theorem and used for solving classification problems.**
- It is **mainly used in text classification that includes a high-dimensional training dataset.**
- Naïve Bayes Classifier is one of the simple and most effective Classification algorithms which helps in building the fast machine learning models that can make quick predictions.
- It is a **probabilistic classifier, which means it predicts on the basis of the probability of an object.**
- Some popular examples of Naïve Bayes Algorithm are **spam filtration, Sentimental analysis, and classifying articles.**

- **Why is it called Naïve Bayes?**
- **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:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

- 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.

Working of Naïve Bayes' Classifier

- Working of Naïve Bayes' Classifier can be understood with the help of the below example:
- Suppose we have a dataset of weather conditions and corresponding target variable "Play". So using this dataset we need to decide that whether we should play or not on a particular day according to the weather conditions. So to solve this problem, we need to follow the below steps:

- Convert the given dataset into frequency tables.
- Generate Likelihood table by finding the probabilities of given features.
- Now, use Bayes theorem to calculate the posterior probability.

- **Problem: If the weather is sunny, then the Player should play or not?**
- **Solution:** To solve this, first consider the dataset given right side:

	Outlook	Play
0	Rainy	Yes
1	Sunny	Yes
2	Overcast	Yes
3	Overcast	Yes
4	Sunny	No
5	Rainy	Yes
6	Sunny	Yes
7	Overcast	Yes
8	Rainy	No
9	Sunny	No
10	Sunny	Yes
11	Rainy	No
12	Overcast	Yes
13	Overcast	Yes

Working of Naïve Bayes' Classifier

- Frequency table for the Weather Conditions:

Weather	Yes	No
Overcast	5	0
Rainy	2	2
Sunny	3	2
Total	10	4

- Likelihood table weather condition:

Weather	No	Yes	
Overcast	0	5	5/14= 0.35
Rainy	2	2	4/14=0.29
Sunny	2	3	5/14=0.35
All	4/14=0.29	10/14=0.71	

- **Applying Bayes' theorem:**

- P(Yes|Sunny)= P(Sunny|Yes)*P(Yes)/P(Sunny)
- P(Sunny|Yes)= 3/10= 0.3
- P(Sunny)= 0.35
- P(Yes)=0.71
- So P(Yes|Sunny) = 0.3*0.71/0.35= **0.60**

- P(No|Sunny)= P(Sunny|No)*P(No)/P(Sunny)
- P(Sunny|NO)= 2/4=0.5
- P(No)= 0.29
- P(Sunny)= 0.35
- So P(No|Sunny)= 0.5*0.29/0.35 = **0.41**

- **Since P(Yes|Sunny)>P(No|Sunny), Hence on a Sunny day, Player can play the game.**

Advantages, Disadvantages and Applications of Naïve Bayes Classifier

- **Advantages:-**
- Naïve Bayes is **one of the fast and easy ML algorithms** to predict a class of datasets.
- It can be used for **Binary as well as Multi-class Classifications.**
- It performs well in Multi-class predictions as compared to the other Algorithms.
- It is the **most popular choice for text classification problems.**

- **Disadvantages:-**
- Naive Bayes assumes that all features are independent or unrelated, so **it cannot learn the relationship between features.**

- **Applications:-**
- It is used for **Credit Scoring.**
- It is used in **medical data classification.**
- It can be used in real-time predictions because Naïve Bayes Classifier is an eager learner.
- It is used in **Text classification such as Spam filtering and Sentiment analysis.**

Types of Naïve Bayes Model

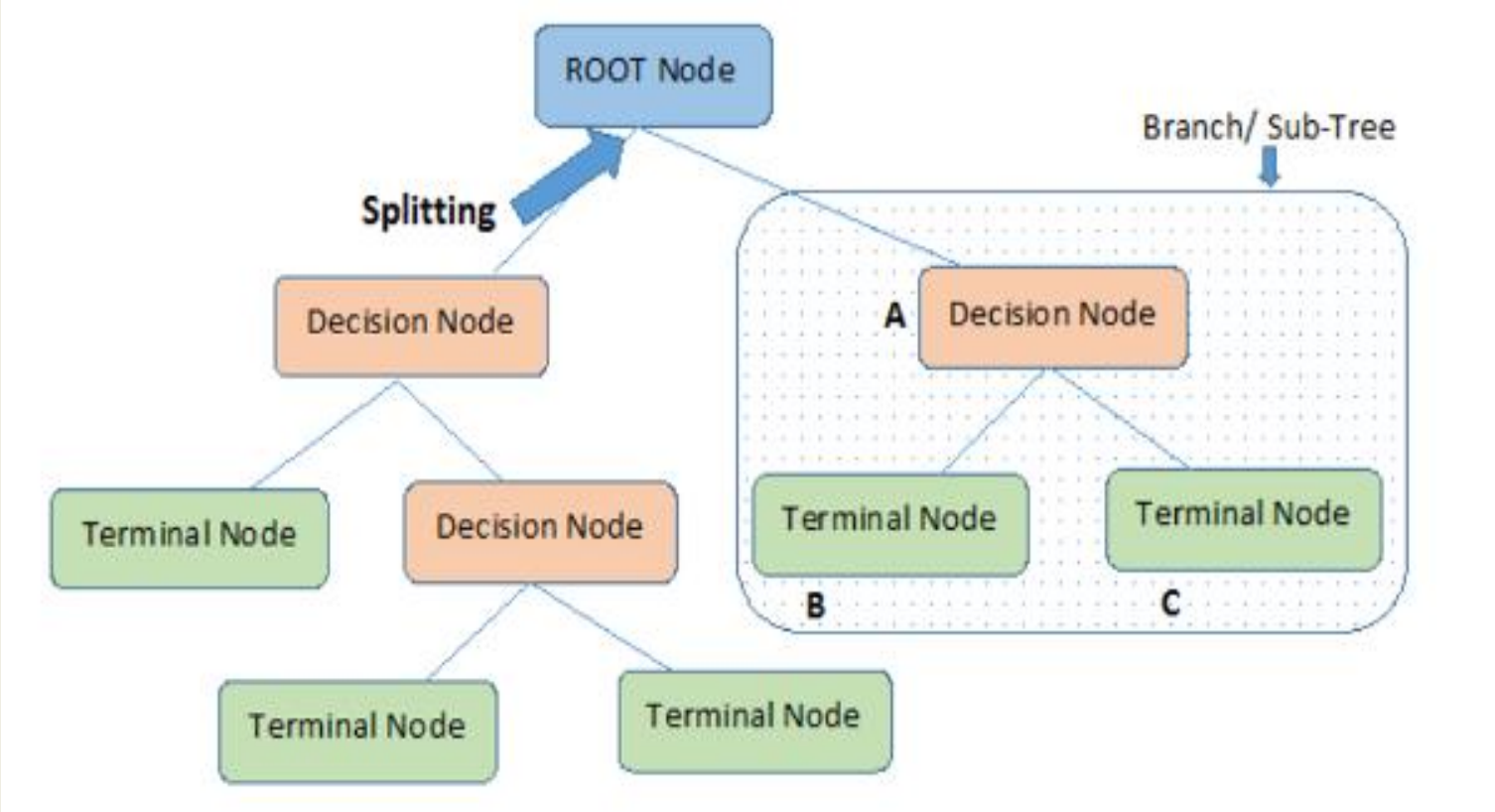
- There are **three types of Naive Bayes Model**, which are given below:
- **Gaussian:** The Gaussian model assumes that **features follow a normal distribution.** This means if predictors take continuous values instead of discrete, then the model assumes that these values are sampled from the Gaussian distribution.
- **Multinomial:** The Multinomial Naïve Bayes classifier is used when the **data is multinomial distributed. It is primarily used for document classification problems**, it means a particular document belongs to which category such as Sports, Politics, education, etc. **The classifier uses the frequency of words for the predictors.**
- **Bernoulli:** The Bernoulli classifier works **similar to the Multinomial classifier, but the predictor variables are the independent Booleans variables.** Such as if a particular word is present or not in a document. This model is **also famous for document classification tasks.**

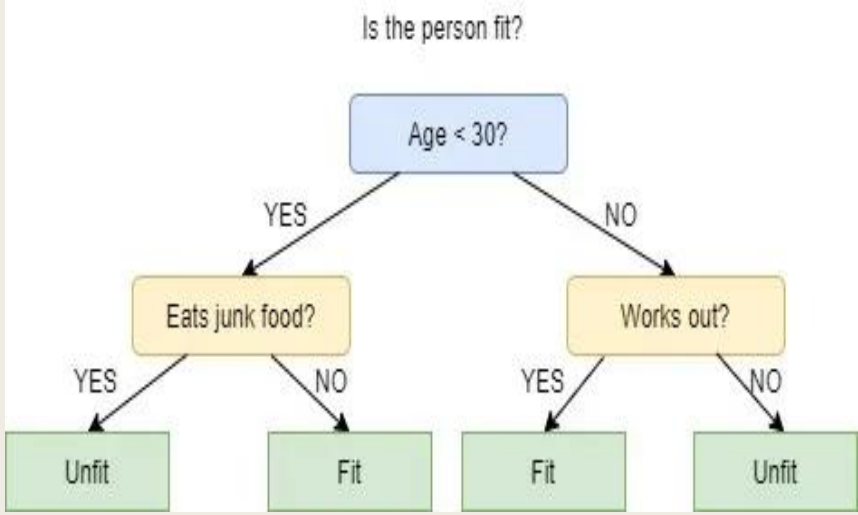
Decision Tree

- 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 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.**
- It is called a decision tree because, similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure.

- **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.

Decision Tree Example





Decision Tree Terminologies

- **Root Node:** Root node is from where the **decision tree starts.** It represents the entire dataset, which further gets divided into two or more homogeneous sets.
- **Leaf Node:** Leaf nodes are the **final output node,** and the tree cannot be segregated further after getting a leaf node.
- **Splitting:** Splitting is the **process of dividing the decision node/root node into sub-nodes according to the given conditions.**
- **Branch/Sub Tree:** A tree formed by **splitting the tree.**
- **Pruning:** Pruning is the process of **removing the unwanted branches** from the tree.
- **Parent/Child node:** A node that is **divided into sub-nodes is known as a parent node, and the sub-nodes emerging from it are referred to as child nodes.** The **parent node represents a decision or condition, while the child nodes represent the potential outcomes or further decisions based on that condition.**

ID3 Algorithm

- ID3 stands for **Iterative Dichotomiser 3** and is named such because the algorithm **iteratively (repeatedly) dichotomizes(divides) features into two or more groups** at each step.

- **Invented by Ross Quinlan,** ID3 uses a top-down greedy approach to build a decision tree.

- In simple words, the **top-down approach means that we start building the tree from the top and the greedy approach means that at each iteration we select the best feature at the present moment to create a node.**

- **Most generally ID3 is only used for classification** problems with nominal features only.

Metrics in ID3

- **ID3 algorithm selects the best feature at each step while building a Decision tree.**

- So the answer to the question: ‘How does ID3 select the best feature?’ is that **ID3 uses Information Gain or just Gain to find the best feature.**

- **Information Gain calculates the reduction in the entropy and measures how well a given feature separates or classifies the target classes.**

- The **feature with the highest Information Gain is selected** as the best one.

- In simple words, **Entropy is the measure of disorder and the Entropy of a dataset is the measure of disorder in the target feature** of the dataset.

- In the case of binary classification (where the target column has only two types of classes) entropy is 0 if all values in the target column are homogenous(similar) and will be 1 if the target column has equal number values for both the classes.

Metrics in ID3

- We denote our dataset as S, entropy is calculated as:

$$\textbf{Entropy(S)} = -\sum p_i * \log_2(p_i) \; ; \; i = 1 \text{ to } n$$

- where,
- n is the total number of classes in the target column (in our case n = 2 i.e YES and NO)
- p_i is the probability of class 'i' or the ratio of "number of rows with class i in the target column" to the "total number of rows" in the dataset.

- Information Gain for a feature column A is calculated as:

$$\textbf{IG(S, A)} = \textbf{Entropy(S)} - \sum((|S_v| / |S|) * \textbf{Entropy}(S_v))$$

- where S_v is the set of rows in S for which the feature column A has value v, $|S_v|$ is the number of rows in S_v and likewise |S| is the number of rows in S.

ID3 Steps

I. Calculate the Information Gain of each feature.

II. Considering that all rows don't belong to the same class, split the dataset S into subsets using the feature for which the Information Gain is maximum.

III. Make a decision tree node using the feature with the maximum Information gain.

IV. If all or most of the rows belong to the same class, make the current node as a leaf node with the class as its label.

V. Repeat for the remaining features until we run out of all features, or the decision tree has all leaf nodes.

Example Dataset

ID	Fever	Cough	Breathing issues	Infected
1	NO	NO	NO	NO
2	YES	YES	YES	YES
3	YES	YES	NO	NO
4	YES	NO	YES	YES
5	YES	YES	YES	YES
6	NO	YES	NO	NO
7	YES	NO	YES	YES
8	YES	NO	YES	YES
9	NO	YES	YES	YES
10	YES	YES	NO	YES
11	NO	YES	NO	NO
12	NO	YES	YES	YES
13	NO	YES	YES	NO
14	YES	YES	NO	NO

Implementation of ID3 on Dataset

- The first step is to find the best feature i.e. the one that has the maximum Information Gain(IG).
- We'll calculate the IG for each of the features now, but for that, we first need to calculate the entropy of S.
- From the total of 14 rows in our dataset S, there are 8 rows with the target value YES and 6 rows with the target value NO. The entropy of S is calculated as:

$$\textbf{Entropy(S)} = -(8/14) * \log_2(8/14) - (6/14) * \log_2(6/14) = 0.99$$

- **Note: If all the values in our target column are same the entropy will be zero (meaning that it has no or zero randomness).**
- We now calculate the Information Gain for each feature.

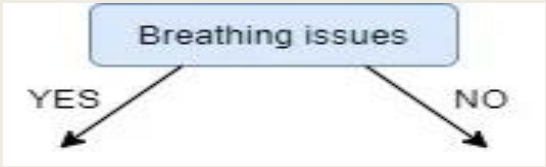
Implementation of ID3 on Dataset

- **IG calculation for Fever:**
- In this(Fever) feature there are 8 rows having value YES and 6 rows having value NO.
- In the 8 rows with YES for Fever, there are 6 rows having target value YES and 2 rows having target value NO.
- In the 6 rows with NO, there are 2 rows having target value YES and 4 rows having target value NO.
- $|S| = 14$
- For v = YES, $|S_v| = 8$
- **$Entropy(S_v) = - (6/8) * \log_2(6/8) - (2/8) * \log_2(2/8) = 0.81$**
- For v = NO, $|S_v| = 6$
- **$Entropy(S_v) = - (2/6) * \log_2(2/6) - (4/6) * \log_2(4/6) = 0.91$**

- **# Expanding the summation in the IG formula:**
- **$IG(S, Fever) = Entropy(S) - (|S_{YES}| / |S|) * Entropy(S_{YES}) - (|S_{NO}| / |S|) * Entropy(S_{NO})$**
- **$\therefore IG(S, Fever) = 0.99 - (8/14) * 0.81 - (6/14) * 0.91 = 0.13$**

Implementation of ID3 on Dataset

- Next, we calculate the IG for the features “Cough” and “Breathing issues”.
- **IG(S, Cough) = 0.04**
- **IG(S, BreathingIssues) = 0.40**
- Since the feature Breathing issues have the highest Information Gain it is used to create the root node.
- Hence, after this initial step our tree looks like this:

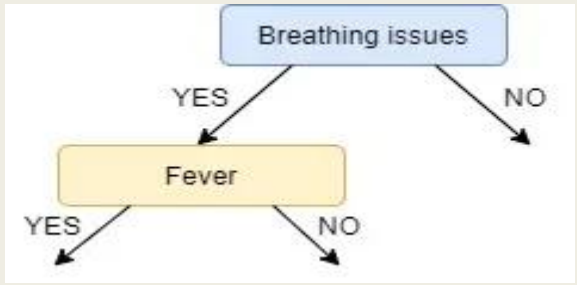


- Next, from the remaining two unused features, namely, Fever and Cough, we decide which one is the best for the left branch of Breathing Issues.
- Since the left branch of Breathing Issues denotes YES, we will work with the subset of the original data i.e the set of rows having YES as the value in the Breathing Issues column. These 8 rows are shown below:

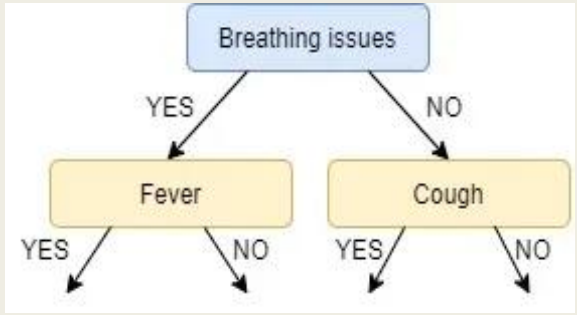
Fever	Cough	Breathing issues	Infected
YES	YES	YES	YES
YES	NO	YES	YES
YES	YES	YES	YES
YES	NO	YES	YES
YES	NO	YES	YES
NO	YES	YES	YES
NO	YES	YES	YES
NO	YES	YES	NO

Implementation of ID3 on Dataset

- Next, we calculate the IG for the features Fever and Cough using the subset S_{BY} (Set Breathing Issues Yes)
- **Note:** For IG calculation the Entropy will be calculated from the subset S_{BY} and not the original dataset S.
- **$IG(S_{BY}, Fever) = 0.20$**
- **$IG(S_{BY}, Cough) = 0.09$**
- **IG of Fever is greater than that of Cough, so we select Fever as the left branch** of Breathing Issues.
- Our tree now looks like this:



- Next, we find the feature with the maximum IG for the right branch of Breathing Issues. But, since there is **only one unused feature left we have no other choice but to make it the right branch of the root node.**
- So our tree now looks like this:



- There are no more unused features, so we stop here and jump to the final step of creating the leaf nodes.

Implementation of ID3 on Dataset

- For the left leaf node of Fever, we see the subset of rows from the original data set that has Breathing Issues and Fever both values as YES.

Fever	Cough	Breathing issues	Infected
YES	YES	YES	YES
YES	NO	YES	YES
YES	YES	YES	YES
YES	NO	YES	YES
YES	NO	YES	YES

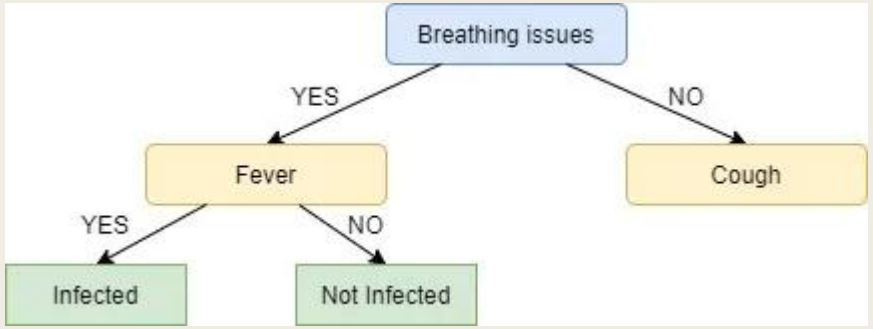
- Since **all the values in the target column are YES, we label the left leaf node as YES, but to make it more logical we label it Infected.**
- Similarly, for the right node of Fever we see the subset of rows from the original data set that have Breathing Issues value as YES and Fever as NO.

Fever	Cough	Breathing issues	Infected
NO	YES	YES	YES
NO	YES	YES	NO
NO	YES	YES	NO

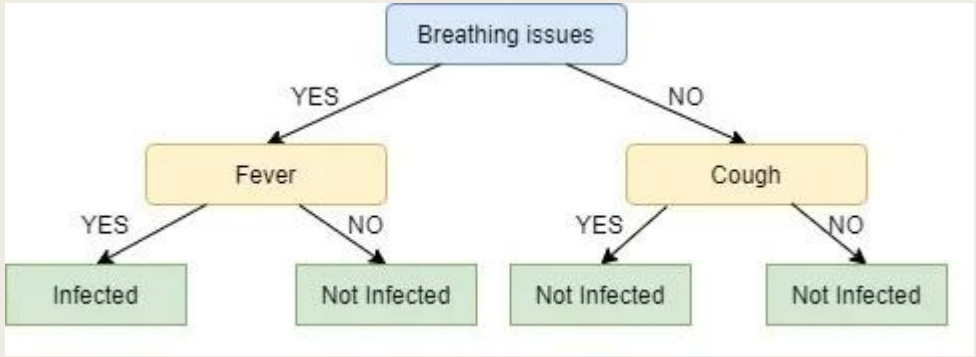
- Here not all but **most of the values are NO, hence NO or Not Infected** becomes our right leaf node.

Implementation of ID3 on Dataset

- Our tree, now, looks like this:



- We repeat the same process for the node Cough, however here **both left and right leaves turn out to be the same i.e. NO or Not Infected** as shown below:



- The **right node of Breathing issues is as good as just a leaf node with class 'Not infected'**. This is one of the Drawbacks of ID3, it doesn't do pruning.
- **Pruning is a mechanism that reduces the size and complexity** of a Decision tree by **removing unnecessary nodes.**
- Another drawback of ID3 is overfitting or high variance i.e. it learns the dataset it used so well that it fails to generalize on new data which **can be resolved using the Random Forest algorithm.**

Advantages and Disadvantages of the Decision Tree

❖ **Advantages of the Decision Tree**

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

❖ **Disadvantages of the Decision Tree**

1. The decision tree **contains lots of layers, which makes it complex.**
2. It may have an **overfitting issue**, which **can be resolved using the Random Forest algorithm.**
3. For more class labels, the **computational complexity of the decision tree may increase.**
4. It may **contain some unnecessary nodes** which can be **solved by prunning.**

KNN Previous Year Questions

- 1.Using KNN algorithm and the given data set, predict the label of the test data point (3,7), where K=3 and Euclidean distance.

X	Y	Label
7	7	1
7	4	1
3	4	2
1	4	2

- Ans :- To predict the label of the test data point, we have to calculate the distance from the test data point to other data in the data set using the Euclidean distance formula.

- **Euclidean distance formula :** $\sqrt{(X_2-X_1)^2+(Y_2-Y_1)^2}$
- Where:
- X_2 = Test data point's X value (3).
- X_1= Existing data's X value.
- Y_2 = Test data point's Y value (7).
- Y_1 = Existing data's Y value.

KNN Previous Year Questions

- **Distance #1**
- For the first row, d1:
- $d1 = \sqrt{(3 - 7)^2 + (7 - 7)^2}$
- $= \sqrt{16 + 0}$
- $= \sqrt{16}$
- $= 4$

- **Distance #2**
- For the second row, d2:
- $d2 = \sqrt{(3 - 7)^2 + (7 - 4)^2}$
- $= \sqrt{16 + 9}$
- $= \sqrt{25}$
- $= 5$

KNN Previous Year Questions

- **Distance #3**
- For the third row, d3:
- $d3 = \sqrt{(3 - 3)^2 + (7 - 4)^2}$
- $= \sqrt{0 + 9}$
- $= \sqrt{9}$
- $= 3$

- **Distance #4**
- For the fourth row, d4:
- $d4 = \sqrt{(3 - 1)^2 + (7 - 4)^2}$
- $= \sqrt{4 + 9}$
- $= \sqrt{13}$
- $= 3.6$

KNN Previous Year Questions

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

X	Y	Label	Distance
7	7	1	4
7	4	1	5
3	4	2	3
1	4	2	3.6

- As we can see, the **majority class within the 3 nearest neighbors to the test data point is label 2. Therefore, we'll classify the test data point as label 2.**

- 2.Perform KNN classification on the following training instances each having two attributes (X1, and X2). Compute the class label for the test instance t1 = (3,7), with K=3 and Euclidean distance.

Training instances	X1	X2	output
I1	7	7	0
I2	7	4	0
I3	3	4	1
I4	1	4	1

- Ans :- Same as solution of question 1.

KNN Previous Year Questions

- 3.Explain the merits and demerits of Cosine distance measure. Find the cosine distance between (1,6,1,0) and (0,1,2,2).
- Ans:- Merits of cosine distance measure:-
- a) Low storage cost,
- b) High computational efficiency
- c) Good retrieval performance.
- Demerits of cosine distance measure:-
- I. It yields the same value regardless of the size of the vectors being compared, as long as the angle between them is the same.
- II. It does not take into account the semantic meanings of words or phrases, even when using techniques like Natural Language Processing.
- **Formula for cosine distance:-**

$$D_c = \frac{p.q}{\|p\|\|q\|}$$

- Where,
- p . q = product (dot) of the vectors ‘p’ and ‘q’.
- ||p|| and ||q|| = length (magnitude) of the two vectors ‘p’ and ‘q’.

KNN Previous Year Questions

- Cosine distance between (1,6,1,0) and (0,1,2,2) is

$$D_c = \frac{1\times 0 + 6\times 1 + 1\times 2 + 0\times 2}{\sqrt{1^2+6^2+1^2+0^2}\times\sqrt{0^2+1^2+2^2+2^2}} = \frac{8}{\sqrt{38}\times\sqrt{9}} = \frac{8}{18.49} = 0.43$$

- 4.Under what conditions Minkowski distance is same as Euclidean distance?
- Ans:- When the **order in the formula of Minkowski distance is 2, it is same as Euclidean Distance.**

- 5.Suppose you have a dataset of animals and you want to use KNN to predict whether a new animal is a cat or a dog based on its weight and height. You have the following dataset.

Animal	Weight (Kg)	Height (Cm)	Species
1	4	35	Cat
2	6	40	Dog
3	3	25	Cat
4	7	45	Dog
5	5	30	Cat
6	8	50	Dog
7	2	20	Cat
8	5	35	Dog

- Predict the species of a new animal that weights 4Kg and is 30 Cm tall.
- Ans: To predict the species of a new animal, we have to calculate the distance of the features of new animal from the features of other animal in the data set using the Euclidean distance formula.

KNN Previous Year Questions

- **Euclidean distance formula : $\sqrt{(X_2-X_1)^2+(Y_2-Y_1)^2}$**
- Where:
- X_2 = New animal's weight (4).
- X_1= Existing animal's weight.
- Y_2 = New animal's height (30).
- Y_1 = Existing animal's height.

- **Distance #1**
- For the first row, d1:
- $d1 = \sqrt{(4 - 4)^2 + (30 - 35)^2}$
- $= \sqrt{0 + 25}$
- $= \sqrt{25}$
- $= 5$

KNN Previous Year Questions

- **Distance #2**
- For the second row, d2:
- $d2 = \sqrt{(4 - 6)^2 + (30 - 40)^2}$
- $= \sqrt{4 + 100}$
- $= \sqrt{104}$
- $= 10.2$

- **Distance #3**
- For the third row, d3:
- $d3 = \sqrt{(4 - 3)^2 + (30 - 25)^2}$
- $= \sqrt{1 + 25}$
- $= \sqrt{26}$
- $= 5.1$

KNN Previous Year Questions

- **Distance #4**
- For the fourth row, d4:
- $d4 = \sqrt{(4 - 7)^2 + (30 - 45)^2}$
- $= \sqrt{9 + 225}$
- $= \sqrt{234}$
- $= 15.3$

- **Distance #5**
- For the first row, d5:
- $d1 = \sqrt{(4 - 5)^2 + (30 - 30)^2}$
- $= \sqrt{1 + 0}$
- $= \sqrt{1}$
- $= 1$

KNN Previous Year Questions

- **Distance #6**
- For the second row, d2:
- $d2 = \sqrt{(4 - 8)^2 + (30 - 50)^2}$
- $= \sqrt{16 + 400}$
- $= \sqrt{416}$
- $= 20.4$

- **Distance #7**
- For the third row, d3:
- $d3 = \sqrt{(4 - 2)^2 + (30 - 20)^2}$
- $= \sqrt{4 + 100}$
- $= \sqrt{104}$
- $= 10.2$

KNN Previous Year Questions

- **Distance #8**
- For the fourth row, d4:
- $d4 = \sqrt{(4 - 5)^2 + (30 - 35)^2}$
- $= \sqrt{1 + 25}$
- $= \sqrt{26}$
- $= 5.1$
- Here's what the table will look like after all the distances have been calculated:

Animal	Weight (Kg)	Height (Cm)	Species	Distance
1	4	35	Cat	5
2	6	40	Dog	10.2
3	3	25	Cat	5.1
4	7	45	Dog	15.3
5	5	30	Cat	1
6	8	50	Dog	20.4
7	2	20	Cat	10.2
8	5	35	Dog	5.1

- As we can see, the **majority class within the 3 nearest neighbors to the new animal is cat. Therefore, we'll classify the new animal as cat.**

KNN Previous Year Questions

- 6.Evaluate the Euclidean distance, Manhattan distance, Minkowski distance and the Cosine distance for the following two points. P1(1,0,2,5,3) and P2(2,1,0,3,-1).
- Ans:- **Euclidean distance formula:-**

$$D_e = \sqrt{\sum_{i=1}^{n} (p_i - q_i)^2}$$

- Euclidean distance between P1(1,0,2,5,3) and P2(2,1,0,3,-1) is

$$D_e = \sqrt{(1-2)^2 + (0-1)^2 + (2-0)^2 + (5-3)^2 + (3-(-1))^2} = \sqrt{26} = 5.1$$

- **Manhattan distance formula:-**

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

- Manhattan distance between P1(1,0,2,5,3) and P2(2,1,0,3,-1) is

$$D_m = |1-2| + |0-1| + |2-0| + |5-3| + |3-(-1)| = 10$$

- **Minkowski distance formula:-**

$$D_{Minkowski} = \left(\sum_{i=1}^{n} |p_i - q_i|^k \right)^{\frac{1}{k}}$$

KNN Previous Year Questions

- Minkowski Distance is the **generalized form of Euclidean and Manhattan Distance. Here, k represents the order of the norm. When the order(k) is 1, it will represent Manhattan Distance and when the order in the above formula is 2, it will represent Euclidean Distance.** So Minkowski distance is10 when k=1 and Minkowski distance is 5.1 when k=2.

- **Cosine distance formula:-**

$$D_c = \frac{p.q}{\|p\|\|q\|}$$

- Where,
- p . q = product (dot) of the vectors ‘p’ and ‘q’.
- ||p|| and ||q|| = length (magnitude) of the two vectors ‘p’ and ‘q’.
- Cosine distance between P1(1,0,2,5,3) and P2(2,1,0,3,-1) is

$$D_c = \frac{1 \times 2 + 0 \times 1 + 2 \times 0 + 5 \times 3 + 3 \times -1}{\sqrt{1^2 + 0^2 + 2^2 + 5^2 + 3^2} \times \sqrt{2^2 + 1^2 + 0^2 + 3^2 + -1^2}} = \frac{14}{\sqrt{39} \times \sqrt{15}}$$

$$= \frac{14}{24.18} = 0.58$$

- 7.Why KNN is called as Lazy Learner algorithm?
- Ans:- KNN is called a lazy learner algorithm because **it does not learn from the training set immediately, instead it stores the dataset, and at the time of classification, it performs an action** on the dataset.

KNN Previous Year Questions

- 8.Perform KNN classification on the following dataset and predict the class for (height = 170, weight = 57), with K=5 using Euclidean distance.

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

- 9.Using KNN algorithm and the given data set, predict the label of the test data point (8,5), where K=3 and Euclidean distance.

X	Y	Label
4.2	3.8	0
6.5	7.7	1
7.3	8.6	1
5.7	5.9	0
8.0	8.1	1
10.0	6.5	1

- 10.The Manhattan distance between two points (10, 10) and (30, 30) is ?

KNN Previous Year Questions

- 11.Using KNN algorithm and the given data set, predict the class label of the test data point (16,8), where K=3 and Euclidean distance.

X	Y	Label
10	5	0
6.5	11	1
7	15	1
12	5	0
8	10	1
15	8	0

- Note:- Question 8-11 is homework.

Naive Bayes Previous Year Question

- 1.Consider the following data set and predict the class of new instance X = {Slow, Rarely, No} using Naive Bayes classification algorithm.

Sl. No	Swim	Fly	Crawl	Class
1.	Fast	No	No	Fish
2.	Fast	No	Yes	Animal
3.	Slow	No	No	Animal
4.	Fast	No	No	Animal
5.	No	Short	No	Bird
6.	No	Short	No	Bird
7.	No	Rarely	No	Animal
8.	Slow	No	Yes	Animal
9.	Slow	No	No	Fish
10.	Slow	No	Yes	Fish
11.	No	Long	No	Bird
12.	Fast	No	No	Bird

- Ans :- P(A|B) = (P(B|A) * P(A))/ P(B)
- **Fish:**
- P(X | Fish) = P(Slow | Fish) * P(Rarely | Fish) * P(No | Fish)
- a)P(Slow | Fish) = (P(Fish | Slow) * P(Slow))/ P (Fish) = ((2/4) * (4/12)) / (3/12) = 0.66
- b)P(Rarely | Fish) = (P(Fish | Rarely) * P(Rarely))/ P (Fish) = ((0/1) * (1/12)) / (3/12) = 0
- c)P(No | Fish) = (P(Fish | No) * P(No))/ P (Fish) = ((2/9) * (9/12)) / (3/12) = 0.66
- Thus, P(X | Fish) = 0.66 * 0 * 0.66 = 0

Naive Bayes Previous Year Question

- **Animal:**
- P(X | Animal) = P(Slow | Animal) * P(Rarely | Animal) * P(No | Animal)
- d)P(Slow | Animal) = (P(Animal | Slow) * P(Slow))/ P (Animal) = ((2/4) * (4/12)) / (5/12) = 0.4
- e)P(Rarely | Animal) = (P(Animal | Rarely) * P(Rarely))/ P (Animal) = ((1/1) * (1/12)) / (5/12) = 0.2
- f)P(No | Animal) = (P(Animal | No) * P(No))/ P (Animal) = ((3/9) * (9/12)) / (5/12) = 0.6
- Thus, P(X | Animal) = 0.4 * 0.2 * 0.6 = 0.048

- **Bird:**
- P(X | Bird) = P(Slow | Bird) * P(Rarely | Bird) * P(No | Bird)
- g)P(Slow | Bird) = (P(Bird | Slow) * P(Slow))/ P (Bird) = ((0/4) * (4/12)) / (4/12) = 0
- h)P(Rarely | Bird) = (P(Bird | Rarely) * P(Rarely))/ P (Bird) = ((0/1) * (1/12)) / (4/12) = 0
- i)P(No | Bird) = (P(Bird | No) * P(No))/ P (Bird) = ((4/9) * (9/12)) / (4/12) = 1
- Thus, P(X | Bird) = 0 * 0 * 1 = 0
- **Since P(X|Animal) has highest value among P(X|Animal), P(X|Fish) and P(X|Bird) , Hence class of new instance is animal.**

Decision Tree Previous Year Question

- 1.Consider the following data set.

Color	Size	Act	Age	Inflated
Yellow	Small	Dip	Adult	F
Yellow	Large	Stretch	Adult	T
Yellow	Large	Stretch	Child	F
Yellow	Large	Dip	Adult	F
Yellow	Large	Dip	Child	F
Purple	Small	Stretch	Adult	T
Purple	Small	Stretch	Adult	T
Purple	Small	Stretch	Child	F
Purple	Small	Dip	Adult	F
Purple	Small	Dip	Child	F

- Calculate the information gain of each attribute. State which attribute should be used as the first root node based on the information gain parameter.

- **Ans:** From the total of 10 rows in our data-set S, there are 3 rows with the target value T and 7 rows with the target value F. The entropy of S is calculated as:

- Entropy(S) = - (3/10) * $\log_2(3/10)$ - (7/10) * $\log_2(7/10)$ = -0.3 * -1.737 - 0.7* -0.5146 = 0.88

Decision Tree Previous Year Question

- **IG calculation for Color:**

- In this(Color) feature there are 5 rows having value Yellow and 5 rows having value Purple.
- In the 5 rows with Yellow for Color, there is 1 row having target value T and 4 rows having target value F.
- In the 5 rows with Purple, there are 2 rows having target value T and 3 rows having target value F.
- $|S| = 10$
- For v = Yellow, $|S_v| = 5$
- $\text{Entropy}(S_v) = - (1/5) * \log_2(1/5) - (4/5) * \log_2(4/5) = -0.2 * -2.322 - 0.8 * -0.3219 = 0.72$
- For v = Purple, $|S_v| = 5$
- $\text{Entropy}(S_v) = - (2/5) * \log_2(2/5) - (3/5) * \log_2(3/5) = -0.4 * -1.322 - 0.6 * -0.737 = 0.971$

- IG(S, Color) = Entropy(S) - (|Syellow| / |S|) * Entropy(Syellow) - (|SPurple| / |S|) * Entropy(SPurple)
- **$\therefore$ IG(S, Color) = 0.88 - (5/10) * 0.72 - (5/10) * 0.971 = 0.0345**

Decision Tree Previous Year Question

- Next, we calculate the IG for the features “Size”, “Act” and “Age”.
- **IG(S, Size) = 0.006**
- **IG(S, Act) = 0.396**
- **IG(S, Age) = 0.281**
- **Since the feature Act have the highest Information Gain it is used to create the root node.**

Regression Previous Year Question

- 1.The fuel efficiency of different cars in miles per gallon (mpg) with respect to its weight is given in the following table.

Weight	Mpg
3504	18
3693	15
3436	18
3433	16
3449	17
4341	15
4354	14
4312	14
4425	14
3850	15

- Find the least square estimation of the line y = β0 + β1X , such that β0 and β1 are the parameters of the line.
- **Ans:**
- Here n =10

$$\bar{x} = 38797/10 = 3879.7$$

$$\bar{y} = 156/10 = 15.6$$

$$Cov(x, y) = \frac{1}{9}(-5218.2) = -579.8$$

$$Var(x) = \frac{1}{9}(1680796.1) = 186755.12$$

- β1 = -579.8/186755.12 = -0.003104, β0 = 15.6 - - (-0.003104) * 3879.7 = 15.6 + 12.04 = 27.64

Regression Previous Year Question

- 2. Fit a straight line Y = a + bX to the data by the method of least square.

X	1	3	4	2	5
Y	3	4	5	2	1

- 3. Fit a straight line Y = a + bX to the data by the method of least square.

X	5	10	15	20	25
Y	16	19	23	20	30

- Q2 & Q3 are homework.

Regression Previous Year Question

❖ 4.Explain different evaluation metrics/errors used in measuring the performance of a regression problem?

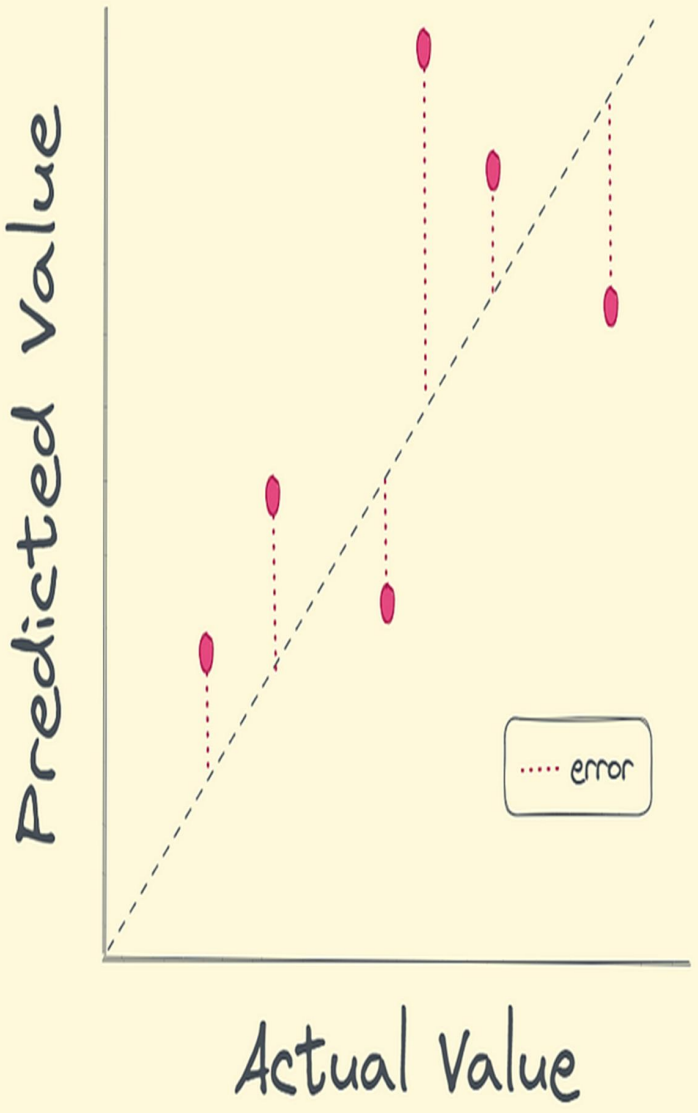
$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

$$R^2 = 1 - \frac{SSR}{SST} = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y})^2}$$

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right|$$



Metric	When to use
MAE	We want a metric that gives equal weight to all errors and is less sensitive to outliers. (Low value is good)
MSE	We want a metric that penalizes larger errors more than smaller ones.(Low value is good)
RMSE	We want a metric that penalizes larger errors more than smaller ones.(Low value is good)
R2	We want to understand the proportion of the variance in the dependent variable that is predictable from the independent variables. (High value is good)
MAPE	We want to express errors as a percentage of the actual values and want a metric that is easy to interpret.(Low value is good)

Regression Previous Year Question

- 5.Write the effect of learning rate on the performance of a Gradient Descent algorithm?
- **Ans:- High Learning Rate:** A high learning rate can lead to **faster convergence,** as the algorithm takes larger steps towards the minimum. However, it **may also cause overshooting**, where the algorithm **may oscillate around the minimum or even fail to converge.**
- **Low Learning Rate:** A low learning rate **slows down the convergence process**, as the algorithm takes smaller steps. While it may **increase the likelihood of convergence**, it can be **computationally expensive and time-consuming.**
- **Optimal Learning Rate:** An optimal learning rate allows the algorithm to **reach a sufficiently accurate solution within a reasonable number of iterations.**
- **Adaptive Learning Rates:** Some advanced optimization algorithms incorporate adaptive learning rates, **adjusting the learning rate during training based on the observed behavior of the optimization process.** This adaptability can **enhance robustness.**