SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

SCHOOL OF COMPUTING

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SCSA3016 DATA SCIENCE

COURSE OBJECTIVES

- To understand the mathematical foundations required for data science.
- To describe a flow process for data science problems.
- To introduce basic data science algorithms and data visualization.
- To learn machine tools and techniques.
- To learn the ideas and tools for data visualization.

UNIT 1 LINEARALGEBRA **9 Hrs.**

Algebraic view – vectors 2D, 3D and nD, matrices, product of matrix & vector, rank, null space, solution of over determined set of equations and pseudo-inverse. Geometric view - vectors, distance, projections, eigenvalue decomposition, Equations of line, plane, hyperplane, circle, sphere, Hypersphere.

UNIT 2 PROBABILITY AND STATISTICS **9 Hrs.**

Introduction to probability and statistics, Population and sample, Normal and Gaussian distributions, Probability Density Function, Descriptive statistics, notion of probability, distributions, mean, variance, covariance, covariance matrix, understanding univariate and multivariate normal distributions, introduction to hypothesis testing, confidence interval for estimates.

UNIT 3 EXPLORATORY DATA ANALYSIS AND THE DATA SCIENCE PROCESS **9 Hrs.**

Exploratory Data Analysis and the Data Science Process - Basic tools (plots, graphs and summary statistics) of EDA - Philosophy of EDA - The Data Science Process - Data Visualization - Basic principles, ideas and tools for data visualization
- Examples of exciting projects- Data Visualization using Tableau.

UNIT 4 MACHINE LEARNING TOOLS, TECHNIQUES AND APPLICATIONS 9 Hrs.

Supervised Learning, Unsupervised Learning, Reinforcement Learning, Dimensionality Reduction, Principal Component Analysis, Classification and Regression models, Tree and Bayesian network models, Neural Networks, Testing, Evaluation and Validation of Models.

UNIT 5 INTRODUCTION TO PYTHON 9 Hrs.

Data structures-Functions-Numpy-Matplotlib-Pandas- problems based on computational complexity-Simple case studies based on python (Binary search, common elements in list), Hash tables, Dictionary.

Max. 45 Hrs.

Course Outcome

On completion of the course, student will be able to

CO1 - Explain the basic terms of Linear Algebra and Statistical Inference.
CO2 - Describe the Data Science process and how its components interact.
CO3 - Apply EDA and the Data Science process in a case study.
CO4 - Classify Data Science problems.
CO5 - Analyse and correlate the results to the solutions.
CO6 - Simulate Data Visualization in exciting projects.

UNIT 1 LINEAR ALGEBRA

Algebraic view – vectors 2D, 3D and nD, matrices, product of matrix & vector, rank, null space, solution of over determined set of equations and pseudo-inverse. Geometric view - vectors, distance, projections, eigenvalue decomposition, Equations of line, plane, hyperplane, circle, sphere, Hypersphere.

1.1.Introduction to Data science

- **Data science** is an interdisciplinary field that uses scientific methods, processes, algorithms and systems to extract knowledge and insights from noisy, structured and unstructured data.
- It is an analyzing method to extract accurate and deep understanding of a raw data using methods in statistics, Machine Learning etc.
- Different process includes in data science are inspecting, cleaning, transforming, modeling, analyzing and interpreting raw data.

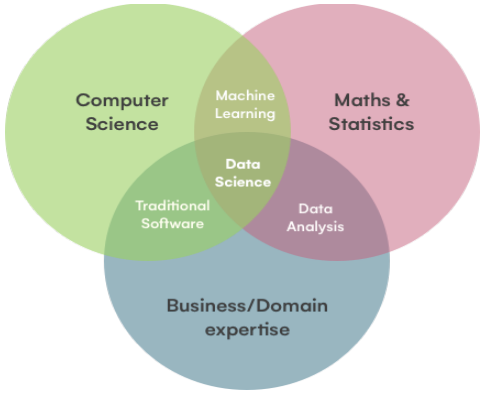


Figure 1 : Collaboration of data science with other branch of studies

Data Science Life Cycle

Figure 2 :Data science life cycle process

Applications of Data Science

- Fraud and risk detection
- Healthcare
- Airline Route Planning
- Image and speech recognition
- Augmented reality

1.2.Linear Algebra

- Linear algebra is one of the most important mathematical and computational tools in data science.
- It is a branch of mathematics that deals with the theory of systems of linear equations, matrices, vector spaces, determinants, and linear transformations.

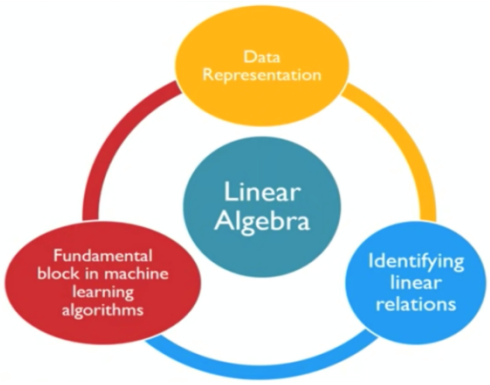


Figure 3 : Process in Linear Algebra

1.3.Algebraic View: Vector

- A vector is an object that has both a magnitude and a direction.
- Geometrically, we can picture a vector as a directed line segment, whose length is the magnitude of the vector and with an arrow indicating the direction. The direction of the vector is from its tail to its head.
- Vectors are used to represent numeric or symbolic characteristics, called features, of an object in a mathematical, easily analyzable way.

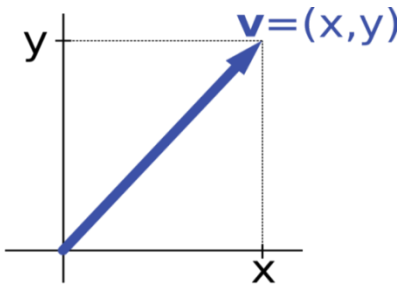


Figure 4 : Vector representation

Perpendicular vector and orthogonal vector

- Two vectors are perpendicular if the angle between them is 90 degrees.
- If two vectors are nonzero and their dot product is equal to 0, then they are perpendicular.

$$\vec{x} \text{ and } \vec{y} \text{ are perpendicular} \rightarrow \vec{x}.\vec{y} = 0$$

$$\vec{x}.\vec{y} = 0\,,\ \vec{x},\ \vec{y} \neq \vec{0} \rightarrow \vec{x} \text{ and } \vec{y} \text{ are perpendicular}$$

- All perpendicular vectors are orthogonal.
- The 0 vector is orthogonal to everything else (even to itself)

$$\vec{0}.\vec{x} = 0$$

1.4.Defining a 2D point/Vector:

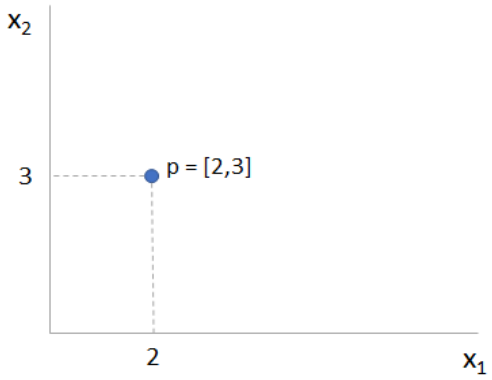


Figure 5: 2D vector plot

In 2D space, a point is defined as the (x,y) coordinates as shown above. Here, the x_1 coordinate (x coordinate) is 2, and the x_2 coordinate (y coordinate) is 3.

Defining a 3D point/Vector:

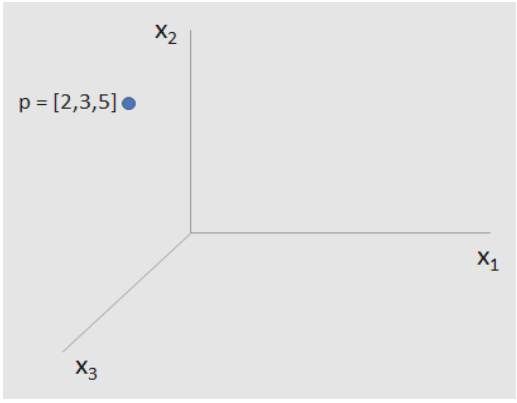


Figure 6 : 3D vector plot

Extending the 2D concept in 3D space point 'p' is defined by (x,y,z) coordinates, where 2 is x_1 coordinate(x coordinate), 3 is x_2 coordinate (y coordinate), and 5 is x_3 coordinate (z coordinate).

Distance of a point from Origin:

a) In 2D

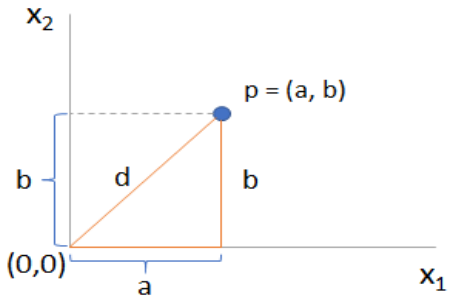


Figure 7 : Distance of a point from origin in 2D space

In 2D space, the distance d is given by,

$$d = \sqrt{(a^2 + b^2)} \quad \text{where, } p = (a, b)$$

b) In 3D

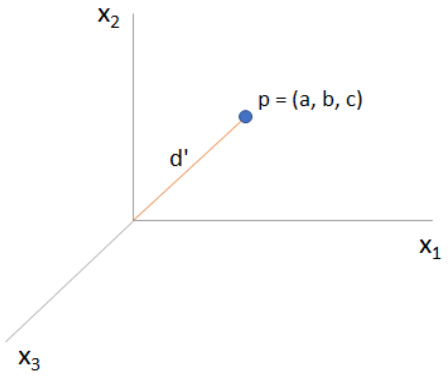


Figure 8 : Distance of a point from origin in 3D space

In 3D space, the distance d is given by,

$$d' = \sqrt{(a^2 + b^2 + c^2)} \quad \text{where } p = (a, b, c)$$

c) **In nD:**

In n-Dimensional space applying Pythagoras theorem on point 'p' we get,

$$d = \sqrt{(a_1^2 + a_2^2 + ... + a_n^2)} \left\{ \quad \text{where } p = (a_1, a_2, .., a_n) \right.$$

Distance between two points

a) In 2D

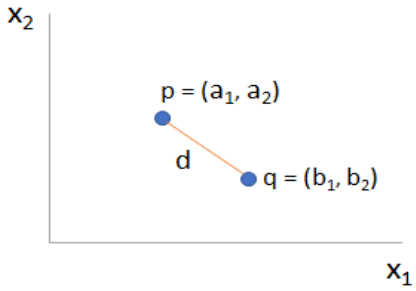


Figure 9 : Distance between two points in 2D space

Consider we have two points say, p and q then the distance d is given by,

$$d = \sqrt{((a_1 - b_1)^2 + (a_2 - b_2)^2)} \left\{ \begin{array}{l} \text{where } p = (a_1, a_2) \\ \text{and } q = (b_1, b_2) \end{array} \right.$$

b) In 3D

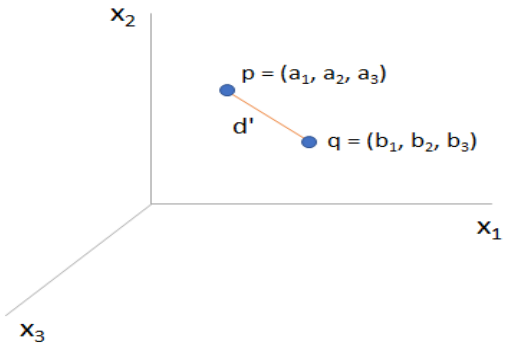


Figure 10 : Distance between two points in 3D space

Extending the same concept in 3D space we get the distance d' for the points p and q as follows:

$$d = \sqrt{((a_1 - b_1)^2 + (a_2 - b_2)^2 + (a_3 - b_3)^2)} \left\{ \begin{array}{l} \text{where } p = (a_1, a_2, a_3) \\ \text{and } q = (b_1, b_2, b_3) \end{array} \right.$$

c) In nD

Extending the above concept in nD space, we get the distance formulae as,

$$d_{pq} = \sqrt{\sum_{i=1}^{n}(a_i - b_i)^2} \left\{ \begin{array}{l} \text{where } p = (a_1, a_2, \ldots, a_n) \\ \text{and } q = (b_1, b_2, \ldots, b_n) \end{array} \right.$$

Row-Vector Representation:

The row vector is a (1xn) matrix where the number of rows is 1 and the number of columns is 'n'.

$$\begin{bmatrix} a_1 & a_2 & a_3 \ldots a_n \end{bmatrix}$$

Column-Vector Representation:

The column vector is an (nx1) matrix where the number of rows is 'n' and the number of columns is 1.

$$\begin{bmatrix} b_1 \\ b_2 \\ . \\ . \\ b_n \end{bmatrix}$$

1.5.Dot Product of Vectors

If we have two vectors **u** and **v**:

$$\mathbf{u} = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{bmatrix} \qquad \mathbf{v} = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix}$$

The dot product (or inner product) of these vectors is defined as the transpose of **u** multiplied by **v**:

$$\mathbf{u}.\mathbf{v} = \mathbf{u}^T\mathbf{v} = [u_1 u_2 \cdots u_n] \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix} = u_1 v_1 + u_2 v_2 + \cdots + u_n v_n$$

Based on this definition the dot product is commutative so:

$$\mathbf{u}.\mathbf{v} = \mathbf{v}.\mathbf{u}$$

1.6.Matrix Theory and Linear Algebra

- Matrices can be used to represent samples with multiple attributes in a compact form.
- Matrices can also be used to represent linear equations in a compact and simple fashion.
- Linear algebra provides tools to understood and manipulate matrices to derive useful knowledge from data.

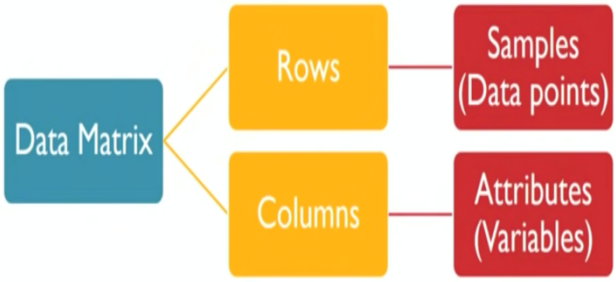


Figure 11 : Matrix elements

- A matrix is a rectangular array of numbers, symbols, or expressions, arranged in rows and columns.

- A matrix with m rows and n columns is called an m×n matrix or m-by-n matrix, where m and n are called the matrix dimensions.
- Matrices can be used to compactly write and work with multiple linear equations, that is, a system of linear equations.
- A matrix is a 2-D array of shape (m×n) with m rows and n columns is as given below:

$$A = \begin{bmatrix} A_{1,1} & A_{1,2} & \cdots & A_{1,n} \\ A_{2,1} & A_{2,2} & \cdots & A_{2,n} \\ \cdots & \cdots & \cdots & \cdots \\ A_{m,1} & A_{m,2} & \cdots & A_{m,n} \end{bmatrix}$$

Tensor

Generally, an n-dimensional array where n>2 is called a Tensor. But a matrix or a vector is also a valid tensor. A tensor is an algebraic object that describes a multilinear relationship between sets of algebraic objects related to a vector space.

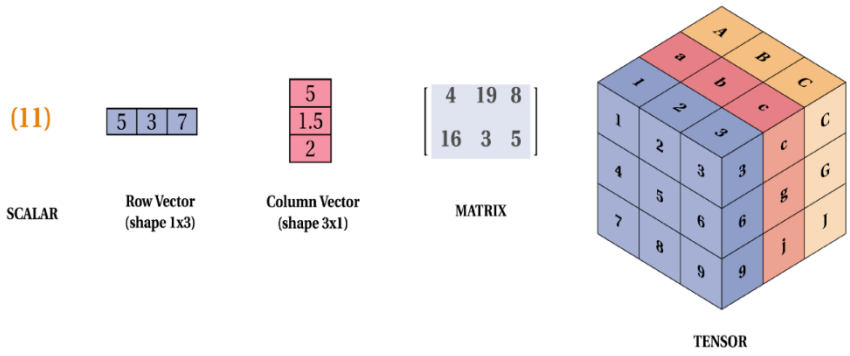


Figure 12 : Matrix and tensor

Terms related to Matrix

- **Order of matrix** – If a matrix has 3 rows and 4 columns, order of the matrix is 3*4 i.e. row*column.
- **Square matrix** – The matrix in which the number of rows is equal to the number of columns.
- **Diagonal matrix** – A matrix with all the non-diagonal elements equal to 0 is called a diagonal matrix.
- **Upper triangular matrix** – Square matrix with all the elements below diagonal equal to 0.
- **Lower triangular matrix** – Square matrix with all the elements above the diagonal equal to 0.
- **Scalar matrix** – Square matrix with all the diagonal elements equal to some constant k.
- **Identity matrix** – Square matrix with all the diagonal elements equal to 1 and all the non-diagonal elements equal to 0.
- **Column matrix** – The matrix which consists of only 1 column. Sometimes, it is used to represent a vector.

- **Row matrix** – A matrix consisting only of row.
- **Trace** – It is the sum of all the diagonal elements of a square matrix.

Basic operations on matrix

- **Addition** – Addition of matrices is almost similar to basic arithmetic addition.
  Eg : Suppose we have 2 matrices 'A' and 'B' and the resultant matrix after the addition is 'C'. Then

  $$Cij = Aij + Bij$$

  For example, let's take two matrices and solve them.

  $$A = \begin{matrix} 1 & 0 \\ 2 & 3 \end{matrix} \qquad B = \begin{matrix} 4 & -1 \\ 0 & 5 \end{matrix} \qquad \text{Then} \qquad C = \begin{matrix} 5 & -1 \\ 2 & 8 \end{matrix}$$

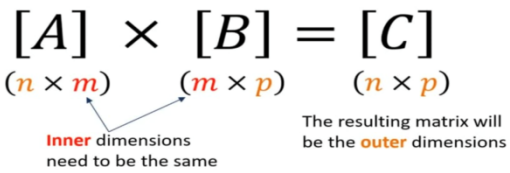
- **Subtraction** – Subtraction of matrices is almost similar to basic arithmetic subtraction.
  Eg : Suppose we have 2 matrices 'A' and 'B' and the resultant matrix after the subtraction is 'D'. Then

  $$Dij = Aij - Bij$$

  For example, let's take two matrices and solve them.

  $$A = \begin{matrix} 1 & 0 \\ 2 & 3 \end{matrix} \qquad B = \begin{matrix} 4 & -1 \\ 0 & 5 \end{matrix} \qquad \text{Then} \qquad C = \begin{matrix} -3 & 1 \\ 2 & -2 \end{matrix}$$

- **Multiplication** – In matrix multiplication the matrices don't need to be quadric, but the inner dimension needs to be same. The size of the resulting matrix will be the outer dimensions.



A

A11	A12
A21	A22

B

B11	B12
B21	B22

A*B

A11*B11 + A12*B21	A11*B12+A12*B22
A21*B11+A22*B21	A21*B12+A22*B22

Figure 13 : Matrix multiplication

Eg :

A

2	4
5	1

B

3	8
7	0

A*B

34	16
22	40

Applying Python on matrices

- In python, matrix can be implemented as 2D list or 2D Array.
- Matrix operations and array are defines in module "**numpy**".

  - **add() :-** This function is used to perform element wise matrix addition.
  - **subtract()** :- This function is used to perform element wise matrix subtraction.
  - **multiply() :-** This function is used to perform element wise matrix multiplication.
  - **dot()** :- This function is used to compute the matrix multiplication, rather than element wise multiplication.

Transpose of a matrix

A general matrix is given by:

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1m} \\ a_{21} & a_{22} & \cdots & a_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nm} \end{bmatrix}$$

Where n is number of rows and m is number of columns

$(n \times m)$

The transpose of matrix A is then:

$$A^T = \begin{bmatrix} a_{11} & a_{21} & \cdots & a_{n1} \\ a_{12} & a_{22} & \cdots & a_{n2} \\ \vdots & \vdots & \ddots & \vdots \\ a_{1m} & a_{2m} & \cdots & a_{nm} \end{bmatrix} \quad (m \times n)$$

Eg :

$$A = \begin{bmatrix} 1 & 3 & 7 & 2 \\ 5 & 8 & -9 & 0 \\ 6 & -7 & 11 & 12 \end{bmatrix} \Rightarrow A^T = \begin{bmatrix} 1 & 5 & 6 \\ 3 & 8 & -7 \\ 7 & -9 & 11 \\ 2 & 0 & 12 \end{bmatrix}$$

$$B = \begin{bmatrix} 1 & 5 \\ 4 & 5 \\ 3 & 2 \\ 7 & 8 \end{bmatrix} \Rightarrow B^T = \begin{bmatrix} 1 & 4 & 3 & 7 \\ 5 & 5 & 2 & 8 \end{bmatrix}$$

Determinant of a matrix

The determinant of a matrix can be calculated from square matrices.

Given a matrix A the Determinant is given by:

$$det(A) = |A|$$

For a $2x2$ matrix we have:

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \Rightarrow det(A) = |A| = a_{11}\, a_{22} - a_{21}\, a_{12}$$

Example:

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \Rightarrow det(A) = |A| = 1 \cdot 4 - 3 \cdot 2 = 4 - 6 = \underline{\underline{-2}}$$

For a $3x3$ matrix we have:

We develop the determinant along a row or a column.

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

$$\Rightarrow det(A) = |A| = a_{11} \begin{vmatrix} a_{22} & a_{23} \\ a_{32} & a_{33} \end{vmatrix} - a_{21} \begin{vmatrix} a_{12} & a_{13} \\ a_{32} & a_{33} \end{vmatrix} + a_{31} \begin{vmatrix} a_{12} & a_{13} \\ a_{22} & a_{23} \end{vmatrix}$$

Applying Python on matrices – Determinant

Given the following Matrices:

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \qquad B = \begin{bmatrix} -1 & 3 & 0 \\ 2 & 1 & -5 \\ 1 & 4 & -2 \end{bmatrix}$$

$$det(A) = -2$$

$$det(B) = -21$$

Python Solution:

```
-2.0000000000000004
-21.00000000000001
```

```
import numpy as np
import numpy.linalg as la

A = np.array([[1, 2],
              [3, 4]])

Adet = la.det(A)

print(Adet)

B = np.array([[-1, 3, 0],
              [2, 1, -5],
              [1, 4, -2]])

Bdet = la.det(B)

print(Bdet)
```

Inverse of a Matrix

The **inverse** of a quadratic matrix A is defined by: A^{-1} Note: $AA^{-1} = A^{-1}A = I$

For a $2x2$ matrix we have:

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$$

The inverse A^{-1} is then given by

$$A^{-1} = \frac{1}{det(A)} \begin{bmatrix} a_{22} & -a_{12} \\ -a_{21} & a_{11} \end{bmatrix}$$

Example:

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \Rightarrow A^{-1} = \frac{1}{det(A)} \begin{bmatrix} 4 & -2 \\ -3 & 1 \end{bmatrix}$$

Where:

$$det(A) = |A| = 1 \cdot 4 - 3 \cdot 2 = 4 - 6 = -2$$

This gives: $A^{-1} = \frac{1}{-2} \begin{bmatrix} 4 & -2 \\ -3 & 1 \end{bmatrix} = \begin{bmatrix} -2 & 1 \\ 1.5 & -0.5 \end{bmatrix}$

The formula to find the inverse of a matrix is

$$\mathbf{A^{-1} = 1/det(A) * Adj(A)}$$

1.7.Rank of a matrix

- Rank of a matrix is equal to the maximum number of linearly independent row vectors in a matrix.
- A set of vectors is linearly dependent if we can express at least one of the vectors as a linear combination of remaining vectors in the set.
- Note : Rank is the number of rows with non zero vectors.
- **Rank of a matrix** – Rank of a matrix is equal to the maximum number of linearly independent row vectors in a matrix.
- A set of vectors is linearly dependent if we can express at least one of the vectors as a linear combination of remaining vectors in the set.
- **To Calculate Rank of Matrix There are Two Methods:**
  1. Minor method
  2. Echelon form
- The maximum number of linearly independent rows in a matrix A is called the **row rank** of A, and the maximum number of linearly independent columns in A is called the **column rank** of A.
- If A is an m by n matrix, that is, if A has m rows and n columns, then it is obvious that

$$\left.\begin{array}{r}\text{row rank of } A \le m \\ \text{column rank of } A \le n\end{array}\right\} \quad (*)$$

- To find the rank of a matrix, we will transform that matrix into its echelon form.
- Then determine the rank by the number of non zero rows.
- Consider the following matrix.

$$A = \begin{bmatrix} 2 & 4 & 6 \\ 4 & 8 & 12 \end{bmatrix}$$

- While observing the rows, we can see that the second row is two times the first row. Here we have two rows. But it does not count. The rank is considered as 1.

Consider the unit matrix

$$A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

We can see that the rows are independent. Hence the rank of this matrix is 3.
The rank of a unit matrix of order m is m.
If A matrix is of order m×n, then $\rho(A) \le \min\{m, n\}$ = minimum of m, n.
If A is of order n×n and $|A| \neq 0$, then the rank of A = n.
If A is of order n×n and $|A| = 0$, then the rank of A will be less than n

Rank of a Matrix by Row- Echelon Form

- We can transform a given non-zero matrix to a simplified form called a Row-echelon form, using the row elementary operations. In this form, we may have rows all of whose entries are zero. Such rows are called zero rows. A non-zero row is one in which at least one of the elements is not zero.

A matrix is said to be in **row-echelon form** if the following rules are satisfied.

- All the leading entries in each row of the matrix is 1
- If a column contains a leading entry then all the entries below the leading entry should be zero
- If any two consecutive non-zero rows, the leading entry in the upper row should occur to the left of the leading entry in the lower row.
- All rows which consist only of zeros should occur in the bottom of the matrix

A matrix A of order $m \times n$ is said to be in echelon form if

(i) Every row of A which has all its entries 0 occurs below every row which has a non-zero entry.

(ii) The number of zeros before the first non-zero element in a row is less then the number of such zeros in the next row.

- For example, consider the following matrix.
- Here R_1 and R_2 are non zero rows.
- R_3 is a zero row.

- Note: A non-zero matrix is said to be in a row-echelon form, if all zero rows occur as bottom rows of the matrix and if the first non-zero element in any lower row occurs to the right of the first non-zero entry in the higher row.
- If a matrix is in row-echelon form, then all elements below the leading diagonal are zeros.
- Consider the following matrix.

$$A = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 5 \\ 0 & 0 & 0 \end{bmatrix}$$

- Check the rows from the last row of the matrix. The third row is a zero row. The first non-zero element in the second row occurs in the third column and it lies to the right of the first non-zero element in the first row which occurs in the second column. Hence the matrix A is in row echelon form.

Find the rank of the matrix A= $\begin{pmatrix} 0 & 1 & 2 & 1 \\ 1 & 2 & 3 & 2 \\ 3 & 1 & 1 & 3 \end{pmatrix}$

Solution:

The order of A is 3 × 4.

$\therefore \rho(A) \leq 3$.

Let us transform the matrix A to an echelon form

Matrix A	Elementary Transformation
$A = \begin{pmatrix} 0 & 1 & 2 & 1 \\ 1 & 2 & 3 & 2 \\ 3 & 1 & 1 & 3 \end{pmatrix}$	
$A = \begin{pmatrix} 1 & 2 & 3 & 2 \\ 0 & 1 & 2 & 1 \\ 3 & 1 & 1 & 3 \end{pmatrix}$	$R_1 \leftrightarrow R_2$
$\sim \begin{pmatrix} 1 & 2 & 3 & 2 \\ 0 & 1 & 2 & 1 \\ 0 & -5 & -8 & -3 \end{pmatrix}$	$R_3 \rightarrow R_3 - 3R_1$
$\sim \begin{pmatrix} 1 & 2 & 3 & 2 \\ 0 & 1 & 2 & 1 \\ 0 & 0 & 2 & 2 \end{pmatrix}$	$R_3 \rightarrow R_3 + 5R_2$

The number of non zero rows is 3. $\therefore \rho(A) = 3$.

Find the rank of the matrix A= $\begin{pmatrix} 1 & 1 & 1 & 1 \\ 3 & 4 & 5 & 2 \\ 2 & 3 & 4 & 0 \end{pmatrix}$

Solution:

The order of A is 3×4.

$\therefore \rho(A) \leq 3$.

Let us transform the matrix A to an echelon form

Matrix A	Elementary Transformation
$A = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 3 & 4 & 5 & 2 \\ 2 & 3 & 4 & 0 \end{pmatrix}$	
$\sim \begin{pmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 2 & -1 \\ 0 & 1 & 2 & -2 \end{pmatrix}$	$R_2 \to R_2 - 3R_1$ $R_3 \to R_3 - 2R_1$
$\sim \begin{pmatrix} 1 & 1 & 1 & 1 \\ 0 & -1 & -2 & 1 \\ 0 & 0 & 0 & -1 \end{pmatrix}$	$R_3 \to R_3 - R_2$

The number of non zero rows is 3.

$\therefore \rho(A) = 3$.

Find the rank of matrix A by using the row echelon form.

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 1 & 4 \\ 3 & 0 & 5 \end{bmatrix}$$

Solution:

$$\text{Given } A = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 1 & 4 \\ 3 & 0 & 5 \end{bmatrix}$$

Now we apply elementary transformations.

$R_2 \rightarrow R_2 - 2R_1$

$R_3 \rightarrow R_3 - 3R_1$

We get

$$\begin{bmatrix} 1 & 2 & 3 \\ 0 & -3 & -2 \\ 0 & -6 & -4 \end{bmatrix}$$

$R_3 \rightarrow R_3 - 2R_2$

$$\begin{bmatrix} 1 & 2 & 3 \\ 0 & -3 & -2 \\ 0 & 0 & 0 \end{bmatrix}$$

The above matrix is in row echelon form.

Number of non-zero rows = 2

Hence the rank of matrix A = 2

Find the rank of the matrix.

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Solution:

Given

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

$R_2 \rightarrow R_2 - R_1$

$R_3 \rightarrow R_3 - R_1$

We get

$$\begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Here number of non zero rows = 1

Hence the rank of the matrix = 1

Find the rank of the matrix A = $\begin{bmatrix} 1 & 2 & 3 \\ 2 & 3 & 4 \\ 3 & 5 & 7 \end{bmatrix}$

Solution:

Given A = $\begin{bmatrix} 1 & 2 & 3 \\ 2 & 3 & 4 \\ 3 & 5 & 7 \end{bmatrix}$

Now we transform the matrix A to echelon form by using elementary transformation.

$R_2 \rightarrow R_2 - 2R_1$

$R_3 \rightarrow R_3 - 3R_1$

$$\begin{bmatrix} 1 & 2 & 3 \\ 0 & -1 & -2 \\ 0 & -1 & -2 \end{bmatrix}$$

$R_3 \rightarrow R_3 - R_2$

$$\begin{bmatrix} 1 & 2 & 3 \\ 0 & -1 & -2 \\ 0 & 0 & 0 \end{bmatrix}$$

Number of non-zero rows = 2

Hence the rank of matrix A = 2

Find the rank of the matrix.

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Solution:

Given

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

$R_2 \rightarrow R_2 - R_1$

$R_3 \rightarrow R_3 - R_1$

We get

$$\begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Here number of non zero rows = 1

Hence the rank of the matrix = 1

Given $A = \begin{bmatrix} 4 & 7 \\ 8 & 14 \end{bmatrix}$

Find the rank of matrix A.

Solution:

Given

$$A = \begin{bmatrix} 4 & 7 \\ 8 & 14 \end{bmatrix}$$

By observing the rows, we can see that elements of the second row are twice the elements of the first row.

$R_1 \rightarrow 2R_1 - R_2$

$$\begin{bmatrix} 0 & 0 \\ 8 & 14 \end{bmatrix}$$

Number of non zero rows = 1

Rank of matrix A = 1.

The rank of the following matrix is

$$\begin{bmatrix} 1 & 1 & 0 & -2 \\ 2 & 0 & 2 & 2 \\ 4 & 1 & 3 & 1 \end{bmatrix}$$

a) 1

b) 2

c) 3

4) 4

Solution:

Given $\begin{bmatrix} 1 & 1 & 0 & -2 \\ 2 & 0 & 2 & 2 \\ 4 & 1 & 3 & 1 \end{bmatrix}$

We transform the matrix using elementary row operations.

$R_2 \rightarrow R_2 - 2R_1$

$$\begin{bmatrix} 1 & 1 & 0 & -2 \\ 0 & -2 & 2 & 6 \\ 0 & -3 & 3 & 9 \end{bmatrix}$$

$R_2 \rightarrow R_2/-2$

$$\begin{bmatrix} 1 & 1 & 0 & -2 \\ 0 & 1 & -1 & -3 \\ 0 & -3 & 3 & 9 \end{bmatrix}$$

$R_3 \rightarrow R_3 + 3R_2$

$$\begin{bmatrix} 1 & 1 & 0 & -2 \\ 0 & 1 & -1 & -3 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Since the number of non zero rows is 2, rank = 2

1.8.Null Space and Nullity of a Matrix

- Null space is a concept in linear algebra identifies the linear relationship among attributes.
- The null space of a matrix A consists of all vectors B such that AB -= 0 and B ≠ 0.
- Size of null space of matrix – number of linear relation among attributes

Consider

$$A = \begin{pmatrix} X_{11} & \dots & X_{1n} \\ \dots & \dots & \dots \\ X_{m1} & \dots & X_{mn} \end{pmatrix}$$

Size of A is m * n and

$$B = \begin{pmatrix} b_1 \\ . \\ . \\ . \\ . \\ b_n \end{pmatrix}$$ Size of B is n * 1

Then the set of linear equations are

$$X_{11}b_1 + X_{12}b_2 + \ldots\ldots X_{1n}b_n = 0$$
$$\ldots\ldots$$
$$\ldots\ldots$$
$$X_{m1}b_1 + X_{m2}b_2 + \ldots\ldots X_{mn}b_n = 0$$

Eg : **Find the null space for given Matrix A**

$$A = \begin{bmatrix} 1 & 2 & 1 & 1 \\ 1 & 2 & 2 & -1 \\ 2 & 4 & 0 & 6 \end{bmatrix}$$

To find *N(A)* we should find all vectors *x such that:*

$$\vec{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}, A\vec{x} = \vec{0} \rightarrow \begin{bmatrix} 1 & 2 & 1 & 1 \\ 1 & 2 & 2 & -1 \\ 2 & 4 & 0 & 6 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

By multiplying matrix *A* to vector *x* we have:

$$\begin{bmatrix} x_1 + 2x_2 + x_3 + x_4 \\ x_1 + 2x_2 + 2x_3 - x_4 \\ 2x_1 + 4x_2 + 6x_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Now, we should solve this equation by considering it as a system of linear equations:

$$\begin{cases} x_1 + 2x_2 + x_3 + x_4 = 0 \\ x_1 + 2x_2 + 2x_3 - x_4 = 0 \\ 2x_1 + 4x_2 + 6x_4 = 0 \end{cases}$$

Reduce the equation using Echelon form.

Therefore, the solution to the above equation is:

$$\begin{cases} x_1 + 2x_2 + 3x_4 = 0 \\ x_3 - 2x_4 = 0 \end{cases} \rightarrow \begin{cases} x_1 = -2x_2 - 3x_4 \\ x_3 = 2x_4 \end{cases}$$

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = x_2 \begin{bmatrix} -2 \\ 1 \\ 0 \\ 0 \end{bmatrix} + x_4 \begin{bmatrix} -3 \\ 0 \\ 2 \\ 1 \end{bmatrix}$$

The x2 and x4 are free variables in $\mathbb{R}$. So, all of the vectors in *Null-Space(A),* which satisfy the original equation, *Ax=0,* can be represented as a linear combination of these two vectors.

1.9.Rank - Nullity Theorem

The rank – nullity theorem helps us to relate the nullity of the data matrix to the rank and the number of attributes in the data

Figure 14 : Rank – Nullity Theorem

Exercise 1:

Find the Rank and Nullity of the matrix $A = \begin{bmatrix} 1 & 2 & 1 \\ -2 & -3 & 1 \\ 3 & 5 & 0 \end{bmatrix}$

Method 1: Normal method

Given $A = \begin{bmatrix} 1 & 2 & 1 \\ -2 & -3 & 1 \\ 3 & 5 & 0 \end{bmatrix}$ and order $n = 3$

$|A| = 1(0-5) - 2(0-3) + 1(-10+9)$

$= -5 + 6 - 1 = 0$

i. e. $|A| = 0 \Rightarrow$ rank of matrix is less than order $= 3$ i. e. $\rho(A) < 3$

Consider $\begin{vmatrix} 1 & 2 \\ -2 & -3 \end{vmatrix} = (-3) + 4 = 1 \neq 0$

Rank of the matrix r=2

Nullity of a matrix =n-r= 3-2=1

1.10. Linear Equations

Given the following linear equations:

$$a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + \cdots = b_1$$
$$a_{21}x_1 + a_{21}x_2 + a_{23}x_3 + \cdots = b_2$$
$$\cdots$$

These equations can be set on the following general form:

$$Ax = b$$

Where A is a matrix, x is a vector with the unknowns and b is a vector of constants

$$A = \begin{bmatrix} a_{11} & \cdots & a_{1m} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nm} \end{bmatrix} \quad x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \quad b = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}$$

Solution:

$$x = A^{-1}b$$

(assuming A^{-1} is possible)

1.11. Solution to over determined set of equations

- If there are fewer equations than variables, then the system is called **underdetermined** and cannot have a unique solution. In this case, there are either infinitely many or no solutions.
- A system with more equations than variables is called **overdetermined.**
- If the number of equations equals the number of variables, it is a **balanced or square** system.
- A balanced system or an overdetermined system may have a unique solution.

Example 1: This system has infinitely many solutions. You can tell because these two lines are the same. (The second one is scaled by a factor of 3.)

x+2y=4

3x+6y=12

$$\begin{bmatrix} 1 & 2 & 4 \\ 3 & 6 & 12 \end{bmatrix}$$

$$\frac{1}{3}R_2 \to R_2 \begin{bmatrix} 1 & 2 & 4 \\ 1 & 2 & 4 \end{bmatrix}$$

$$-R_1 + R_2 \to R_2 \begin{bmatrix} 1 & 2 & 4 \\ 0 & 0 & 0 \end{bmatrix}$$

Notice that there is only one pivot column in this row-reduced matrix. The second column is not a pivot column, so we call y a free variable. A system with free variables is called dependent. Variables that do correspond to a pivot column are called fixed variables. In general, we can express the solution of the system as the fixed variables in terms of the free variables.

$$x=4-2yx=4-2y$$

This is a dependent system, and there are infintely many solutions, depending on the value of the free variable y. Once a value of yy is selected, the value of x is automatically fixed.

Example 2: This system has no solution, so we call it inconsistent.

$$x+y+z=13$$
$$x-y-z=4$$
$$x+5y+5z=-1$$

$$\left[\begin{array}{ccc|c} 1 & 1 & 1 & 1 \\ 3 & -1 & -1 & 4 \\ 1 & 5 & 5 & -1 \end{array}\right]$$

$$\begin{matrix} -3R_1 + R_2 \to R_2 \\ -R_1 + R_3 \to R_3 \end{matrix} \left[\begin{array}{ccc|c} 1 & 1 & 1 & 1 \\ 0 & -4 & -4 & 1 \\ 0 & 4 & 4 & -2 \end{array}\right]$$

$$R_2 + R_3 \to R_3 \left[\begin{array}{ccc|c} 1 & 1 & 1 & 1 \\ 0 & -4 & -4 & 1 \\ 0 & 0 & 0 & -1 \end{array}\right]$$

We don't need to go any further than this. The last row reads 0x+0y+0z=−10x+0y+0z=−1, that is, 0=−10=−1. Such a false statement reveals that this system of equations has no solution. It is inconsistent.

1.12. Eigen Values and Eigen Vectors

The mathematical formulation is, Ax = λx

- The constant λ (positive) represents the amount of stretch or shrinkage the attributes x go through in the x direction.
- The solution x are known as eigen vectors and λ is eigen values.

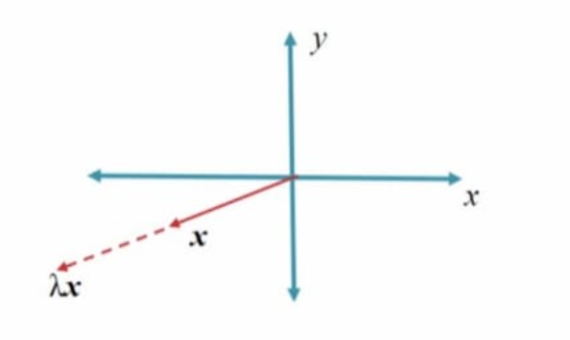


Figure 15: Eigen value λ

Characteristic Equation

The equation $|A - \lambda I| = 0$ is called the characteristic equation of the matrix A

Note:

1. Solving $|A - \lambda I| = 0$, we get n roots for λ and these roots are called characteristic roots or eigen values or latent values of the matrix A
2. Corresponding to each value of λ ,the equation AX = λX has a non-zero solution vector X.

If X_r be the non-zero vector satisfying AX=λX , when$\lambda = \lambda_r$, X_r is said to be the latent vector or eigen vector of a matrix A corresponding to λ_r.

Working rule to find characteristic equation:

Fora 3x3matrix:

Method1:

The characteristic equation is $|A - \lambda I| = 0$

Method2:

Its characteristic equation can be written as $\lambda^3 - S_1\lambda^2 + S_2\lambda - S_3 = 0$where S1 = sum of the main diagonal elements, S2=sum of the minors of the main diagonal elements, S3=Determinant of A =$\|A\|$

Fora2x2matrix

Method1:

The characteristic equation is $|A - \lambda I| = 0$

Method2:

Its characteristic equation can be written as $\lambda^2 - S_1\lambda + S_2 = 0$ where S1= sum of the main diagonal elements, S2=Determinant of A = |A|

Problems:

1. **Find the characteristic equation of** $\begin{pmatrix} 8 & -6 & 2 \\ -6 & 7 & -4 \\ 2 & -4 & 3 \end{pmatrix}$

   Solution:

   Its characteristic equation is $\lambda^3 - S_1\lambda^2 + S_2\lambda - S_3 = 0$

   Where S1=sum of the main diagonal elements=8+7+3=18,

   S2=sum of the minors of the main diagonal elements=45

   S3=Determinant of A= | |A=0

Therefore, the characteristic equation is $\lambda^3 - 18\lambda^2 + 45\lambda = 0$.

2. **Find the characteristic equation of** $\begin{pmatrix} 3 & 1 \\ -1 & 2 \end{pmatrix}$

   Solution:

   The characteristic equation of A is

   $\lambda^2 - S_1\lambda + S_2.S_1 = sumofthemaindiagonalelements$

   S1 = 3+2=5and$S_2 = DeterminantofA = |A|$=3(2)–1(-1)=7

   Therefore, the characteristic equation is $\lambda^2 - 5\lambda + 7$=0.

Steps to find out the eigen value and eigen vector for variable x,

1. Find the characteristic equation $|A - \lambda I| = 0$
2. Solve the characteristic equation to get characteristic roots. They are called Eigen values
3. To find the Eigen vectors, solve$[A - \lambda I]X = 0$ for different values ofλ

Problems:

1. Find the eigen values and eigenvectors of the matrix $\begin{pmatrix} 1 & 1 \\ 3 & -1 \end{pmatrix}$

Solution:

Let A= $\begin{pmatrix} 1 & 1 \\ 3 & -1 \end{pmatrix}$ which is a non-symmetric matrix

To find the characteristic equation:

The characteristic equation of A is $\lambda^2 - S_1\lambda + S_2 = 0$

Where,

$S_1 = sum of the main diagonal elements = 1 - 1 = 0$,

$S_2 = Determinant of A =$ $|A|$=1(-1)–1(3)=-4

Therefore, the characteristic equation is $\lambda^2 - 4 = 0$

i.e.,$\lambda^2 = 4$ or $\lambda = \pm 2$

Therefore, the eigen values are 2,-2

To find the eigen vectors:

$[A - \lambda I]X = 0$

$$\left[\begin{pmatrix} 1 & 1 \\ 3 & -1 \end{pmatrix} - \lambda\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}\right]\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \Rightarrow \left[\begin{pmatrix} 1 & 1 \\ 3 & -1 \end{pmatrix} - \begin{pmatrix} \lambda & 0 \\ 0 & \lambda \end{pmatrix}\right]\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} 1-\lambda & 1 \\ 3 & -1-\lambda \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$----------(1)

Case1: If $\boldsymbol{\lambda = -2}$, $\begin{bmatrix} 1-(-2) & 1 \\ 3 & -1-(-2) \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$ [From(1)]

i.e., $\begin{bmatrix} 3 & 1 \\ 3 & 1 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$

i.e.,$3x_1 + x_2 = 0$,

i.e., we get only one equation

$3x_1 + x_2 = 0 \Rightarrow 3x_1 = -x_2 \Rightarrow \frac{x_1}{1} = \frac{x_2}{-3}$ $X_1 = \begin{bmatrix} 1 \\ -3 \end{bmatrix}$

Therefore

Case2:If $\boldsymbol{\lambda = 2}$, $\begin{bmatrix} 1-(2) & 1 \\ 3 & -1-(2) \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$ [From(1)]

i.e., $\begin{bmatrix} -1 & 1 \\ 3 & -3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$

i.e., $-x_1 + x_2 = 0 \Rightarrow x_1 - x_2 = 0$

$3x_1 - 3x_2 = 0 \Rightarrow x_1 - x_2 = 0$

i.e., we get only one equation $x_1 - x_2 = 0$

$\Rightarrow x_1 = x_2 \Rightarrow \frac{x_1}{1} = \frac{x_2}{1}$

Hence, $X_2 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$

2.Find the eigen values and eigen vectors of

$$\begin{bmatrix} \mathbf{2} & \mathbf{2} & \mathbf{-7} \\ \mathbf{2} & \mathbf{1} & \mathbf{2} \\ \mathbf{0} & \mathbf{1} & \mathbf{-3} \end{bmatrix}$$

Solution:

Let A=$\begin{bmatrix} \mathbf{2} & \mathbf{2} & \mathbf{-7} \\ \mathbf{2} & \mathbf{1} & \mathbf{2} \\ \mathbf{0} & \mathbf{1} & \mathbf{-3} \end{bmatrix}$

To find the characteristic equation:

Its characteristic equation can be written as $\lambda^3 - S_1\lambda^2 + S_2\lambda - S_3 = 0$

where

$S_1 = sum of the main diagonal elements = 2 + 1 - 3 = 0$,

$S_2 = Sum of the minors of the main diagonal elements = \begin{vmatrix} 1 & 2 \\ 1 & -3 \end{vmatrix} + \begin{vmatrix} 2 & -7 \\ 0 & -3 \end{vmatrix} + \begin{vmatrix} 2 & 2 \\ 2 & 1 \end{vmatrix} = -5 + (-6) + (-2) = -5 - 6 - 2 = -13$

$S_3 = Determinant of A = |A|$=2(-5)-2(-6)-7(2)=-10+12–14=-12

Therefore, the characteristic equation of A is $\lambda^3 - 13\lambda + 12 = 0$

3	1	0	-13	12
	1	3	31	-4	0

0 3 9 −12

$$(\lambda-3)(\lambda^2+3\lambda-4)=0 \Rightarrow \lambda=3, \lambda=\frac{-3\pm\sqrt{3^2-4(1)(-4)}}{2(1)}=\frac{-3\pm\sqrt{25}}{2}=\frac{-3\pm5}{2}$$
$$=\frac{-3+5}{2},\frac{-3-5}{2}=1,-4$$

Therefore, the eigenvaluesare3,1,and-4

To find the eigen vectors: Let $[A-\lambda I]X=0$

$$\begin{bmatrix}2-\lambda & 2 & -7\\ 2 & 1-\lambda & 2\\ 0 & 1 & -3-\lambda\end{bmatrix}\begin{bmatrix}x_1\\x_2\\x_3\end{bmatrix}=\begin{bmatrix}0\\0\\0\end{bmatrix}$$

Case1:If$\boldsymbol{\lambda=1}$, $\begin{bmatrix}2-1 & 2 & -7\\ 2 & 1-1 & 2\\ 0 & 1 & -3-1\end{bmatrix}\begin{bmatrix}x_1\\x_2\\x_3\end{bmatrix}=\begin{bmatrix}0\\0\\0\end{bmatrix}$

i.e., $\begin{bmatrix}1 & 2 & -7\\ 2 & 0 & 2\\ 0 & 1 & -4\end{bmatrix}\begin{bmatrix}x_1\\x_2\\x_3\end{bmatrix}=\begin{bmatrix}0\\0\\0\end{bmatrix}$

$\Rightarrow x_1+2x_2-7x_3=0$ (1)

$2x_1+0x_2+2x_3=0$ (2)

$0x_1+x_2-4x_3=0$ (3)

Considering equations (1) and (2) and using method of cross-multiplication, we get, x1 x2 x3

2 -7 1 2

0 2 2 0

x x x x x x

1_2_3⇒1_2_3

4 −16 −4 1 | |

(1)

$-4 \quad -1$

Therefore, $X_1 = \begin{pmatrix} -4 \\ -1 \end{pmatrix}$

Case 2:If $\boldsymbol{\lambda = 3}$, $\begin{bmatrix} 2-3 & 2 & -7 \\ 2 & 1-3 & 2 \\ 0 & 1 & -3-3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$

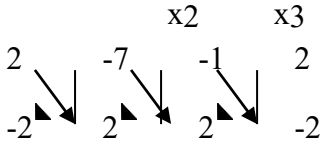
i.e., $\begin{bmatrix} -1 & 2 & -7 \\ 2 & -2 & 2 \\ 0 & 1 & -6 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$

$\Rightarrow -x_1 + 2x_2 - 7x_3 = 0$ .....(1)

$2x_1 - 2x_2 + 2x_3 = 0$.....(2)

$0x_1 + x_2 - 6x_3 = 0$.....(3)

Considering equations (1) and (2) and using method of cross-multiplication, we get,x1



$x \quad x \quad x \quad x \quad x \quad x$

$1\_2\_3 \Rightarrow 1\_2\_3$

$-10 \quad -12 \quad -2 \quad 5 \quad 6 \quad 1$

$$\Rightarrow \frac{x_1}{-10} = \frac{x_2}{-12} = \frac{x_3}{-2} \Rightarrow \frac{x_1}{5} = \frac{x_2}{6} = \frac{x_3}{1}$$

Therefore, $X_2 = \begin{bmatrix} 5 \\ 6 \\ 1 \end{bmatrix}$

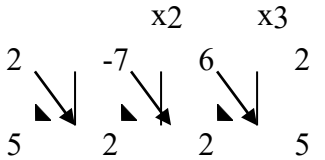
Case3:If $\lambda = -4$, $\begin{bmatrix} 6 & 2 & -7 \\ 2 & 5 & 2 \\ 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$

$\Rightarrow 6x_1 + 2x_2 - 7x_3 = 0$.....(1)

$2x_1 + 5x_2 + 2x_3 = 0$.....(2)

$0x_1 + x_2 + x_3 = 0$.....(3)

Considering equations (1) and (2) and using method of cross-multiplication, we get,x1



$$\Rightarrow \frac{x_1}{39} = \frac{x_2}{-26} = \frac{x_3}{26} \Rightarrow \frac{x_1}{3} = \frac{x_2}{-2} = \frac{x_3}{2}$$

Therefore, $X_3 = \begin{bmatrix} 3 \\ -2 \\ 2 \end{bmatrix}$

1.13. Singular Value Decomposition

- In linear algebra, the Singular Value Decomposition (SVD) of a matrix is a factorization of that matrix into three matrices.

$$A = U\Sigma V^T$$

- A is the input matrix
- U are the left singular vectors,
- sigma are the diagonal/eigenvalues
- V are the right singular vectors.

Example: Find the SVD of A, $U\Sigma V^T$, where $A = \begin{pmatrix} 3 & 2 & 2 \\ 2 & 3 & -2 \end{pmatrix}$.

First we compute the singular values σ_i by finding the eigenvalues of AA^T.

$$AA^T = \begin{pmatrix} 17 & 8 \\ 8 & 17 \end{pmatrix}.$$

The characteristic polynomial is $det(AA^T - \lambda I) = \lambda^2 - 34\lambda + 225 = (\lambda - 25)(\lambda - 9)$, so the singular values are $\sigma_1 = \sqrt{25} = 5$ and $\sigma_2 = \sqrt{9} = 3$.

Now we find the right singular vectors (the columns of V) by finding an orthonormal set of eigenvectors of A^TA. It is also possible to proceed by finding the left singular vectors (columns of U) instead. The eigenvalues of A^TA are 25, 9, and 0, and since A^TA is symmetric we know that the eigenvectors will be orthogonal.

For $\lambda = 25$, we have

$$A^TA - 25I = \begin{pmatrix} -12 & 12 & 2 \\ 12 & -12 & -2 \\ 2 & -2 & -17 \end{pmatrix}$$

which row-reduces to $\begin{pmatrix} 1 & -1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}$. A unit-length vector in the kernel of that matrix is $v_1 = \begin{pmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \\ 0 \end{pmatrix}$.

For $\lambda = 9$ we have $A^TA - 9I = \begin{pmatrix} 4 & 12 & 2 \\ 12 & 4 & -2 \\ 2 & -2 & -1 \end{pmatrix}$ which row-reduces to $\begin{pmatrix} 1 & 0 & -\frac{1}{4} \\ 0 & 1 & \frac{1}{4} \\ 0 & 0 & 0 \end{pmatrix}$.

A unit-length vector in the kernel is $v_2 = \begin{pmatrix} 1/\sqrt{18} \\ -1/\sqrt{18} \\ 4/\sqrt{18} \end{pmatrix}$.

For the last eigenvector, we could compute the kernel of A^TA or find a unit vector perpendicular to v_1 and v_2. To be perpendicular to $v_1 = \begin{pmatrix} a \\ b \\ c \end{pmatrix}$ we need $-a = b$. Then the condition that $v_2^T v_3 = 0$ becomes $2a/\sqrt{18} + 4c/\sqrt{18} = 0$ or $-a = 2c$. So $v_3 = \begin{pmatrix} a \\ -a \\ -a/2 \end{pmatrix}$ and for it to be unit-length we need $a = 2/3$ so $v_3 = \begin{pmatrix} 2/3 \\ -2/3 \\ -1/3 \end{pmatrix}$.

So at this point we know that

$$A = U\Sigma V^T = U \begin{pmatrix} 5 & 0 & 0 \\ 0 & 3 & 0 \end{pmatrix} \begin{pmatrix} 1/\sqrt{2} & 1/\sqrt{2} & 0 \\ 1/\sqrt{18} & -1/\sqrt{18} & 4/\sqrt{18} \\ 2/3 & -2/3 & -1/3 \end{pmatrix}.$$

Finally, we can compute U by the formula $\sigma u_i = Av_i$, or $u_i = \frac{1}{\sigma} Av_i$. This gives $U = \begin{pmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ 1/\sqrt{2} & -1/\sqrt{2} \end{pmatrix}$. So in its full glory the SVD is:

$$A = U\Sigma V^T = \begin{pmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ 1/\sqrt{2} & -1/\sqrt{2} \end{pmatrix} \begin{pmatrix} 5 & 0 & 0 \\ 0 & 3 & 0 \end{pmatrix} \begin{pmatrix} 1/\sqrt{2} & 1/\sqrt{2} & 0 \\ 1/\sqrt{18} & -1/\sqrt{18} & 4/\sqrt{18} \\ 2/3 & -2/3 & -1/3 \end{pmatrix}.$$

1.14. Pseudo Inverse

Pseudo inverse or Moore – Penrose inverse is the generalization of the matrix inverse that may not be invertible. If the matrix is invertible then its inverse will be equal to pseudo inverse and denoted by A^+.

- If the columns of a matrix A are linearly independent, so $A^T \cdot A$ is invertible and we obtain with the following formula the pseudo inverse:

$$\mathbf{A^+ = (A^T \cdot A)^{-1} \cdot A^T}$$

- Here A^+ is a left inverse of A, what means: $A^+ \cdot A = E$.
- However, if the rows of the matrix are linearly independent, we obtain the pseudo inverse with the formula:

$$\boldsymbol{A^+ = A^T \cdot (A \cdot A^T)^{-1}}$$

- This is a right inverse of A, what means: $A \cdot A^+ = E$.
- If both the columns and the rows of the matrix are linearly independent, then the matrix is invertible and the pseudo inverse is equal to the inverse of the matrix.

If A has rank deficient, then the Pseudo inverse of A id defined as

$$A^{+} = (U\Sigma V^{T})^{-1} = (V^{T})^{-1}\Sigma^{-1}U^{-1} = V\Sigma^{-1}U^{T}$$

$$\text{If } \Sigma = \begin{bmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \\ 0 & 0 \end{bmatrix} \text{ then } \Sigma^{-1} = \begin{bmatrix} \frac{1}{\sigma_1} & 0 & 0 \\ 0 & \frac{1}{\sigma_2} & 0 \end{bmatrix}$$

Problems:

1. *Find the pseudo inverse of* $A = \begin{bmatrix} 1 & 2 & 1 & 3 \\ 4 & 3 & 2 & 1 \end{bmatrix}$

Sol:

$$Given\ A = \begin{bmatrix} 1 & 2 & 1 & 3 \\ 4 & 3 & 2 & 1 \end{bmatrix}$$

$$Here \begin{vmatrix} 1 & 2 \\ 4 & 3 \end{vmatrix} = 3 - 8 = -5 \neq 0$$

$$rank(A) = 2$$

Then the pseudo inverse of A *is* $A^{+} = A^{T}(AA^{T})^{-1}$

$$A = \begin{bmatrix} 1 & 2 & 1 & 3 \\ 4 & 3 & 2 & 1 \end{bmatrix} \quad A^{T} = \begin{bmatrix} 1 & 4 \\ 2 & 3 \\ 1 & 2 \\ 3 & 1 \end{bmatrix}$$

$$AA^{T} = \begin{bmatrix} 1 & 2 & 1 & 3 \\ 4 & 3 & 2 & 1 \end{bmatrix} \begin{bmatrix} 1 & 4 \\ 2 & 3 \\ 1 & 2 \\ 3 & 1 \end{bmatrix} = \begin{bmatrix} 15 & 15 \\ 15 & 30 \end{bmatrix}$$

$$|AA^{T}| = 15(30 - 15) = 225$$

$$(AA^{T})^{-1} = \frac{1}{225} \begin{vmatrix} 30 & -15 \\ -15 & 15 \end{vmatrix} = \begin{bmatrix} 2/15 & -1/15 \\ -1/15 & 1/15 \end{bmatrix}$$

$$A^{+} = A^{T}(AA^{T})^{-1} = \begin{bmatrix} 1 & 4 \\ 2 & 3 \\ 1 & 2 \\ 3 & 1 \end{bmatrix} \begin{bmatrix} 2/15 & -1/15 \\ -1/15 & 1/15 \end{bmatrix} = \begin{bmatrix} -2/15 & 3/15 \\ 1/15 & 1/15 \\ 0 & 1/15 \\ 5/15 & -2/15 \end{bmatrix}$$

$$= \frac{1}{15} \begin{bmatrix} -2 & 3 \\ 1 & 1 \\ 0 & 1 \\ 5 & -2 \end{bmatrix}$$

3. *Find the pseudo inverse of* $A = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 4 & 6 \end{bmatrix}$

Sol:

$$Given\ A = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 4 & 6 \end{bmatrix}$$

Here all min *ors of order two are zero*

$\therefore rank(A) = 1$

Since for the given matrix the rank is not equal to the number of rows or columns

therefore it has to be solved by singular value decomposition.

Then the pseudo inverse of A using SVD is $A^{+} = V\Sigma^{+}U^{T} = V\Sigma^{-1}U^{T}$

Compute $A^{T}A = \begin{bmatrix} 5 & 10 & 15 \\ 10 & 20 & 30 \\ 15 & 30 & 45 \end{bmatrix}$

It has eigen values $\lambda_1 = 70, \lambda_2 = 0, \lambda_3 = 0$ and the corresponding

eigen vectors are $\begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}, \begin{bmatrix} -2 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 3 \\ 6 \\ -5 \end{bmatrix}$

The singular values of A are

$$\sigma_1 = \sqrt{\lambda_1} = \sqrt{70}, \sigma_2 = \sqrt{\lambda_1} = \sqrt{0}, \sigma_3 = \sqrt{\lambda_3} = \sqrt{0}$$

Normalized vectors are $v_1 = \begin{bmatrix} 1/\sqrt{14} \\ 2/\sqrt{14} \end{bmatrix}$, $v_2 = \begin{bmatrix} -2/\sqrt{5} \\ 1/\sqrt{5} \end{bmatrix}, v_2 = \begin{bmatrix} 3/\sqrt{70} \\ 6/\sqrt{70} \end{bmatrix}$

Normalized vectors are $v_1 = \begin{bmatrix} 1/\sqrt{14} \\ 2/\sqrt{14} \\ 3/\sqrt{14} \end{bmatrix}$, $v_2 = \begin{bmatrix} -2/\sqrt{5} \\ 1/\sqrt{5} \\ 0 \end{bmatrix}$, $v_2 = \begin{bmatrix} 3/\sqrt{70} \\ 6/\sqrt{70} \\ -5/\sqrt{70} \end{bmatrix}$

So Thus $V = \begin{bmatrix} 1/\sqrt{14} & -2/\sqrt{5} & 3/\sqrt{70} \\ 2/\sqrt{14} & 1/\sqrt{5} & 6/\sqrt{70} \\ 3/\sqrt{14} & 0 & -5/\sqrt{70} \end{bmatrix}$ and $\Sigma = \begin{bmatrix} \sqrt{70} & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$

Also we can find $u_1 = \frac{1}{\sigma_1} A v_1 = \begin{bmatrix} 1/\sqrt{5} \\ 2/\sqrt{5} \end{bmatrix}$ and $u_2 = \frac{1}{\sigma_2} A v_2 = \begin{bmatrix} -2/\sqrt{5} \\ 1/\sqrt{5} \end{bmatrix}$

$$U = \begin{bmatrix} 1/\sqrt{5} & -2/\sqrt{5} \\ 2/\sqrt{5} & 1/\sqrt{5} \end{bmatrix}$$

$$\therefore A^+ = V \Sigma^+ U^T$$

$$= \begin{bmatrix} 1/\sqrt{14} & -2/\sqrt{5} & 3/\sqrt{70} \\ 2/\sqrt{14} & 1/\sqrt{5} & 6/\sqrt{70} \\ 3/\sqrt{14} & 0 & -5/\sqrt{70} \end{bmatrix} \begin{bmatrix} 1/\sqrt{70} & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} 1/\sqrt{5} & 2/\sqrt{5} \\ -2/\sqrt{5} & 1/\sqrt{5} \end{bmatrix}$$

$$= \frac{1}{70} \begin{bmatrix} 1 & 2 \\ 2 & 4 \\ 3 & 6 \end{bmatrix}$$

1.15. Eigen Value Decomposition:

- A few applications of eigen values and eigenvectors that are very useful when handing the data in a matrix form because you could decompose them into matrices that are easy to manipulate.

- In order for the matrix "A" to be either diagonalized or eigen decomposed, it has to meet the following criteria:

  - Must be a Square matrix

  - Has to have linearly independent eigenvectors

- Let $\mathbf{S} \in \mathbb{R}^{m \times m}$ be a square matrix with m linearly independent eigenvectors (a "non-defective" matrix)
- Theorem: Exists an eigen decomposition $\mathbf{S} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^{-1}$ ($\boldsymbol{\Lambda}$: diagonal)
  - (cf. matrix diagonalization theorem)
- Columns of U are eigenvectors of S

Unique for distinct eigen-values

Problem:

Recall $S = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}; \lambda_1 = 1, \lambda_2 = 3.$

The eigenvectors $\begin{pmatrix} 1 \\ -1 \end{pmatrix}$ and $\begin{pmatrix} 1 \\ 1 \end{pmatrix}$ form $U = \begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix}$

Inverting, we have $U^{-1} = \begin{bmatrix} 1/2 & -1/2 \\ 1/2 & 1/2 \end{bmatrix}$ ← Recall $UU^{-1} = 1$.

Then, $S = UAU^{-1} = \begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 \\ 0 & 3 \end{bmatrix}\begin{bmatrix} 1/2 & -1/2 \\ 1/2 & 1/2 \end{bmatrix}$

Let's divide U (and multiply U^{-1}) by $\sqrt{2}$

Then, $S = \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ -1/\sqrt{2} & 1/\sqrt{2} \end{bmatrix}\begin{bmatrix} 1 & 0 \\ 0 & 3 \end{bmatrix}\begin{bmatrix} 1/\sqrt{2} & -1/\sqrt{2} \\ 1/\sqrt{2} & 1/\sqrt{2} \end{bmatrix}$

1.16. Equations of Line

- The equation of a line means an equation in x and y whose solution set is a line in the (x,y) plane.
- The most popular form in algebra is the "slope-intercept" form **y = mx + b.**
- This in effect uses x as a parameter and writes y as a function of x: y = f(x) = mx+b. When x = 0, y = b and the point (0,b) is the intersection of the line with the y-axis.
- Line as a geometrical object and not the graph of a function, it makes sense to treat x and y more even handedly. The general equation for a line (normal form) is **ax + by = c,**
- This can easily be converted to slope-intercept form by solving for y:

y = (-a/b) + c/b,

except for the special case b = 0, when the line is parallel to the y-axis.

Finding the equation of a line through 2 points in the plane

- For any two points P and Q, there is exactly one line PQ through the points. If the coordinates of P and Q are known, then the coefficients a, b, c of an equation for the line can be found by solving a system of linear equations.

Example: For P = (1, 2), Q = (-2, 5), find the equation ax + by = c of line PQ.

- Since P is on the line, its coordinates satisfy the equation:

$$a(1) + b(2) = c, \text{ or } a + 2b = c$$

Since Q is on the line, its coordinates satisfy the equation: a(-2) + b5 = c,

Or $-2a + 5b = c$.

- Multiply the first equation by 2 and add to eliminate a from the equation:

$4b + 5b = 9b = 2c + c = 3c$, so $b = (1/3)c$.

- Then substituting into the first equation, $a = c - 2b = c - (2/3)c = (1/3)c$.

- This gives the equation **[(1/3)c]x + [(1/3)c}y = c**.

1.17. Equations of Plane

- A plane in 3D-space has the equation

$$ax + by + cz = d,$$

- where at least one of the numbers a, b, c must be nonzero.

- If c is not zero, it is often useful to think of the plane as the graph of a function z of x and y. The equation can be rearranged like this:

$$z = -(a/c)x + (-b/c)\, y + d/c$$

- Another useful choice, when d is not zero, is to divide by d so that the constant term = 1.

$$(a/d)x + (b/d)y + (c/d)z = 1.$$

Example: Finding the equation of a plane through 3 points in space

- Given points P, Q, R in space, find the equation of the plane through the 3 points.

Example: P = (1, 1, 1), Q = (1, 2, 0), R = (-1, 2, 1). We seek the coefficients of an equation ax + by + cz = d, where P, Q and R satisfy the equations, thus:

$$a + b + c = d$$

$$a + 2b + 0c = d$$

$$-a + 2b + c = d$$

- Subtracting the first equation from the second and then adding the second equation to the third, we eliminate a to get

$$b - c = 0$$
$$4b + c = 2d$$

- Adding the equations gives 5b = 2d, or b = (2/5)d, then solving for c = b = (2/5)d and then a = d - b - c = (1/5)d.

- So the equation (with a nonzero constant left in to choose) is d(1/5)x + d(2/5)y + d(2/5)z = d, so one choice of constant gives

$$x + 2y + 2z = 5$$

- or another choice would be (1/5)x + (2/5)y + (2/5)z = 1

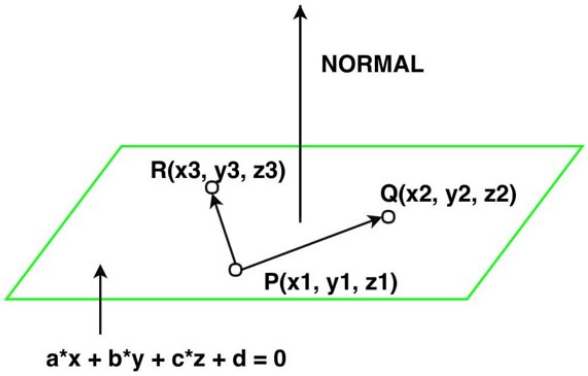


Figure 16. Equation of a plane

Example 2: P(x1, y1, z1), Q(x2, y2, z2), and R (x3, y3, z3) are three non-collinear points on a plane. Find equation of plane.

We know that: $ax + by + cz + d = 0$ ——————(i)

By plugging in the values of the points P, Q, and R into equation (i), we get the following:

$a(x_1) + b(y_1) + c(z_1) + d = 0$

$a(x_2) + b(y_2) + c(z_2) + d = 0$

$a(x_3) + b(y_3) + c(z_3) + d = 0$

Suppose, P = (1,0,2), Q = (2,1,1), and R = (−1,2,1)

Then, by substituting the values in the above equations, we get the following:

$a(1) + b(0) + c(2) + d = 0$

$a(2) + b(1) + c(1) + d = 0$

$a(-1) + b(2) + c(1) + d = 0$

Solving these equations gives us b = 3a, c = 4a, and d = (-9)a——————(ii)

By plugging in the values from (ii) into (i), we end up with the following:

$ax + by + cz + d = 0$

$ax + 3ay + 4az - 9a$

$x + 3y + 4z - 9$

Therefore, the equation of the plane with the three non-collinear points P, Q, and R is $x + 3y + 4z - 9$.

Example 3: A (3,1,2), B (6,1,2), and C (0,2,0) are three non-collinear points on a plane. Find the equation of the plane.

Solution:

We know that: $ax + by + cz + d = 0$ ——————(i)

By plugging in the values of the points A, B, and C into equation (i), we get the following:

$a(3) + b(1) + c(2) + d = 0$

$a(6) + b(1) + c(2) + d = 0$

$a(0) + b(2) + c(0) + d = 0$

Solving these equations gives us

a = 0, c = 1/2b, d = —2b ——————(ii)

By plugging in the values from (ii) into (i), we end up with the following:

ax + by + cz + d = 0

(0)x + (—by) + ½ bz — 2b = 0

x - y + ½ z —2 = 0

2x-2y + z-4 = 0

Therefore, the equation of the plane with the three non-collinear points A, B and C is

2x-2y + z-4 = 0.

1.18. Equation of hyperplane

- A hyperplane is a higher-dimensional generalization of lines and planes.
- The equation of a hyperplane is w · x + b = 0, where w is a vector normal to the hyperplane and b is an offset.

Half Space

- It can be seen that

$$X^T n + b = 0 \; \forall \; X \in line$$

$$X^T n + b > 0 \; \forall \; X \in subspace \; in \; the \; n \; direction \; (X_3)$$

$$X^T n + b < 0 \; \forall \; X \in subspace \; in \; the \; -n \; direction \; (X_2)$$

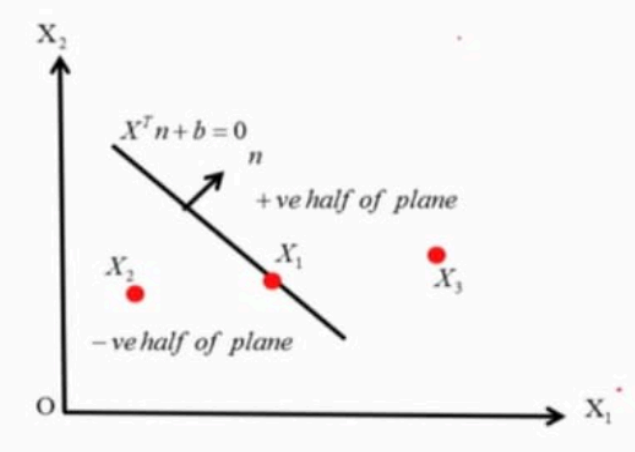


Figure 17: Halfspace

Example

Let us consider a 2D geometry with $n = \begin{bmatrix}1\\3\end{bmatrix}$ and $b = 4$

$$X = \begin{bmatrix}x_1\\x_2\end{bmatrix}$$

$$X^T n + b = 0$$

$$[x_1\ x_2]\begin{bmatrix}1\\3\end{bmatrix} + 4 = 0$$

$$x_1 + 3x_2 + 4 = 0$$

The hyperplane is the equation of a line

The halfspaces corresponding to this hyperplane are

$x_1 + 3x_2 + 4 > 0$: Positive halfspace

$x_1 + 3x_2 + 4 < 0$: Negative halfspace

1.19. Equation of circle

A circle is a closed curve that is drawn from the fixed point called the center, in which all the points on the curve are having the same distance from the center point of the center. The equation of a circle with (h, k) center and r radius is given by:

$$(x-h)^2 + (y-k)^2 = r^2$$

This is the standard form of the equation.Thus, if we know the coordinates of the center of the circle and its radius as well, we can easily find its equation.

Example:

1. Consider a circle whose center is at the origin and radius is equal to 8 units.

Solution:

Given: Centre is (0, 0), radius is 8 units.

We know that the equation of a circle when the center is origin:

$$\mathbf{x^2 + y^2 = a^2}$$

For the given condition, the equation of a circle is given as

$$x^2 + y^2 = 8^2$$

$x^2 + y^2 = 64$, which is the equation of a circle

2. Find the equation of the circle whose center is (3,5) and the radius is 4 units.

Solution:

Here, the center of the circle is not an origin.

Therefore, the general equation of the circle is,

$(x-3)^2 + (y-5)^2 = 42$

$x^2 - 6x + 9 + y^2 - 10y + 25 = 16$

$x^2 + y^2 - 6x - 10y + 18 = 0$ is the equation of circle

3. Equation of a circle is $x^2+y^2-12x-16y+19=0$. Find the center and radius of the circle.

Solution:

Given equation is of the form $x^2 + y^2 + 2gx + 2fy + c = 0$,

$2g = -12,\ 2f = -16, c = 19$

$g = -6, f = -8$

Centre of the circle is (6,8)

Radius of the circle = $\sqrt{[(-6)^2 + (-8)^2 - 19]} = \sqrt{[100 - 19]} = \sqrt{81} = 9$ units.

Therefore, the radius of the circle is 9 units.

1.20. Equation of sphere:

A sphere is a geometrical object in three-dimensional space that resembles the surface of a ball. Similar to a circle in two-dimensional space, a sphere can be mathematically defined as the set of all points that are at the same distance from a given point. This given point is called the center of the sphere. The distance between the center and any point on the surface of the sphere is called the radius, represented by r.

$$x^2 + y^2 + z^2 = r^2$$

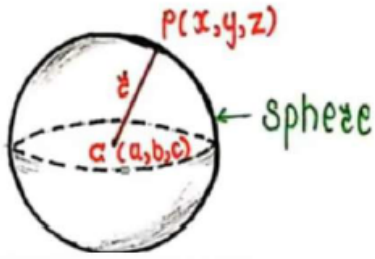
This is called the equation of a sphere, also known as the general equation of a sphere or the equation of a sphere through the circle.

i) Center - Radius form

Let $P(x, y, z)$ be any point on the sphere, $C(a, b, c)$ be the centre of the sphere and r be the radius.

Then by distance formula,

$$(x-a)^2 + (y-b)^2 + (z-c)^2 = r^2$$



ii) **General Form**

Let $P(x, y, z)$ be any point on the sphere, $C(a, b, c)$ be the centre of the sphere and r be the radius.

Then equation of sphere by Centre-Radius form is:

$$(x-a)^2+(y-b)^2+(z-c)^2=r^2$$

$$\Rightarrow x^2+y^2+z^2-2ax-2by-2cz+(a^2+b^2+c^2-r^2)=0$$

Put $a=-u, b=-v, c=-w$ & $d=a^2+b^2+c^2-r^2$

∴ Equation of the sphere becomes,

$$x^2+y^2+z^2+2ux+2vy+2wz+d=0$$

Centre $C\equiv(a,b,c)\equiv(-u,-v,-w)$ & Radius: $r=\sqrt{u^2+v^2+w^2-d}$

1.21. Equation of hypersphere

- A hypersphere is a four-dimensional analog of a sphere; also known as a 4-sphere.
- The intersection of a sphere with a plane is a circle; the intersection of a hypersphere with a hyperplane is a sphere. These analogies are reflected in the underlying mathematics.
- $x^2 + y^2 = r^2$ is the Cartesian equation of a circle of radius r; $x^2 + y^2 + z^2 = r^2$ is the corresponding equation of a sphere; $x^2 + y^2 + z^2 + w^2 = r^2$ is the equation of a hypersphere, where w is measured along a fourth dimension at right angles to the x-, y-, and z-axes.
- The n-hypersphere (often simply called the n-sphere) is a generalization of the circle (called by geometers the 2-sphere) and usual sphere (called by geometers the 3-sphere) to dimensions n>=4. The n-sphere is therefore defined (again, to a geometer; see below) as the set of n-tuples of points $(x_1, x_2, ..., x_n)$ such that

  $$x_1^2+x_2^2+...+x_n^2=R^2,$$

  where R is the radius of the hypersphere.
- The hypersphere has a **hypervolume** (analogous to the volume of a sphere) of $\pi^2 r^4/2$, and a surface volume (analogous to the sphere's surface area) of $2\pi^2 r^3$.
- A solid angle of a hypersphere is measured in **hypersteradians**, of which the hypersphere contains a total of $2\pi^2$. The apparent pattern of 2π **radians** in a circle and 4π **steradians** in a sphere does not continue with 8π hypersteradians because the n-volume, n-area, and number of n-radians of an n-sphere are all related to gamma function and the way it can cancel out powers of π halfway between integers. In general, the term "hypersphere" may be used to refer to any n-sphere.

TEXT / REFERENCE BOOKS

1. Cathy O'Neil and Rachel Schutt. Doing Data Science, Straight Talk From The Frontline. O'Reilly. 2014.
2. Introduction to Linear Algebra - By Gilbert Strang, Wellesley-Cambridge Press, 5th Edition.2016.
3. Avrim Blum, John Hopcroft and Ravindran Kannan. Foundations of Data Science.

QUESTION BANK

Part-A			
Q.No	**Questions**	**Competence**	**BT Level**
1.	Define Data science.	Remember	BTL 1
2.	Distinguish between between a vector and a scalar point	Analysis	BTL 4
3.	List out the applications of data science.	Remember	BTL 1
4.	Differentiate between matrices and tensors.	Analysis	BTL 4
5.	When two vector will become perpendicular?	Analysis	BTL 4
6.	Define Linear algebra.	Understand	BTL 2
7.	Illustrate relationship diagram between data science and other branch of studies.	Analysis	BTL 4
8.	Illustrate the geometrical representation of a vector point.	Analysis	BTL 4
9.	Is all orthogonal vector are perpendicular vectors? Justify your answer.	Analysis	BTL 4
10.	Given two points A and B with coordinates (3,5) and (6,8) respectively. What is the distance between A and B?	Apply	BTL 3
11.	. Define 1D vector?	Understand	BTL 2
12.	Define 2D vector?	Understand	BTL 2
13.	Enumerate null space.	Understand	BTL 2
14.	Define Rank of a matrix with an example.	Understand	BTL 2
15.	Describe Rank-Nullity theorem.	Understand	BTL 2
16.	Differentiate between underdetermined and overdetermined set of equations.	Analysis	BTL 4
17.	When a set of equations can be termed as balanced system?	Analysis	BTL 4
18.	Compare and contrast hyperplane and halfspace.	Analysis	BTL 4
19.	Find out the Eigen value for the matrix $\begin{bmatrix} -5 & 2 \\ -7 & 4 \end{bmatrix}$	Apply	BTL 3

20.	Write the equation of a circle when the center is at the origin?	Apply	BTL 3
	PART B		
Q.No	**Questions**	**Competence**	**BT Level**
1.	Find the inverse of the given matrix A = $\begin{bmatrix} 4 & -2 & 1 \\ 5 & 0 & 3 \\ -1 & 2 & 6 \end{bmatrix}$	Apply	BTL 3
2.	Find the rank and nullity of the given matrix $\begin{bmatrix} 1 & 2 & 1 \\ -2 & -3 & 1 \\ 3 & 5 & 0 \end{bmatrix}$	Apply	BTL 3
3.	Calculate the Null Space for the matrix A $\begin{bmatrix} 1 & 2 & 1 & 1 \\ 1 & 2 & 2 & -1 \\ 2 & 4 & 0 & 6 \end{bmatrix}$	Apply	BTL 3
4.	Find out the eigen values and eigen vectors for the given matrix $\begin{bmatrix} 8 & 7 \\ 2 & 3 \end{bmatrix}$	Apply	BTL 3
5.	Perform diagonal decomposition to the given matrix A to make the sub matrices. A = $\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$	Apply	BTL 3
6.	Do Singular Value Decomposition in the given matrix $\begin{bmatrix} 4 & 0 \\ 3 & -5 \end{bmatrix}$	Apply	BTL 3
7.	Calculate the pseudo inverse for the matrix B = $\begin{bmatrix} 1 & 2 & 3 \\ 2 & 4 & 6 \end{bmatrix}$	Apply	BTL 3
8.	Find the equation of the sphere which passes through the points (2,1,1) and (0,3,2) and has its center on the line $2x + y + 3z = 0 = x + 2y + 2z$	Apply	BTL 3
9.	Explain about the Equation of line, plane, and hyperplane with an example?	Analysis	BTL 4
10.	Explain about the pseudo inverse with an example?	Analysis	BTL 4
11.	Explain 1D,2D and nD vectors?	Analysis	BTL 4

SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

UNIT – II – Probability and Statistics – SCSA3016

Unit -2
PROBABILITY AND STATISTICS

Introduction to probability and statistics, Population and sample, Normal and Gaussian distributions, Probability Density Function, Descriptive statistics, notion of probability, distributions, mean, variance, covariance, covariance matrix, understanding univariate and multivariate normal distributions, introduction to hypothesis testing, confidence interval for estimates.

1.1.What is Data?

Data is the information collected through different sources which can be qualitative or quantitative in nature. Mostly, the data collected is used to analyse and draw insights on a particular topic.

Types of Data.

Numerical Data

Numerical data is the information in numbers i.e., numeric which poses as a quantitative measurement of things.

For example:

1. Heights and weights of people
2. Stock Prices

a) Discrete Data

Discrete data is the information that often counts of some event i.e., can only take specific values. These are often integer-based, but not necessarily.

For example:

1. Number of times a coin was flipped
2. Shoe sizes of people

b) Continuous Data

Continuous Data is the information that has the possibility of having infinite values i.e., can take any value within a range.

For example:

How many centimeters of rain fell on a given day?

Categorical Data

This type of data is qualitative in nature which has no inherent mathematical significance. It is sort of a fixed value under which a unit of observation is assigned or “categorized”.

For example:

1. Gender

2. Binary Data (Yes/No)
3. Attributes of a vehicle like color, mileage, number of doors, etc.

1.2.What are Statistics?

The field of **Statistics** deals with the collection, presentation, analysis, and use of data to make decisions, solve problems, and design products and processes. **Statistics** is the science of learning from data, and of measuring, controlling, and communicating uncertainty; and it thereby provides the navigation essential for controlling the course of scientific and societal advances. In simple terms, statistics is the science of data. **Statistics is defined as collection, compilation, analysis and interpretation of numerical data.**

1.2.1. Statistics is the science of data

The most important aspect of any Data Science approach is how the information is processed. When we talk about developing insights out of data it is basically digging out the possibilities. Those possibilities in Data Science are known as **Statistical Analysis**. Most of us wonder how can data in the form of text, images, videos, and other highly unstructured formats get easily processed by Machine Learning models. But the truth is we actually convert that data into a numerical form which is not exactly our data but the numerical equivalent of it. So, this brings us to the very important aspect of Data Science. With data in numerical format, it provides us with infinite possibilities to understand the information out it. Statistics acts as a pathway to understand your data and process that for successful results. Not only the power of statistics is limited to understanding the data it also provides methods to measure the success of our insights, getting different approaches for the same problem, getting the right mathematical approach for your data.

- In an agricultural study, researchers want to know which of four fertilizers (which vary in their nitrogen contents) produces the highest corn yield. In a clinical trial, physicians want to determine which of two drugs is more effective for treating HIV in the early stages of the disease. In a public health study, epidemiologists want to know whether smoking is linked to a particular demographic class in high school students.
- To develop an appreciation for variability and how it effects product, process and system.
- It is estimating the present; predicting the future
- Study methods that can be used to solve problems, build knowledge.
- Statistics make data into information
- Develop an understanding of some basic ideas of statistical reliability, stochastic process (probability concepts).
- Statistics is very important in every aspect of society (Govt., People or Business)

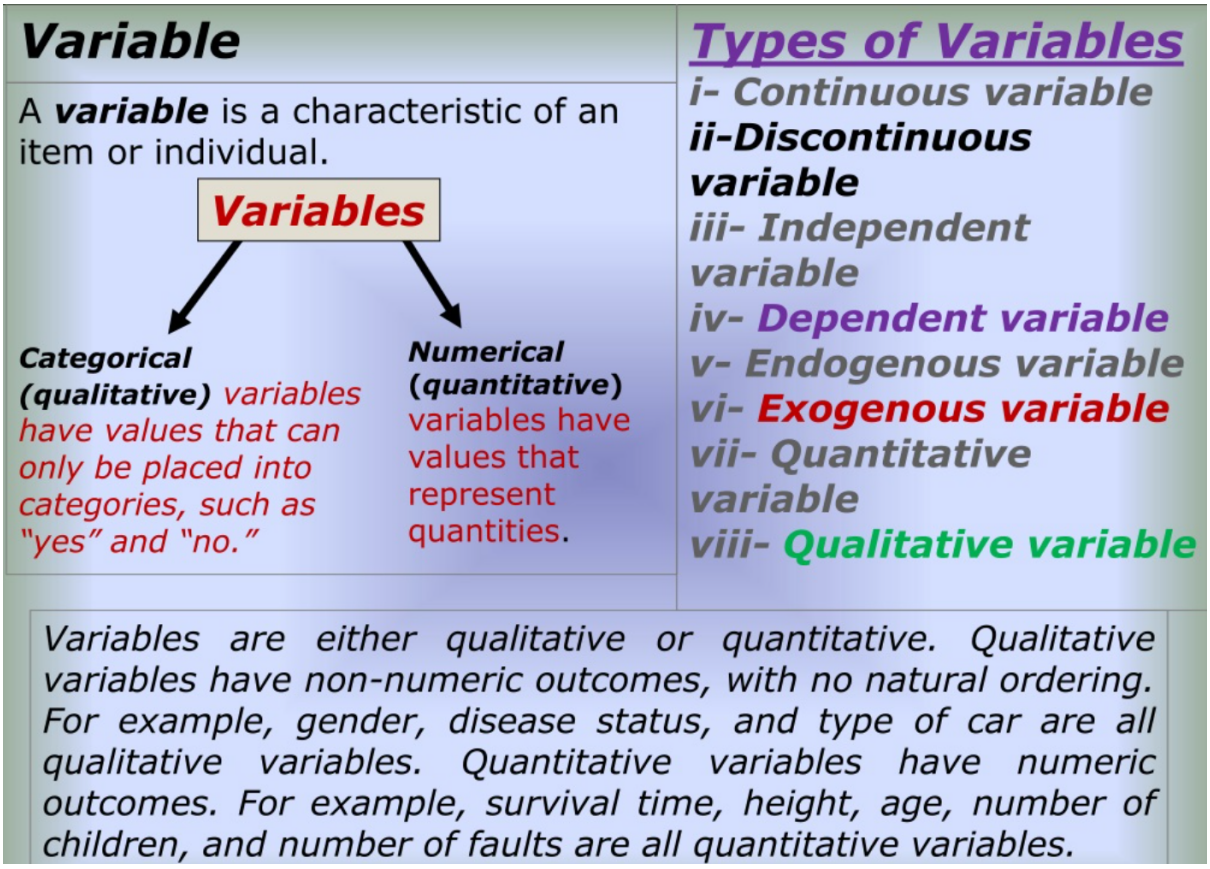
1.2.2. Basic terms

Variable: Property with respect to which data from a sample differ in some measurable way
Measurement: assignment of numbers to something
Data: collection of measurements
Population: all possible data

Sample: collected data

1. Variable

A **variable** is a characteristic or condition that can change or take on different values. Most research begins with a general question about the relationship between two variables for a specific group of individuals.



Types of Variables

Variables can be classified as discrete or continuous. **Discrete variables** (such as class size) consist of indivisible categories, and **continuous variables** (such as time or weight) are infinitely divisible into whatever units a researcher may choose. For example, time can be measured to the nearest minute, second, half-second, etc.

2. Measuring Variables

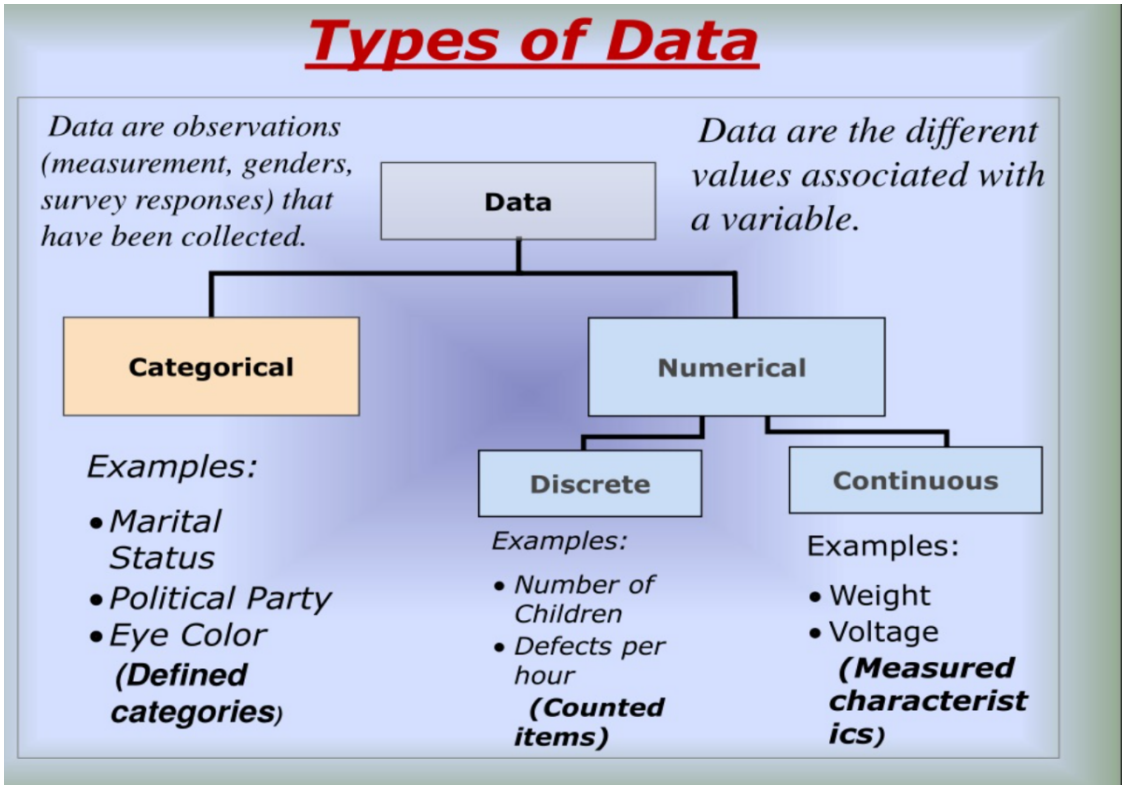
To establish relationships between variables, researchers must observe the variables and record their observations. This requires that the variables be **measured**. The process of measuring a variable requires a set of categories called a **scale of measurement** and a process that classifies each individual into one category.

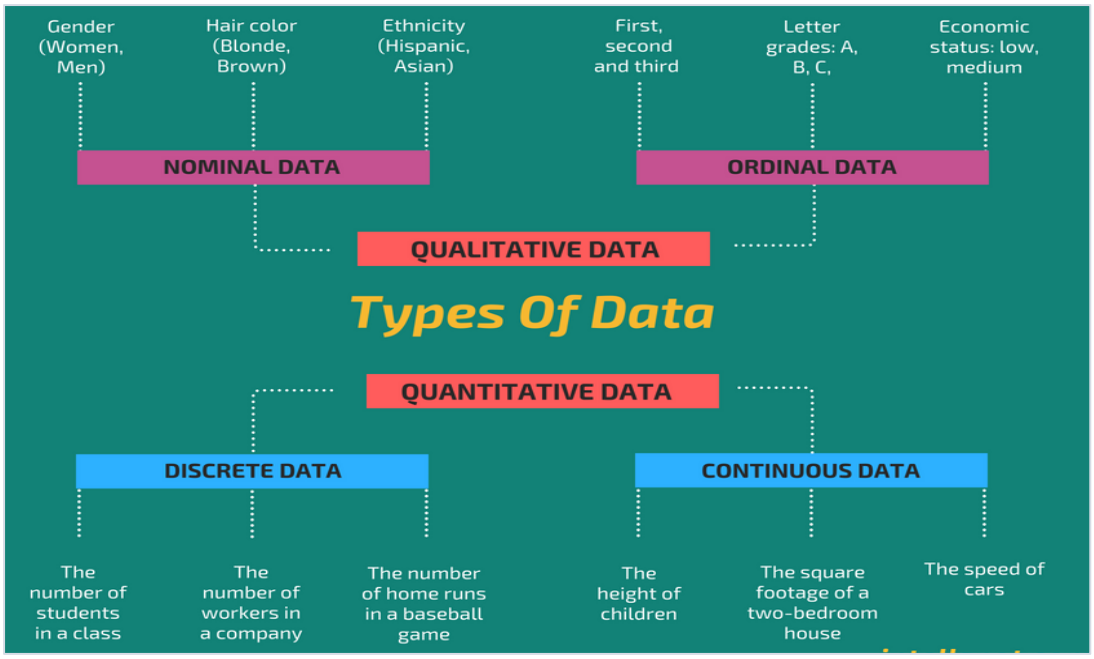
4 Types of Measurement Scales

A **nominal scale** is an unordered set of categories identified only by name. Nominal measurements only permit you to determine whether two individuals are the same or different.
An **ordinal scale** is an ordered set of categories. Ordinal measurements tell you the direction of difference between two individuals.
An **interval scale** is an ordered series of equal-sized categories. Interval measurements identify the direction and magnitude of a difference. The zero point is located arbitrarily on an interval scale.
A **ratio scale** is an interval scale where a value of zero indicates none of the variable. Ratio measurements identify the direction and magnitude of differences and allow ratio comparisons of measurements.

3. Data

The measurements obtained in a research study are called the **data**. The goal of statistics is to help researchers organize and interpret the data.





Quantitative

The data which are statistical or numerical are known as Quantitative data. Quantitative data is generated through. Quantitative data is also known as Structured data. Experiments, Tests, Surveys, Market Report. Quantitative data is again divided into **Continuous data and Discrete data.**

Continuous Data

Continuous data is the data which can have any value. That means Continuous data can give infinite outcomes so it should be grouped before representing on a graph.

Examples

- The speed of a vehicle as it passes a checkpoint
- The mass of a cooking apple
- The time taken by a volunteer to perform a task

Discrete Data

- Discrete data can have certain values. That means only a finite number can be categorized as discrete data.
- Numbers of cars sold at a dealership during a given month
- Number of houses in certain block
- Number of fish caught on a fishing trip
- Number of complaints received at the office of airline on a given day
- Number of customers who visit at bank during any given hour
- Number of heads obtained in three tosses of a coin

Differences between Discrete and Continuous data

- Numerical data could be either discrete or continuous

- Continuous data can take any numerical value (within a range); For example, weight, height, etc.
- There can be an infinite number of possible values in continuous data
- Discrete data can take only certain values by finite 'jumps', i.e., it 'jumps' from one value to another but does not take any intermediate value between them (For example, number of students in the class

Qualitative

Data that deals with description or quality instead of numbers are known as Quantitative data. Qualitative data is also known as **unstructured data.** Because this type of data is loosely compact and can't be analyzed conventionally.

4. **Population**

The entire group of individuals is called the **population**. For example, a researcher may be interested in the relation between class size (variable 1) and academic performance (variable 2) for the population of third-grade children.

5. **Sample**

Usually, **populations** are so large that a researcher cannot examine the entire group. Therefore, a **sample** is selected to represent the population in a research study. The goal is to use the results obtained from the sample to help answer questions about the population.

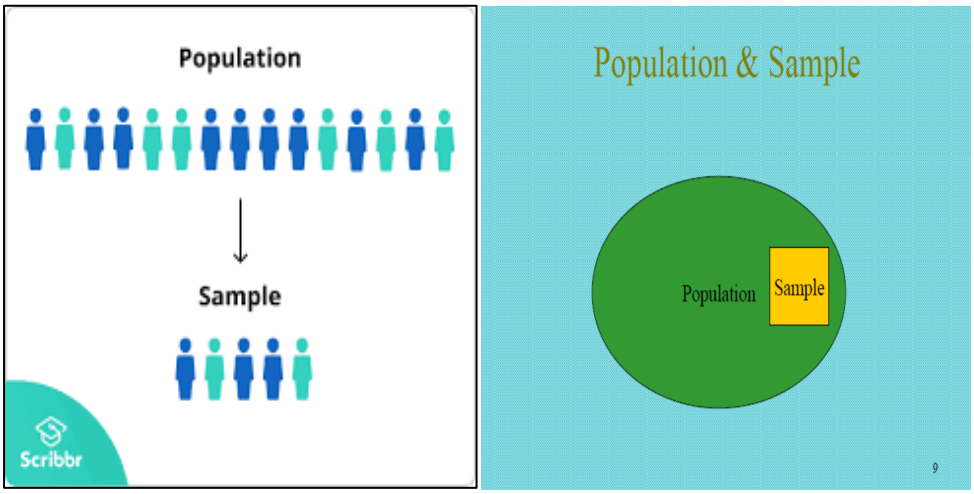
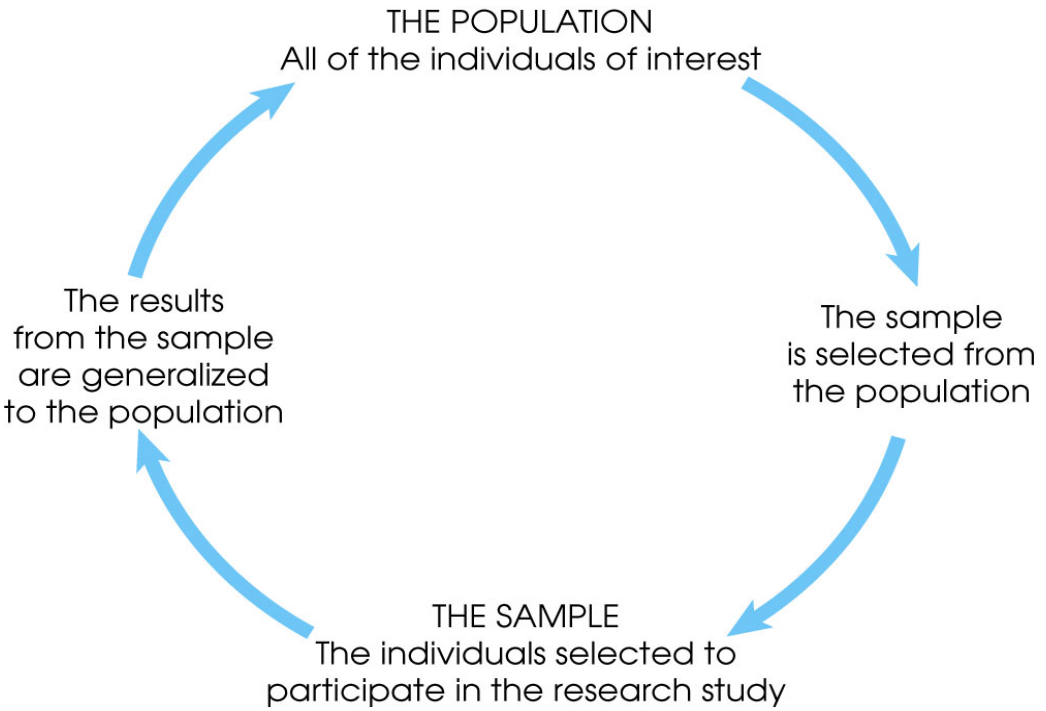


Figure 2.1: Population and Sample

POPULATION	SAMPLE
▪ The measurable quality is called a parameter.	▪ The measurable quality is called a statistic.
▪ The population is a complete set.	▪ The sample is a subset of the population.
▪ Reports are a true representation of opinion.	▪ Reports have a margin of error and confidence interval.
▪ It contains all members of a specified group.	▪ It is a subset that represents the entire population.

Sampling Error

- The discrepancy between a sample statistic and its population parameter is called **sampling error**.
- Defining and measuring sampling error is a large part of inferential statistics.



1.3.Frequency Distribution

Frequency Distribution (or Frequency Table)

Shows how a data set is partitioned among all of several categories (or classes) by listing all of the categories along with the number (frequency) of data values in each of them

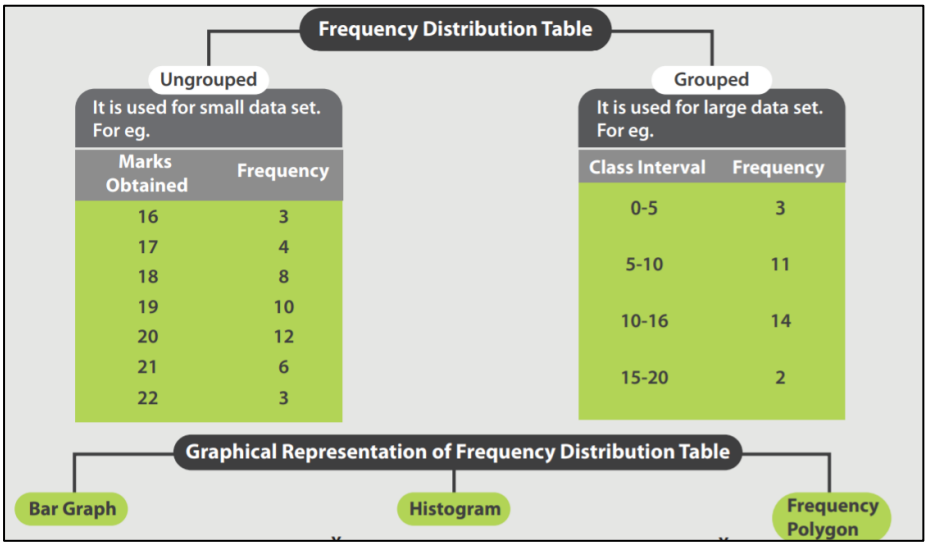
Frequency Distribution

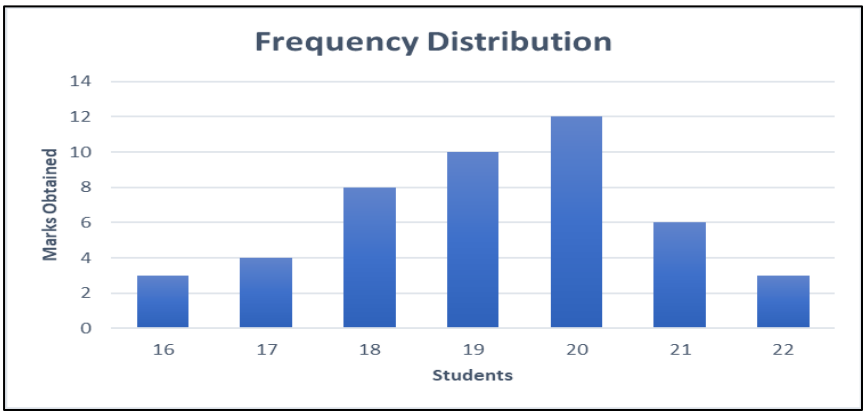
When data are in original form, they are called raw data

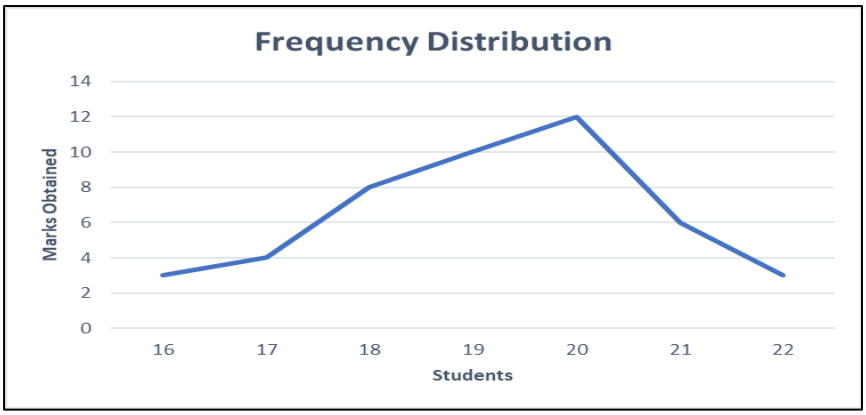
Organizing Data:

Categorical distribution

Grouped distribution
Ungrouped distribution





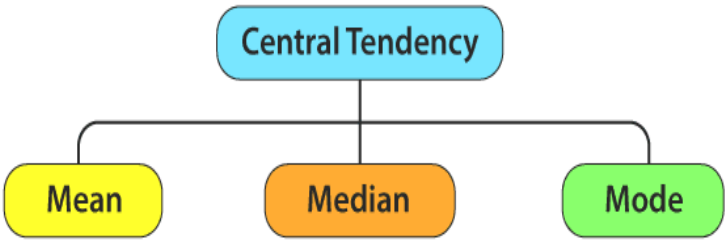


Frequency distribution refers to data classified on the basis of some variable that can be measured such as prices, weight, height, wages etc.

Weights (in kg)	Number of students
31 - 35	9
36 - 40	5
41 - 45	14
46 - 50	3
51 - 55	1
56 - 60	2
61 - 65	2
66 - 70	1
71 - 75	1
Total	38

Measures of Centre Tendency

- In statistics, the **central tendency** is the descriptive summary of a data set.
- Through the single value from the dataset, it reflects the centre of the data distribution.
- Moreover, it does not provide information regarding individual data from the dataset, where it gives a summary of the dataset. Generally, the central tendency of a dataset can be defined using some of the measures in statistics.



Mean

- The mean represents the average value of the dataset.
- It can be calculated as the sum of all the values in the dataset divided by the number of values. In general, it is considered as the **arithmetic mean**.
- Some other measures of mean used to find the central tendency are as follows:
- **Geometric Mean** (nth root of the product of n numbers)
- **Harmonic Mean** (the reciprocal of the average of the reciprocals)
- **Weighted Mean** (where some values contribute more than others)
- It is observed that if all the values in the dataset are the same, then all geometric, arithmetic and harmonic mean values are the same. If there is variability in the data, then the mean value differs.

Calculating the Mean

Calculate the mean of the following data:

1 5 4 3 2

Sum the scores (ΣX):

1 + 5 + 4 + 3 + 2 = 15

Divide the sum (ΣX = 15) by the number of scores (N = 5):

15 / 5 = 3

Mean = X = 3

The Median

- The *median* is simply another name for the 50th percentile
- Sort the data from highest to lowest
- Find the score in the middle
- If N, the number of scores, is even the median is the average of the middle two scores

Median Example

What is the median of the following scores:

10 8 14 15 7 3 3 8 12 10 9

Sort the scores:

15 14 12 10 10 9 8 8 7 3 3

Determine the middle score:

middle = (N + 1) / 2 = (11 + 1) / 2 = 6

Middle score = median = 9

Median Example

What is the median of the following scores:

24 18 19 42 16 12

- Sort the scores:

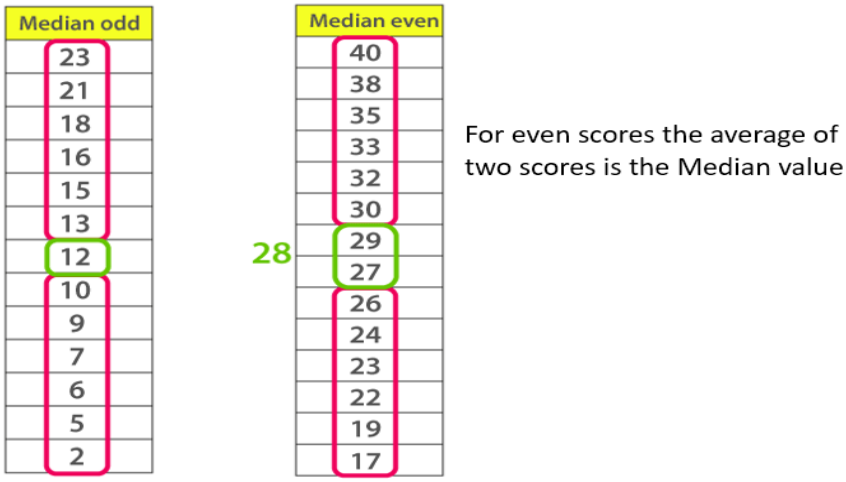
  42 24 19 18 16 12

- Determine the middle score:

  middle = (N + 1) / 2 = (6 + 1) / 2 = 3.5

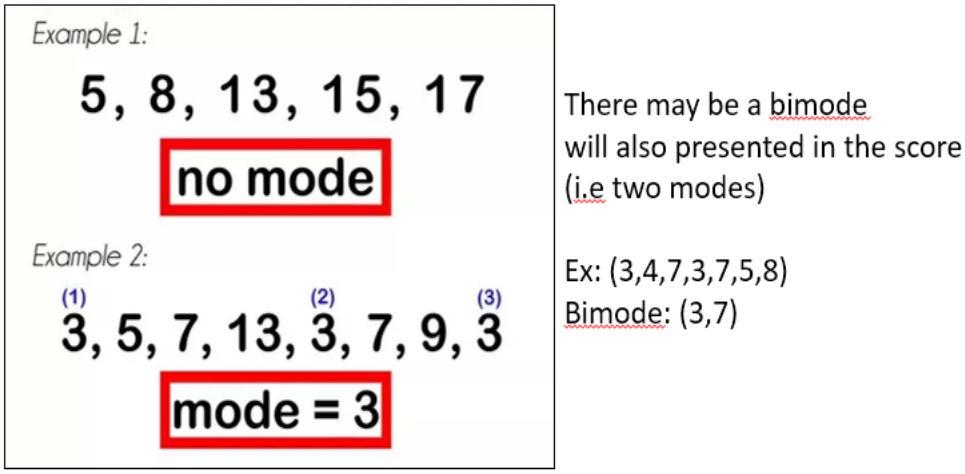
- Median = average of 3rd and 4th scores:

  (19 + 18) / 2 = 18.5



Mode

The *mode* is the score that occurs most frequently in a set of data.



Variance

- **Variance** is the average squared deviation from the mean of a set of data.
- It is used to find the Standard deviation.
- $\sigma^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2$

Variance

- This is a good measure of how much variation exists in the sample, normalized by sample size.
- It has the nice property of being additive.
- The only problem is that the variance is measured in units squared

How to find Variance

- Find the **Mean** of the data.
- Subtract the mean from each value – the result is called the **deviation from the mean.**

- Square each deviation of the mean.
- Find the sum of the squares.
- Divide the total by the number of items.

How to find Variance? - Example

- Suppose you're given the data set 1, 2, 2, 4, 6. (X = 1,2,2,4,6) One Variable X
- Calculate the mean of your data set. The mean of the data is (1+2+2+4+6)/5
- Mean= 15/5 = 3.
- Subtract the mean from each of the data values and list the differences. Subtract 3 from each of the values 1, 2, 2, 4, 6
- 1-3 = -2 2-3 = -1 2-3 = -1 4-3 = 1 6-3 = 3
- Your list of differences is -2, -1, -1, 1, 3 (deviation)
- You need to square each of the numbers -2, -1, -1, 1, 3
  $(-2)^2 = 4$, $(-1)^2 = 1$, $(-1)^2 = 1$, $(1)^2 = 1$, $(3)^2 = 9$
- Your list of squares is 4, 1, 1, 1, 9, Add the squares 4+1+1+1+9 = 16
- Subtract one from the number of data values you started with. You began this process (it may seem like a while ago) with five data values. One less than this is 5-1 = 4.
- Divide the sum from step four by the number from step five. The sum was 16, and the number from the previous step was 4. You divide these two numbers 16/4 = 4.

Variation in one variable

- So, these four measures all describe aspects of the variation in a single variable:
- a. Sum of the squared deviations
- b. Variance
- c. Standard deviation
- d. Standard error
- Can we adapt them for thinking about the way in which two variables might vary together?

Covariance

- In mathematics and statistics, **covariance** is a measure of the relationship between two random variables. (X, Y)
- More precisely, covariance refers to **the measure of how two random variables in a data set will change together**.
- **Positive covariance**: Indicates that two variables tend to move in the same direction.
- **Negative covariance**: Reveals that two variables tend to move in inverse directions.

- The covariance between two random variables X and Y can be calculated using the following **formula** (for **population**):

$$\text{Cov}(X, Y) = \frac{\sum(X_i - \bar{X})(Y_j - \bar{Y})}{n}$$

- For a **sample** covariance, the formula is slightly adjusted:

$$\text{Cov}(X, Y) = \frac{\sum(X_i - \bar{X})(Y_j - \bar{Y})}{n - 1}$$

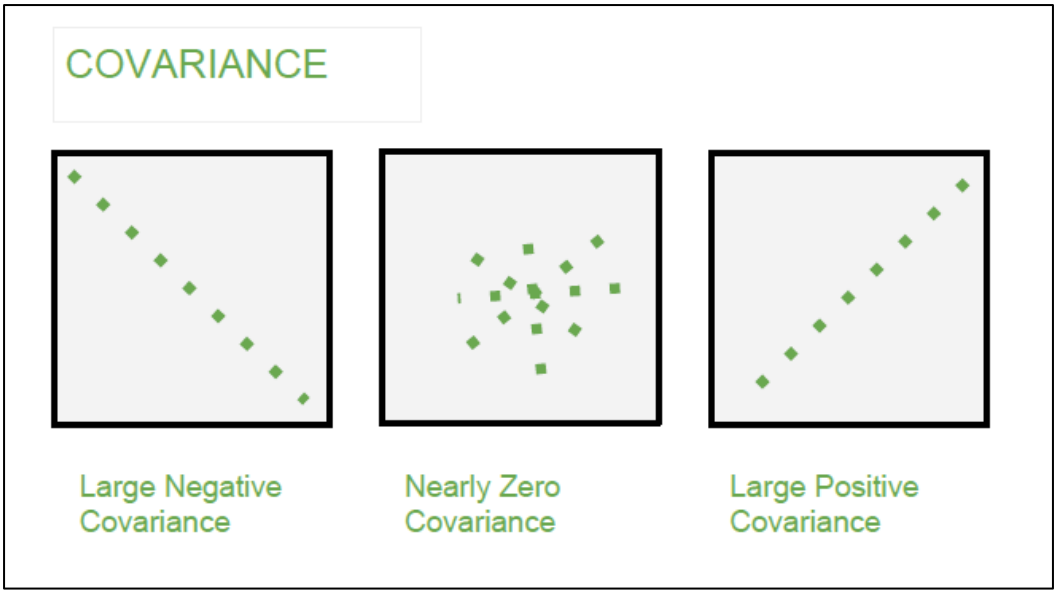
Where:
X_i – the values of the X-variable
Y_j – the values of the Y-variable
$\bar{X}$ – the mean (average) of the X-variable
$\bar{Y}$ – the mean (average) of the Y-variable
n – the number of data points

Covariance Example

Example 1: Find covariance for following data set (Two Variables X and Y)
X = {2,5,6,8,9}, Y = {4,3,7,5,6}

Solution:

- Given data sets X = {2,5,6,8,9}, Y = {4,3,7,5,6} and N = 5
- **Mean(X)** = (2 + 5 + 6 + 8 + 9) / 5 = 30 / 5 = **6**
- **Mean(Y)** = (4 + 3 +7 + 5 + 6) / 5 = 25 / 5 = **5**
- **Sample covariance Cov(X,Y) = $\sum(X_i - X) \times (Y_i - Y)/ (N - 1)$**
- = [(2 - 6)(4 - 5) + (5 - 6)(3 - 5) + (6 - 6)(7 - 5) + (8 - 6)(5 - 5) +
- (9 - 6)(6 - 5)] / 5 - 1
- = 4 + 2 + 0 + 0 + 3 / 4 = 9 / 4 = **2.25**
- **Population covariance Cov(X,Y) = $\sum(X_i - X) \times (Y_i - Y)/ (N)$**
- = [(2 - 6)(4 - 5) + (5 - 6)(3 - 5) + (6 - 6)(7 - 5) + (8 - 6)(5 - 5) + (9 - 6)(6 - 5)] / 5
- = 4 + 2 + 0 + 0 + 3 /5
- = 9 / 5
- = 1.8
- **Answer:** The **sample covariance** is 2.25 and the **population covariance** is 1.8
- **Positive and Negative Covariance**



Covariance Matrix

- The **covariance matrix** is a math concept that occurs in several areas of machine learning.If you have a set of n numeric data items, where each data item has d dimensions, then the covariance matrix is a d-by-d symmetric square matrix where there are variance values on the diagonal and covariance values off the diagonal.
- Suppose you have a set of n=5 data items, representing 5 people, where each data item has a Height (X), test Score (Y), and Age (Z) (therefore d = 3):
- Covariance Matrix

	X	Y	Z
	Height	Score	Age
	64.0	580.0	29.0
	66.0	570.0	33.0
	68.0	590.0	37.0
	69.0	660.0	46.0
	73.0	600.0	55.0
mean =	68.0	600.0	40.0
	n=5		

Covariance Matrix

- The covariance matrix for this data set is:

```
    X         Y         Z
X   11.50     50.00    34.75
Y   50.00   1250.00   205.00
Z   34.75    205.00   110.00
```

- The 11.50 is the variance of X, 1250.0 is the variance of Y, and 110.0 is the variance of Z. For variance, in words, subtract each value from the dimension mean. Square, add them up, and divide by n-1. For example, for X:
- Var(X) = [(64–68.0)^2 + (66–68.0^2 + (68-68.0)^2 + (69-68.0)^2 +(73-68.0)^2] / (5-1) = (16.0 + 4.0 + 0.0 + 1.0 + 25.0) / 4 = 46.0 / 4 = 11.50.

Covariance Matrix

- Covar(XY) =
- [(64-68.0)*(580-600.0) + (66-68.0)*(570-600.0) + (68-68.0)*(590-600.0) + (69-68.0)*(660-600.0) + (73-68.0)*(600-600.0)] / (5-1) =
- [80.0 + 60.0 + 0 + 60.0 + 0] / 4 =
- 200 / 4 = 50.0
- If you examine the calculations carefully, you'll see the pattern to compute the covariance of the XZ and YZ columns. And you'll see that Covar(XY) = Covar(YX).

Standard Deviation

- Variability is a term that describes how spread out a distribution of scores (or darts) is.
- **Variance** and **standard deviation** are closely related ways of measuring, or quantifying, variability.
- Standard deviation is simply the square root of variance
- **Find the mean** (or arithmetic average) of the scores. To find the mean, add up the scores and divide by n where n is the number of scores.
- **Find the sum of squared deviations** (abbreviated SSD). To get the SSD, find the sum of the squares of the differences (or deviations) between the mean and all the individual scores.
- **Find the variance**. If you are told that the set of scores constitute a population, divide the SSD by n to find the variance. If instead you are told, or can infer, that the set of scores constitute a sample, divide the SSD by (n – 1) to get the variance.
- **Find the standard deviation**. To get the standard deviation, take the square root of the variance.

How to find Standard Deviation – Example (in Population score)

- **Example 1: Find the SSD, variance, and standard deviation** for the following **population of scores**: 1, 2, 3, 4, 5 using the list of steps given above.
- **Find the mean**. The mean of these five numbers (the population mean) is (1+2+3+4+5)/5 = 15/5 = 3.
- Let's use the definitional formula for SSD for its calculation: SSD is the sum of the squares of the differences (squared deviations) between the mean and the individual scores. The squared deviations are $(3-1)^2$, $(3-2)^2$, $(3-3)^2$, $(3-4)^2$, and $(3-5)^2$. That is, 4, 1, 0, 1, and 4. The SSD is then 4 + 1 + 0 + 1 + 4 = 10.
- Divide SSD by n, since this is a population of scores, to **get the variance**. So the variance is 10/5 = 2.
- The standard deviation is the square root of the variance. So the standard deviation is the square root of 2. = $\sqrt{2}$ =1.4142
- For practice, let's also compute the SSD using the computational formula, $\sum_i (x_i)^2 - (1/N)(\sum_i x_i)^2$. $\sum_i (x_i)^2 = 12 + 22 + 32 + 42 + 52 = 1 + 4 + 9 + 16 + 25 = 55$. $(1/N)(\sum_i x_i)^2 = (1/5)(1 + 2 + 3 + 4 + 5)^2 = (1/5)(15^2) = 45$. So SSD = 55 – 45 = 10, just like before.

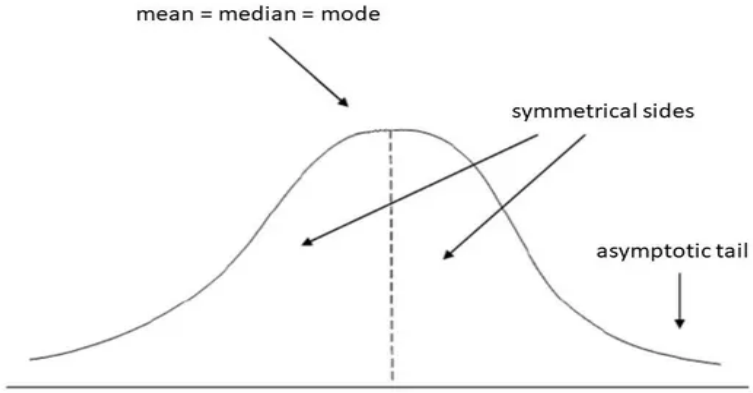
How to find Standard Deviation – Example (in Sample score)

- **Example 2:** Find the SSD, variance, and standard deviation for the following **sample of scores**: 1, 3, 3, 5.
- The average of these four numbers (the sample mean) is (1+3+3+5)/4 = 12/4 = 3.
- So, SSD = $(3-1)^2 + (3-3)^2 + (3-3)^2 + (3-5)^2 = 4 + 0 + 0 + 4 = 8$.
- Now, because we were told that these scores constitute a sample, we'll divide SSD by n-1 to get the sample variance.
- In our case we have four scores, so n = 4 so n-1 = 3. Therefore, our sample variance is 8/3.
- And the **sample standard deviation** is square root of 8/3 = $\sqrt{2.6}$ (SQRT 0F 2.6) = 1.6124

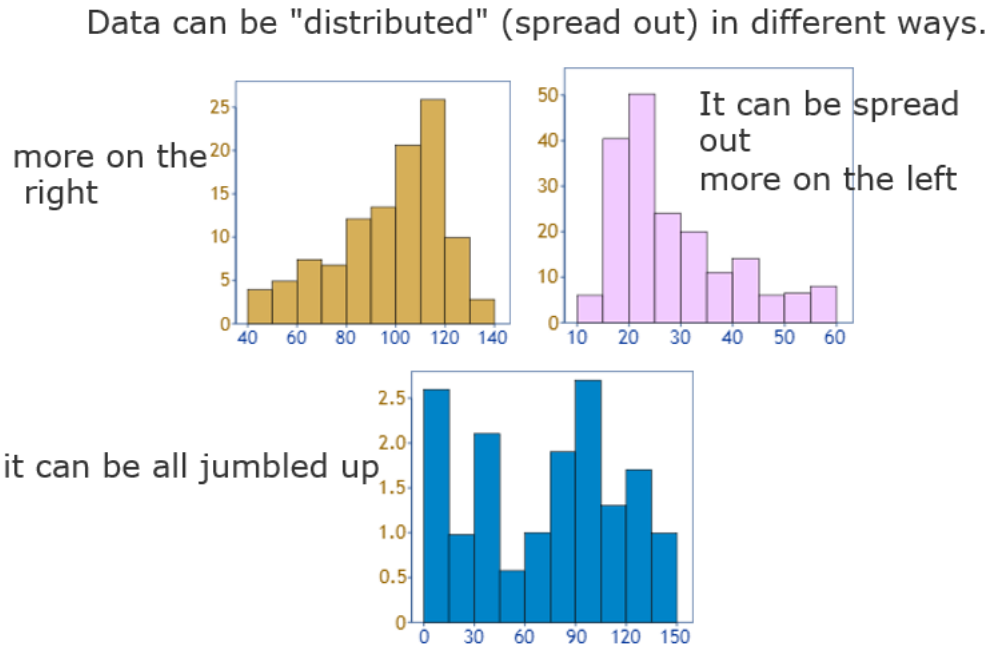
1.4.What is Distribution in statistics?

- A **distribution** is simply a collection of data, or scores, on a variable. Usually, these scores are arranged in order from smallest to largest and then they can be presented graphically.
- **A distribution** is an arrangement of values of a variable showing their observed or theoretical frequency of occurrence.
- A **bell curve** showing how the class did on our last exam would be an example of a distribution.

- All distributions can be characterized by the following two dimensions:
- **Central Tendency:** Mean, Median and Mode(s) of the distribution
- **Variability:** All distributions have a variance and standard deviation
- **Bell Curve**
- The term **bell curve** is used to describe the mathematical concept called normal distribution, sometimes referred to as Gaussian distribution.
- "Bell curve" refers to the bell shape that is created when a line is plotted using the data points for an item that meets the criteria of **normal distribution.**
- In a bell curve, the center contains the greatest number of a value and, therefore, it is the highest point on the arc of the line. This point is referred to the mean, but in simple terms, it is the highest number of occurrences of an element (in statistical terms, the mode).



Distribution



- But there are many cases where the data tends to be around a central value with no bias left or right, and it gets close to a **"Normal Distribution"** like this:

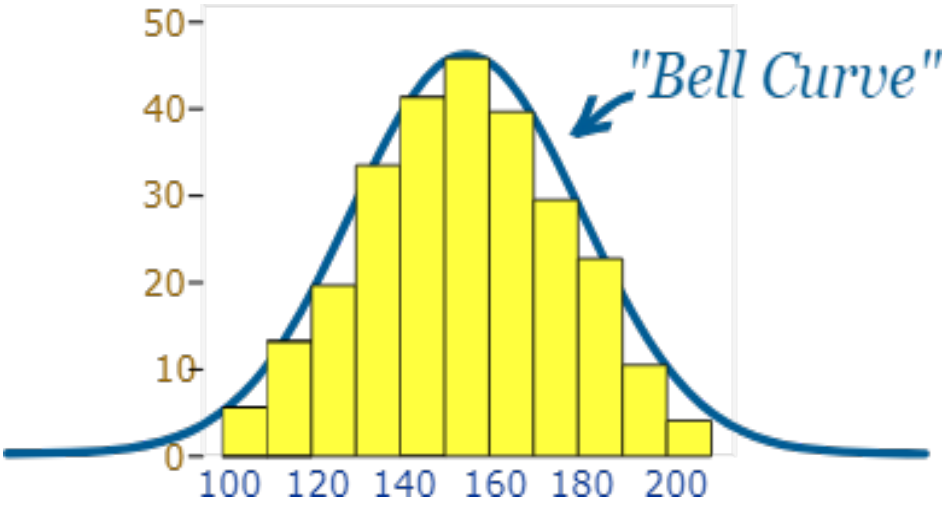


Figure 2.2: Bell cure

1.4.1. Normal Distribution

- **Normal distribution:** a bell-shaped, symmetrical distribution in which the mean, median and mode are all equal Z scores (also known as standard scores): the number of standard deviations that a given raw score falls above or below the mean Standard normal distribution: a normal distribution represented in z scores. The standard normal distribution always has a mean of zero and a standard deviation of one.
- The normal distribution is an important class of **Statistical Distribution** that has a wide range of applications. This distribution applies in most Machine Learning Algorithms and the concept of the Normal Distribution is a must for any **Statistician**, **Machine Learning Engineer**, and **Data Scientist.**

Parameters of Normal Distribution

- Mean
- Standard Deviation

Properties of Normal Distribution

- Symmetricity
- Measures of Central Tendencies are equal
- Empirical Rule
- Skewness and Kurtosis
- The area under the curve

Properties of Normal Distribution

- All forms of the normal distribution share the following characteristics:

1. It is symmetric

- The shape of the normal distribution is perfectly symmetrical.

- This means that the curve of the normal distribution can be divided from the middle and we can produce two equal halves. Moreover, the symmetric shape exists when an equal number of observations lie on each side of the curve.

2. The mean, median, and mode are equal

- The midpoint of normal distribution refers to the point with maximum frequency i.e., it consists of most observations of the variable.
- The midpoint is also the point where all three measures of central tendency fall. These measures are usually equal in a perfectly shaped normal distribution.

3. Empirical rule

- In normally distributed data, there is a constant proportion of data points lying under the curve between the mean and a specific number of standard deviations from the mean.
- Thus, for a normal distribution, almost all values lie within **3 standard deviations** of the mean.
- These check buttons of normal distribution will help you realize the appropriate percentages of the area under the curve.
- Remember that this empirical rule applies to all normal distributions. Also, note that these rules are applied only to the normal distributions.

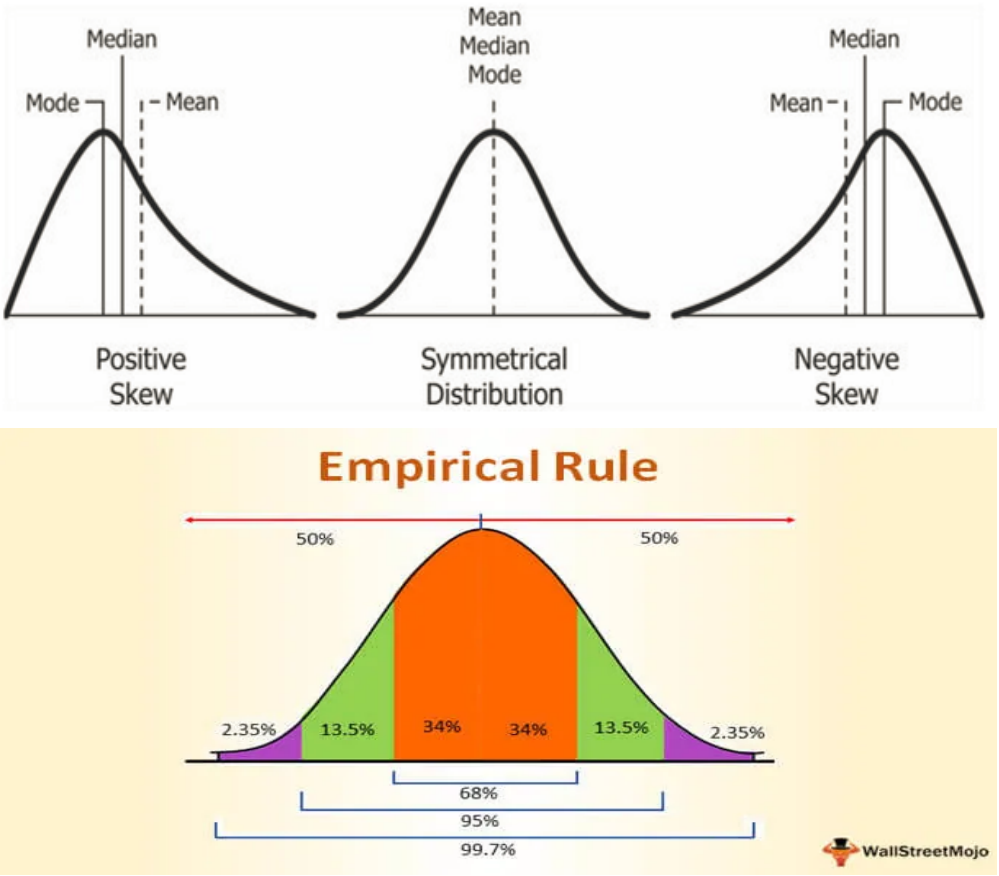
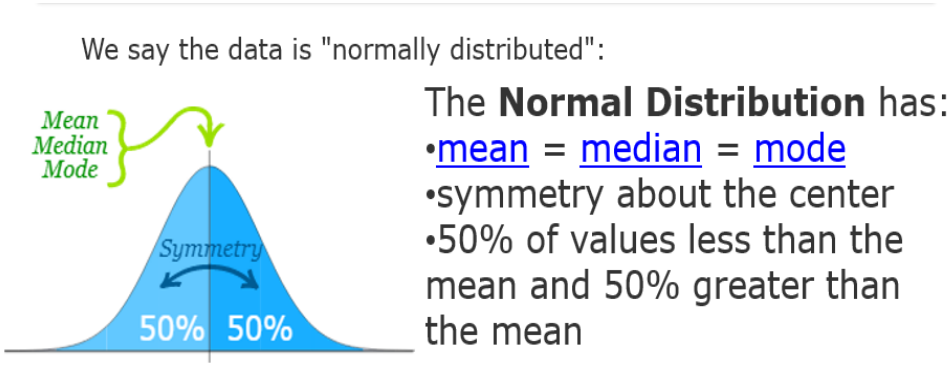


Figure 2.3: Empirical rule

Many things closely follow a Normal Distribution

Example:

- Heights of people
- Size of things produced by machines
- Errors in measurements
- Blood pressure
- Marks on a test



The normal distribution is a bell-shaped, symmetrical distribution in which the mean, median and mode are all equal. If the mean, median and mode are unequal, the distribution will be either positively or negatively skewed. Consider the illustration below:

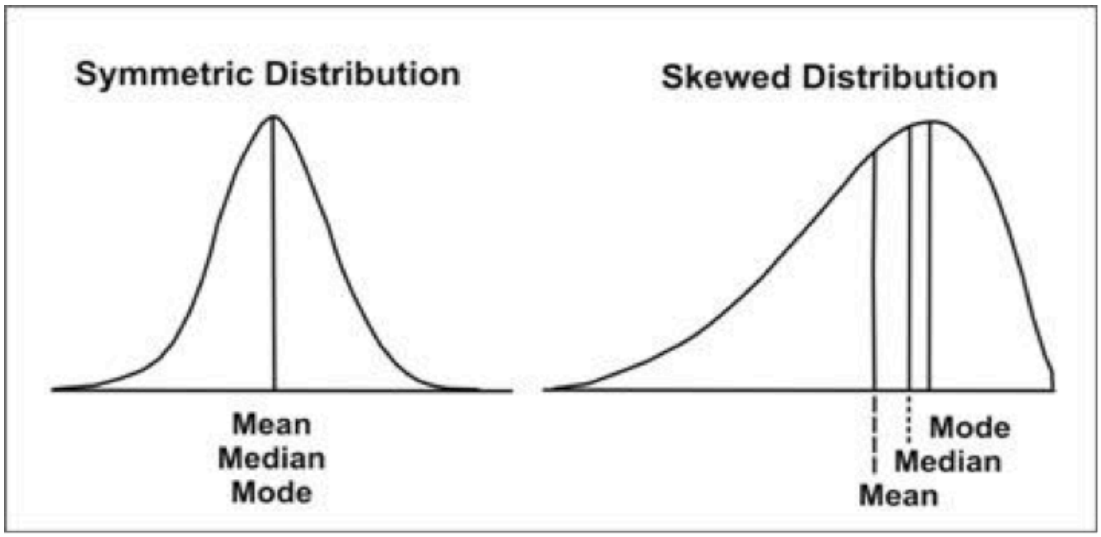


Figure 2.4: Symmetric and Skewed distribution

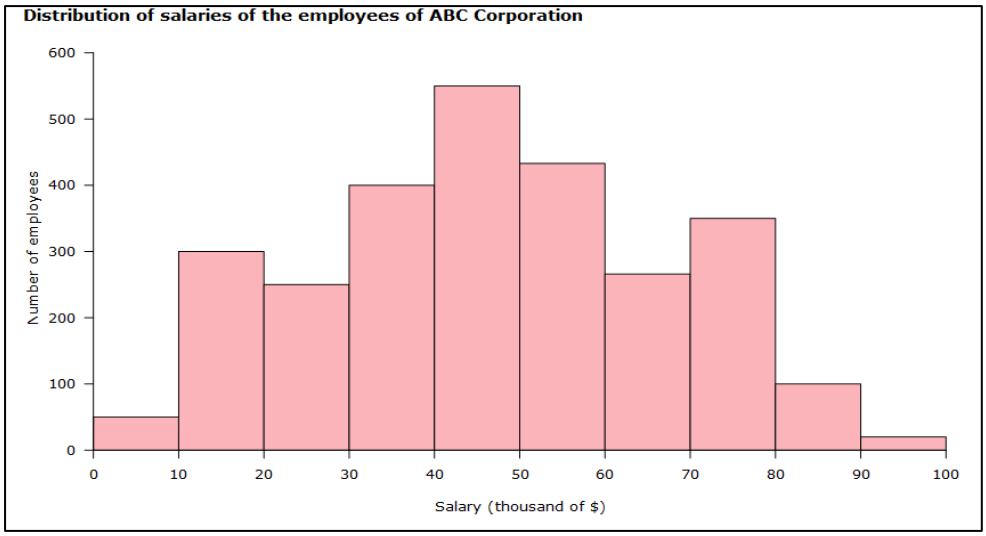
1.5.Probability Density

- Given a random variable, we are interested in the density of its probabilities.
- For example, given a random sample of a variable, we might want to know things like the shape of the probability distribution, the most likely value, the spread of values, and other properties.
- Knowing the probability distribution for a random variable can help to **calculate** moments of the distribution, like the **mean and variance**, but can also be useful for other more general

considerations, like determining whether an observation is unlikely or very unlikely and might be an outlier or anomaly.

- The problem is, we may not know the probability distribution for a random variable.
- We rarely do know the distribution because we don't have access to all possible outcomes for a random variable. In fact, all we have access to is a sample of observations. As such, we must select a probability distribution.
- This problem is referred to as probability density estimation, or **simply "*density estimation*,"** as we are using the observations in a random sample to estimate the general density of probabilities beyond just the sample of data, we have available.
- A random variable *x* has a probability distribution *p(x)*.
- The relationship between the outcomes of a random variable and its probability is referred to as the probability density, or simply the "*density*."
- If a random variable is continuous, then the probability can be calculated via probability density function, or PDF for short.
- The shape of the probability density function across the domain for a random variable is referred to as the probability distribution and common probability distributions have names, such as uniform, normal, exponential, and so on.
- There are a few **steps in the process of density estimation** for a random variable.
- The first step is to review the density of observations in the random sample with a simple histogram.
- From the histogram, we might be able to identify a common and well-understood probability distribution that can be used, such as a normal distribution. If not, we may have to fit a model to estimate the distribution.

Histogram



Density With a Histogram

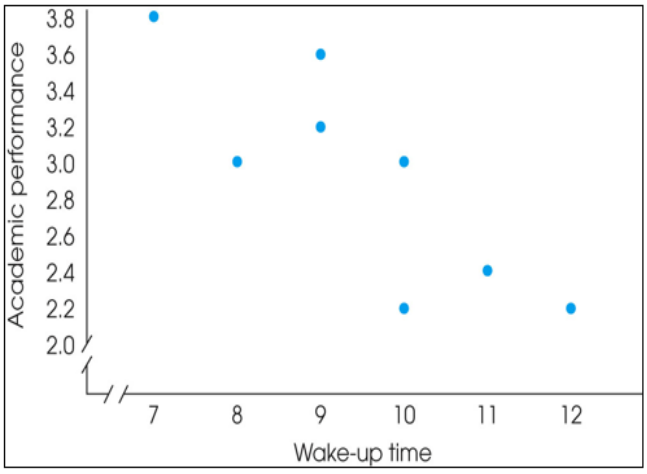
- The first step in density estimation is to create a histogram of the observations in the random sample.
- A histogram is a plot that involves first grouping the observations into bins and counting the number of events that fall into each bin.
- The counts, or frequencies of observations, in each bin are then plotted as a bar graph with the bins on the x-axis and the frequency on the y-axis.
- The choice of the number of bins is important as it controls the coarseness of the distribution (number of bars) and, in turn, how well the density of the observations is plotted.
- It is a good idea to experiment with different bin sizes for a given data sample to get multiple perspectives or views on the same data.

Correlational Studies

- The goal of a **correlational** study is to determine whether there is a relationship between two variables and to describe the relationship.
- A **correlational** study simply observes the two variables as they exist naturally.

Correlational Studies

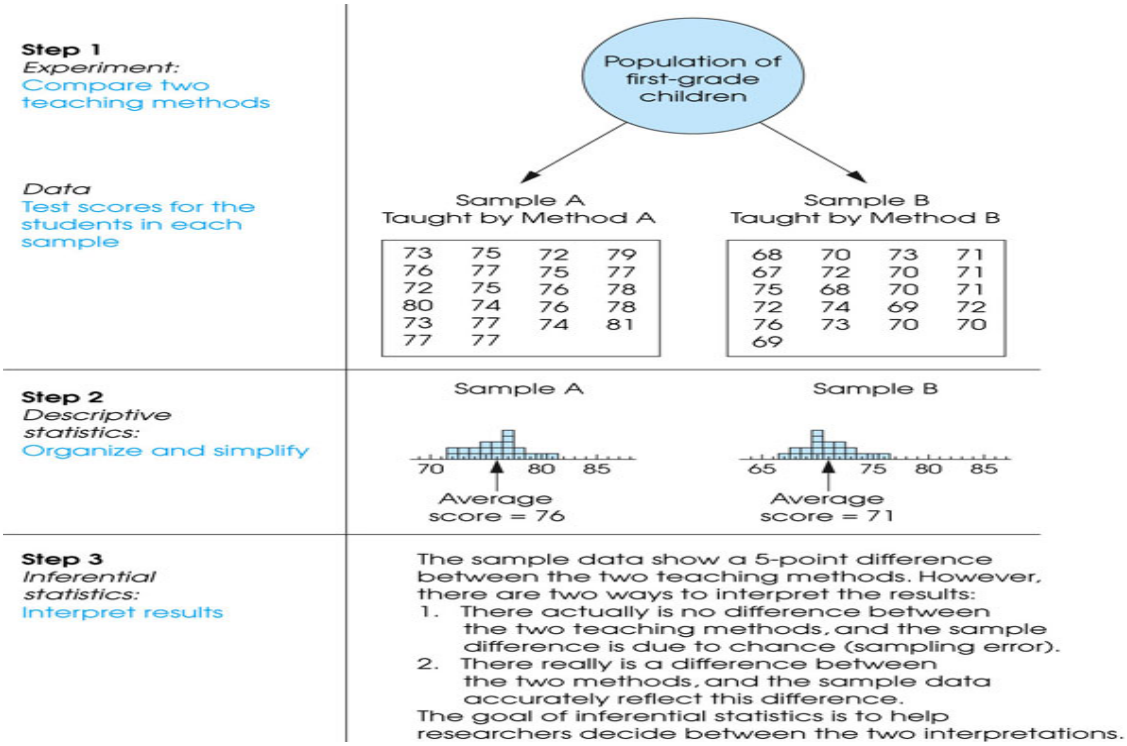
Child	Wake-up Time	Academic Performance
A	11	2.4
B	9	3.6
C	9	3.2
D	12	2.2
E	7	3.8
F	10	2.2
G	10	3.0
H	8	3.0



Experiment

- The goal of an **experiment** is to demonstrate a cause-and-effect relationship between two variables; that is, to show that changing the value of one variable causes change to occur in a second variable.
- In an **experiment**, one variable is manipulated to create treatment conditions.
- A second variable is observed and measured to obtain scores for a group of individuals in each of the treatment conditions.
- The measurements are then compared to see if there are differences between treatment conditions.
- All other variables are controlled to prevent them from influencing the results.

- In an experiment, the manipulated variable is called the **independent variable** and the observed variable is the **dependent variable**.



1.6. What Is a Probability Density Function (PDF)?

A probability distribution can be described in various forms, such as by a probability density function or a cumulative distribution function. Probability density functions, or PDFs, are mathematical functions that usually apply to continuous and discrete values. PDFs are very commonly used in statistical analysis, and thus are quite commonly used for Data Science. Generally, PDFs are a necessary tool when studying data with applied science using statistics. However, there are some PDFs that extend beyond this basic usage and have slightly different uses than one might be assume on first glance. For example, the PDF of the T distribution is often used to calculate a T-statistic. This T statistic, along with the degrees of freedom (n minus one) (v,) are then usually put into the regularized lower incomplete beta function, which happens to be the cumulative distribution function for the T distribution. While the absolute likelihood for a continuous random variable to take on any particular value is 0, the value of the PDF can be used to infer, in any particular sample of random variables, how much more likely it is statistically that the random variable would equal one sample compared to the other sample.

A function that defines the relationship between a random variable and its probability, such that you can find the probability of the variable using the function, is called a Probability Density Function (PDF) in statistics.

The different types of variables. They are mainly of two types:

1. Discrete Variable: A variable that can only take on a certain finite value within a specific range is called a discrete variable. It usually separates the values by a finite interval, e.g.,

a sum of two dice. On rolling two dice and adding up the resulting outcome, the result can only belong to a set of numbers not exceeding 12 (as the maximum result of a dice throw is 6). The values are also definite.

2. Continuous Variable: A continuous random variable can take on infinite different values within a range of values, e.g., amount of rainfall occurring in a month. The rain observed can be 1.7cm, but the exact value is not known. It can, in actuality, be 1.701, 1.7687, etc. As such, you can only define the range of values it falls into. Within this value, it can take on infinite different values.

Now, consider a continuous random variable x, which has a probability density function, that defines the range of probabilities taken by this function as f(x). After plotting the pdf, you get a graph as shown below:

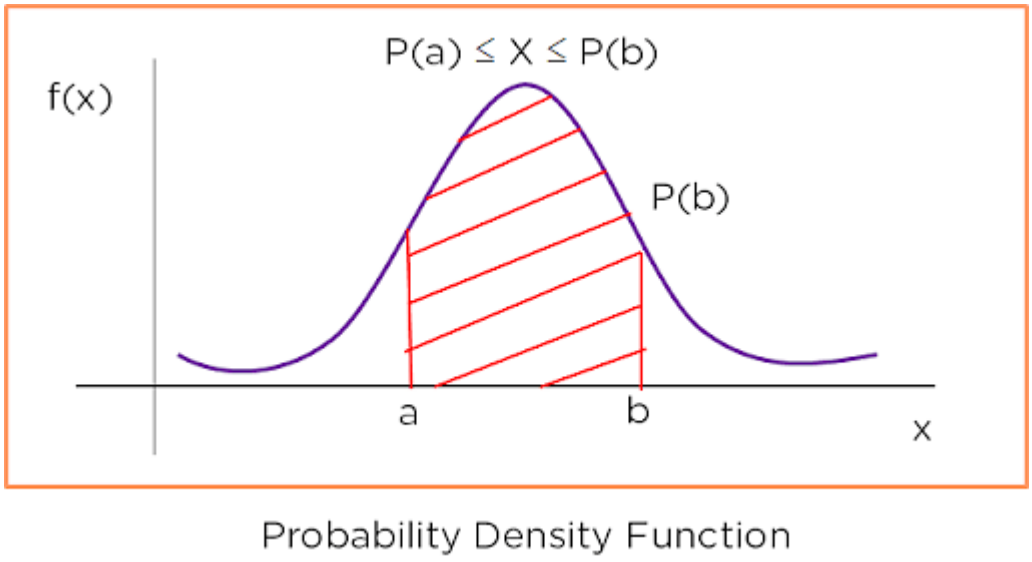


Figure 2.5: Probability Density Function

In the above graph, you get a bell-shaped curve after plotting the function against the variable. The blue curve shows this. Now consider the probability of a point b. To find it, you need to find the area under the curve to the left of b. This is represented by P(b). To find the probability of a variable falling between points a and b, you need to find the area of the curve between a and b. As the probability cannot be more than P(b) and less than P(a), you can represent it as:

P(a) <= X <= P(b).

Consider the graph below, which shows the rainfall distribution in a year in a city. The x-axis has the rainfall in inches, and the y-axis has the probability density function. The probability of some amount of rainfall is obtained by finding the area of the curve on the left of it.

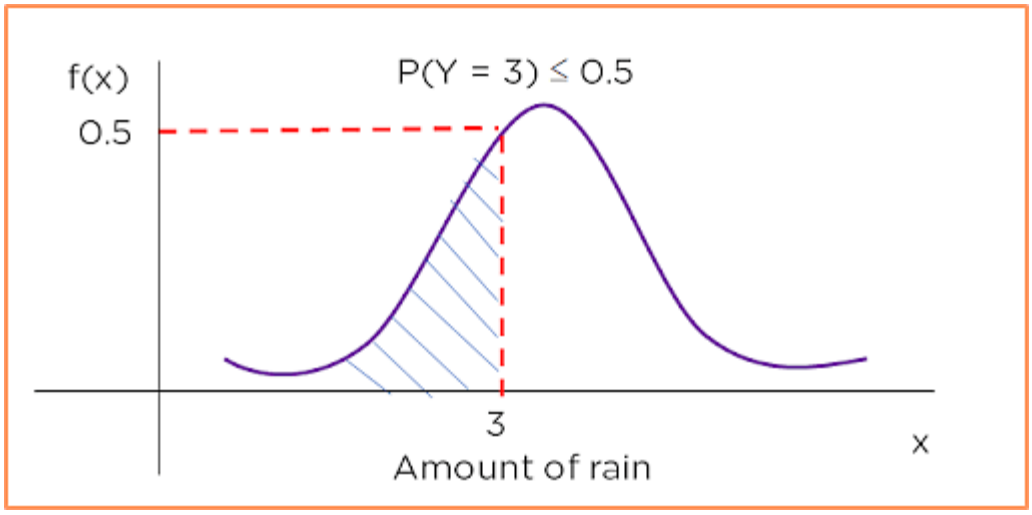


Figure 2.6: Probability Density Function of the amount of rainfall

For the probability of 3 inches of rainfall, you plot a line that intersects the y-axis at the same point on the graph as a line extending from 3 on the x-axis does. This tells you that the probability of 3 inches of rainfall is less than or equal to 0.5.

1.7. Descriptive Statistics

What is Statistics?

Statistics is the science of collecting data and analyzing them to infer proportions (sample) that are representative of the population. In other words, statistics is interpreting data in order to make predictions for the population.

Descriptive Statistics

Descriptive Statistics is summarizing the data at hand through certain numbers like mean, median etc. so as to make the understanding of the data easier. It does not involve any generalization or inference beyond what is available. This means that the descriptive statistics are just the representation of the data (sample) available and not based on any theory of probability.

Commonly Used Measures

1. Measures of Central Tendency
2. Measures of Dispersion (or Variability)

Measures of Central Tendency

A Measure of Central Tendency is a one number summary of the data that typically describes the center of the data. These one number summary is of three types.

1. **Mean :** Mean is defined as the ratio of the sum of all the observations in the data to the total number of observations. This is also known as Average. Thus mean is a number around which the entire data set is spread.
2. **Median :** Median is the point which divides the entire data into two equal halves. One-half of the data is less than the median, and the other half is greater than the same. Median is calculated by first arranging the data in either ascending or descending order.

- If the number of observations are odd, median is given by the middle observation in the sorted form.

- If the number of observations are even, median is given by the mean of the two middle observation in the sorted form.

An important point to note that the order of the data (ascending or descending) does not effect the median.

3. Mode : Mode is the number which has the maximum frequency in the entire data set, or in other words,mode is the number that appears the maximum number of times. A data can have one or more than one mode.

- If there is only one number that appears maximum number of times, the data has one mode, and is called **Uni-modal**.
- If there are two numbers that appear maximum number of times, the data has two modes, and is called **Bi-modal**.
- If there are more than two numbers that appear maximum number of times, the data has more than two modes, and is called **Multi-modal**.

Example to compute the Measures of Central Tendency

Consider the following data points.

17, 16, 21, 18, 15, 17, 21, 19, 11, 23

- Mean — Mean is calculated as

$$Mean = \frac{17+16+21+18+15+17+21+19+11+23}{10} = \frac{178}{10} = 17.8$$

- Median — To calculate Median, lets arrange the data in ascending order.

11, 15, 16, 17, 17, 18, 19, 21, 21, 23

Since the number of observations is even (10), median is given by the average of the two middle observations (5th and 6th here).

$$Median = \frac{5^{th}\ Obs + 6^{th}\ Obs}{2} = \frac{17+18}{2} = 17.5$$

- Mode — Mode is given by the number that occurs maximum number of times. Here, 17 and 21 both occur twice. Hence, this is a Bimodal data and the modes are 17 and 21.
- Since Median and Mode does not take all the data points for calculations, these are robust to outliers, i.e. these are not effected by outliers.
- At the same time, Mean shifts towards the outlier as it considers all the data points. This means if the outlier is big, mean overestimates the data and if it is small, the data is underestimated.
- If the distribution is symmetrical, Mean = Median = Mode. Normal distribution is an example.

1.8.Notion of probability, distributions, mean, variance, covariance, covariance matrix,

Probability and Statistics form the basis of Data Science. The probability theory is very much helpful for making the prediction. Estimates and predictions form an important part of Data science. With the help of statistical methods, we make estimates for the further analysis. Thus, statistical methods are largely dependent on the theory of probability. And all of probability and statistics is dependent on Data.

1.8.1. Data

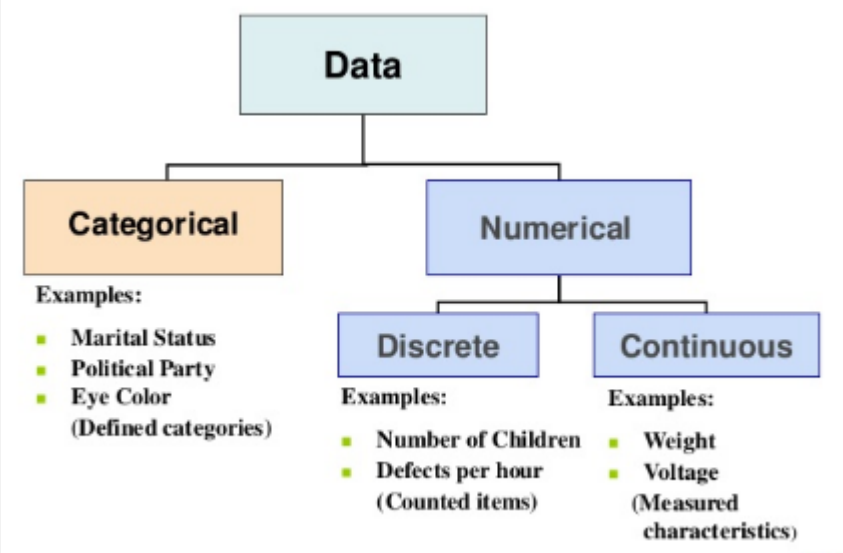
Data is the collected information(observations) we have about something or facts and statistics collected together for reference or analysis.

Data — a collection of facts (numbers, words, measurements, observations, etc) that has been translated into a form that computers can process

Why does Data Matter?

- Helps in understanding more about the data by identifying relationships that may exist between 2 variables.
- Helps in predicting the future or forecast based on the previous trend of data.
- Helps in determining patterns that may exist between data.
- Helps in detecting fraud by uncovering anomalies in the data.

Data matters a lot nowadays as we can infer important information from it. Now let's delve into how data is categorized. Data can be of 2 types categorical and numerical data. For Example in a bank, we have regions, occupation class, gender which follow categorical data as the data is within a fixed certain value and balance, credit score, age, tenure months follow numerical continuous distribution as data can follow an unlimited range of values.

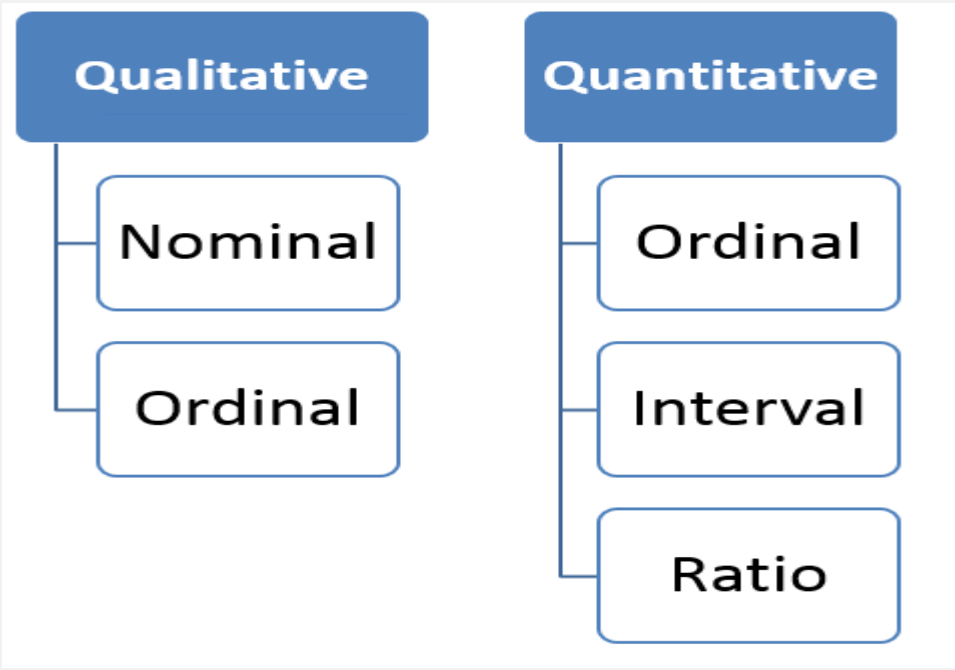


Note: Categorical Data can be visualized by Bar Plot, Pie Chart, Pareto Chart. Numerical Data can be visualized by Histogram, Line Plot, Scatter Plot

1.8.2. Descriptive Statistics

A descriptive statistic is a summary statistic that quantitatively describes or summarizes features of a collection of information. It helps us in knowing our data better. It is used to describe the characteristics of data.

Measurement level of Data



The qualitative and quantitative data is very much similar to the above categorical and numerical data.

Nominal: Data at this level is categorized using names, labels or qualities. eg: Brand Name, ZipCode, Gender.

Ordinal: Data at this level can be arranged in order or ranked and can be compared. eg: Grades, Star Reviews, Position in Race, Date

Interval: Data at this level can be ordered as it is in a range of values and meaningful differences between the data points can be calculated. eg: Temperature in Celsius, Year of Birth

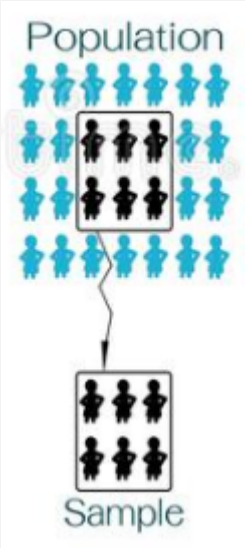
Ratio: Data at this level is similar to interval level with added property of an inherent zero. Mathematical calculations can be performed on these data points. eg: Height, Age, Weight

1.8.3. Population or Sample Data

Before performing any analysis of data, we should determine if the data we're dealing with is population or sample.

Population: Collection of all items (N) and it includes each and every unit of our study. It is hard to define and the measure of characteristic such as mean, mode is called parameter.

Sample: Subset of the population (n) and it includes only a handful units of the population. It is selected at random and the measure of the characteristic is called as statistics.



For Example, say you want to know the mean income of the subscribers to a movie subscription service(parameter). We draw a random sample of 1000 subscribers and determine that their mean income($\bar{x}$) is $34,500 (statistic). We conclude that the population mean income (μ) is likely to be close to $34,500 as well.

1.8.4. Measures of Central Tendency

The measure of central tendency is a single value that attempts to describe a set of data by identifying the central position within that set of data. As such, measures of central tendency are sometimes called measures of central location. They are also classed as summary statistics.

1.8.5. Mean: The mean is equal to the sum of all the values in the data set divided by the number of values in the data set i.e the calculated average. **It susceptible to outliers** when unusual values are added it gets skewed i.e deviates from the typical central value.

$$\bar{x} = \frac{(x_1 + x_2 + \cdots + x_n)}{n}$$

1.8.6. Median: The median is the middle value for a dataset that has been arranged in order of magnitude. Median is a better alternative to mean as it is less affected by outliers and skewness of the data. The median value is much closer than the typical central value.

If the total number of values is odd then

$$\text{Median} = \left(\frac{n+1}{2}\right)^{th} \text{term}$$

If the total number of values is even then

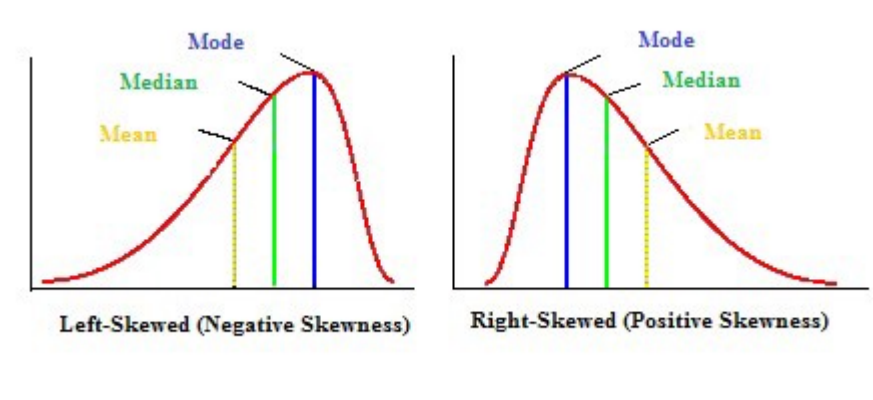
$$\text{Median} = \left(\frac{\left(\frac{n}{2}\right)^{th} term + \left(\frac{n}{2}+1\right)^{th} term}{2}\right)^{\text{th}} term$$

1.8.7. Mode: The mode is the most commonly occurring value in the dataset. The mode can, therefore sometimes consider the mode as being the most popular option.

For Example, In a dataset containing {13,35,54,54,55,56,57,67,85,89,96} values. Mean is 60.09. Median is 56. Mode is 54.

1.8.8. Measures of Asymmetry

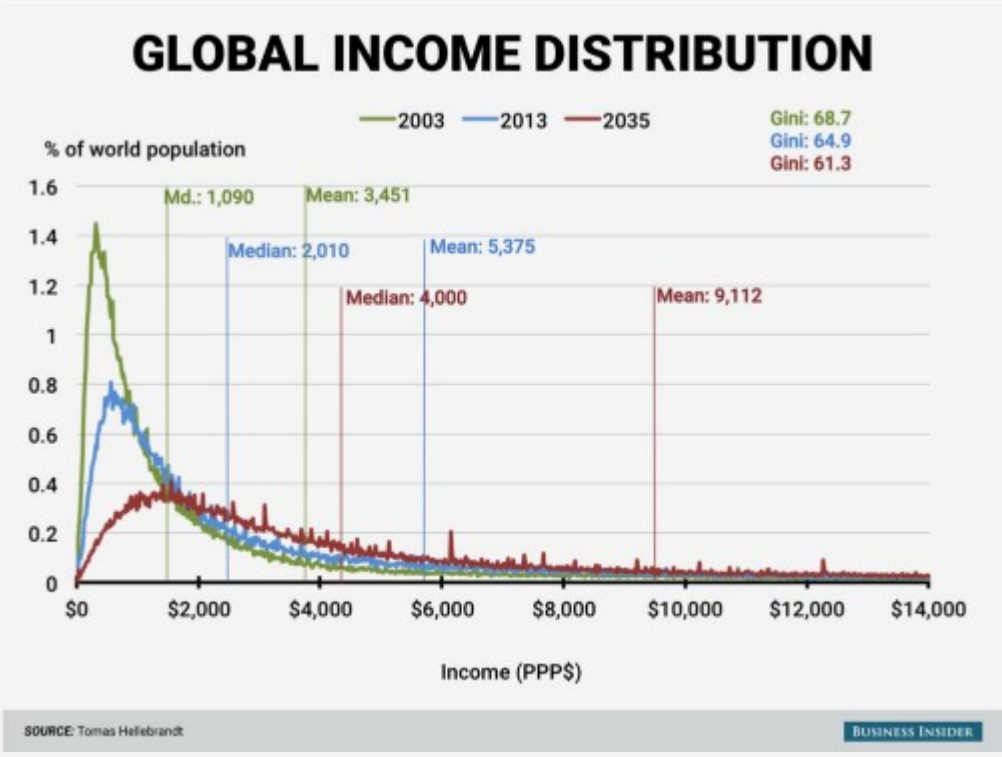
Skewness: Skewness is the asymmetry in a statistical distribution, in which the curve appears distorted or skewed towards to the left or to the right. Skewness indicates whether the data is concentrated on one side.



Positive Skewness: Positive Skewness is when the mean>median>mode. The outliers are skewed to the right i.e the tail is skewed to the right.

Negative Skewness: Negative Skewness is when the mean<median<mode. The outliers are skewed to the left i.e the tail is skewed to the left.

Skewness is important as it tells us about where the data is distributed.



For eg: Global Income Distribution in 2003 is highly right-skewed.We can see the mean $3,451 in 2003(green) is greater than the median $1,090. It suggests that the global income is not evenly distributed. Most individuals incomes are less than $2,000 and less number of people with income

above $14,000, so the skewness. But it seems in 2035 according to the forecast income inequality will decrease over time.

1.8.9. Measures of Variability(Dispersion)

The measure of central tendency gives a single value that represents the whole value; however, the central tendency cannot describe the observation fully. The measure of dispersion helps us to study the variability of the items i.e the spread of data.

Remember: Population Data has N data points and Sample Data has (n-1) data points. (n-1) is called Bessel's Correction and it is used to reduce bias.

1.8.10. Range: The difference between the largest and the smallest value of a data, is termed as the range of the distribution. Range does not consider all the values of a series, i.e. it takes only the extreme items and middle items are not considered significant. eg: For {13,33,45,67,70} the range is 57 i.e(70–13).

1.8.11. Variance: Variance measures how far is the sum of squared distances from each point to the mean i.e the dispersion around the mean.

Variance is the average of all squared deviations.

$$\sigma^2 = \frac{\sum (x_i - \mu)^2}{n} \text{ for populations}$$

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

Note: *The units of values and variance is not equal so we use another variability measure.*

1.8.12. Standard Deviation: As Variance suffers from unit difference so standard deviation is used. The square root of the variance is the standard deviation. It tells about the concentration of the data around the mean of the data set.

Population standard deviation: σ

= square root of the population variance

$$\sigma=\sqrt{\sigma^2}$$

Sample standard deviation: s

= square root of the sample variance, so that

$$s=\sqrt{s^2}$$

For eg: {3,5,6,9,10} are the values in a dataset.

$$\text{Mean} = \frac{3+5+6+9+10}{5} = 6.6$$

$$\text{Variance} = \frac{(3-6.6)^2+(5-6.6)^2+(6-6.6)^2+(9-6.6)^2+(10-6.6)^2}{5}$$

$$= \frac{12.96+2.56+0.36+5.76+11.56}{5} = \frac{33.2}{5} = 6.64$$

$$\text{Standard Deviation} = \sqrt{\text{Variance}} = \sqrt{6.64} = 2.576$$

Given measurements on a sample, what is the difference between a standard deviation and a standard error?

A **standard deviation is** a sample estimate of the population parameter; that is, it is an estimate of the variability of the observations. Since the population is unique, it has a unique standard deviation, which may be large or small depending on how variable the observations are. We would not expect the sample standard deviation to get smaller because the sample gets larger. However, a large sample would provide a more precise estimate of the population standard deviation than a small sample.
A **standard error**, on the other hand, is a measure of precision of an estimate of a population parameter. A standard error is always attached to a parameter, and one can have standard errors of any estimate, such as mean, median, fifth centile, even the standard error of the standard deviation. Since one would expect the precision of the estimate to increase with the sample size, the standard error of an estimate will decrease as the sample size increases.

1.8.13. Coefficient of Variation(CV): It is also called as the relative standard deviation. It is the ratio of standard deviation to the mean of the dataset.

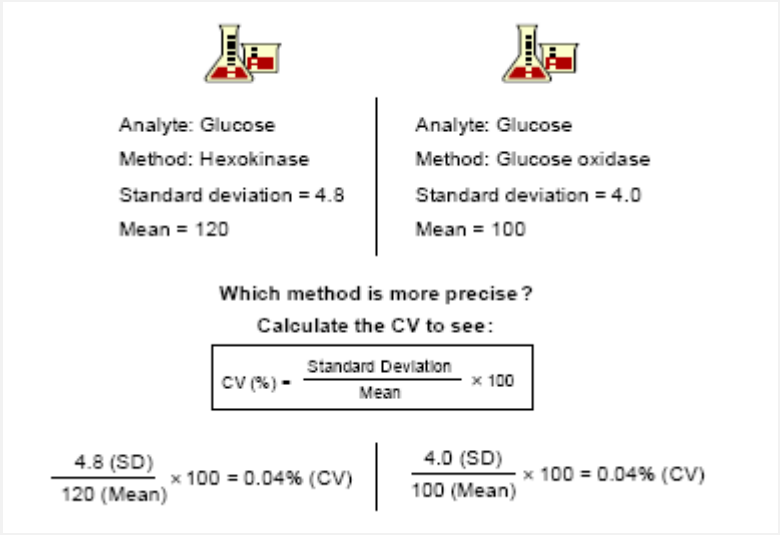
CV for a population:

$$CV = \frac{\sigma}{\mu} * 100\%$$

CV for a sample:

$$CV = \frac{s}{\bar{x}} * 100\%$$

Standard deviation is the variability of a single dataset. Whereas the coefficient of variance can be used for comparing 2 datasets.



From the above example, we can see that the CV is the same. Both methods are precise. So it is perfect for comparisons.

1.8.14. Measures of Quartiles

Quartiles are better at understanding as every data point considered.

Measures of Relationship

Measures of relationship are used to find the comparison between 2 variables.

1.8.15. Covariance: Covariance is a measure of the relationship between the variability of 2 variables i.e It measures the degree of change in the variables, when one variable changes, will there be the same/a similar change in the other variable.

A population covariance is

$$Cov(x,y) = \sigma_{xy} = \frac{\sum_{i=1}^{n}(x_i - \mu_x)(y_i - \mu_y)}{N}$$

where x_i and y_i are the observed values, μ_x and μ_y are the population means, and N is the population size.

A sample covariance is

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

where x_i and y_i are the observed values, $\bar{x}$ and $\bar{y}$ are the sample means, and n is the sample size.

Covariance does not give effective information about the relation between 2 variables as it is not normalized.

1.8.16. Correlation: Correlation gives a better understanding of covariance. It is normalized covariance. Correlation tells us how correlated the variables are to each other. It is also called as Pearson Correlation Coefficient.

$$Correlation = \rho = \frac{\text{cov}(X,Y)}{\sigma_X \sigma_Y}$$

The value of correlation ranges from -1 to 1. -1 indicates negative correlation i.e with an increase in 1 variable independent there is a decrease in the other dependent variable.1 indicates positive correlation i.e with an increase in 1 variable independent there is an increase in the other dependent variable.0 indicates that the variables are independent of each other.

For Example,

Height	Weight	$x-\bar{x}$	$y-\bar{y}$	$(x-\bar{x})(y-\bar{y})$	$(x-\bar{x})^2$	$(y-\bar{y})^2$
5	45	−0.14	−5	0.7	0.019	25
5.5	53	−0.36	3	−1.08	0.129	9
6	70	0.86	20	17.2	0.739	400
4.7	42	−0.44	−8	3.52	0.193	64
4.5	40	−0.64	−10	6.4	0.409	100

Sum(Height) = 25.7 Mean(Height) = 5.14

Sum(Weight) = 250 Mean(Weight) = 50

$\sum(x-\bar{x})(y-\bar{y}) = 26.74$

$\sum(x-\bar{x})^2 = 1.489$

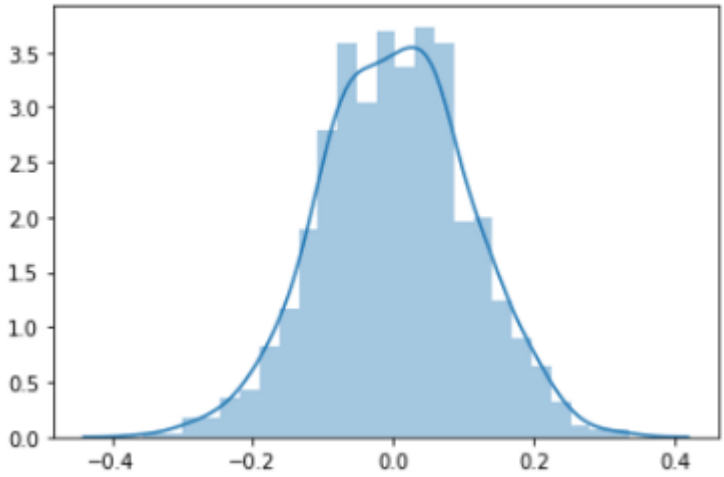
$\sum(y-\bar{y})^2 = 598$

$$\text{Correlation} = \frac{\sum(x-\bar{x})(y-\bar{y})}{\sqrt{\sum(x-\bar{x})^2}\sqrt{\sum(y-\bar{y})^2}} = \frac{26.74}{\sqrt{1.489}\sqrt{598}} = \frac{26.54}{1.220 * 24.454} = 0.889$$

Correlation 0.889 tells us Height and Weight has a positive correlation. It is obvious that as the height of a person increases weight too increases.

1.9.Understanding Univariate and Multivariate Normal Distribution

Gaussian distribution is a synonym for normal distribution. S is a set of random values whose probability distribution looks like the picture below.



This is a bell-shaped curve. If a probability distribution plot forms a bell-shaped curve like above and the mean, median, and mode of the sample are the same that distribution is called normal distribution or Gaussian distribution.

The Gaussian distribution is parameterized by two parameters:

- The mean and The variance

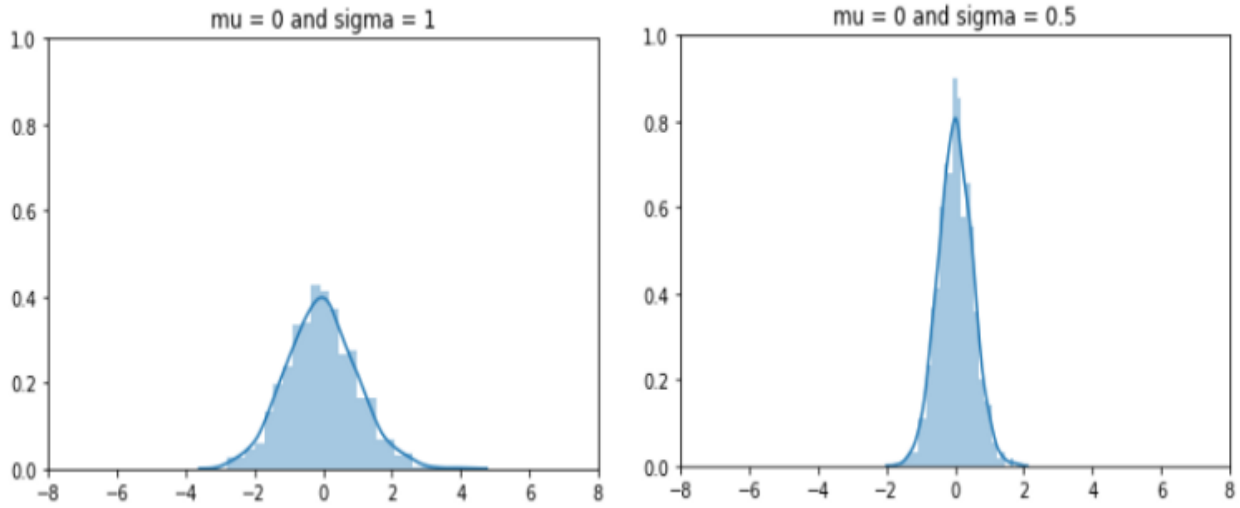
So, the Gaussian density is the highest at the point of μ or mean, and further, it goes from the mean, the Gaussian density keeps going lower.

Here is the formula for the Gaussian distribution:

$$p(x;\ \mu, \sigma^2) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(x-\ \mu)^2}{2\sigma^2}\right)$$

This is the formula for the bell-shaped curve where sigma square is called the variance.

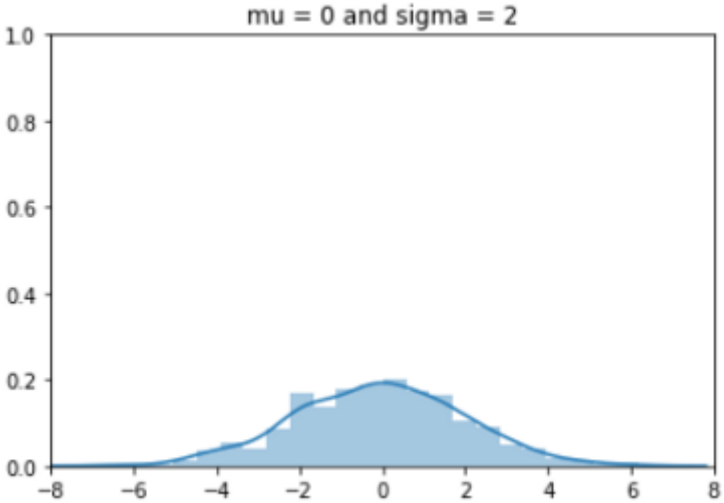
Mean =0, and different sigmas



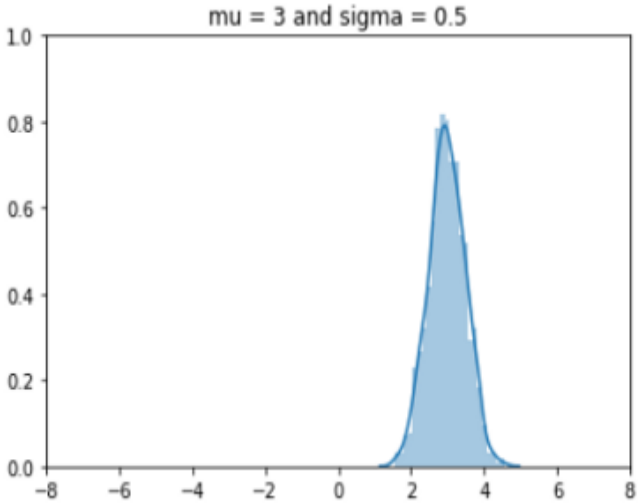
This is the probability distribution of a set of random numbers with μ is equal to 0 and sigma is 1. In the first picture, μ is 0 which means the highest probability density is around 0 and the sigma is one. means the width of the curve is 1. the height of the curve is about 0.5 and the range is -4 to 4 (look at x-axis). The variance sigma square is 1.

Here is another set of random numbers that has a μ of 0 and sigma 0.5 in the second figure. Because the μ is 0, like the previous picture the highest probability density is at around 0 and the sigma is 0.5. So, the width of the curve is 0.5. The variance sigma square becomes 0.25.

As the width of the curve is half the previous curve, the height became double. The range changed to -2 to 2 (x-axis) which is the half of the previous picture.



In this picture, sigma is 2 and μ is 0 as the previous two pictures. Compare it to figure 1 where sigma was 1. This time height became half of figure 1. Because the width became double as the sigma became double. The variance sigma square is 4, four times bigger than figure 1. Look at the range in the x-axis, it's -8 to 8.



Here, we changed μ to 3 and sigma is 0.5 as figure 2. So, the shape of the curve is exactly the same as figure 2 but the center shifted to 3. Now the highest density is at around 3.

It changes shapes with the different values of sigma but the area of the curve stays the same.

One important property of probability distribution is, the area under the curve is integrated to one.

Parameter Estimation

Calculating μ is straight forward. it's simply the average. Take the summation of all the data and divide it by the total number of data.

$$\mu = \frac{1}{m}\sum_{i=1}^{m} x^i$$

The formula for the variance (sigma square) is:

$$\sigma^2 = \frac{1}{m}\sum_{i=1}^{m}(x^i - \mu)^2$$

1.9.1. Univariate Normal Distributions

- Before defining the multivariate normal distribution, we will visit the univariate normal distribution. A random variable *X* is normally distributed with mean μ and variance σ2 if it has the probability density function of *X* as:
- ϕ(x)=12πσ2exp⁡{−12σ2(x−μ)2}
- This result is the usual bell-shaped curve that you see throughout statistics.
- In this expression, you see the squared difference between the variable *x* and its mean, μ. This value will be minimized when x is equal to μ. The quantity −σ−2(x−μ)2 will take its largest value when *x* is equal to μ or likewise, since the exponential function is a monotone function, the normal density takes a maximum value when x is equal to μ.
- The variance σ2 defines the spread of the distribution about that maximum. If σ2 is large, then the spread is going to be large, otherwise, if the σ2 value is small, then the spread will be small.

1.9.2. Multivariate Gaussian Distribution

- **Multivariate analysis** is a branch of statistics concerned with the analysis of multiple measurements, made on one or several samples of individuals. For example, we may wish to measure length, width, and weight of a product.
- **Multivariate statistical analysis** is concerned with data that consist of sets of measurements on a number of individuals or objects.

- The sample data may be heights and weights of some individuals drawn randomly from a population of school children in a given city, or the statistical treatment may be made on a collection of measurements

"Why is the multivariate normal distribution so important? "

- There are three reasons why this might be so:
- *Mathematical Simplicity*. It turns out that this distribution is relatively easy to work with, so it is easy to obtain multivariate methods based on this particular distribution.
- *Multivariate version of the Central Limit Theorem*. You might recall in the univariate course that we had a central limit theorem for the sample mean for large samples of random variables. A similar result is available in multivariate statistics that says if we have a collection of random vectors X1, X2, ⋯Xn that are independent and identically distributed, then the sample mean vector, $\bar{x}$, is going to be approximately multivariate normally distributed for large samples.
- Many natural phenomena may also be modeled using this distribution, just as in the univariate case.

Multivariate normal model

When multivariate data are analyzed, the multivariate normal model is the most commonly used model.

The multivariate normal distribution model extends the univariate normal distribution model to fit vector observations.

Definition of multivariate normal distribution

A p-dimensional vector of random variables,

$$\mathbf{X} = X_1, X_2, \ldots, X_p \qquad -\infty < X_i < \infty, \; i = 1, \ldots, p,$$

is said to have a multivariate normal distribution if its density function $f(\mathbf{X})$ is of the form

$$\begin{aligned} f(\mathbf{X}) &= f(X_1, X_2, \ldots, X_p) \\ &= \left(\frac{1}{2\pi}\right)^{p/2} |\mathbf{\Sigma}|^{-1/2} \exp\left[-\frac{1}{2}(\mathbf{X}-\mathbf{m})'\mathbf{\Sigma}^{-1}(\mathbf{X}-\mathbf{m})\right], \end{aligned}$$

where $\mathbf{m} = (m_1, \ldots, m_p)$ is the vector of means and $\mathbf{\Sigma}$ is the variance-covariance matrix of the multivariate normal distribution. The shortcut notation for this density is

$$\mathbf{X} = \mathrm{N}_p(\mathbf{m}, \mathbf{\Sigma}).$$

Instead of having one set of data, what if we have two sets of data and we need a multivariate Gaussian distribution. Suppose we have two sets of data; x1 and x2.

Separately modeling p(x1) and p(x2) is probably not a good idea to understand the combined effect of both the dataset. In that case, you would want to combine both the dataset and model only p(x).

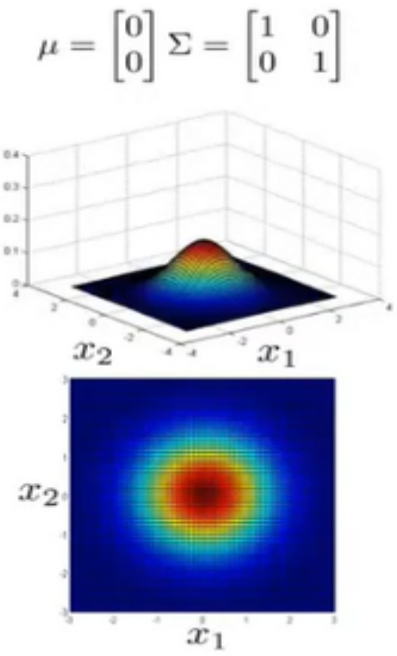
Here is the formula to calculate the probability for multivariate Gaussian distribution,

$$p(x;\ \mu, \Sigma) = \frac{1}{(2\pi)^{n/2}|\Sigma|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(x - \mu)^T \Sigma^{-1}(x - \mu)\right)$$

The summation symbol in this equation is the determinant of sigma which is actually an n x n matrix of sigma.

Visual Representation of Multivariate Gaussian Distribution

Standard Normal Distribution



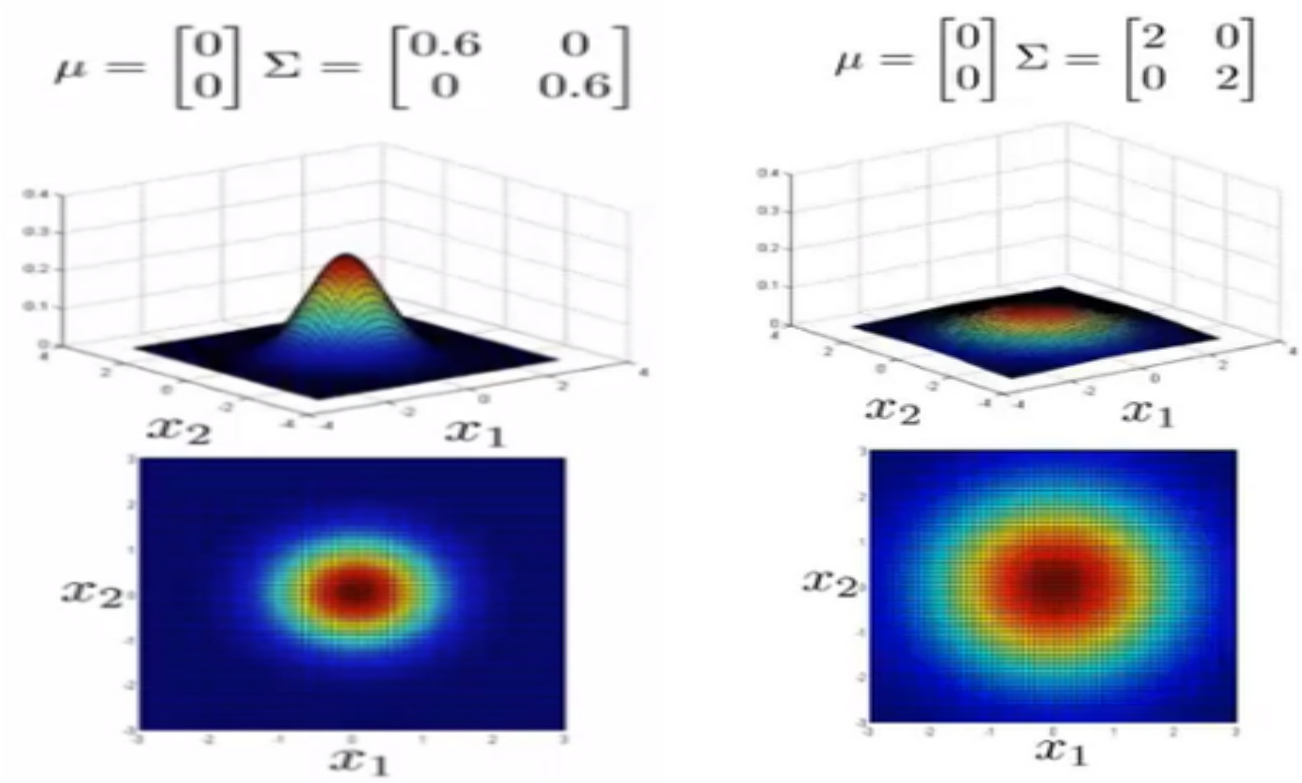
The picture represents a probability distribution of a multivariate Gaussian distribution where μ of both x1 and x2 are zeros.

Summation symbol is an identity matrix that contains sigma values as diagonals. The 1s in the diagonals are the sigma for both x1 and x2. And the zeros in the off diagonals show the correlation between x1 and x2. So, x1 and x2 are not correlated in this case.

In both x1 and x2 direction, the highest probability density is at 0 as the μ is zero.

The dark red color area in the center shows the highest probability density area. The probability density keeps going lower in the lighter red, yellow, green, and cyan areas. It's the lowest in the dark blue color zone.

Changing the Standard Deviation - Sigma



when the standard deviation sigma shrinks, the range also shrinks. At the same time, the height of the curve becomes higher to adjust the area.

In the contrast, when sigma is larger, the variability becomes wider. So, the height of the curve gets lower.

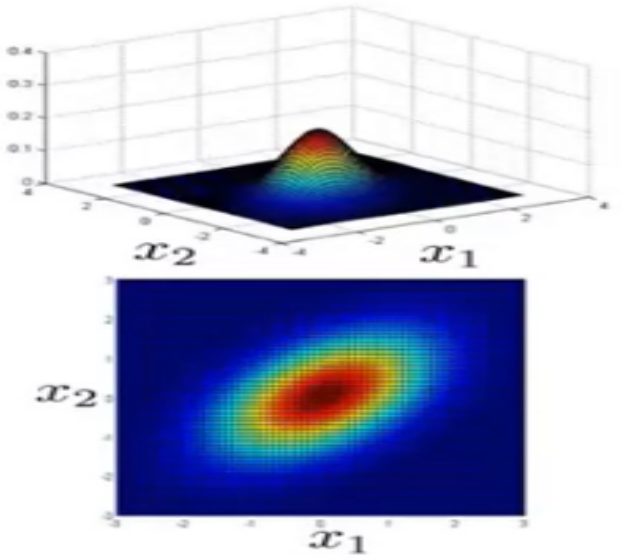
The sigma values for both x1 and x2 will not be the same always.

$$\mu = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \Sigma = \begin{bmatrix} 0.6 & 0 \\ 0 & 1 \end{bmatrix}$$

the range looks like an eclipse. It shrunk for the x1 as the standard deviation sigma is smaller for sigma.

Change the Correlation Factor Between the Variables

$$\mu = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \Sigma = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1 \end{bmatrix}$$



This is a completely different scenario. The off-diagonal values are not zeros anymore. It's 0.5. It shows that x1 and x2 are correlated by a factor of 0.5.

The eclipse has a diagonal direction now. x1 and x2 are growing together as they are positively correlated.

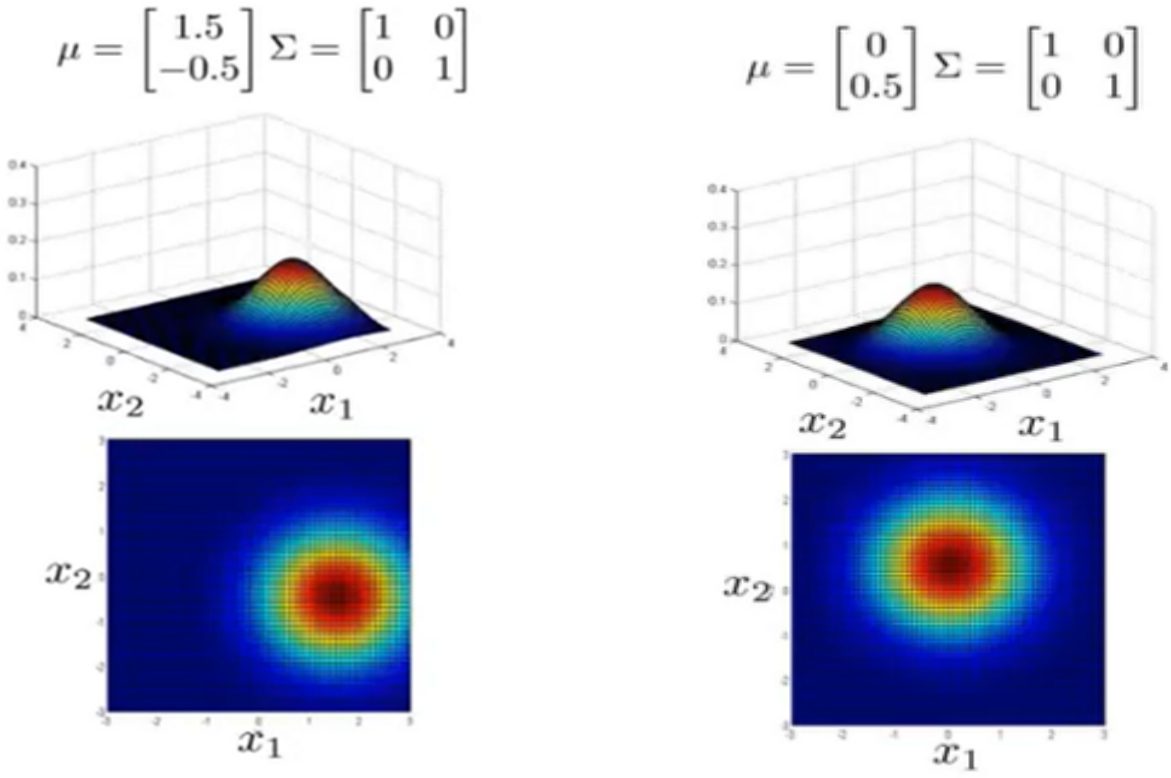
When x1 is large x2 also large and when x1 is small, x2 is also small.

Different Means

The center of the curve shifts from zero for x2 now.

The center position or the highest probability distribution area should be at 0.5 now.

The center of the highest probability in the x1 direction is 1.5. At the same time, the center of the highest probability is -0.5 for x2 direction.



1.10. Hypothesis Testing

Hypothesis testing is a part of statistical analysis, where we test the assumptions made regarding a population parameter.

It is generally used when we were to compare:

- a single group with an external standard
- two or more groups with each other

A Parameter is a number that describes the data from the population whereas, a Statistic is a number that describes the data from a sample.

Terminologies

1.10.1. Null Hypothesis: Null hypothesis is a statistical theory that suggests there is no statistical significance exists between the populations.

It is denoted by H0 and read as H-naught.

1.10.2. Alternative Hypothesis: An Alternative hypothesis suggests there is a significant difference between the population parameters. It could be greater or smaller. Basically, it is the contrast of the Null Hypothesis.

It is denoted by Ha or H1.

H0 must always contain equality(=). Ha always contains difference(≠, >, <).

For example, if we were to test the equality of average means (μ) of two groups:

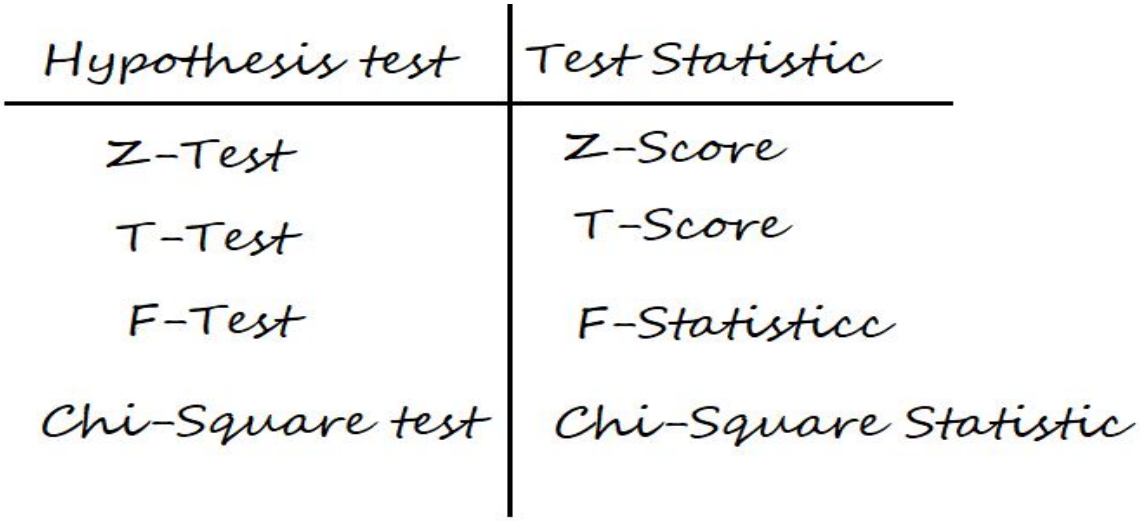
for a two-tailed test, we define H0: μ1 = μ2 and Ha: μ1≠μ2

for a one-tailed test, we define H0: μ1 = μ2 and Ha: μ1 > μ2 or Ha: μ1 < μ2

1.10.3. Level of significance: Denoted by alpha or α. It is a fixed probability of wrongly rejecting a True Null Hypothesis. For example, if α=5%, that means we are okay to take a 5% risk and conclude there exists a difference when there is no actual difference.

1.10.4. Test Statistic: It is denoted by t and is dependent on the test that we run. It is deciding factor to reject or accept Null Hypothesis.

The four main test statistics are given in the below table:



Test Type	Distribution	Test Parameters
Z-test	Normal	Mean
T-test	Student-t	Mean
ANOVA	F distribution	Means
Chi-Square	Chi-squared distribution	Association between two categorical variables

Each hypothesis test uses these basic principles.

Element	Example	Description
Hypothesis with hypothesized value	$\mu > 45$	The H_o here is that the population mean is greater than 45
Test value	45	This is used as a benchmark to test how likely a mean of 45 is given the population mean and SD
Confidence interval	$\alpha = 0.05$	At the 95% confidence level (1-0.95 = 0.05), we can be certain that our test gets the true answer 95% of the time
Test statistic	$z\ score = 1.5$	The test statistic gives you the standardized value of your test value on your test distribution

P-value	$p - value = 0.06$	The p-value is the calculated probability of your value occuring

In hypothesis testing, the following rules are used to either reject or accept the hypothesis given a α of 0.05. Keep in mind that if you were to have an α of 0.1, you're results would be given with **90% confidence** and the example above, with a p-value of 0.06, would reject H_o.

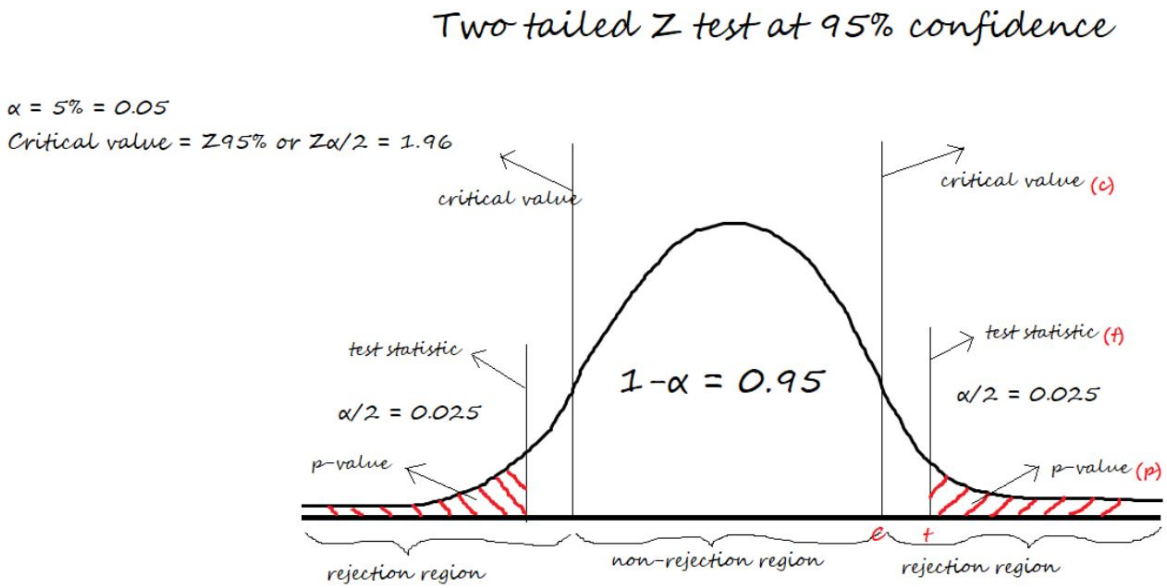
P-value < 0.05	Region of rejection	Reject H_o
P-value > 0.05	Region of acceptance	Fail to reject H_o

p-value: It is the proportion of samples (assuming the Null Hypothesis is true) that would be as extreme as the test statistic. It is denoted by the letter p.

Now, assume we are running a two-tailed Z-Test at 95% confidence. Then, the level of significance (α) = 5% = 0.05. Thus, we will have (1-α) = 0.95 proportion of data at the center, and α = 0.05 proportion will be equally shared to the two tails. Each tail will have (α/2) = 0.025 proportion of data.

The critical value i.e., Z95% or Zα/2 = 1.96 is calculated from the Z-scores table.

Now, take a look at the below figure for a better understanding of critical value, test-statistic, and p-value.

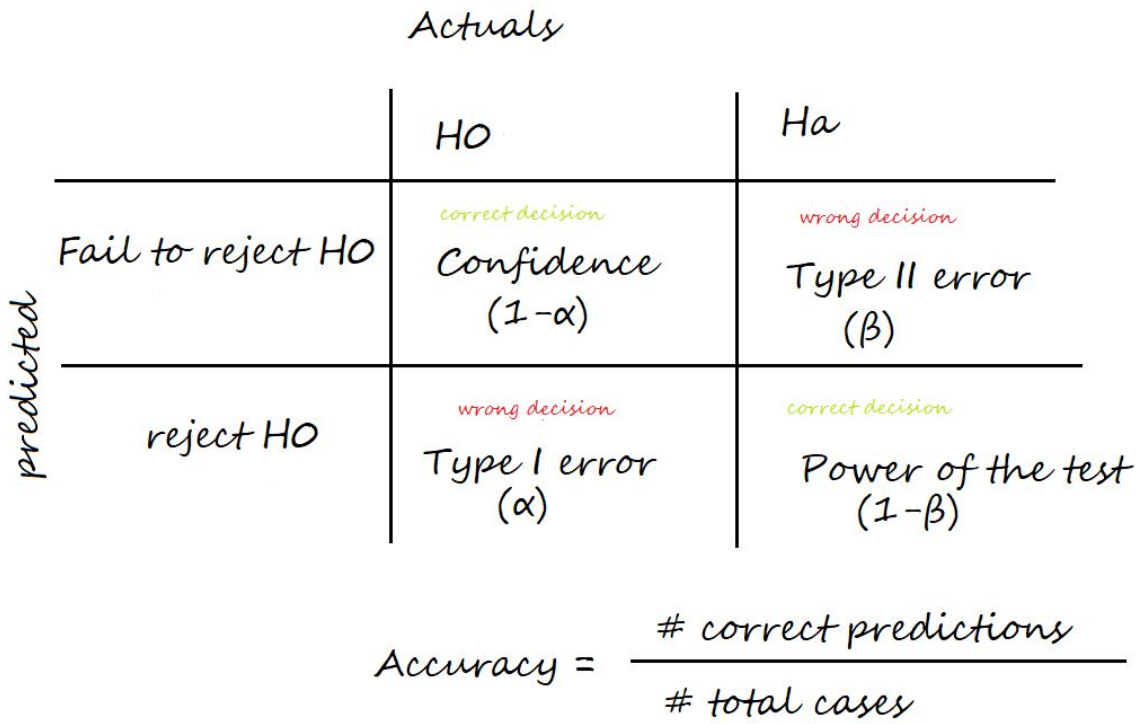


Steps of Hypothesis testing

For a given business problem,

1. Start with specifying Null and Alternative Hypotheses about a population parameter
2. Set the level of significance (α)
3. Collect Sample data and calculate the Test Statistic and P-value by running a Hypothesis test that well suits our data
4. Make Conclusion: Reject or Fail to Reject Null Hypothesis
5. Confusion Matrix in Hypothesis testing

To plot a confusion matrix, we can take actual values in columns and predicted values in rows or vice versa.



$$\text{Accuracy} = \frac{\#\ \text{correct predictions}}{\#\ \text{total cases}}$$

Confidence: The probability of accepting a True Null Hypothesis. It is denoted as (1-α)

Power of test: The probability of rejecting a False Null Hypothesis i.e., the ability of the test to detect a difference. It is denoted as (1-β) and its value lies between 0 and 1.

Type I error: Occurs when we reject a True Null Hypothesis and is denoted as α.

Type II error: Occurs when we accept a False Null Hypothesis and is denoted as β.

Accuracy: Number of correct predictions / Total number of cases

The factors that affect the power of the test are sample size, population variability, and the confidence (α).

Confidence and power of test are directly proportional. Increasing the confidence increases the power of the test.

Type 1 and 2 errors occur when we **reject** or **accept** our null hypothesis when, in reality, we shouldn't have. This happens because, while statistics is powerful, there is a certain chance that you may be wrong. The table below summarizes these types of errors.

	Accept H_o	**Reject** H_o
In reality, H_o **is actually true**	Correct: H_o is true and statistical test accepts H_o	Incorrect: Type 1 error - H_o is true and statistical test rejects H_o
In reality, H_o **is actually false**	Incorrect: Type 2 error - H_o is false and statistical test accepts H_o	Correct: H_o is false and statistical test rejects H_o

Confidence Interval

A confidence interval, in statistics, refers to the probability that a population parameter will fall between a set of values for a certain proportion of times.

A confidence interval is the mean of your estimate plus and minus the variation in that estimate. This is the range of values you expect your estimate to fall between if you redo your test, within a certain level of confidence.

Confidence, in statistics, is another way to describe probability. For example, if you construct a confidence interval with a 95% confidence level, you are confident that 95 out of 100 times the estimate will fall between the upper and lower values specified by the confidence interval.

The desired confidence level is usually one minus the alpha (a) value you used in the statistical test:

Confidence level = 1 − a

So if you use an alpha value of $p < 0.05$ for statistical significance, then your confidence level would be $1 − 0.05 = 0.95$, or 95%.

When to use confidence intervals?

Confidence intervals can be calculated for many kinds of statistical estimates, including:

- Proportions
- Population means
- Differences between population means or proportions
- Estimates of variation among groups

These are all point estimates, and don't give any information about the variation around the number. Confidence intervals are useful for communicating the variation around a point estimate.

Example: Variation around an estimate

You survey 100 Brits and 100 Americans about their television-watching habits, and find that both groups watch an average of 35 hours of television per week.

However, the British people surveyed had a wide variation in the number of hours watched, while the Americans all watched similar amounts.

Even though both groups have the same point estimate (average number of hours watched), the British estimate will have a wider confidence interval than the American estimate because there is more variation in the data.

Calculating a confidence interval

Most statistical programs will include the confidence interval of the estimate when you run a statistical test.

If you want to calculate a confidence interval on your own, you need to know:

- The point estimate you are constructing the confidence interval for
- The critical values for the test statistic
- The standard deviation of the sample
- The sample size

Once you know each of these components, you can calculate the confidence interval for your estimate by plugging them into the confidence interval formula that corresponds to your data.

Point estimate

The point estimate of your confidence interval will be whatever statistical estimate you are making (e.g. population mean, the difference between population means, proportions, variation among groups).

Example: Point estimate - In the TV-watching example, the point estimate is the mean number of hours watched: 35.

Finding the critical value

Critical values tell you how many standard deviations away from the mean you need to go in order to reach the desired confidence level for your confidence interval.

There are three steps to find the critical value.

1. **Choose your alpha (a) value.**

The alpha value is the probability threshold for statistical significance. The most common alpha value is p = 0.05, but 0.1, 0.01, and even 0.001 are sometimes used. It's best to look at the papers published in your field to decide which alpha value to use.

2. **Decide if you need a one-tailed interval or a two-tailed interval.**

You will most likely use a two-tailed interval unless you are doing a one-tailed t-test.

For a two-tailed interval, divide your alpha by two to get the alpha value for the upper and lower tails.

3. **Look up the critical value that corresponds with the alpha value.**

If your data follows a normal distribution, or if you have a large sample size ($n > 30$) that is approximately normally distributed, you can use the z-distribution to find your critical values.

For a z-statistic, some of the most common values are shown in this table:

Confidence level	90%	95%	99%
alpha for one-tailed CI	0.1	0.05	0.01
alpha for two-tailed CI	0.05	0.025	0.005
z-statistic	1.64	1.96	2.57

If you are using a small dataset ($n \leq 30$) that is approximately normally distributed, use the t-distribution instead.

The t-distribution follows the same shape as the z-distribution, but corrects for small sample sizes. For the t-distribution, you need to know your degrees of freedom (sample size minus 1).

Check out this set of t tables to find your t-statistic. The author has included the confidence level and p-values for both one-tailed and two-tailed tests to help you find the t-value you need.

For normal distributions, like the t-distribution and z-distribution, the critical value is the same on either side of the mean.

Example: Critical value In the TV-watching survey, there are more than 30 observations and the data follow an approximately normal distribution (bell curve), so we can use the z-distribution for our test statistics.

For a two-tailed 95% confidence interval, the alpha value is 0.025, and the corresponding critical value is 1.96.

This means that to calculate the upper and lower bounds of the confidence interval, we can take the mean ±1.96 standard deviations from the mean.

Finding the standard deviation

Most statistical software will have a built-in function to calculate your standard deviation, but to find it by hand you can first find your sample variance, then take the square root to get the standard deviation.

1.Find the sample variance

Sample variance is defined as the sum of squared differences from the mean, also known as the mean-squared-error (MSE):

$$S^2 = \sum_{i=1}^{n} \frac{(Xi - \overline{X})^2}{n-1}$$

To find the MSE, subtract your sample mean from each value in the dataset, square the resulting number, and divide that number by $n - 1$ (sample size minus 1).

Then add up all of these numbers to get your total sample variance (s2). For larger sample sets, it's easiest to do this in Excel.

2.Find the standard deviation.

The standard deviation of your estimate (s) is equal to the square root of the sample variance/sample error (s2):

$$S = \sqrt{(S^2)}$$

Example: Standard deviation In the television-watching survey, the variance in the GB estimate is 100, while the variance in the USA estimate is 25. Taking the square root of the variance gives us a sample standard deviation (s) of:

10 for the GB estimate.

5 for the USA estimate.

Sample size

The sample size is the number of observations in your data set.

Example: Sample size In our survey of Americans and Brits, the sample size is 100 for each group.

Confidence interval for the mean of normally-distributed data

Normally-distributed data forms a bell shape when plotted on a graph, with the sample mean in the middle and the rest of the data distributed fairly evenly on either side of the mean.

The confidence interval for data which follows a standard normal distribution is:

$$CI = \overline{X} \pm Z^* \frac{\sigma}{\sqrt{n}}$$

Where:

CI = the confidence interval

$\bar{X}$ = the population mean

Z* = the critical value of the z-distribution

σ = the population standard deviation

√n = the square root of the population size

The confidence interval for the t-distribution follows the same formula, but replaces the Z* with the t*.

In real life, you never know the true values for the population (unless you can do a complete census). Instead, we replace the population values with the values from our sample data, so the formula becomes:

$$CI = \hat{x} \pm Z^* \frac{s}{\sqrt{n}}$$

Where:

ˆx = the sample mean

s = the sample standard deviation

Example: Calculating the confidence interval- In the survey of Americans' and Brits' television watching habits, we can use the sample mean, sample standard deviation, and sample size in place of the population mean, population standard deviation, and population size.

To calculate the 95% confidence interval, we can simply plug the values into the formula.

For the USA:

$$\begin{aligned} CI &= 35 \pm 1.96 \frac{5}{\sqrt{100}} \\ &= 35 \pm 1.96(0.5) \\ &= 35 \pm 0.98 \end{aligned}$$

So for the USA, the lower and upper bounds of the 95% confidence interval are 34.02 and 35.98.

For GB:

$$\begin{aligned} CI &= 35 \pm 1.96 \frac{10}{\sqrt{100}} \\ &= 35 \pm 1.96(1) \\ &= 35 \pm 1.96 \end{aligned}$$

So for the GB, the lower and upper bounds of the 95% confidence interval are 33.04 and 36.96.

1. The confidence 'level' refers to the long term success rate of the method i.e. how often this type of interval will capture the parameter of interest.

2. A specific confidence interval gives a range of plausible values for the parameter of interest.

3. A larger margin of error produces a wider confidence interval that is more likely to contain the parameter of interest(increased confidence)

4. Increasing the confidence will increase the margin of error resulting in a wider interval.

1.11. PROBLEMS ON PROBABILITY AND STATISTICS

1) Find the probability of drawing two cards that belong to different colors or different shapes (suits) from a shuffled deck of 52 cards?

Solution:

We will have a match only after the second card is drawn. That decides whether it is a match or not. After we have drawn any card from the deck, there will be 51 cards left. Of these 51 cards, have 12 cards that belong to the same suit of the card we have already drawn. We need to avoid these 12 cards.

So, number of favorable cases = 51 -12 = 39 and all possible cases = 51

Hence, the probability = 39 / 51 = 13 / 17

2) Suppose there are two events, A and B, with P(A) = 0.50, P(B) = 0.60, and P(A ∩∩ B) = 0.40.

a. Find P(A|B).

b. Find P(B|A).

c. Are A and B independent? Why or why not?

Solution:

We know that,

P(A|B)=P(A∩B)P(B)P(A|B)=P(A∩B)P(B)
Putting values, we get -

P(A|B)=0.400.60P(A|B)=0.400.60
P(A|B)=23P(A|B)=23
P(A|B)=0.666667

3)

Find the mean and variance of the distribution $P[X=x]=2^{-x}$, x=1,2,3….

Solution:

$$P[X=x]= \frac{1}{2^x}=\left(\frac{1}{2}\right)^{x-1}\frac{1}{2}, x=1,2,3.....$$

$$\therefore p=\frac{1}{2}\ and\ \ q=\frac{1}{2}$$

$$Mean=\frac{q}{p}=1;\ Variance=\frac{q}{p^2}=2$$

Find the expected value and the variance of the number of times one must throw a die until the outcome 1 has occurred 4 times.

Solution:

X follows the negative bionomial distribution with parameter r = 4 and p=1/6

E(X)= mean = r P = rqQ = r (1-p) (1/p)= 20. (p=1/Q and q=P/Q)

Variance = rPQ = r(1-p)/p^2 = 120.

5)

. If a boy is throwing stones at a target, what is the probability that his 10^{th} throw is his 5^{th} hit, if the probability of hitting the target at any trial is ½?

Solution:

Since 10^{th} throw should result in the 5^{th} successes ,the first 9 throws ought to have resulted in 4 successes and 5 faliures.

$$n = 5, r = 5,\ p=\frac{1}{2}=q$$

$$\therefore \text{Required probability} = P(X=5)= (5+5-1)C_5\ (1/2)^5\ (1/2)^5$$

$$=9C_4\ (1/2^{10}) = 0.123$$

6) Find the median for the data set: 34, 22, 15, 25, 10.

Step 1: Arrange data in increasing order 10, 15, 22, 25, 34

Step 2: There are 5 numbers in the data set, n = 5.

Step 3: n = 5, so n is an odd number Median = middle number, median is 22

7) Find the median for the data set: 19, 34, 22, 15, 25, 10.

Step 1: Arrange data in increasing order 10, 15, 19, 22, 25, 34
Step 2: There are 6 numbers in the data set, n = 6.
Step 3: n = 6, so n is an even number Median = average of two middle numbers median = (19+22)/2= 20.5

Notes: Mean and median don't have to be numbers from the data set!Mean and median can only take one value each.Mean is influenced by extreme values, while median is resistant.

8) Find the mode for the data set: 19, 19, 34, 3, 10, 22, 10, 15, 25, 10, 6.

The number that occurs the most is number 10, mode = 10.

9) Find the mode for the data set: 19, 19, 34, 3, 10, 22, 10, 15, 25, 10, 6, 19.

Number 10 occurs 3 times, but also number 19 occurs 3 times, since there is no number that occur 4 times both numbers 10 and 19 are mode, mode = {10, 19}.

Notes: Mode is always the number from the data set.Mode can take zero, one, or more than one values. (There can be zero modes, one mode, two modes, ...)

10) The set S = { 5,10,15,20,30}, Mean of set S = 5+10+15+20+30/5 = 80/5 = 16

Sometimes, the question includes frequency of the values. In that case, the formula changes to

Mean = $\sum FiXi / \sum Fi$,

Where, Fi = frequency of the ith value of the distribution, Xi = ith value of the distribution

11) Find the mean, median, mode, and range for the following list of values: 13, 18, 13, 14, 13, 16, 14, 21, 13

Solution: The mean is the usual average, so we'll add and then divide:
(13 + 18 + 13 + 14 + 13 + 16 + 14 + 21 + 13) ÷ 9 = 15

Note that the mean, in this case, isn't a value from the original list. This is a common result. You should not assume that your mean will be one of your original numbers.

The median is the middle value, so first we'll have to rewrite the list in numerical order:

13, 13, 13, 13, 14, 14, 16, 18, 21

There are nine numbers in the list, so the middle one will be the (9 + 1) ÷ 2 = 10 ÷ 2 = 5th number:

13, 13, 13, 13, 14, 14, 16, 18, 21

So the median is 14.

The mode is the number that is repeated more often than any other, so 13 is the mode, since 13 is being repeated 4 times.

The largest value in the list is 21, and the smallest is 13, so the range is 21 – 13 = 8.

Mean: 15 |median: 14 |mode: 13 |range: 8

12) Find the mean, median, mode, and range for the following list of values: 1, 2, 4, 7

Solution: The mean is the usual average:

(1 + 2 + 4 + 7) ÷ 4 = 14 ÷ 4 = 3.5

The median is the middle number. In this example, the numbers are already listed in numerical order, so we don't have to rewrite the list. But there is no "middle" number, because there are even number of numbers. Because of this, the median of the list will be the mean (that is, the usual average) of the middle two values within the list. The middle two numbers are 2 and 4, so:

(2 + 4) ÷ 2 = 6 ÷ 2 = 3

So the median of this list is 3, a value that isn't in the list at all.

The mode is the number that is repeated most often, but all the numbers in this list appear only once, so there is no mode.

The largest value in the list is 7, the smallest is 1, and their difference is 6, so the range is 6.

Mean: 3.5 | median: 3 |mode: none |range: 6

13) Calculate covariance for the following data set:
x: 2.1, 2.5, 3.6, 4.0 (mean = 3.1)
y: 8, 10, 12, 14 (mean = 11)

Substitute the values into the formula and solve:
$Cov(X,Y) = \Sigma E((X-\mu)(Y-\nu)) / n-1$

= (2.1-3.1)(8-11)+(2.5-3.1)(10-11)+(3.6-3.1)(12-11)+(4.0-3.1)(14-11) /(4-1)
= (-1)(-3) + (-0.6)(-1)+(.5)(1)+(0.9)(3) / 3
= 3 + 0.6 + .5 + 2.7 / 3
= 6.8/3
= 2.267

14) Let A and B be events on the same sample space, with P (A) = 0.6 and P (B) = 0.7. Can these two events be disjoint?

Answer:

No

These two events cannot be disjoint because P(A)+P(B) >1.

P(A∪B) = P(A)+P(B)-P(A∩B).

An event is disjoint if P(A∩B) = 0. If A and B are disjoint P(A∪B) = 0.6+0.7 = 1.3

And Since probability cannot be greater than 1, these two mentioned events cannot be disjoint.

15) Alice has 2 kids and one of them is a girl. What is the probability that the other child is also a girl?

You can assume that there are an equal number of males and females in the world.

The outcomes for two kids can be {BB, BG, GB, GG}

Since it is mentioned that one of them is a girl, we can remove the BB option from the sample space. Therefore the sample space has 3 options while only one fits the second condition. Therefore the probability the second child will be a girl too is 1/3.

=0.333

16) A player is randomly dealt a sequence of 13 cards from a deck of 52-cards. All sequences of 13 cards are equally likely. In an equivalent model, the cards are chosen and dealt one at a time.
Solution:
When choosing a card, the dealer is equally likely to pick any of the cards that remain in the deck. Since we are not told anything about the first 12 cards that are dealt, the probability that the 13th card dealt is a King, is the same as the probability that the first card dealt, or in fact any particular card dealt is a King, and this equals: 4/52
Answer: 1/13

17) Suppose a life insurance company sells a $240,000 one year term life insurance policy to a 25-year old female for $210. The probability that the female survives the year is .999592. Find the expected value of this policy for the insurance company.

Solution:

P(company loses the money) = 0.99592

P(company does not lose the money) = 0.000408

The amount of money company loses if it loses = 240,000 – 210 = 239790

While the money it gains is $210

Expected money the company will have to give = 239790*0.000408 = 97.8

Expect money company gets = 210.

Therefore the value = 210 – 98 = $112

18) Find the variance and mean of the number obtained on a throw of an unbiased die.

In Statistics, we have studied that the variance is a measure of the spread or scatter in data.
Likewise, the variability or spread in the values of a random variable may be measured by variance.

For a random variable X which takes on values x1, x2, x3 … xn with probabilities p1, p2, p3 … p n and the expectation is E[X]
The variance of **X** or Var(**X**) is denoted by

$$E[X-u]^2 = \sum(x_i-\mu)^2 p_{x_i} = E[X^2] - (E[X])^2$$

We know that the sample space of this experiment is {1,2,3,4,5,6}

Let's define our random variable X, which represents the number obtained on a throw.

So, the probabilities of the values which our random variable can take are,

P(1) = P(2) = P(3) = P(4) = P(5) = P(6) =

Therefore, the probability distribution of the random variable is,

X	2	3	4	5	6

Probabilities

$$\frac{1}{6} \quad \frac{1}{6} \quad \frac{1}{6} \quad \frac{1}{6} \quad \frac{1}{6} \quad \frac{1}{6}$$

E[X] = $$\sum p_{x_i} x_i = 1 \times \frac{1}{6} + 2 \times \frac{1}{6} + 3 \times \frac{1}{6} + 4 \times \frac{1}{6} + 5 \times \frac{1}{6} + 6 \times \frac{1}{6} = \frac{21}{6}$$

Also, E[X2] $$1^2 \times \frac{1}{6} + 2^2 \times \frac{1}{6} + 3^2 \times \frac{1}{6} + 4^2 \times \frac{1}{6} + 5^2 \times \frac{1}{6} + 6^2 \times \frac{1}{6} = \frac{91}{6}$$

Thus, Var(X) = E[X2] – (E[X])2

$$= (\frac{91}{6}) - (\frac{21}{6})^2 = \frac{91}{6} - \frac{441}{36} = \frac{35}{12}$$

So, therefore mean is $\frac{21}{6}$

and variance is $\frac{35}{12}$

19) The mean population IQ for adults is 100 with an SD of 15. You want to see whether those born prematurely have a lower IQ. To test this, you attain a sample of the IQ's adults that were born prematurely with a sample mean of 95. Your hypothesis is that prematurely born people do not have lower IQs.

Because we know the population mean and standard deviation, as well as the distribution (IQ's are generally normally distributed), we can use a z-test.

Null Hypothesis	H_o: IQ of 95 or above is normal $\mu > 95$
Alternative Hypothesis	H_o: IQ of 95 is not normal

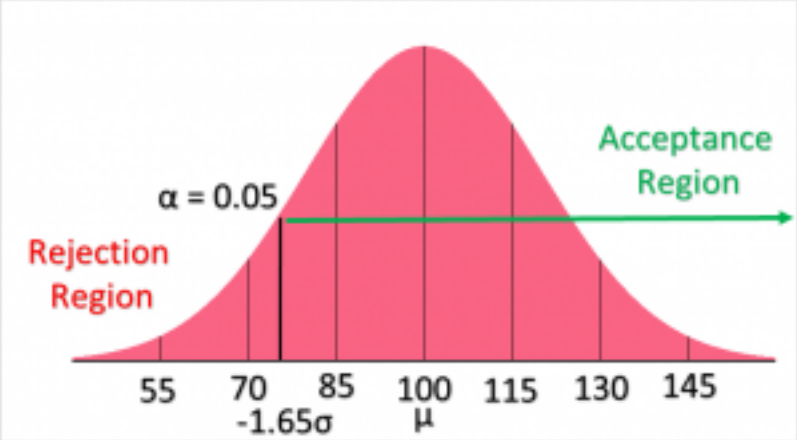
First, we find the z-score

$$Z = \frac{x - \mu}{\sigma} \longrightarrow Z = \frac{95 - 100}{15} = -0.333$$

Next, we find the z-score on a z-table for **negative** values.

Z	**0.00**	**0.01**	**0.02**	**0.03**
0.2	0.42074	0.41683	0.41294	0.40905
0.3	0.38209	0.37828	0.37448	0.37070

With a p-value of **0.3707**, we the fail to reject the null hypothesis.



Type 1 Error Example

In the example above, you can see that we have chosen our confidence level to be at 95%. This gives us an alpha of 0.05. As explained above, a type 1 error occurs when our statistical test rejects the null hypothesis when, **in reality**, the null hypothesis is true.

The main question is, how do we know when a type 1 error has occurred? The only way we could know for certain would be if we had all population values, which we don't. Luckily, we can use the same logic as we do for the confidence level. If we are 95% certain of something occurring, this means that the probability that this thing really didn't occur as the tail end of our rejection region. Therefore, the type 1 error is calculated simply as the **1 minus the probability** that our hypothesis occurred, which is simply our p-value 0.3707.

20) Jane has just begun her new job as on the sales force of a very competitive company. In a sample of 16 sales calls it was found that she closed the contract for an average value of 108 dollars with a standard deviation of 12 dollars. Test at 5% significance that the population mean is at least 100 dollars against the alternative that it is less than 100 dollars. Company policy requires that new members of the sales force must exceed an average of ?100 per contract

during the trial employment period. Can we conclude that Jane has met this requirement at the significance level of 95%?

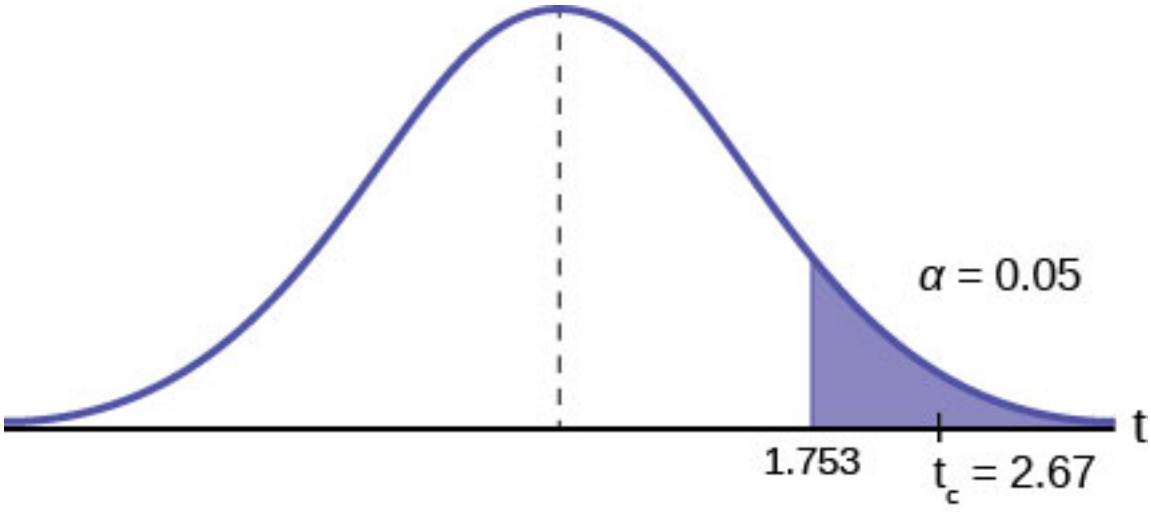
1. H_0: $\mu \leq 100$
   H_a: $\mu > 100$

The null and alternative hypothesis are for the parameter μ because the number of dollars of the contracts is a continuous random variable. Also, this is a one-tailed test because the company has only an interested if the number of dollars per contact is below a particular number not "too high" a number. This can be thought of as making a claim that the requirement is being met and thus the claim is in the alternative hypothesis.

2. Test statistic: $t_c = \frac{\bar{x}-0}{\frac{s}{\sqrt{n}}} = \frac{108-100}{\left(\frac{12}{\sqrt{16}}\right)} = 2.67$

3. Critical value: $t_a = 1.753$ with n-1 degrees of freedom= 15

The test statistic is a Student's t because the sample size is below 30; therefore, we cannot use the normal distribution. Comparing the calculated value of the test statistic and the critical value of (t_a) at a 5% significance level, we see that the calculated value is in the tail of the distribution. Thus, we conclude that 108 dollars per contract is significantly larger than the hypothesized value of 100 and thus we cannot accept the null hypothesis. There is evidence that supports Jane's performance meets company standards.



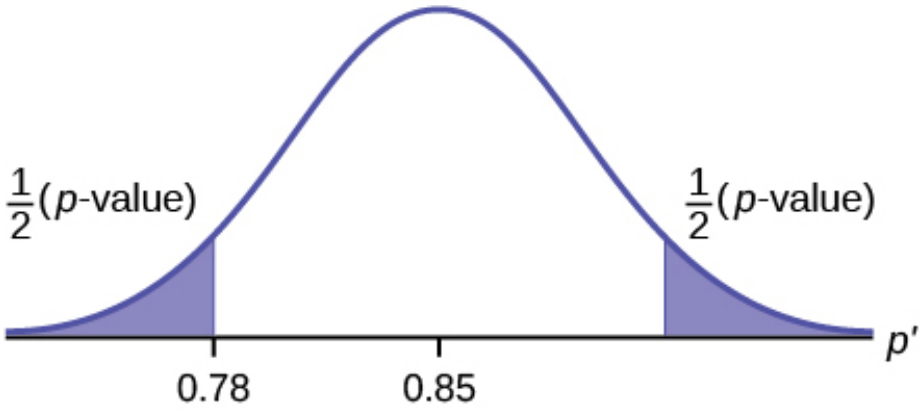
21) A teacher believes that 85% of students in the class will want to go on a field trip to the local zoo. She performs a hypothesis test to determine if the percentage is the same or different from 85%. The teacher samples 50 students and 39 reply that they would want to go to the zoo. For the hypothesis test, use a 1% level of significance.

Since the problem is about percentages, this is a test of single population proportions.

$H_0 : p = 0.85$

$H_a: p \neq 0.85$

$p = 0.7554$



Because $p > \alpha$, we fail to reject the null hypothesis. There is not sufficient evidence to suggest that the proportion of students that want to go to the zoo is not 85%.

TEXT / REFERENCE BOOKS

4. Cathy O'Neil and Rachel Schutt. Doing Data Science, Straight Talk From The Frontline. O'Reilly. 2014.
5. Applied Statistics and Probability For Engineers – By Douglas Montgomery.2016.
6. Avrim Blum, John Hopcroft and Ravindran Kannan. Foundations of Data Science.

QUESTION BANK

Part-A			
Q.No	**Questions**	**Competence**	**BT Level**
1.	Define Statistics	Remember	BTL 1
2.	Differentiate discrete data and continuous data	Analysis	BTL 4
3.	Define Normal Distribution	Remember	BTL 1
4.	List the properties of Normal distribution	Understand	BTL 2
5.	Enumerate the measure of central tendency?	Understand	BTL 2
6.	Differentiate population and sample	Analysis	BTL 4
7.	Define mean and median	Remember	BTL 1
8.	Define Standard Deviation	Remember	BTL 1
9.	Find the mean, median, mode, and range for the following list of values: 1, 2, 4, 7	Apply	BTL 3
10.	Calculate covariance for the following data set: x: 2.1, 2.5, 3.6, 4.0 (mean = 3.1) y: 8, 10, 12, 14 (mean = 11)	Apply	BTL 3
11.	Find the median for the data set: 34, 22, 15, 25, 10	Analysis	BTL 4
12.	Enumerate covariance matrix	Remember	BTL 2
13.	Define covariance?	Remember	BTL 1
14.	Differentiate positive and negative covariance?	Analysis	BTL 4
15.	Describe the measures of Variability	Understand	BTL 2
16.	Why is the multivariate normal distribution so important?	Analysis	BTL 4
17.	Enumerate measures of asymmetry	Understand	BTL 2
18.	Differentiate Null and Alternate Hypothesis	Analysis	BTL 4
19.	Interpret Hypothesis testing?	Understand	BTL 2
20.	Consider the following data points. 17, 16, 21, 18, 15, 17, 21, 19, 11, 23.	Apply	BTL 3

	Find the mean and median.		
PART B			
Q.No	**Questions**	**Competence**	**BT Level**
1.	Illustrate different types of normal distributions	Analysis	BTL 4
2.	Define Hypothesis testing and Discuss how to test the assumptions made regarding a population parameter.	Analysis	BTL 4
3.	Explain the notion of probability, distributions, mean, variance, covariance, covariance matrix with an example.	Analysis	BTL 4
4.	Explain about probability density function?	Analysis	BTL 4
5.	Find the variance and mean of the number obtained on a throw of an unbiased die.	Apply	BTL 3
6.	Explain about T-test, F-test and Z-test?	Analysis	BTL 4
7.	The mean population IQ for adults is 100 with an SD of 15. You want to see whether those born prematurely have a lower IQ. To test this, you attain a sample of the IQ's adults that were born prematurely with a sample mean of 95. Your hypothesis is that prematurely born people do not have lower IQs.	Apply	BTL 3
8.	Explain about descriptive statistics?	Analysis	BTL 4
9.	Suppose a life insurance company sells a $240,000 one year term life insurance policy to a 25-year old female for $210. The probability that the female survives the year is .999592. Find the expected value of this policy for the insurance company.	Apply	BTL 3
10.	A teacher believes that 85% of students in the class will want to go on a field trip to the local zoo. She performs a hypothesis test to determine if the percentage is the same or different from 85%. The teacher samples 50 students and 39 reply that they would want to go to the zoo. For the hypothesis test, use a 1% level of significance.	Apply	BTL 3

SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

SCHOOL OF COMPUTING
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

UNIT – III
Exploratory Data Analysis and the Data Science Process – SCSA3016

UNIT 3 EXPLORATORY DATA ANALYSIS AND THE DATA SCIENCE PROCESS

Exploratory Data Analysis and the Data Science Process - Basic tools (plots, graphs and summary statistics) of EDA -Philosophy of EDA - The Data Science Process – Data Visualization - Basic principles, ideas and tools for data visualization - Examples of exciting projects- Data Visualization using Tableau.

3.1 EXPLORATORY DATA ANALYSIS

Exploratory Data Analysis, or EDA, is an important step in any Data Analysis or Data Science project. EDA is the process of investigating the dataset to discover patterns, and anomalies (outliers), and form hypotheses based on our understanding of the dataset. EDA involves generating summary statistics for numerical data in the dataset and creating various graphical representations to understand the data better.

To explore data in a systematic way, a task that statisticians call exploratory data analysis, or EDA.

EDA is an iterative cycle.

1. Generate questions about your data.
2. Search for answers by visualising, transforming, and modelling your data.
3. Use what you learn to refine your questions and/or generate new questions.

WHAT IS EDA?

Exploratory Data Analysis (EDA) is an approach to analyzing datasets to summarize their main characteristics, often with visual methods. EDA is used for seeing what the data can tell us before the modeling task. It is not easy to look at a column of numbers or a whole spreadsheet and determine important characteristics of the data. It may be tedious, boring, and/or overwhelming to derive insights by looking at plain numbers. Exploratory data analysis techniques have been devised as an aid in this situation. EDA assists Data science professionals in various ways: -

1. Getting a better understanding of data
2. Identifying various data patterns
3. Getting a better understanding of the problem statement.

The EDA is important to,

- Detect outliers and anomalies
- Determine the quality of data
- Determine what statistical models can fit the data
- Find out if the assumptions about the data, that you or your team started out with is correct or way off.
- Extract variables or dimensions on which the data can be pivoted.
- Determine whether to apply univariate or multivariate analytical techniques.
- EDA is typically used for these four goals:

- Exploring a single variable and looking at trends over time.
- Checking data for errors.
- Checking assumptions.
- Looking at relationships between variables

3.1.1. Why is exploratory data analysis important in data science?

The main purpose of EDA is to help look at data before making any assumptions. It can help identify obvious errors, as well as better understand patterns within the data, detect outliers or anomalous events, find interesting relations among the variables.

Data scientists can use exploratory analysis to ensure the results they produce are valid and applicable to any desired business outcomes and goals. EDA also helps stakeholders by confirming they are asking the right questions. EDA can help answer questions about standard deviations, categorical variables, and confidence intervals. Once EDA is complete and insights are drawn, its features can then be used for more sophisticated data analysis or modelling, including machine learning.

3.1.2. Various exploratory data analysis methods like:

- **Descriptive Statistics**, which is a way of giving a brief overview of the dataset we are dealing with, including some measures and features of the sample.
- **Grouping data** (Basic grouping with *group by*)
- **ANOVA**, Analysis Of Variance, which is a computational method to divide variations in an observations set into different components.
- **Correlation and correlation** methods.

Descriptive Statistics: It is a helpful way to understand characteristics of your data and to get a quick summary of it. Pandas in python provide an interesting method **describe()**. The describe function applies basic statistical computations on the dataset like extreme values, count of data points standard deviation etc. Any missing value or NaN value is automatically skipped. describe() function gives a good picture of distribution of data.

Grouping data: Group by is an interesting measure available in pandas which can help us figure out effect of different categorical attributes on other data variables.

ANOVA

- ANOVA stands for Analysis of Variance. It is performed to figure out the relation between the different group of categorical data.
- Under ANOVA we have two measures as result:
  – F-testscore : which shows the variation of groups mean over variation
  – p-value: it shows the importance of the result
- This can be performed using python module scipy method name *f_oneway()*

Correlation and Correlation computation: Correlation is a simple relationship between two variables in a context such that one variable affects the other. Correlation is different from act of *causing*.

3.1.3. Types of EDA

Exploratory data analysis is generally cross-classified in two ways. First, each method is either non-graphical or graphical. And second, each method is either univariate or multivariate (usually just bivariate)

There are broadly two categories of EDA, graphical and non-graphical.

- Univariate Non-graphical
- Multivariate Non-graphical
- Univariate graphical
- Multivariate graphical

Univariate non-graphical: This is the simplest form of data analysis as during this we use just one variable to research the info. The standard goal of univariate non-graphical EDA is to know the underlying sample distribution/ data and make observations about the population. Outlier detection is additionally part of the analysis.

The characteristics of population distribution include:

- **Central tendency:** The central tendency or location of distribution has got to do with typical or middle values. The commonly useful measures of central tendency are statistics called mean, median, and sometimes mode during which the foremost common is mean. For skewed distribution or when there's concern about outliers, the median may be preferred.
- **Spread:** Spread is an indicator of what proportion distant from the middle we are to seek out the find the info values. the quality deviation and variance are two useful measures of spread. The variance is that the mean of the square of the individual deviations and therefore the variance is the root of the variance
- **Skewness and kurtosis:** Two more useful univariates descriptors are the skewness and kurtosis of the distribution. Skewness is that the measure of asymmetry and kurtosis may be a more subtle measure of peakedness compared to a normal distribution

Multivariate non-graphical: Multivariate non-graphical EDA technique is usually wont to show the connection between two or more variables within the sort of either cross-tabulation or statistics.

- For categorical data, an extension of tabulation called cross-tabulation is extremely useful. For 2 variables, cross-tabulation is preferred by making a two-way table with column headings that match the amount of one-variable and row headings that match the amount of the opposite two variables, then filling the counts with all subjects that share an equivalent pair of levels.

- For each categorical variable and one quantitative variable, we create statistics for quantitative variables separately for every level of the specific variable then compare the statistics across the amount of categorical variable.
- Comparing the means is an off-the-cuff version of ANOVA and comparing medians may be a robust version of one-way ANOVA.

Univariate graphical: Non-graphical methods are quantitative and objective, they are doing not give the complete picture of the data; therefore, graphical methods are more involve a degree of subjective analysis also are required.

Common sorts of univariate graphics are:

- **Histogram:** The foremost basic graph is a histogram, which may be a bar plot during which each bar represents the frequency (count) or proportion (count/total count) of cases for a variety of values. Histograms are one of the simplest ways to quickly learn a lot about your data, including central tendency, spread, modality, shape and outliers.
- **Stem-and-leaf plots:** An easy substitute for a histogram may be stem-and-leaf plots. It shows all data values and therefore the shape of the distribution.
- **Boxplots:** Another very useful univariate graphical technique is that the boxplot. Boxplots are excellent at presenting information about central tendency and show robust measures of location and spread also as providing information about symmetry and outliers, although they will be misleading about aspects like multimodality. One among the simplest uses of boxplots is within the sort of side-by-side boxplots.
- **Quantile-normal plots:** The ultimate univariate graphical EDA technique is that the most intricate. it's called the quantile-normal or QN plot or more generally the quantile-quantile or QQ plot. it's wont to see how well a specific sample follows a specific theoretical distribution. It allows detection of non-normality and diagnosis of skewness and kurtosis

Multivariate graphical: A graphical representation always gives you a better understanding of the relationship, especially among multiple variables.

Other common sorts of multivariate graphics are:

- **Scatterplot:** For 2 quantitative variables, the essential graphical EDA technique is that the scatterplot, sohas one variable on the x-axis and one on the y-axis and therefore the point for every case in your dataset.
- **Run chart:** It's a line graph of data plotted over time.
- **Heat map:** It's a graphical representation of data where values are depicted by color.
- **Multivariate chart:** It's a graphical representation of the relationships between factors and response.
- **Bubble chart:** It's a data visualization that displays multiple circles (bubbles) in two-dimensional plot.

3.2. TOOLS in EDA

Non-graphical exploratory data analysis involves data collection and reporting in nonvisual or non-pictorial formats. Some of the most common data science tools used to create an EDA include:

- **Python:** An interpreted, object-oriented programming language with dynamic semantics. Its high-level, built-in data structures, combined with dynamic typing and dynamic binding, make it very attractive for rapid application development, as well as for use as a scripting or glue language to connect existing components together. Python and EDA can be used together to identify missing values in a data set, which is important so you can decide how to handle missing values for machine learning.
- **R:** An open-source programming language and free software environment for statistical computing and graphics supported by the R Foundation for Statistical Computing.

  The R language is widely used among statisticians in data science in developing statistical observations and data analysis.

Graphical exploratory data analysis employs visual tools to display data, such as:

Box plots

- Box plots are used where there is a need to summarize data on an interval scale like the ones on the stock market, where ticks observed in one whole day may be represented in a single box, highlighting the lowest, highest, median and outliers.

Heatmap

- Heatmaps are most often used for the representation of the correlation between variables. Here is an example of a heatmap.
- As you can see from the chart, there is a strong correlation between density and residual sugar and absolutely no correlation between alcohol and residual sugar.

Histograms

- The histogram is the graphical representation of numerical data that splits the data into ranges. The taller the bar, the greater the number of data points falling in that range. A good example here is the height data of a class of students. You would notice that the height data looks like a bell curves for a particular class with most the data lying within a certain range and a few of outside these ranges. There will be outliers too, either very short or very small.

Line graphs: one of the most basic types of charts that plots data points on a graph; has a wealth of uses in almost every field of study.

Pictograms: replace numbers with images to visually explain data. They're common in the design of infographics, as well as visuals that data scientists can use to explain complex findings to non-data-scientist professionals and the public.

Scattergrams or scatterplots: typically used to display two variables in a set of data and then look for correlations among the data. For example, scientists might use it to evaluate the presence of two particular chemicals or gases in marine life in an effort to look for a relationship between the two variables.

3.3 PHILOSOPHY OF EDA

- The father of EDA is John Tukey who officially coined the term in his 1977 masterpiece. Lyle Jones, the editor of the multi-volume "The collected works of John W. Tukey: Philosophy and principles of data analysis" describes EDA as "an attitude towards flexibility that is absent of prejudice".
- The key frame of mind when engaging with EDA and thus VDA is to approach the dataset with little to no expectation, and not be influenced by rigid parameterisations. EDA commands to let the data speak for itself. To use the words of Tukey (1977, preface):
- "It is important to understand what you CAN DO before you learn to measure how WELL you seem to have DONE it… Exploratory data analysis can never be the whole story, but nothing else can serve as the foundation stone –as the first step."
- Since the inception of EDA as unifying class of methods, it has influenced the development of several other major statistical developments including in non-parametric statistics, robust analysis, data mining, and visual data analytics. These classes of methods are motivated by the need to stop relying on rigid assumption-driven mathematical formulations that often fail to be confirmed by observables.
- EDA is not identical to statistical graphics although the two terms are used almost interchangeably. Statistical graphics is a collection of techniques--all graphically based and all focusing on one data characterization aspect. EDA encompasses a larger venue; EDA is an approach to data analysis that postpones the usual assumptions about what kind of model the data follow with the more direct approach of allowing the data itself to reveal its underlying structure and model. EDA is not a mere collection of techniques; EDA is a philosophy as to how we dissect a data set; what we look for; how we look; and how we interpret. It is true that EDA heavily uses the collection of techniques that we call "statistical graphics", but it is not identical to statistical graphics.

3.4. DATA SCIENCE PROCESS

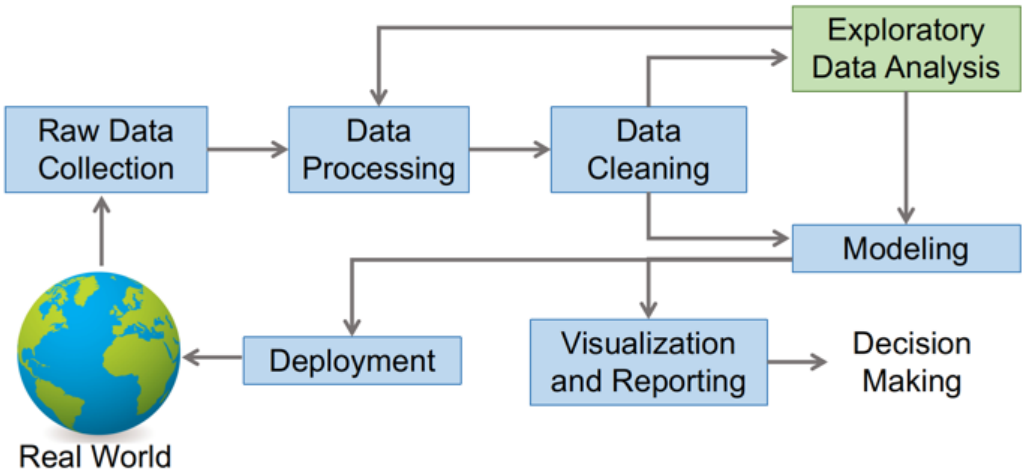


Figure 3.1: Data Science process

The key steps involved in Data Science Modelling are:
Step 1: Understanding the Problem
Step 2: Data Extraction
Step 3: Data Cleaning
Step 4: Exploratory Data Analysis
Step 5: Feature Selection
Step 6: Incorporating Machine Learning Algorithms
Step 7: Testing the Models
Step 8: Deploying the Model

Step 1: Understanding the Problem

The first step involved in Data Science Modelling is understanding the problem. A Data Scientist listens for keywords and phrases when interviewing a line-of-business expert about a business challenge. The Data Scientist breaks down the problem into a procedural flow that always involves a holistic understanding of the business challenge, the Data that must be collected, and various Artificial Intelligence and Data Science approach that can be used to address the problem.

Step 2: Data Extraction

- The next step in Data Science Modelling is Data Extraction. Not just any Data, but the Unstructured Data pieces you collect, relevant to the business problem you're trying to address. The Data Extraction is done from various sources online, surveys, and existing Databases.

Step 3: Data Cleaning

Data Cleaning is useful as you need to sanitize Data while gathering it. Data cleaning is the process of detecting, correcting and ensuring that your given data set is free from error, consistent and usable by identifying any errors or corruptions in the data, correcting or deleting them, or manually processing them as needed to prevent the error from corrupting our final analysis.
The following are some of the most typical causes of Data Inconsistencies and Errors:

- Duplicate items are reduced from a variety of Databases.
- The error with the input Data in terms of Precision.
- Changes, Updates, and Deletions are made to the Data entries.
- Variables with missing values across multiple Databases.

Steps In Data Preprocessing:

- Gathering the data
- Import the dataset & Libraries
- Dealing with Missing Values
- Divide the dataset into Dependent & Independent variable
- dealing with Categorical values
- Split the dataset into training and test set
- Feature Scaling

Gathering the data

- Data is raw information, its the representation of both human and machine observation of the world. Dataset entirely depends on what type of problem you want to solve. Each problem in machine learning has its own unique approach.

Some website to get the dataset :

- Kaggle:
  https://www.kaggle.com/datasets
- UCI Machine Learning Repository: One of the oldest sources on the web to get the dataset.
  http://mlr.cs.umass.edu/ml/
- This awesome GitHub repository has high-quality datasets.
  https://github.com/awesomedata/awesome-public-datasets

Import the dataset & Libraries

- First step is usually importing the libraries that will be needed in the program. A library is essentially a collection of modules that can be called and used.
- Pandas offer tools for cleaning and process your data. It is the most popular Python library that is used for data analysis. In pandas, a data table is called a dataframe.

Dealing with Missing Values

- Sometimes we may find some data are missing in the dataset. if we found then we will remove those rows or we can calculate either **mean, mode or median** of the feature and replace it with missing values. This is an approximation which can add variance to the dataset.

  #**Check for null values-** dataset.isna() or dataset.isnull() to see the null values in dataset.

#**Drop Null values-** Pandas provide a **dropna()** function that can be used to drop either row or columns with missing data.

#Replacing Null values with Strategy: For replacing null values we use the strategy that can be applied on a feature which has numeric data. We can calculate the *Mean, Median or Mode* of the feature and replace it with the missing values.

- De-Duplicate means remove all duplicate values. There is no need for duplicate values in data analysis. These values only affect the accuracy and efficiency of the analysis result. To find duplicate values in the dataset we will use a simple dataframe function i.e. duplicated(). Let's see the example:

  dataset.duplicated()

Feature Scaling

- The final step of data preprocessing is to apply the very important feature scaling.
- Feature Scaling is a technique to standardize the independent features present in the data in a fixed range. It is performed during the data pre-processing.
- **Why Scaling** :- Most of the times, your dataset will contain features highly varying in magnitudes, units and range. But since, most of the machine learning algorithms use Euclidean distance between two data points in their computations, this is a problem.

Standardization and Normalization

- Data Standardization and Normalization is a common practice in machine learning.
- Standardization is another scaling technique where the values are centered around the mean with a unit standard deviation. This means that the mean of the attribute becomes zero and the resultant distribution has a unit standard deviation.
- Normalization is a scaling technique in which values are shifted and rescaled so that they end up ranging between 0 and 1. It is also known as Min-Max scaling.

Step 4: Exploratory Data Analysis

- Exploratory Data Analysis (EDA) is a robust technique for familiarising yourself with Data and extracting useful insights. Data Scientists sift through Unstructured Data to find patterns and infer relationships between Data elements. Data Scientists use Statistics and Visualisation tools to summarise Central Measurements and variability to perform EDA.

Step 5: Feature Selection

- Feature Selection is the process of identifying and selecting the features that contribute the most to the prediction variable or output that you are interested in, either automatically or manually.
- The presence of irrelevant characteristics in your Data can reduce the Model accuracy and cause your Model to train based on irrelevant features. In other words, if the features are strong enough, the Machine Learning Algorithm will give fantastic outcomes.
- Two types of characteristics must be addressed:
    - Consistent characteristics that are unlikely to change.

Variable characteristics whose values change over time

Step 6: Incorporating Machine Learning Algorithms

- This is one of the most crucial processes in Data Science Modelling as the Machine Learning Algorithm aids in creating a usable Data Model. There are a lot of algorithms to pick from, the Model is selected based on the problem. There are three types of Machine Learning methods that are incorporated:

1) Supervised Learning

- It is based on the results of a previous operation that is related to the existing business operation. Based on previous patterns, Supervised Learning aids in the prediction of an outcome. Some of the Supervised Learning Algorithms are:
    - Linear Regression
    - Random Forest
    - Support Vector Machines

2) Unsupervised Learning

- This form of learning has no pre-existing consequence or pattern. Instead, it concentrates on examining the interactions and connections between the presently available Data points. Some of the Unsupervised Learning Algorithms are:
    - KNN (k-Nearest Neighbors)
    - K-means Clustering
    - Hierarchical Clustering
    - Anomaly Detection

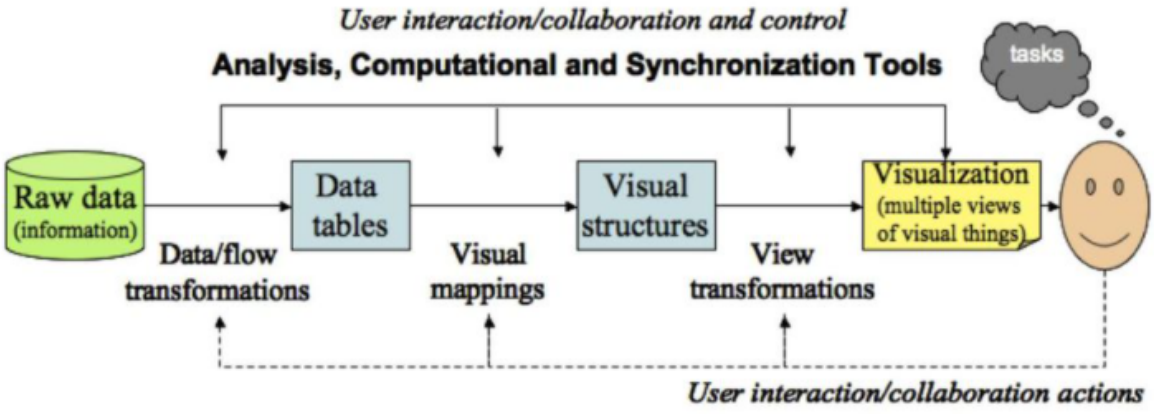
3.5 DATA VISUALIZATION

- Data visualization is the process of translating large data sets and metrics into charts, graphs and other visuals.

- The resulting visual representation of data makes it easier to identify and share real-time trends, outliers, and new insights about the information represented in the data.
- Data visualization is one of the steps of the data science process, which states that after data has been collected, processed and modeled, it must be visualized for conclusions to be made.
- Data visualization is also an element of the broader data presentation architecture (DPA) discipline, which aims to identify, locate, manipulate, format and deliver data in the most efficient way possible.

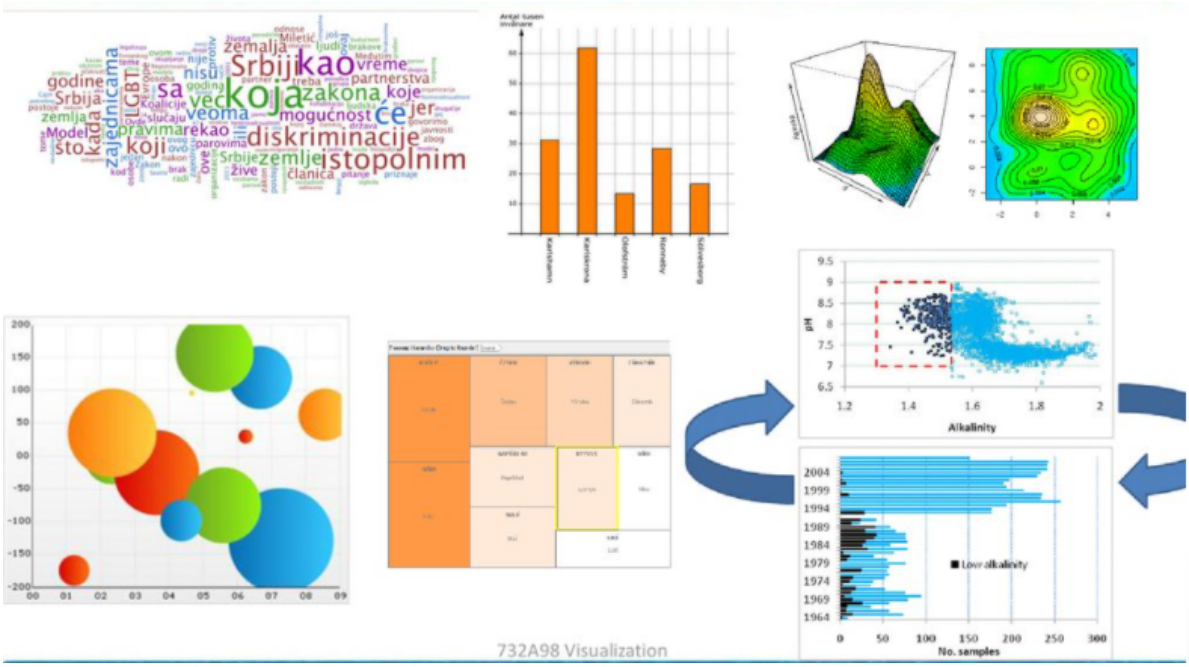
Why Data Visualization is important?

- It's hard to think of a professional industry that doesn't benefit from making data more understandable. Every STEM field benefits from understanding data—and so do fields in government, finance, marketing, history, consumer goods, service industries, education, sports, and so on. And, since visualization is so prolific, it's also one of the most useful professional skills to develop. The better we can convey the points visually, whether in a dashboard or a slide deck, the better we can leverage that information. The concept of the citizen data scientist is on the rise. Skill sets are changing to accommodate a data-driven world. It is increasingly valuable for professionals to be able to use data to make decisions and use visuals to tell stories of when data informs the who, what, when, where, and how. While traditional education typically draws a distinct line between creative storytelling and technical analysis, the modern professional world also values those who can cross between the two: data visualization sits right in the middle of analysis and visual storytelling.



Key ingredient: mapping data columns to visual structures (aesthetics)

- **Some examples of Data Visualization**



Common general types of data visualization:

- Charts
- Tables
- Graphs
- Maps
- Infographics
- Dashboards
- **More specific examples of methods to visualize data:**
- Area Chart
- Bar Chart
- Box-and-whisker Plots
- Bubble Cloud
- Bullet Graph
- Cartogram
- Circle View
- Dot Distribution Map
- Gantt Chart

- Heat Map
- Highlight Table
- Histogram
- Matrix
- Network
- Polar Area
- Radial Tree
- Scatter Plot (2D or 3D)
- Streamgraph
- Text Tables
- Timeline
- Treemap
- Wedge Stack Graph
- Word Cloud
- And any mix-and-match combination in a dashboard

Challenges in Data Visualization

- Which graphs can be used for analysis of my data?
- How to create these graphs
- How should these graphs be analysed?
- How to make these graphs looking good for publication or presentation?

3.6. Data Visualization Tools

1. Tableau

- It is a business intelligence service that aids people in visualizing as well as understanding their data it's also one of those very widely used services in the field of business intelligence. It allows you to design an interactive reports dashboard and worksheets to obtain business visions it has outstanding visualization capabilities and has a great performance.

Pros:

- Outstanding visual library

- User friendly
- Great performance
- Connectivity to data
- Powerful computation
- Quick insights

Cons:

- Inflexible pricing
- No option for auto-refresh
- Restrictive imports
- Manual updates for static features

2. Power BI

- Power BI, Microsoft's easy-to-use data visualization tool, is available for both on-premise installation and deployment on the cloud infrastructure. Power BI is one of the most complete data visualization tools that supports a myriad of backend databases, including Teradata, Salesforce, PostgreSQL, Oracle, Google Analytics, Github, Adobe Analytics, Azure, SQL Server, and Excel. The enterprise-level tool creates stunning visualizations and delivers real-time insights for fast decision-making.

The Pros of Power BI:

- No requirement for specialized tech support
- Easily integrates with existing applications
- Personalized, rich dashboard
- High-grade security
- No speed or memory constraints
- Compatible with Microsoft products

The Cons of Power BI:

- Cannot work with varied, multiple datasets

3. Dundas BI

- Dundas BI offers highly-customizable data visualizations with interactive scorecards, maps, gauges, and charts, optimizing the creation of ad-hoc, multi-page reports. By providing users full control over visual elements, Dundas BI simplifies the complex operation of cleansing, inspecting, transforming, and modeling big datasets.

The Pros of Dundas BI:

- Exceptional flexibility

- A large variety of data sources and charts
- Wide range of in-built features for extracting, displaying, and modifying data

The Cons of Dundas BI:

- No option for predictive analytics
- 3D charts not supported

4. JupyteR

- A web-based application, JupyteR, is one of the top-rated data visualization tools that enable users to create and share documents containing visualizations, equations, narrative text, and live code. JupyteR is ideal for data cleansing and transformation, statistical modeling, numerical simulation, interactive computing, and machine learning.

The Pros of JupyteR:

- Rapid prototyping
- Visually appealing results
- Facilitates easy sharing of data insights

The Cons of JupyteR:

- Tough to collaborate
- At times code reviewing becomes complicated

5. Zoho Reports

- Zoho Reports, also known as Zoho Analytics, is a comprehensive data visualization tool that integrates Business Intelligence and online reporting services, which allow quick creation and sharing of extensive reports in minutes. The high-grade visualization tool also supports the import of Big Data from major databases and applications.

The Pros of Zoho Reports:

- Effortless report creation and modification
- Includes useful functionalities such as email scheduling and report sharing
- Plenty of room for data
- Prompt customer support.

The Cons of Zoho Reports:

- User training needs to be improved

- The dashboard becomes confusing when there are large volumes of data

6. GoogleCharts

- One of the major players in the data visualization market space, Google Charts, coded with SVG and HTML5, is famed for its capability to produce graphical and pictorial data visualizations. Google Charts offers zoom functionality, and it provides users with unmatched cross-platform compatibility with iOS, Android, and even the earlier versions of the Internet Explorer browser.

The Pros of Google Charts:

- User-friendly platform
- Easy to integrate data
- Visually attractive data graphs
- Compatibility with Google products.

The Cons of Google Charts:

- The export feature needs fine-tuning
- Inadequate demos on tools
- Lacks customization abilities
- Network connectivity required for visualization

7. Sisense

Regarded as one of the most agile data visualization tools, Sisense gives users access to instant data analytics anywhere, at any time. The best-in-class visualization tool can identify key data patterns and summarize statistics to help decision-makers make data-driven decisions.

The Pros of Sisense:

- Ideal for mission-critical projects involving massive datasets
- Reliable interface
- High-class customer support
- Quick upgrades
- Flexibility of seamless customization

The Cons of Sisense:

- Developing and maintaining analytic cubes can be challenging
- Does not support time formats
- Limited visualization versions

8. Plotly

- An open-source data visualization tool, Plotly offers full integration with analytics-centric programming languages like Matlab, Python, and R, which enables complex visualizations. Widely used for collaborative work, disseminating, modifying, creating, and sharing interactive, graphical data, Plotly supports both on-premise installation and cloud deployment.

The Pros of Plotly:

- Allows online editing of charts
- High-quality image export
- Highly interactive interface
- Server hosting facilitates easy sharing

The Cons of Plotly:

- Speed is a concern at times
- Free version has multiple limitations
- Various screen-flashings create confusion and distraction

9. Data Wrapper

- Data Wrapper is one of the very few data visualization tools on the market that is available for free. It is popular among media enterprises because of its inherent ability to quickly create charts and present graphical statistics on Big Data. Featuring a simple and intuitive interface, Data Wrapper allows users to create maps and charts that they can easily embed into reports.

The Pros of Data Wrapper:

- Does not require installation for chart creation
- Ideal for beginners
- Free to use

The Cons of Data Wrapper:

- Building complex charts like Sankey is a problem
- Security is an issue as it is an open-source tool

10. QlikView

A major player in the data visualization market, Qlikview provides solutions to over 40,000 clients in 100 countries. Qlikview's data visualization tool, besides enabling accelerated, customized visualizations, also incorporates a range of solid features, including analytics, enterprise reporting, and Business Intelligence capabilities.

The Pros of QlikView:

- User-friendly interface
- Appealing, colorful visualizations
- Trouble-free maintenance
- A cost-effective solution

The Cons of QlikView:

- RAM limitations
- Poor customer support
- Does not include the 'drag and drop' feature

3.7. DATA VISUALIZATION WITH PYTHON

Python offers multiple great graphing libraries that come packed with lots of different features.

Here are a few popular plotting libraries:

- Matplotlib: low level, provides lots of freedom
- Pandas Visualization: easy to use interface, built on Matplotlib
- Seaborn: high-level interface, great default styles
- ggplot: based on R's ggplot2, uses Grammar of Graphics
- Plotly: can create interactive plots

Matplotlib

- Matplotlib is a visualization library in Python for 2D plots of arrays. Matplotlib is written in Python and makes use of the NumPy library. It can be used in Python and IPython shells, Jupyter notebook, and web application servers. Matplotlib comes with a wide variety of plots like line, bar, scatter, histogram, etc. which can help us, deep-dive, into understanding trends, patterns, correlations. It was introduced by John Hunter in 2002.

Seaborn

- Conceptualized and built originally at the Stanford University, this library sits on top of *matplotlib*. In a sense, it has some flavors of *matplotlib* while from the visualization point, its is much better than *matplotlib* and has added features as well. Below are its advantages
    - Built-in themes aid better visualization
    - Statistical functions aiding better data insights
    - Better aesthetics and built-in plots
    - Helpful documentation with effective examples

Bokeh

- Bokeh is an interactive visualization library for modern web browsers. It is suitable for large or streaming data assets and can be used to develop interactive plots and dashboards. There

is a wide array of intuitive graphs in the library which can be leveraged to develop solutions. It works closely with PyData tools. The library is well-suited for creating customized visuals according to required use-cases. The visuals can also be made interactive to serve a what-if scenario model. All the codes are open source and available on GitHub.

plotly

- plotly.py is an interactive, open-source, high-level, declarative, and browser-based visualization library for Python. It holds an array of useful visualization which includes scientific charts, 3D graphs, statistical charts, financial charts among others. Plotly graphs can be viewed in Jupyter notebooks, standalone HTML files, or hosted online. Plotly library provides options for interaction and editing. The robust API works perfectly in both local and web browser mode.

plotly

- plotly.py is an interactive, open-source, high-level, declarative, and browser-based visualization library for Python. It holds an array of useful visualization which includes scientific charts, 3D graphs, statistical charts, financial charts among others. Plotly graphs can be viewed in Jupyter notebooks, standalone HTML files, or hosted online. Plotly library provides options for interaction and editing. The robust API works perfectly in both local and web browser mode.

ggplot

- ggplot is a Python implementation of the grammar of graphics. The Grammar of Graphics refers to the mapping of data to aesthetic attributes (colour, shape, size) and geometric objects (points, lines, bars). The basic building blocks according to the grammar of graphics are data, geom (geometric objects), stats (statistical transformations), scale, coordinate system, and facet.
- Using ggplot in Python allows you to develop informative visualizations incrementally, understanding the nuances of the data first, and then tuning the components to improve the visual representations.

3.9. Examples Of Exciting Projects- Exploratory Data Analysis : Iris Dataset

Importing relevant libraries

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn import metrics
sns.set()
```

Source Of Data

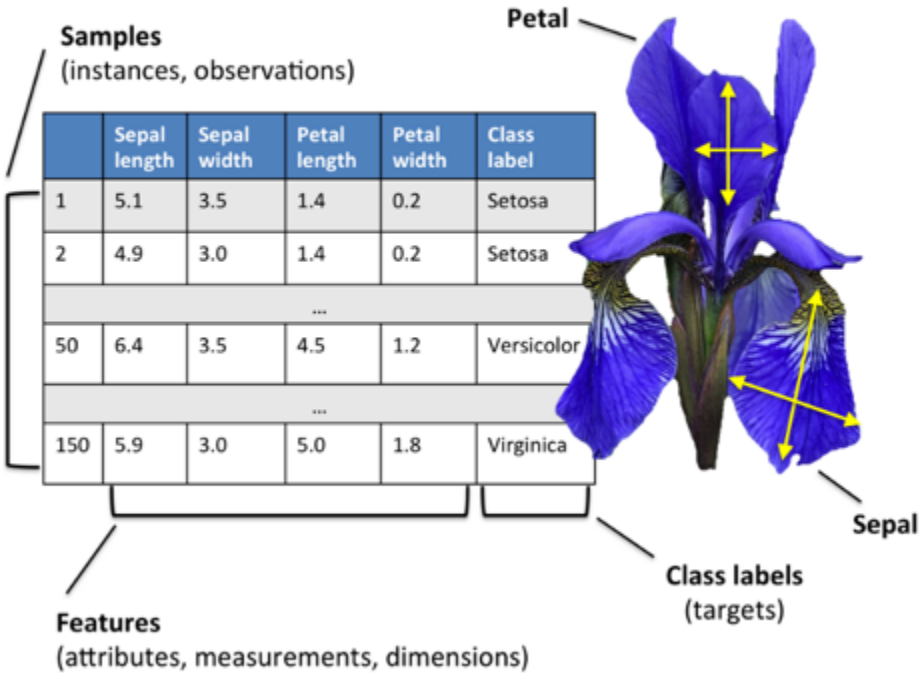
- Data has been stored inside a csv file namely ‘iris.csv’

Loading data

- *iris_data = pd.read_csv('iris.csv')*
- *iris_data*

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa
...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	virginica
146	6.3	2.5	5.0	1.9	virginica
147	6.5	3.0	5.2	2.0	virginica
148	6.2	3.4	5.4	2.3	virginica
149	5.9	3.0	5.1	1.8	virginica

150 rows × 5 columns



	Sepal length	Sepal width	Petal length	Petal width	Class label
1	5.1	3.5	1.4	0.2	Setosa
2	4.9	3.0	1.4	0.2	Setosa
...					
50	6.4	3.5	4.5	1.2	Versicolor
...					
150	5.9	3.0	5.0	1.8	Virginica

Getting Information about the Dataset

- We will use the shape parameter to get the shape of the dataset.

  iris_data.shape

  Output:

- (150, 6)We can see that the dataframe contains 6 columns and 150 rows.

Gaining information from data

iris_data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
Column Non-Null Count Dtype
--- ------ -------------- -----
0 sepal_length 150 non-null float64
1 sepal_width 150 non-null float64
2 petal_length 150 non-null float64
3 petal_width 150 non-null float64
4 species 150 non-null object
dtypes: float64(4), object(1)
memory usage: 6.0+ KB

We can see that only one column has categorical data and all the other columns are of the numeric type with non-Null entries.

- We can see that only one column has categorical data and all the other columns are of the numeric type with non-Null entries.

Data Insights:

- *1 All columns are not having any Null Entries*
- *2 Four columns are numerical type*
- *3 Only Single column categorical type*

Statistical Insight

- *iris_data.describe()*

	sepal_length	sepal_width	petal_length	petal_width
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.054000	3.758667	1.198667
std	0.828066	0.433594	1.764420	0.763161
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.350000	1.300000
75%	6.400000	3.300000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

Data Insights:

- Mean values
- Standard Deviation ,
- Minimum Values
- Maximum Values

Checking Missing Values

- We will check if our data contains any missing values or not. Missing values can occur when no information is provided for one or more items or for a whole unit. We will use the isnull() method.

- *iris_data*.isnull().sum()

```
Id               0
SepalLengthCm    0
SepalWidthCm     0
PetalLengthCm    0
PetalWidthCm     0
Species          0
dtype: int64
```

We can see that no column as any missing value.

Checking For Duplicate Entries

- *iris_data[iris_data.duplicated()]*

	sepal_length	sepal_width	petal_length	petal_width	species
34	4.9	3.1	1.5	0.1	setosa
37	4.9	3.1	1.5	0.1	setosa
142	5.8	2.7	5.1	1.9	virginica

There are 3 duplicates, therefore we must check whether each species data set is balanced in no's or no

Checking the balance

iris_data['species'].value_counts()

```
Species
Iris-setosa        50
Iris-versicolor    50
Iris-virginica     50
dtype: int64
```

Therefore we shouldn't delete the entries as it might imbalance the data sets and hence will prove to be less useful for valuable insights

Data Visualization

Visualizing the target column

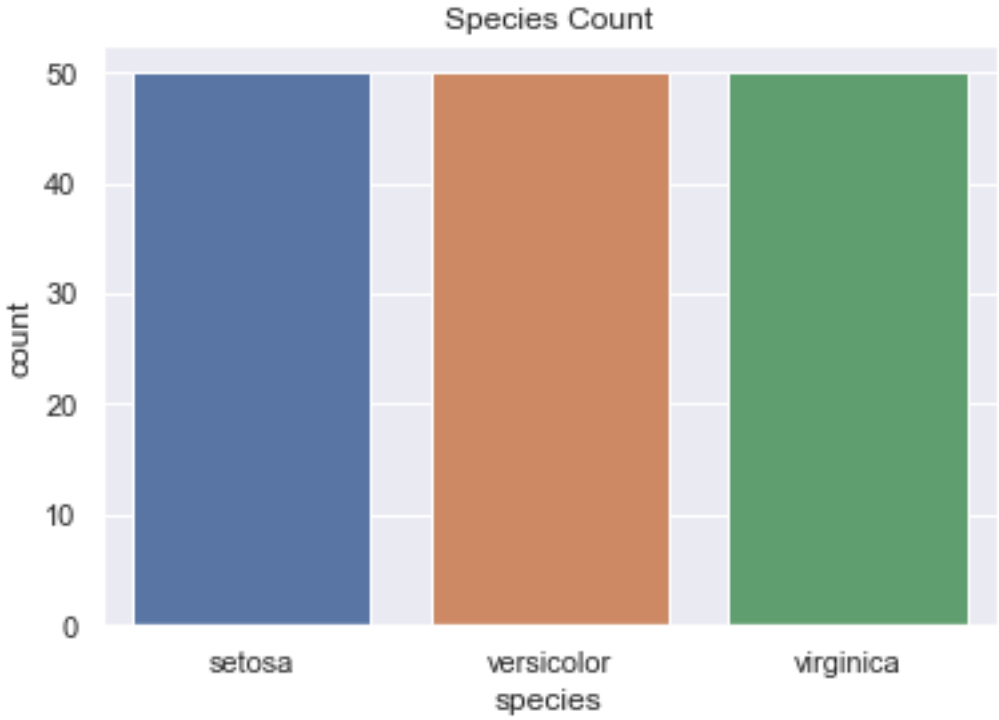
- Our target column will be the Species column because at the end we will need the result according to the species only. **Note:** We will use Matplotlib and Seaborn library for the data visulalization.

importing packages

import seaborn as sns

import matplotlib.pyplot as plt

plt.title('Species Count')
sns.countplot(iris_data['species'])



Data Insight:

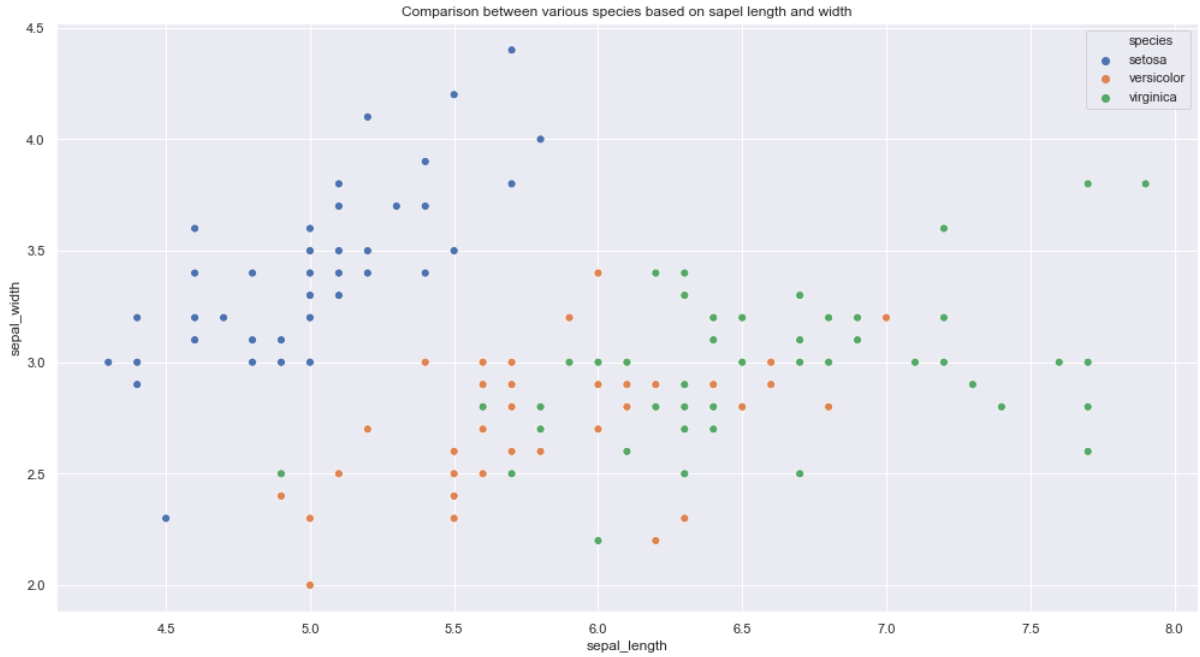
- This further visualizes that species are well balanced
- Each species (Iris virginica, setosa, versicolor) has 50 as it's count

Uni-variate Analysis

Comparison between various species based on sepal length and width

plt.figure(figsize=(17,9))
plt.title('Comparison between various species based on sapel length and width')

sns.scatterplot(iris_data[‘sepal_length’],iris_data[‘sepal_width’],hue =iris_data[‘species’],s=50)

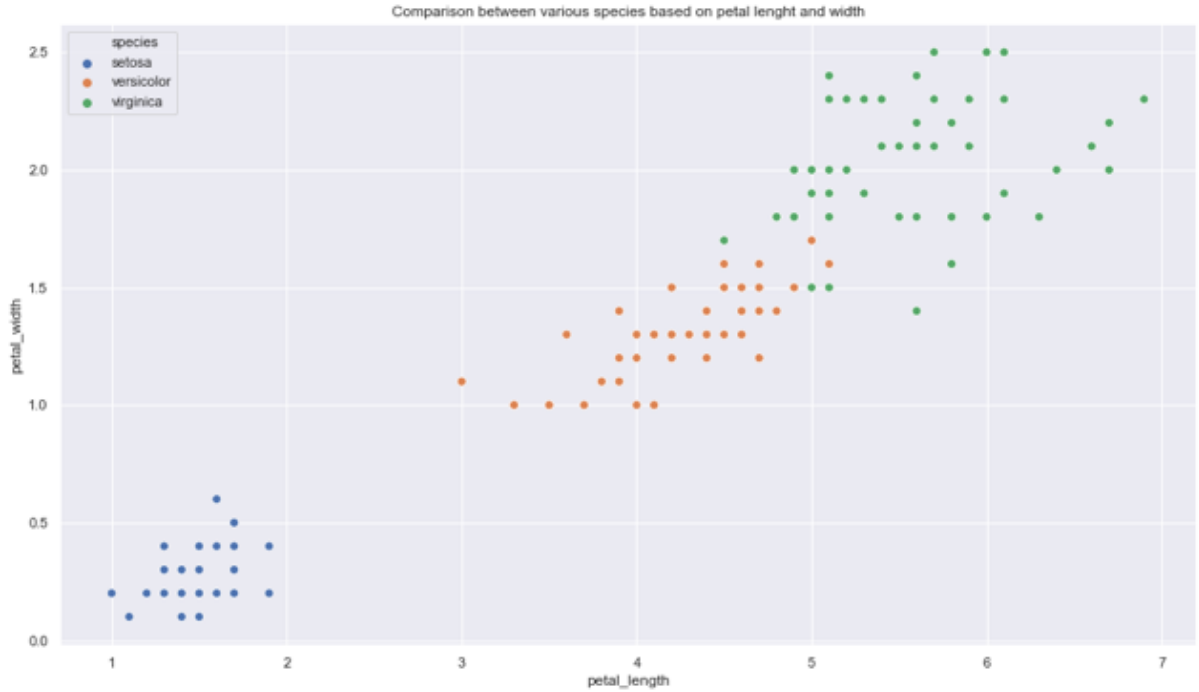


Data Insights:

- Iris Setosa species has smaller sepal length but higher width.
- Versicolor lies in almost middle for length as well as width
- Virginica has larger sepal lengths and smaller sepal widths

Comparison between various species based on petal length and width

plt.figure(figsize=(16,9))
plt.title(‘Comparison between various species based on petal lenght and width’)
sns.scatterplot(iris_data[‘petal_length’], iris_data[‘petal_width’], hue = iris_data[‘species’], s= 50)

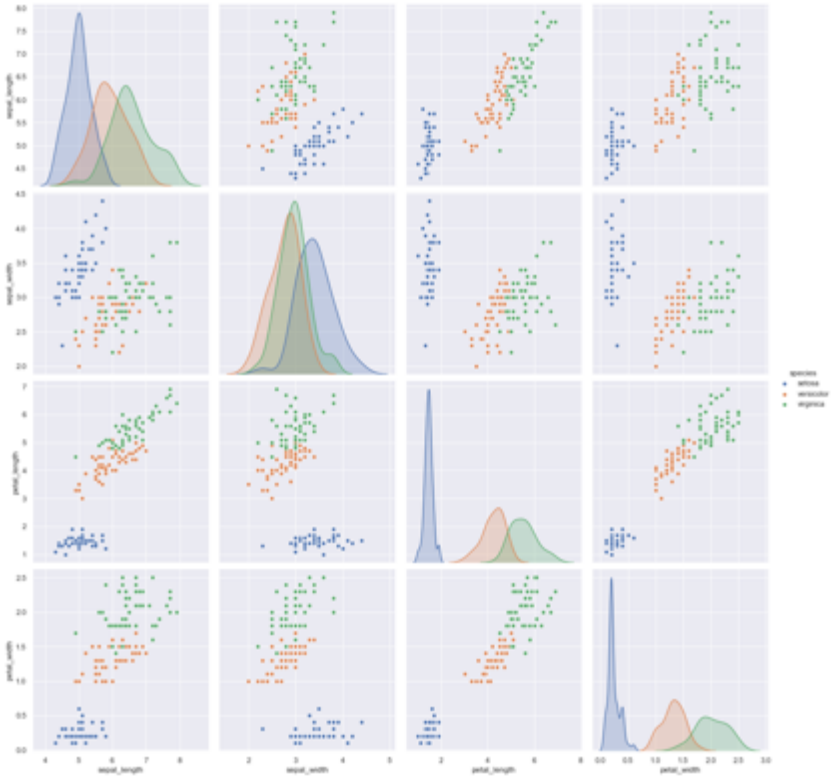


Data Insights

- Setosa species have the smallest petal length as well as petal width
- Versicolor species have average petal length and petal width
- Virginica species have the highest petal length as well as petal width

Let's plot all the column's relationships using a pairplot. It can be used for multivariate analysis.

- *sns.pairplot(iris_data,hue="species",height=4)*



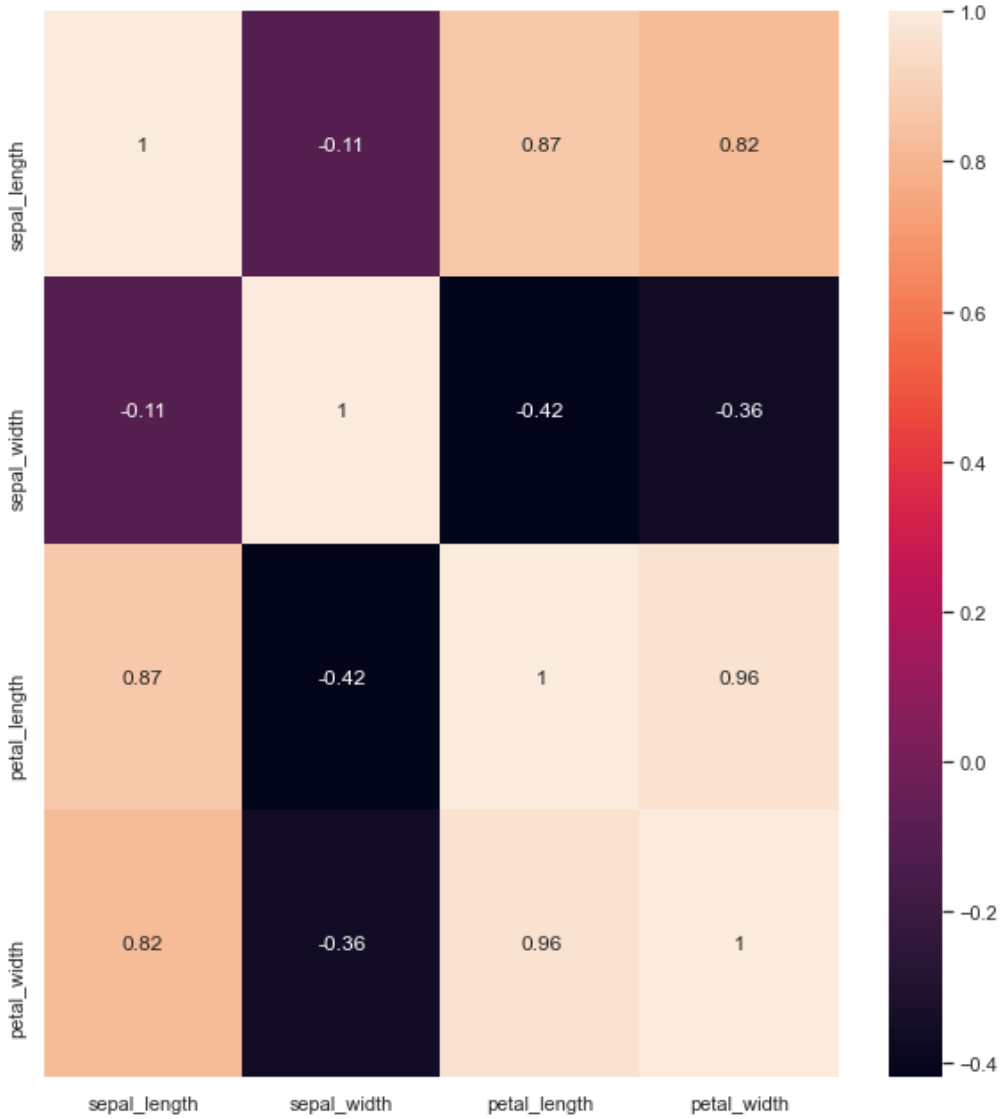
Data Insights:

- High co relation between petal length and width columns.
- Setosa has both low petal length and width
- Versicolor has both average petal length and width
- Virginica has both high petal length and width.
- Sepal width for setosa is high and length is low.
- Versicolor have average values for for sepal dimensions.
- Virginica has small width but large sepal length

The **heatmap** is a data visualization technique that is used to analyze the dataset as colors in two dimensions. Basically, it shows a correlation between all numerical variables in the dataset. In simpler terms, we can plot the above-found correlation using the heatmaps.

Checking Correlation

- *plt.figure(figsize=(10,11))*
  sns.heatmap(iris_data.corr(),annot=True)
  plt.plot()



Data Insights:

Sepal Length and Sepal Width features are slightly correlated with each other

Checking Mean & Median Values for each species

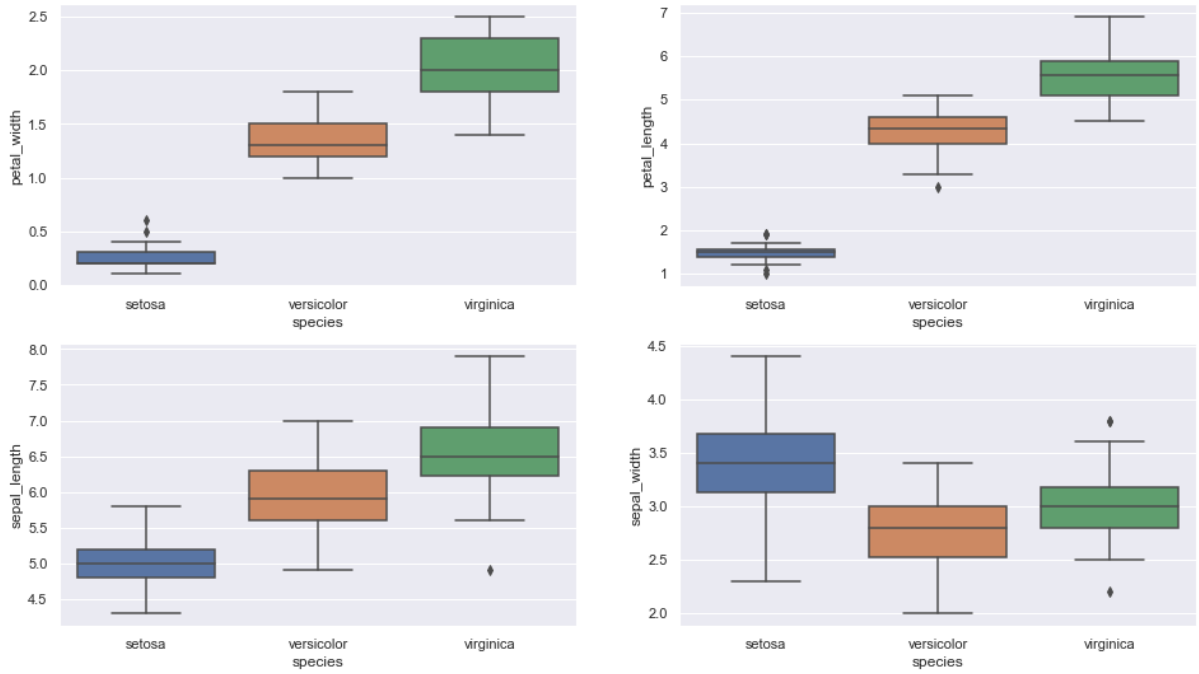
- *iris.groupby('species').agg(['mean', 'median'])*

	sepal_length		sepal_width		petal_length		petal_width	
	mean	median	mean	median	mean	median	mean	median
species								
setosa	5.006	5.0	3.418	3.4	1.464	1.50	0.244	0.2
versicolor	5.936	5.9	2.770	2.8	4.260	4.35	1.326	1.3
virginica	6.588	6.5	2.974	3.0	5.552	5.55	2.026	2.0

visualizing the distribution , mean and median using box plots & violin plots

Box plots to know about distribution

- boxplot to see how the categorical feature “Species” is distributed with all other four input variables
- *fig, axes = plt.subplots(2, 2, figsize=(16,9))*
  sns.boxplot(y=”petal_width”, x= “species”, data=iris_data, orient=’v’ , ax=axes[0, 0])
  sns.boxplot(y=”petal_length”, x= “species”, data=iris_data, orient=’v’ , ax=axes[0, 1])
  sns.boxplot(y=”sepal_length”, x= “species”, data=iris_data, orient=’v’ , ax=axes[1, 0])
  sns.boxplot(y=”sepal_width”, x= “species”, data=iris_data, orient=’v’ , ax=axes[1, 1])
  plt.show()



Data Insights:

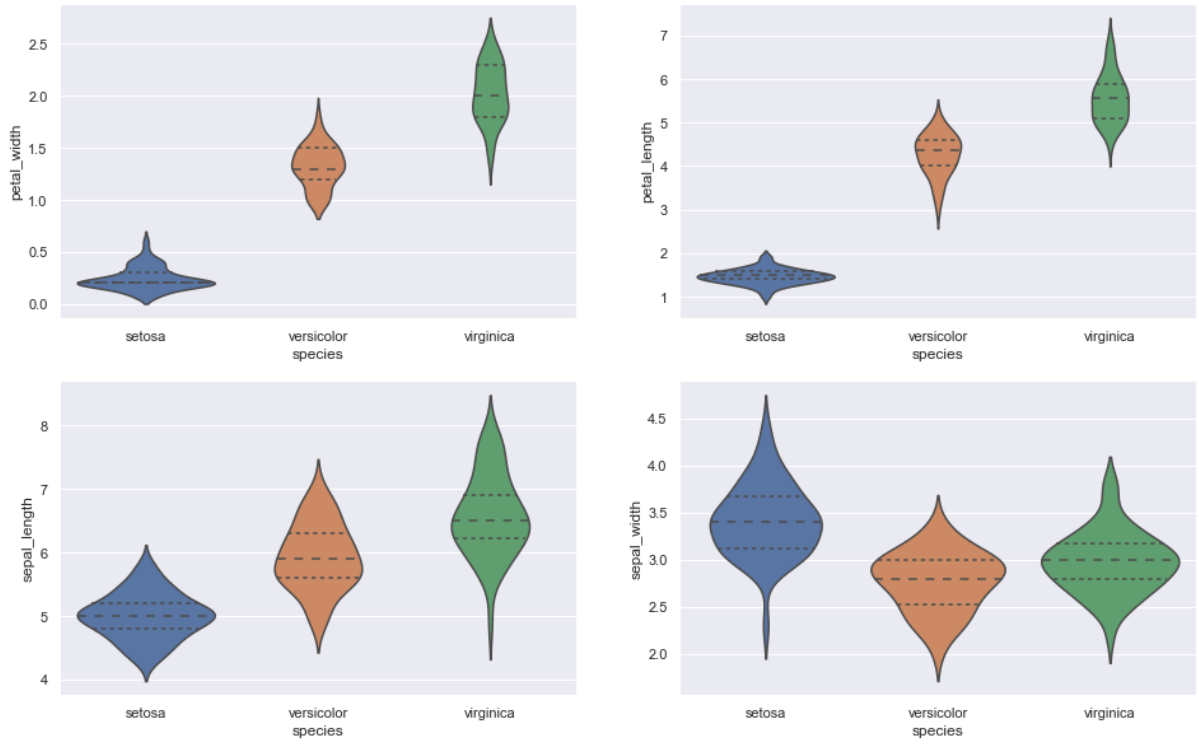
- Setosa is having smaller feature and less distributed
- Versicolor is distributed in a average manner and average features
- Virginica is highly distributed with large no .of values and features
- Clearly the mean/ median values are being shown by each plots for various features(sepal length & width, petal length & width)

Violin Plot for checking distribution

- The violin plot shows density of the length and width in the species. The thinner part denotes that there is less density whereas the fatter part conveys higher density

fig, axes = plt.subplots(2, 2, figsize=(16,10))
sns.violinplot(y=”petal_width”, x= “species”, data=iris_data, orient=’v’ , ax=axes[0, 0],inner=’quartile’)

sns.violinplot(y="petal_length", x= "species", data=iris_data, orient='v' , ax=axes[0, 1],inner='quartile')
sns.violinplot(y="sepal_length", x= "species", data=iris_data, orient='v' , ax=axes[1, 0],inner='quartile')
sns.violinplot(y="sepal_width", x= "species", data=iris_data, orient='v' , ax=axes[1, 1],inner='quartile')
plt.show()



Data Insights:

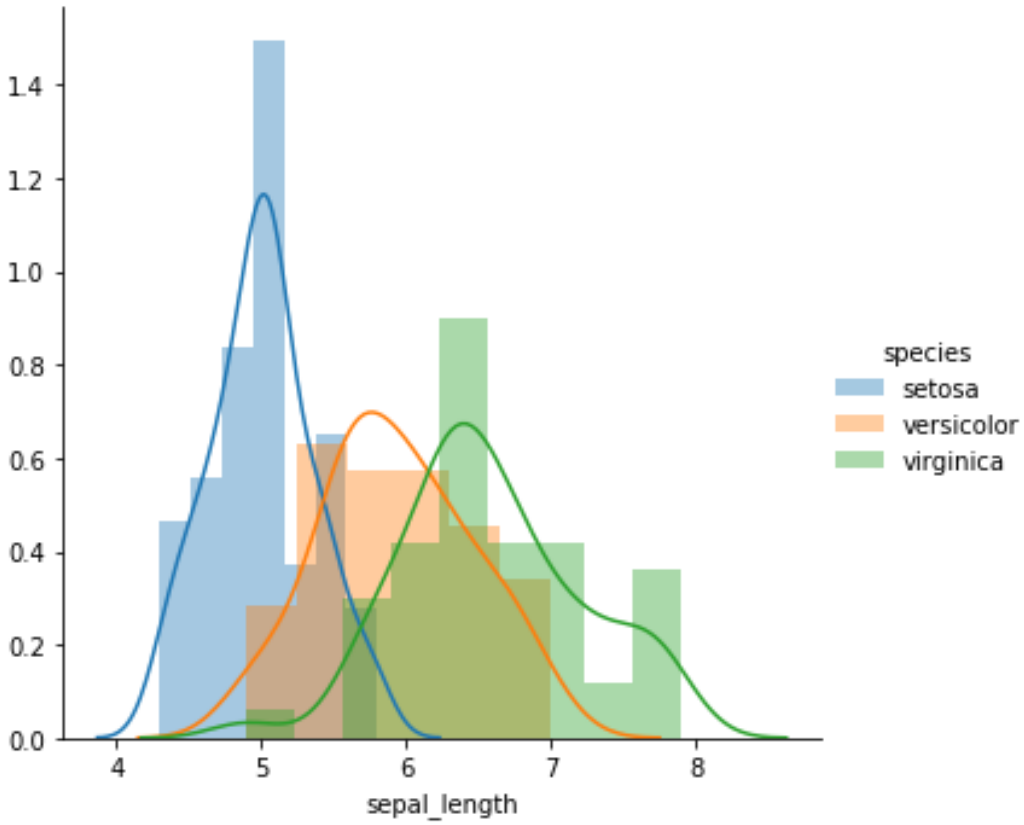
- Setosa is having less distribution and density in case of petal length & width
- Versicolor is distributed in a average manner and average features in case of petal length & width
- Virginica is highly distributed with large no .of values and features in case of sepal length & width
- High density values are depicting the mean/median values, for example: Iris Setosa has highest density at 5.0 cm (sepal length feature) which is also the median value(5.0) as per the table

Mean / Median Table for reference

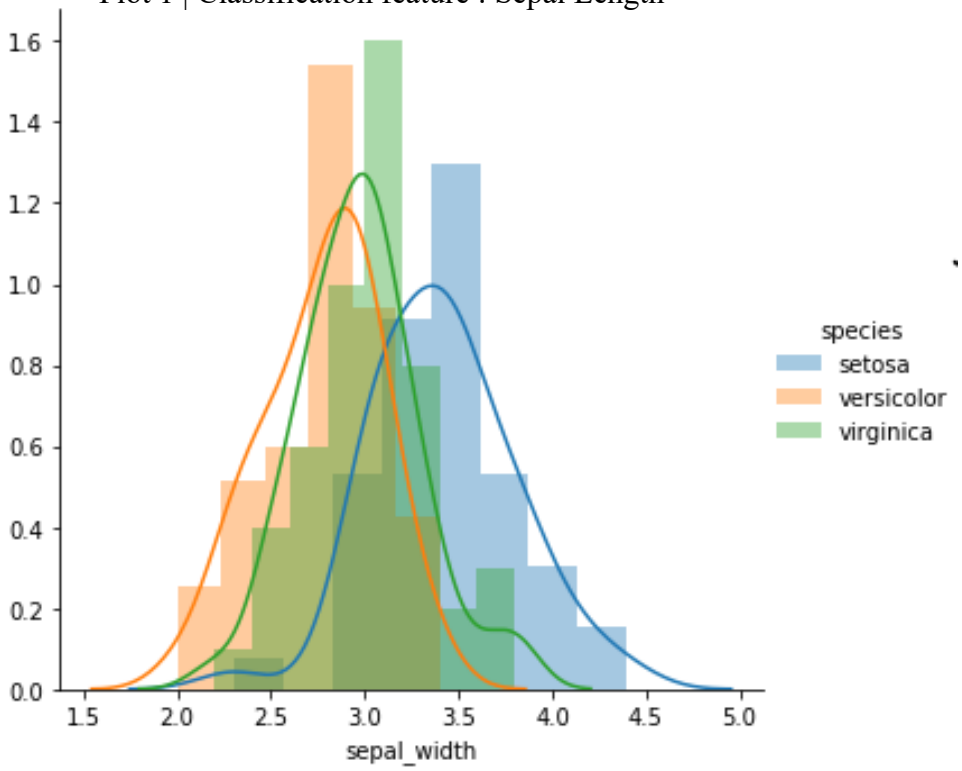
	sepal_length		sepal_width		petal_length		petal_width	
	mean	median	mean	median	mean	median	mean	median
species								
setosa	5.006	5.0	3.418	3.4	1.464	1.50	0.244	0.2
versicolor	5.936	5.9	2.770	2.8	4.260	4.35	1.326	1.3
virginica	6.588	6.5	2.974	3.0	5.552	5.55	2.026	2.0

Plotting the Histogram & Probability Density Function (PDF)

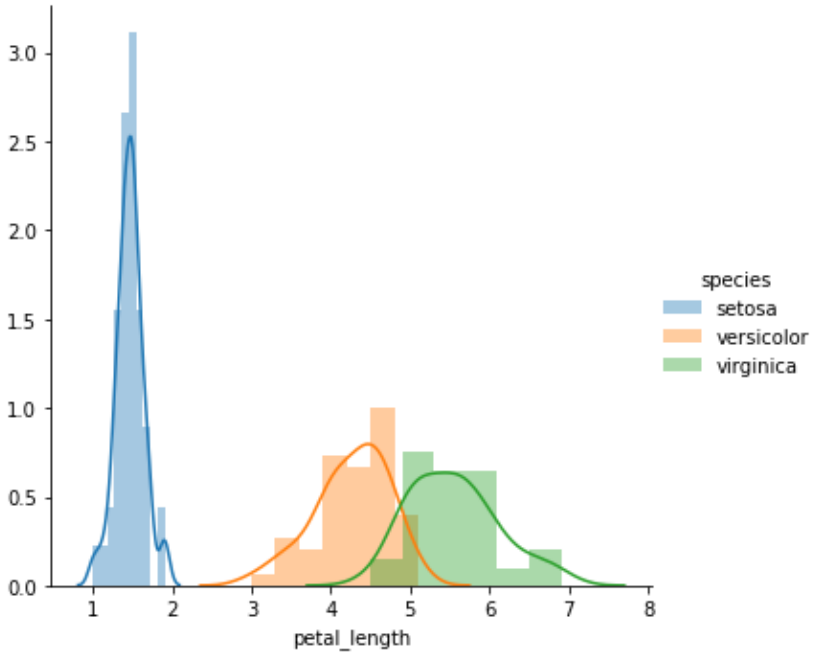
- plotting the probability density function(PDF) with each feature as a variable on X-axis and it's histogram and corresponding kernel density plot on Y-axis.
- *sns.FacetGrid(iris, hue="species", height=5) *
  *.map(sns.distplot, "sepal_length") *
  .add_legend()
- *sns.FacetGrid(iris, hue="species", height=5) *
  *.map(sns.distplot, "sepal_width") *
  .add_legend()
- *sns.FacetGrid(iris, hue="species", height=5) *
  *.map(sns.distplot, "petal_length") *
  .add_legend()
- *sns.FacetGrid(iris, hue="species", height=5) *
  *.map(sns.distplot, "petal_width") *
  .add_legend()
  plt.show()



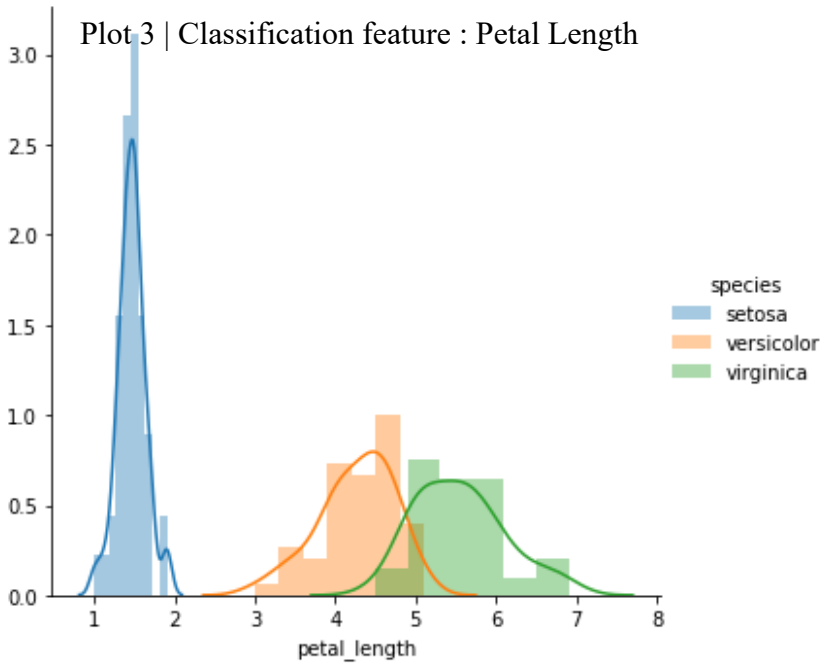
Plot 1 | Classification feature : Sepal Length



Plot 2 | Classification feature : Sepal Width



Plot 3 | Classification feature : Petal Length



Plot 4 | Classification feature : Petal Width

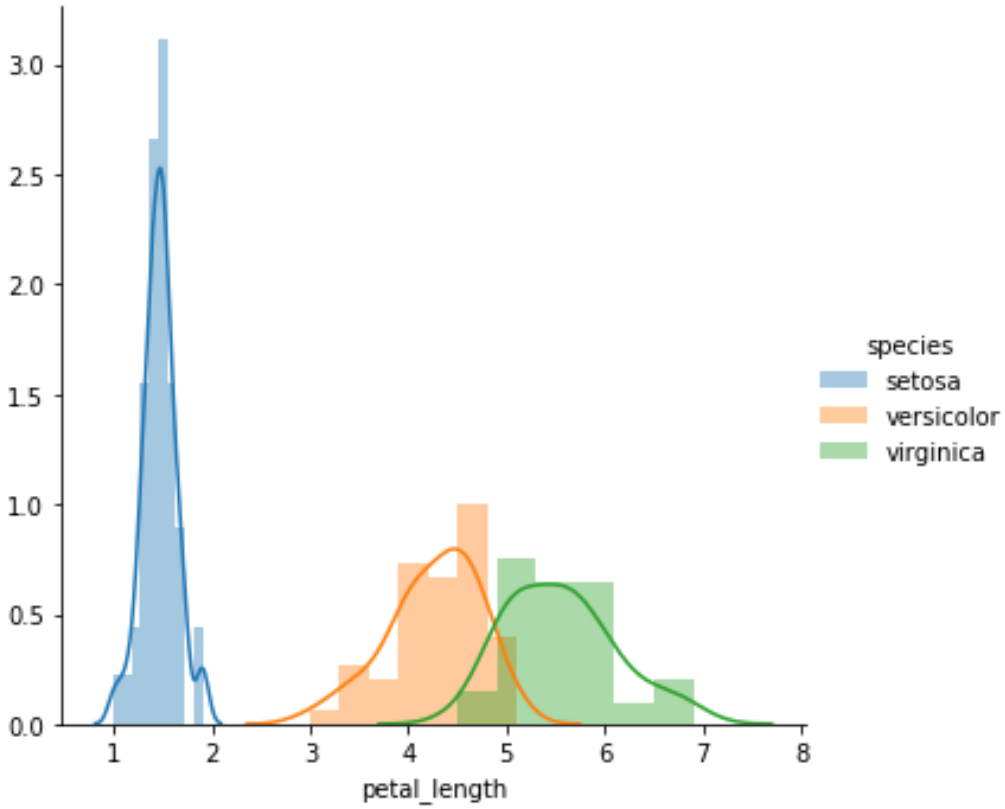
Data Insights:

- Plot 1 shows that there is a significant amount of overlap between the species on sepal length, so it is not an effective Classification feature
- Plot 2 shows that there is even higher overlap between the species on sepal width, so it is not an effective Classification feature
- Plot 3 shows that petal length is a good Classification feature as it clearly separates the species . The overlap is extremely less (between Versicolor and Virginica) , Setosa is well separated from the rest two

- Just like Plot 3, Plot 4 also shows that petal width is a good Classification feature . The overlap is significantly less (between Versicolor and Virginica) , Setosa is well separated from the rest two

Choosing Plot 3 (Classification feature as Petal Length)to distinguish among the species

Choosing Plot 3 (Classification feature as Petal Length)to distinguish among the species



Plot 3 | Classification feature : Petal Length

Data Insights:

- The pdf curve of Iris Setosa ends roughly at 2.1
- If petal length < 2.1, then species is Iris Setosa
- The point of intersection between pdf curves of Versicolor and Virginica is roughly at 4.8
- If petal length > 2.1 and petal length < 4.8 then species is Iris Versicolor
- If petal length > 4.8 then species is Iris Virginica

3.8 DATA VISUALIZATION USING TABLEAU

- Tableau is a Data Visualisation tool that is widely used for Business Intelligence but is not limited to it. It helps create interactive graphs and charts in the form of dashboards and worksheets to gain business insights. And all of this is made possible with gestures as simple as drag and drop

- Tableau is a powerful and fastest growing data visualization tool used in the Business Intelligence Industry. It helps in simplifying raw data in a very easily understandable format. Tableau helps create the data that can be understood by professionals at any level in an organization. It also allows non-technical users to create customized dashboards.
- Data analysis is very fast with Tableau tool and the visualizations created are in the form of dashboards and worksheets.
- The best features of Tableau software are
    - Data Blending
    - Real time analysis
    - Collaboration of data
- **What Products does Tableau offer?**

	Key Features	Other Features	Operating System	License
Tableau Desktop	Creating Dashboards and Stories locally	Tableau Personal - limited data sources, non connectivity to Tableau Server Tableau Professional - Full enterprise capabilities	Windows, Mac	Personal - $999 Professional - $1999
Tableau Public	A Massive, public, non commercial Tableau Server	All data published in public	-	Free
Tableau Online	Creating Dashboards and Stories on the Cloud	Live Connections	-	$500 per year per
Tableau Reader	View Dashboards and Sheets locally	Cannot modify workbooks or connect to the server	Windows, Mac	Free
Tableau Server	Connect to Data sources and share Dashboards	Users can directly interact with Dashboards via browser	Windows	Core Licensing

Why Tableau?

- Tableau is greatly used because data can be analyzed very quickly with it. Also, visualizations are generated as dashboards and worksheets. Tableau allows one to create dashboards that provide actionable insights and drive the business forward. Tableau products always operate in virtualized environments when they are configured with the proper underlying operating system and hardware. Tableau is used by data scientists to explore data with limitless visual analytics.

Features of Tableau

- Tableau Dashboard
- Collaboration and Sharing
- Live and In-memory Data
- Data Sources in Tableau
- Advanced Visualizations

- Mobile View
- Revision History
- Licensing Views
- Subscribe others
- ETL Refresh and many more make Tableau one of the most famous Data Visualization tools.

3.8.1. Tableau Product Suite

- The Tableau Product Suite consists of
    - Tableau Desktop
    - Tableau Public
    - Tableau Online
    - Tableau Server
    - Tableau Reader

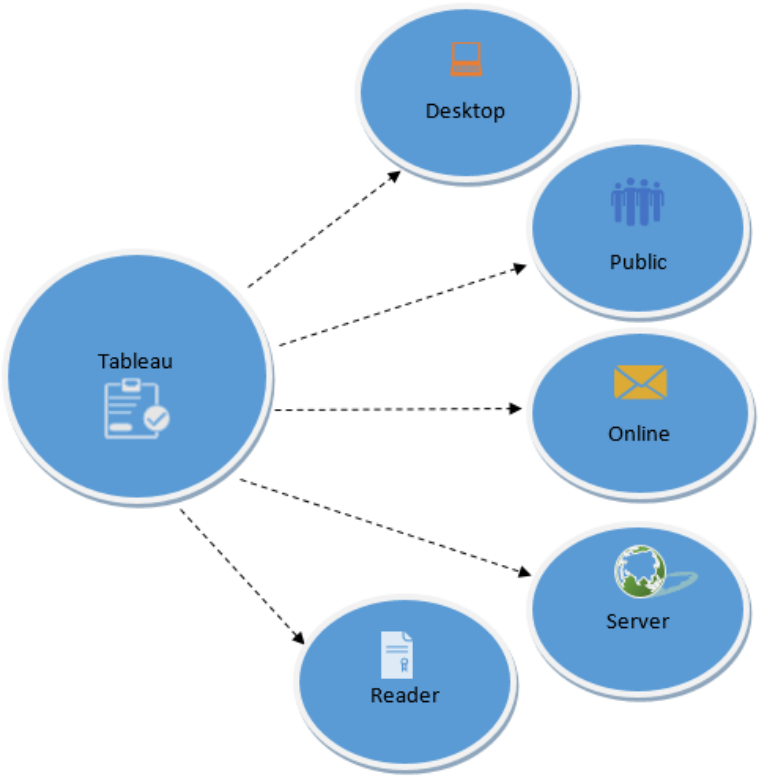


Figure 3.2: Tableau Product Suite

For a clear understanding, data analytics in Tableau tool can be classified into two section.

- **Developer Tools:** The Tableau tools that are used for development such as the creation of dashboards, charts, report generation, visualization fall into this category. The Tableau products, under this category, are the Tableau Desktop and the Tableau Public.

- **Sharing Tools:** As the name suggests, the purpose of these Tableau products is sharing the visualizations, reports, dashboards that were created using the developer tools. Products that fall into this category are Tableau Online, Server, and Reader.

Tableau Desktop

- Tableau Desktop has a rich feature set and allows you to code and customize reports. Right from creating the charts, reports, to blending them all together to form a dashboard, all the necessary work is created in Tableau Desktop.
- For live data analysis, Tableau Desktop provides connectivity to Data Warehouse, as well as other various types of files. The workbooks and the dashboards created here can be either shared locally or publicly.
- Based on the connectivity to the data sources and publishing option, Tableau Desktop is classified into
    - **Tableau Desktop Personal:** The development features are similar to Tableau Desktop. Personal version keeps the workbook private, and the access is limited. The workbooks cannot be published online. Therefore, it should be distributed either Offline or in Tableau Public.
    - **Tableau Desktop Professional:** It is pretty much similar to Tableau Desktop. The difference is that the work created in the Tableau Desktop can be published online or in Tableau Server. Also, in Professional version, there is full access to all sorts of the datatype. It is best suitable for those who wish to publish their work in Tableau Server.

Tableau Public

- It is Tableau version specially build for the cost-effective users. By the word "Public," it means that the workbooks created cannot be saved locally; in turn, it should be saved to the Tableau's public cloud which can be viewed and accessed by anyone.
- There is no privacy to the files saved to the cloud since anyone can download and access the same. This version is the best for the individuals who want to learn Tableau and for the ones who want to share their data with the general public.

Tableau Server

- The software is specifically used to share the workbooks, visualizations that are created in the Tableau Desktop application across the organization. To share dashboards in the Tableau Server, you must first publish your work in the Tableau Desktop. Once the work has been uploaded to the server, it will be accessible only to the licensed users.
- However, It's not necessary that the licensed users need to have the Tableau Server installed on their machine. They just require the login credentials with which they can check reports via a web browser. The security is high in Tableau server, and it is much suited for quick and effective sharing of data in an organization.
- The admin of the organization will always have full control over the server. The hardware and the software are maintained by the organization.

Tableau Online

- As the name suggests, it is an online sharing tool of Tableau. Its functionalities are similar to Tableau Server, but the data is stored on servers hosted in the cloud which are maintained by the Tableau group.
- There is no storage limit on the data that can be published in the Tableau Online. Tableau Online creates a direct link to over 40 data sources that are hosted in the cloud such as the MySQL, Hive, Amazon Aurora, Spark SQL and many more.
- To publish, both Tableau Online and Server require the workbooks created by Tableau Desktop. Data that is streamed from the web applications say Google Analytics, Salesforce.com are also supported by Tableau Server and Tableau Online.

Tableau Reader

- Tableau Reader is a free tool which allows you to view the workbooks and visualizations created using Tableau Desktop or Tableau Public. The data can be filtered but editing and modifications are restricted. The security level is zero in Tableau Reader as anyone who gets the workbook can view it using Tableau Reader.
- If you want to share the dashboards that you have created, the receiver should have Tableau Reader to view the document.

3.8.2. How does Tableau work?

- Tableau connects and extracts the data stored in various places. It can pull data from any platform imaginable. A simple database such as an excel, pdf, to a complex database like Oracle, a database in the cloud such as Amazon webs services, Microsoft Azure SQL database, Google Cloud SQL and various other data sources can be extracted by Tableau.
- When Tableau is launched, ready data connectors are available which allows you to connect to any database. Depending on the version of Tableau that you have purchased the number of data connectors supported by Tableau will vary.
- The pulled data can be either connected live or extracted to the Tableau's data engine, Tableau Desktop. This is where the Data analyst, data engineer work with the data that was pulled up and develop visualizations. The created dashboards are shared with the users as a static file. The users who receive the dashboards views the file using Tableau Reader.
- The data from the Tableau Desktop can be published to the Tableau server. This is an enterprise platform where collaboration, distribution, governance, security model, automation features are supported. With the Tableau server, the end users have a better experience in accessing the files from all locations be it a desktop, mobile or email.

3.8.3. Tableau Uses- Following are the main uses and applications of Tableau:

- Business Intelligence
- Data Visualization
- Data Collaboration

- Data Blending
- Real-time data analysis
- Query translation into visualization
- To import large size of data
- To create no-code data queries
- To manage large size metadata

3.8.4. Excel Vs. Tableau

- Both Excel and Tableau are data analysis tools, but each tool has its unique approach to data exploration. However, the analysis in Tableau is more potent than excel.
- Excel works with rows and columns in spreadsheets whereas Tableau enables in exploring excel data using its drag and drop feature. Tableau formats the data in Graphs, pictures that are easily understandable.

Parameters	Excel	Tableau
Purpose	Spreadsheet application used for manipulating the data.	Perfect visualization tool used for analysis.
Usage	Most suitable for statistical analysis of structured data.	Most suitable for quick and easy representation of big data which helps in resolving the big data issues.
Performance	Moderate speed with no option to quicken.	Moderate speed with options to optimize and enhance the progress of an operation.
Security	The inbuilt security feature is weak when compared to Tableau. The security update needs to be installed on a regular basis.	Extensive options to secure data without scripting. Security features like row level security and permission are inbuilt.
User Interface	To utilize excel to full potential, macro and visual basic scripting knowledge is required.	The tool can be used without any coding knowledge.

Business need	Best for preparing on-off reports with small data	Best while working with big data.
Products	Bundled with MS Office tools	Comes with different versions such as the Tableau server, cloud, and desktop.
Integration	Excel integrates with around 60 applications	Tableaus integrated with over 250 applications
Real time data exploration	When you are working in excel, you need have an idea of where your data takes you to get to know the insights	In Tableaus, you are free to explore data without even knowing the answer that you want. With the in-built features like data blending and drill-down, you will be able to determine the variations and data patterns.
Easy Visualizations	When working in excel, we first manipulate the data that is present and then the visualization such as the different charts, graphs are created manually. To make the visualizations easily understandable, you should understand the features of excel well.	Whereas in Tableau, the data is visualized from the beginning.

3.8.5. Creating Visuals in Tableau

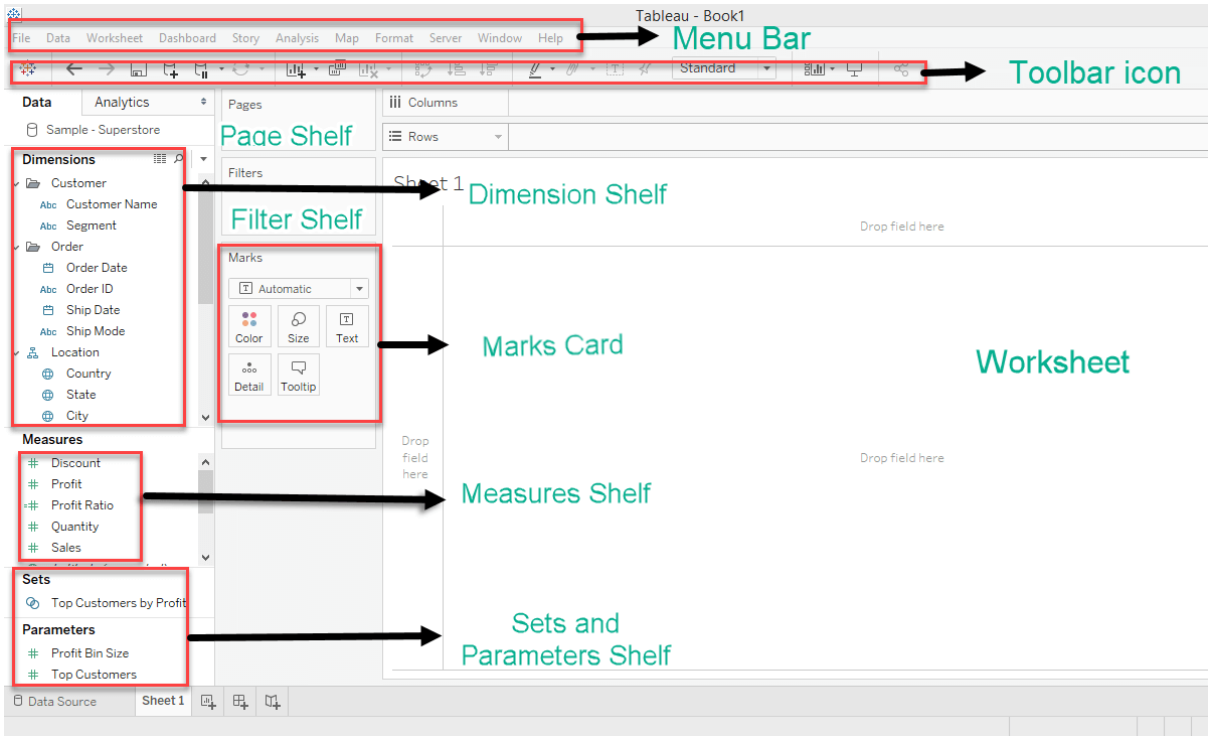
Tableau supports the following data types:

- **Boolean:** True and false can be stored in this data type.
- **Date/Datetime:**
  This data type can help in leveraging Tableau's default date hierarchy behavior when applied to valid date or DateTime fields.
- **Number:** These are values that are numeric. Values can be integers or floating-point numbers (numbers with decimals).
- **String:** This is a sequence of characters encased in single or double quotation marks.

- **Geolocation:** These are values that we need to plot maps.

3.8.6. Understanding different Sections in Tableau

- Tableau work-page consist of different section.



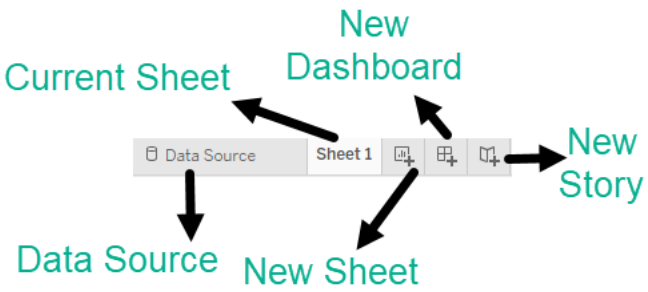


Figure 3.3: Tableau Work page

Source: Local

- **Menu Bar:** Here you'll find various commands such as File, Data, and Format.
- **Toolbar Icon**: The toolbar contains a number of buttons that enable you to perform various tasks with a click, such as Save, Undo, and New Worksheet.
- **Dimension Shelf:** This shelf contains all the categorical columns under it. example: categories, segments, gender, name, etc
- **Measure Shelf:** This shelf contains all numerical columns under it like profit, total sales, discount, etc
- **Page Shelf:** This shelf is used for joining pages and create animations. we will come on it later

- **Filter Shelf:** You can choose which data to include and exclude using the Filters shelf, for example, you might want to analyze the profit for each customer segment, but only for certain shipping containers and delivery times. You may make a view like this by putting fields on the Filters tier.
- **Marks Card:** The visualization can be designed using the Marks card. The markings card can be used to change the data components of the visualization, such as color, size, shape, path, label, and tooltip.
- **Worksheet:** In the workbook, the worksheet is where the real visualization may be seen. The worksheet contains information about the visual's design and functionality.
- **Data Source:** Using Data Source we can add new data, modify, remove data.
- **Current Sheet:** The current sheets are those sheets which we have created and to those, we can give some names.
- **New Sheet:** If we want to create a new worksheet (blank canvas) we can do using this tab.
- **New Dashboard:** This button is used to create a dashboard canvas.
- **New Storyboard:** It is used to create a new story

TEXT / REFERENCE BOOKS

1. Avrim Blum, John Hopcroft and Ravindran Kannan. Foundations of Data Science.
2. Trevor Hastie, Robert Tibshirani and Jerome Friedman. Elements of Statistical Learning, 2nd Edition. ISBN 0387952845. 2009. (free online)

QUESTION BANK

Part-A			
Q.No	**Questions**	**Competence**	**BT Level**
1.	Define EDA	Remember	BTL 1
2.	Why is exploratory data analysis important in data science?	Analysis	BTL 4
3.	Define Descriptive Statistics	Remember	BTL 1
4.	List the various methods in EDA	Understand	BTL 2
5.	List the types of EDA?	Understand	BTL 2
6.	Enumerate the characteristics of population distribution	Understand	BTL 2
7.	State the philosophy of EDA?	Understand	BTL 2
8.	List the steps involved in Data Science Process?	Remember	BTL 1
9.	List the steps involved in data preprocessing	Understand	BTL 2
10.	Define feature scaling	Understand	BTL 2
11.	List some popular plotting libraries of python in data visualization?	Understand	BTL 2
12.	Define Data Visualization	Remember	BTL 2
13.	List the data visualization tools	Remember	BTL 1
14.	Difference between tableau desktop, tableau server and tableau public	Analysis	BTL 4
15.	How does tableau work?	Understand	BTL 2
16.	Enumerate the challenges in data visualization?	Analysis	BTL 4

17.	How to clean the data?	Understand	BTL 2
18.	Differentiate Excel and Tableau	Analysis	BTL 4
19.	Why Data Visualization is important**?**	Analysis	BTL 4
20.	Enumerate the most typical causes of Data Inconsistencies and Errors	Analysis	BTL 4

PART B

Q.No	**Questions**	**Competence**	**BT Level**
1.	Explain the steps involved in data science process?	Analysis	BTL 4
2.	Explain various types of EDA?	Analysis	BTL 4
3.	Explain various univariate and multivariate graphs?	Analysis	BTL 4
4.	Explain about graphical exploratory data analysis?	Analysis	BTL 4
5.	Explain about the different stages of preprocessing?	Analysis	BTL 4
6.	Discuss in detail about preprocessing and data cleaning stages?	Analysis	BTL 4
7.	Explain the following 1. Univariate Non-graphical 2. Multivariate Non-graphical 3. Univariate graphical 4. Multivariate graphical	Analysis	BTL 4
8.	Discuss about the different data visualization tools	Analysis	BTL 4
9.	Explain about the tableau product suit?	Analysis	BTL 4
10.	How to analyse the data insights for the isis dataset?	Create	BTL 5

SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

SCHOOL OF COMPUTING
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

UNIT IV
Machine Learning Tools, Techniques And Applications- SCSA3016

UNIT 4
MACHINE LEARNING TOOLS, TECHNIQUES AND APPLICATIONS

Supervised Learning, Unsupervised Learning, Reinforcement Learning, Dimensionality Reduction, Principal Component Analysis, Classification and Regression models, Tree and Bayesian network models, Neural Networks, Testing, Evaluation and Validation of Models.

4.1. MACHINE LEARNING

Machine learning is **a method of data analysis that automates analytical model building**. It is a branch of artificial intelligence based on the idea that systems can learn from data, identify patterns and make decisions with minimal human intervention.
Example: Image recognition, Speech recognition, Medical diagnosis, Statistical arbitrage, Predictive analytics, etc.

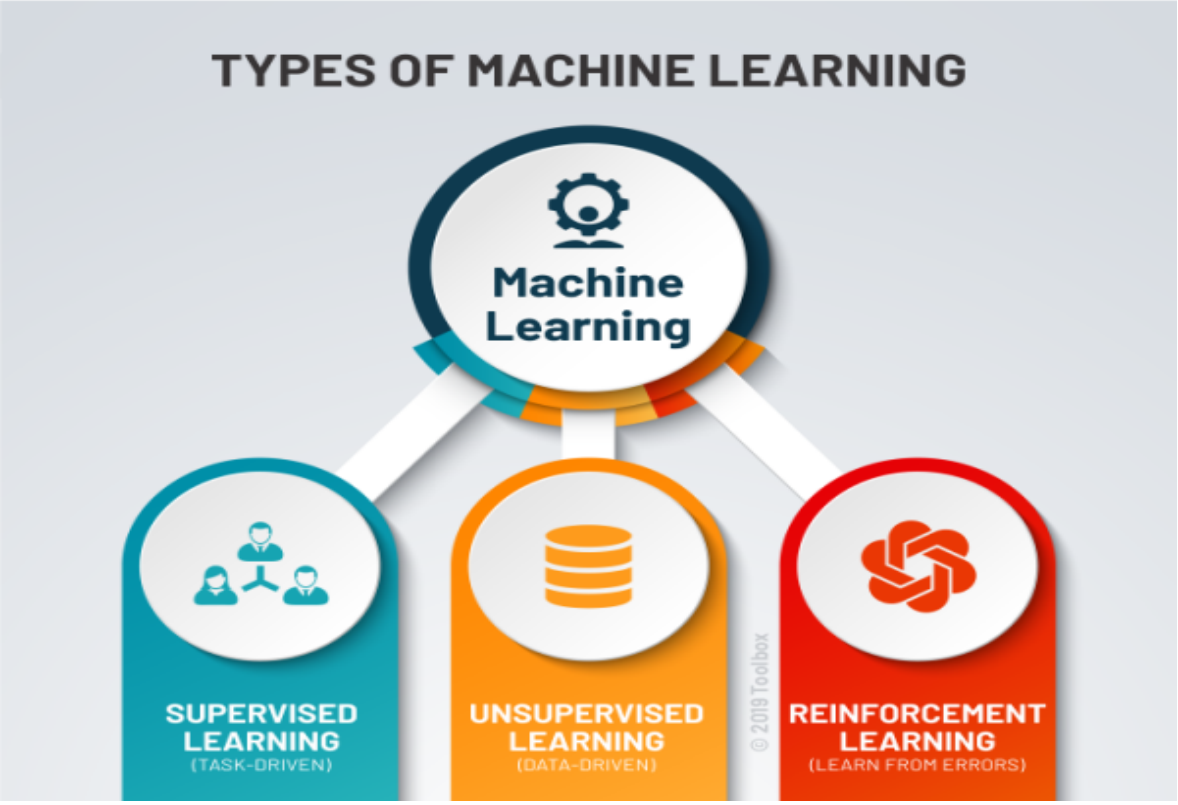
Artificial Intelligence, Machine Learning and Deep Learning

- AI is defined as a program that exhibits cognitive ability similar to that of a human being.
- Making computers think like humans and solve problems the way we do is one of the main tenets of artificial intelligence.
- Any computer program that shows characteristics, such as self-improvement, learning through inference, or even basic human tasks, such as image recognition and language processing, is considered to be a form of AI.
- The field of artificial intelligence includes within it the sub-fields of **machine learning and deep learning**.
- Deep Learning is a more specialized version of machine learning that utilizes more complex methods for difficult problems.
- However, the difference between machine learning and artificial intelligence is that machine learning is probabilistic (output can be explained, thereby ruling out the black box nature of AI), deep learning is deterministic.

4.2. TYPES OF MACHINE LEARNING

These are three types of machine learning:

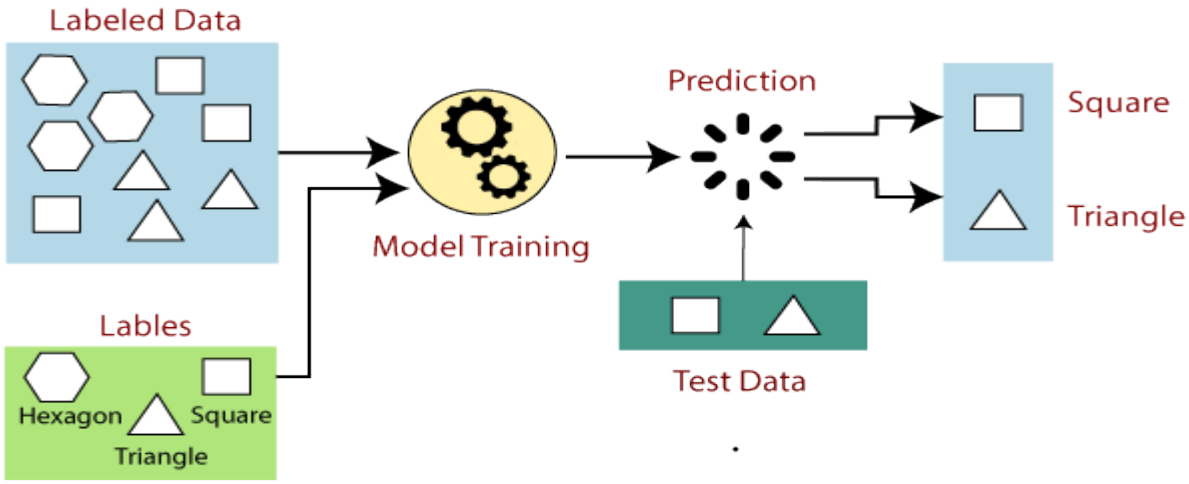
1. Supervised Learning
2. Unsupervised Learning
3. Reinforcement Learning



1. **Supervised Learning:**
   - Supervised learning is one of the most basic types of machine learning.
   - In this type, the machine learning algorithm is trained on **labelled data**.
   - In supervised learning, the ML algorithm is given a small training dataset to work with.
   - This training dataset is a smaller part of the bigger dataset and serves to give the algorithm a basic idea of the problem, solution, and data points to be dealt with.
   - The algorithm then finds relationships between the parameters given, essentially establishing a cause and effect relationship between the variables in the dataset.
   - At the end of the training, the algorithm has an idea of how the data works and the relationship between the input and the output.
   - This solution is then deployed for use with the final dataset, which it learns from in the same way as the training dataset.
   - **Example : Risk Assessment, Image classification, Fraud Detection, spam filtering**, etc.

How Supervised Learning Works?

In supervised learning, models are trained using labelled dataset, where the model learns about each type of data. Once the training process is completed, the model is tested on the basis of test data (a subset of the training set), and then it predicts the output.



Suppose we have a dataset of different types of shapes which includes square, rectangle, triangle, and Polygon. Now the first step is that we need to train the model for each shape.

- If the given shape has four sides, and all the sides are equal, then it will be labelled as a **Square**.
- If the given shape has three sides, then it will be labelled as a **triangle**.
- If the given shape has six equal sides then it will be labelled as **hexagon**.

Now, after training, we test our model using the test set, and the task of the model is to identify the shape.

The machine is already trained on all types of shapes, and when it finds a new shape, it classifies the shape on the bases of a number of sides, and predicts the output.

Advantages of Supervised learning:

- With the help of supervised learning, the model can predict the output on the basis of prior experiences.
- In supervised learning, we can have an exact idea about the classes of objects.

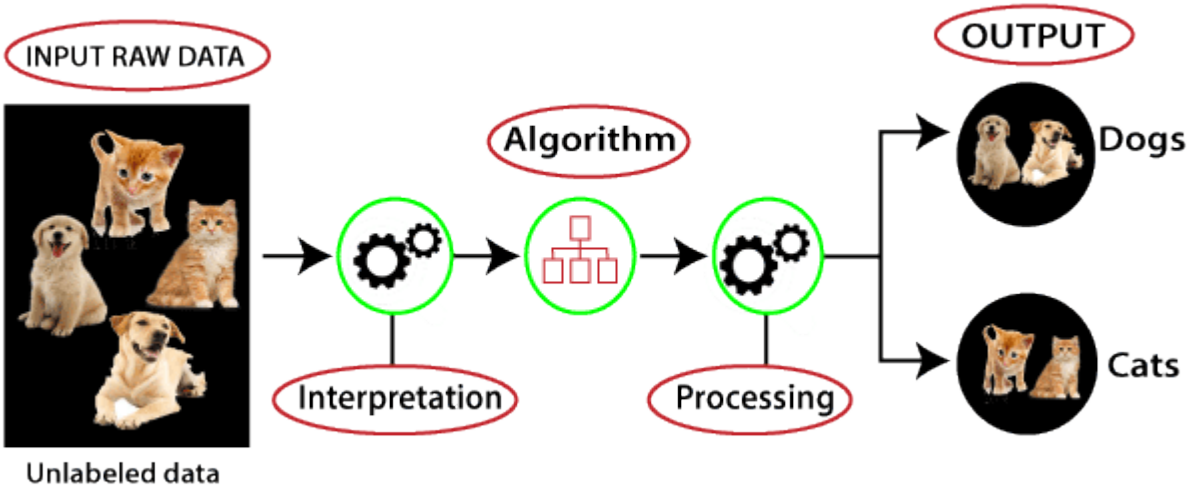
Disadvantages of supervised learning:

- Supervised learning models are not suitable for handling the complex tasks.
- Supervised learning cannot predict the correct output if the test data is different from the training dataset.
- Training required lots of computation times.

2. Unsupervised Learning:

- Unsupervised machine learning holds the advantage of being able to work with **unlabelled data**.
- This means that human labour is not required to make the dataset machine-readable, allowing much larger datasets to be worked on by the program.
- In supervised learning, the labels allow the algorithm to find the exact nature of the relationship between any two data points. However, unsupervised learning does not have labels to work off of, resulting in the creation of hidden structures.
- Relationships between data points are perceived by the algorithm in an abstract manner, with no input required from human beings.
- The creation of these hidden structures is what makes unsupervised learning algorithms versatile.
- Instead of a defined and set problem statement, unsupervised learning algorithms can adapt to the data by dynamically changing hidden structures.
- This offers more post-deployment development than supervised learning algorithms.
- **Example :** Principal Component Analysis, Clustering

How Unsupervised Learning Works?



Here, we have taken an unlabelled input data, which means it is not categorized and corresponding outputs are also not given. Now, this unlabelled input data is fed to the machine learning model in order to train it. Firstly, it will interpret the raw data to find the hidden patterns from the data and then will apply suitable algorithms such as k-means clustering, Decision tree, etc.

Once it applies the suitable algorithm, the algorithm divides the data objects into groups according to the similarities and difference between the objects.

Advantages of Unsupervised Learning

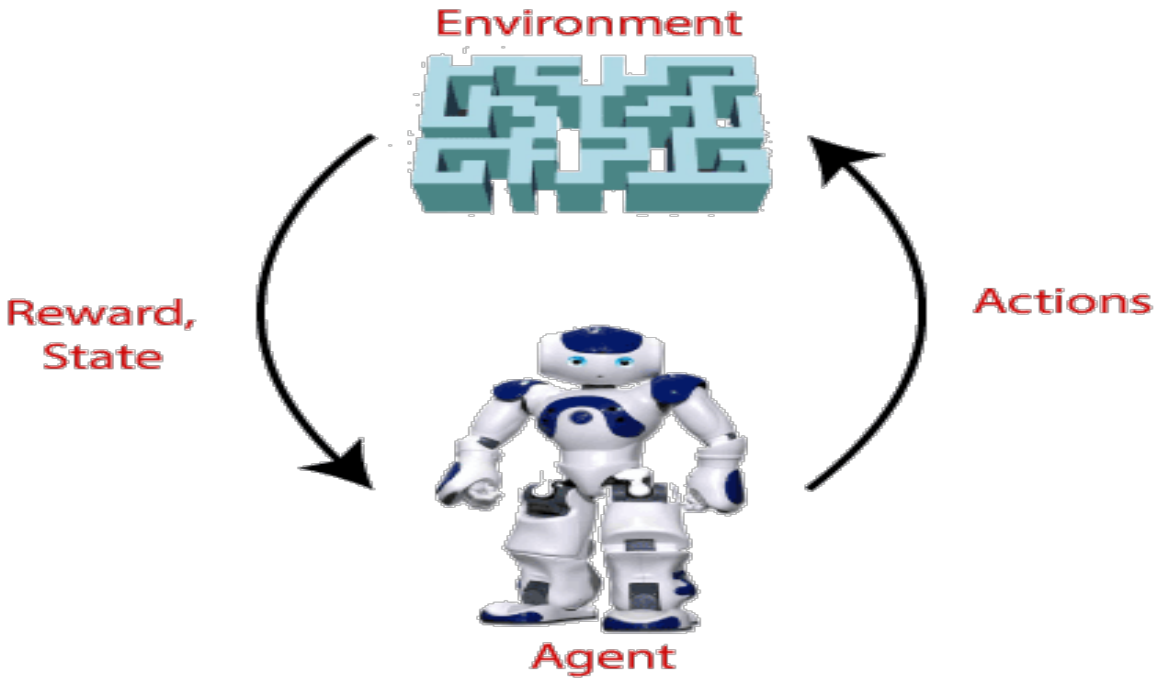
- Unsupervised learning is used for more complex tasks as compared to supervised learning because, in unsupervised learning, we don't have labelled input data.
- Unsupervised learning is preferable as it is easy to get unlabelled data in comparison to labelled data.

Disadvantages of Unsupervised Learning

- Unsupervised learning is intrinsically more difficult than supervised learning as it does not have corresponding output.
- The result of the unsupervised learning algorithm might be less accurate as input data is not labelled, and algorithms do not know the exact output in advance.

3. Reinforcement Learning

- Reinforcement Learning directly takes inspiration from how human beings learn from data in their lives. It features an algorithm that improves upon itself and learns from new situations using a trial-and-error method. Favourable outputs are encouraged or 'reinforced', and non-favourable outputs are discouraged or 'punished'.
- In every iteration of the algorithm, the output result is given to the interpreter, which decides whether the outcome is favourable or not.
- In case of the program finding the correct solution, the interpreter reinforces the solution by providing a reward to the algorithm. If the outcome is not favourable, the algorithm is forced to reiterate until it finds a better result. In most cases, the reward system is directly tied to the effectiveness of the result.
- In typical reinforcement learning use-cases, such as finding the shortest route between two points on a map, the solution is not an absolute value. Instead, it takes on a score of effectiveness, expressed in a percentage value. The higher this percentage value is, the more reward is given to the algorithm.
- Thus, the program is trained to give the best possible solution for the best possible reward.



Terms used in Reinforcement Learning

- **Agent():** An entity that can perceive/explore the environment and act upon it.
- **Environment():** A situation in which an agent is present or surrounded by. In RL, we assume the stochastic environment, which means it is random in nature.
- **Action():** Actions are the moves taken by an agent within the environment.
- **State():** State is a situation returned by the environment after each action taken by the agent.
- **Reward():** A feedback returned to the agent from the environment to evaluate the action of the agent.
- **Policy():** Policy is a strategy applied by the agent for the next action based on the current state.
- **Value():** It is expected long-term retuned with the discount factor and opposite to the short-term reward.
- **Q-value():** It is mostly similar to the value, but it takes one additional parameter as a current action (a).

Types of Reinforcement learning

There are mainly two types of reinforcement learning, which are:

- **Positive Reinforcement**
- **Negative Reinforcement**

i. **Positive Reinforcement:**

The positive reinforcement learning means adding something to increase the tendency that expected behaviour would occur again. It impacts positively on the behaviour of the agent and increases the strength of the behaviour.

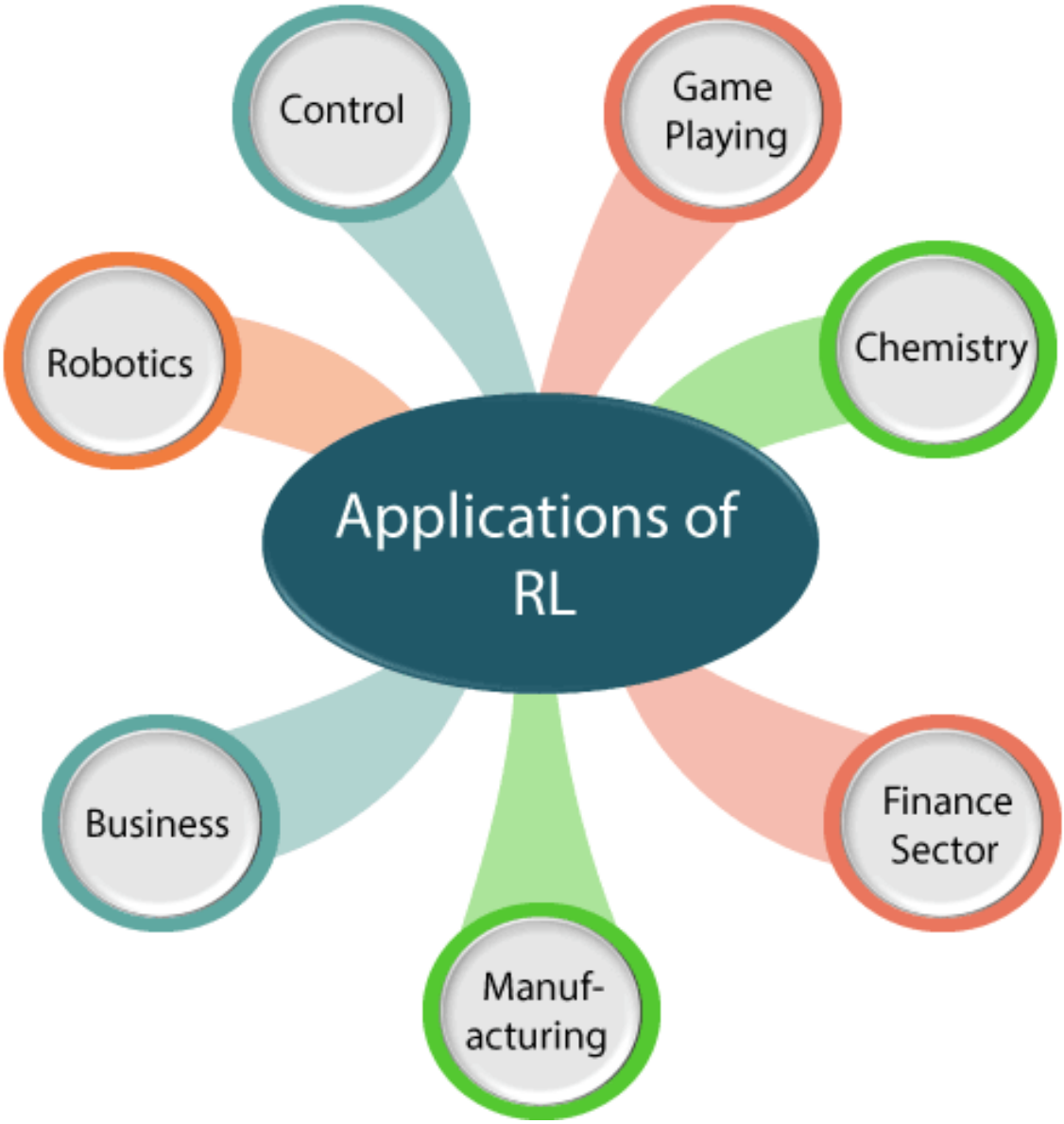
This type of reinforcement can sustain the changes for a long time, but too much positive reinforcement may lead to an overload of states that can reduce the consequences.

ii. **Negative Reinforcement:**

The negative reinforcement learning is opposite to the positive reinforcement as it increases the tendency that the specific behaviour will occur again by avoiding the negative condition.

It can be more effective than the positive reinforcement depending on situation and behaviour, but it provides reinforcement only to meet minimum behaviour.

Reinforcement Learning Applications

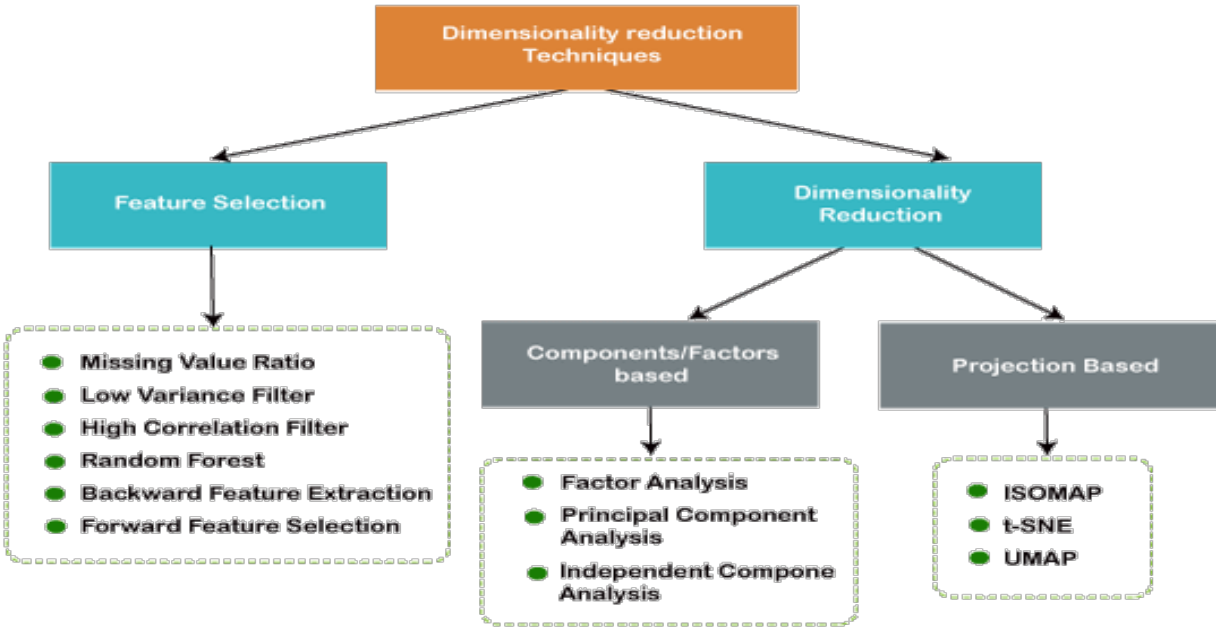


Difference Between Supervised, Unsupervised and Reinforcement Learning

Criteria	Supervised ML	Unsupervised ML	Reinforcement ML
Definition	Learns by using labelled data	Trained using unlabelled data without any guidance.	Works on interacting with the environment
Type of data	Labelled data	Unlabelled data	No – predefined data
Type of problems	Regression and classification	Association and Clustering	Exploitation or Exploration
Supervision	Extra supervision	No supervision	No supervision
Algorithms	Linear Regression, Logistic Regression, SVM, KNN etc.	K – Means, C – Means, Apriori	Q – Learning, SARSA
Aim	Calculate outcomes	Discover underlying patterns	Learn a series of action
Application	Risk Evaluation, Forecast Sales	Recommendation System, Anomaly Detection	Self Driving Cars, Gaming, Healthcare

4.3. DIMENSIONALITY REDUCTION

- The number of input features, variables, or columns present in a given dataset is known as dimensionality, and the process to reduce these features is called dimensionality reduction.
- A dataset contains a huge number of input features in various cases, which makes the predictive modelling task more complicated.
- Because it is very difficult to visualize or make predictions for the training dataset with a high number of features, for such cases, dimensionality reduction techniques are required to use.
- Dimensionality reduction technique can be defined as, ***"It is a way of converting the higher dimensions dataset into lesser dimensions dataset ensuring that it provides similar information."***
- These techniques are widely used in machine learning for obtaining a better fit predictive model while solving the classification and regression problems.
- It is commonly used in the fields that deal with high-dimensional data, such as **speech recognition, signal processing, bioinformatics, etc.**



The Curse of Dimensionality

- Handling the high-dimensional data is very difficult in practice, commonly known as the *curse of dimensionality.*
- If the dimensionality of the input dataset increases, any machine learning algorithm and model becomes more complex.
- As the number of features increases, the number of samples also gets increased proportionally, and the chance of overfitting also increases.
- If the machine learning model is trained on high-dimensional data, it becomes overfitted and results in poor performance.
- Hence, it is often required to reduce the number of features, which can be done with dimensionality reduction.

Benefits of Dimensionality Reduction

- By reducing the dimensions of the features, the space required to store the dataset also gets reduced.
- Less Computation training time is required for reduced dimensions of features.
- Reduced dimensions of features of the dataset help in visualizing the data quickly.
- **I**t removes the redundant features (if present) by taking care of multicollinearity.

Disadvantages of dimensionality Reduction

- Some data may be lost due to dimensionality reduction.

- In the PCA dimensionality reduction technique, sometimes the principal components required to consider are unknown.

4.3.1. Approaches of Dimension Reduction

There are two ways to apply the dimension reduction technique, which are given below:

- Feature Selection
- Feature Extraction

i. Feature Selection:

Feature selection is the process of selecting the subset of the relevant features and leaving out the irrelevant features present in a dataset to build a model of high accuracy. In other words, it is a way of selecting the optimal features from the input dataset.

ii. Feature Extraction:

Feature extraction is the process of transforming the space containing many dimensions into space with fewer dimensions. This approach is useful when we want to keep the whole information but use fewer resources while processing the information.

Common techniques of Dimensionality Reduction

a. **Principal Component Analysis**
b. **Backward Elimination**
c. **Forward Selection**
d. **Score comparison**
e. **Missing Value Ratio**
f. **Low Variance Filter**
g. **High Correlation Filter**
h. **Random Forest**
i. **Factor Analysis**
j. **Auto-Encoder**

4.4. Principal Component Analysis

- Principal Component Analysis is an unsupervised learning algorithm that is used for the dimensionality reduction in machine learning.
- It is a statistical process that converts the observations of correlated features into a set of linearly uncorrelated features with the help of orthogonal transformation.

- These new transformed features are called the **Principal Components**.
- It is one of the popular tools that is used for exploratory data analysis and predictive modelling.
- It is a technique to draw strong patterns from the given dataset by reducing the variances.
- PCA generally tries to find the lower-dimensional surface to project the high-dimensional data.
- PCA works by considering the variance of each attribute because the high attribute shows the good split between the classes, and hence it reduces the dimensionality.
- Example : ***image processing, movie recommendation system, etc.***

Principal Components in PCA

- The transformed new features or the output of PCA are the Principal Components.
- The number of these PCs are either equal to or less than the original features present in the dataset.

Some properties of these principal components are given below:

- The principal component must be the linear combination of the original features.
- These components are orthogonal, i.e., the correlation between a pair of variables is zero.
- The importance of each component decreases when going to 1 to n, it means the 1 PC has the most importance, and n PC will have the least importance.

Algorithm of PCA

The PCA algorithm is based on some mathematical concepts such as:

- Variance and Covariance
- Eigenvalues and Eigen factors

Some common terms used in PCA algorithm:

- **Dimensionality:** It is the number of features or variables present in the given dataset. More easily, it is the number of columns present in the dataset.
- **Correlation:** It signifies that how strongly two variables are related to each other. Such as if one changes, the other variable also gets changed. The correlation value ranges from -1 to +1.

Here, -1 occurs if variables are inversely proportional to each other, and +1 indicates that variables are directly proportional to each other.

- **Orthogonal:** It defines that variables are not correlated to each other, and hence the correlation between the pair of variables is zero.
- **Eigenvectors:** If there is a square matrix M, and a non-zero vector v is given. Then v will be eigenvector if Av is the scalar multiple of v.
- **Covariance Matrix:** A matrix containing the covariance between the pair of variables is called the Covariance Matrix.

Steps for PCA algorithm

1. **Getting the dataset**

   Firstly, we need to take the input dataset and divide it into two subparts X and Y, where X is the training set, and Y is the validation set.

2. **Representing data into a structure**

   Now we will represent our dataset into a structure. Such as we will represent the two-dimensional matrix of independent variable X. Here each row corresponds to the data items, and the column corresponds to the Features. The number of columns is the dimensions of the dataset.

3. **Standardizing the data**

   In this step, we will standardize our dataset. Such as in a particular column, the features with high variance are more important compared to the features with lower variance. If the importance of features is independent of the variance of the feature, then we will divide each data item in a column with the standard deviation of the column. Here we will name the matrix as Z.

4. **Calculating the Covariance of Z**

   To calculate the covariance of Z, we will take the matrix Z, and will transpose it. After transpose, we will multiply it by Z. The output matrix will be the Covariance matrix of Z.

5. **Calculating the Eigen Values and Eigen Vectors**

   Now we need to calculate the eigenvalues and eigenvectors for the resultant covariance matrix Z. Eigenvectors or the covariance matrix are the directions of the axes with high information. And the coefficients of these eigenvectors are defined as the eigenvalues.

6. **Sorting the Eigen Vectors**

   In this step, we will take all the eigenvalues and will sort them in decreasing order, which

means from largest to smallest. And simultaneously sort the eigenvectors accordingly in matrix P of eigenvalues. The resultant matrix will be named as P*.

7. **Calculating the new features Or Principal Components**
   Here we will calculate the new features. To do this, we will multiply the P* matrix to the Z. In the resultant matrix Z*, each observation is the linear combination of original features. Each column of the Z* matrix is independent of each other.
8. **Remove less or unimportant features from the new dataset.**
   The new feature set has occurred, so we will decide here what to keep and what to remove. It means, we will only keep the relevant or important features in the new dataset, and unimportant features will be removed out.

Recipe for Dimension Reduction with PCA

Data $\boldsymbol{D}=\{\boldsymbol{x}_1,\boldsymbol{x}_2,\ldots,\boldsymbol{x}_n\}$. Each $\boldsymbol{x}_i$ is a $\boldsymbol{d}$-dimensional vector. Wish to use PCA to reduce dimension to $\boldsymbol{k}$

1. Find the sample mean $\hat{\boldsymbol{\mu}} = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}_i$
2. Subtract sample mean from the data $\boldsymbol{z}_i = \boldsymbol{x}_i - \hat{\boldsymbol{\mu}}$
3. Compute the scatter matrix $\boldsymbol{S} = \sum_{i=1}^{n} \boldsymbol{z}_i \boldsymbol{z}_i^t$
4. Compute eigenvectors $\boldsymbol{e}_1,\boldsymbol{e}_2,\ldots,\boldsymbol{e}_k$ corresponding to the $\boldsymbol{k}$ largest eigenvalues of $\boldsymbol{S}$
5. Let $\boldsymbol{e}_1,\boldsymbol{e}_2,\ldots,\boldsymbol{e}_k$ be the columns of matrix $\boldsymbol{E} = [\boldsymbol{e}_1 \cdots \boldsymbol{e}_k]$
6. The desired $\boldsymbol{y}$ which is the closest approximation to $\boldsymbol{x}$ is $\boldsymbol{y} = \boldsymbol{E}^t \boldsymbol{z}$

PCA Algorithm :

The steps involved in PCA Algorithm are as follows-

- **Step-01:** Get data.
- **Step-02:** Compute the mean vector (μ).
- **Step-03:** Subtract mean from the given data.
- **Step-04:** Calculate the covariance matrix.
- **Step-05:** Calculate the eigen vectors and eigen values of the covariance matrix.
- **Step-06:** Choosing components and forming a feature vector.
- **Step-07:** Deriving the new data set.

PRACTICE PROBLEMS BASED ON PRINCIPAL COMPONENT ANALYSIS

Problem-01:

Given data = { 2, 3, 4, 5, 6, 7 }; {1, 5, 3, 6, 7, 8 }.

Compute the principal component using PCA Algorithm.

OR

Consider the two dimensional patterns (2, 1), (3, 5), (4, 3), (5, 6), (6, 7), (7, 8).

Compute the principal component using PCA Algorithm.

OR

Compute the principal component of following data-

CLASS 1

X = 2 , 3 , 4

Y = 1 , 5 , 3

CLASS 2

X = 5 , 6 , 7

Y = 6 , 7 , 8

Solution:

Step-01:

Get data.

The given feature vectors are-

- $x_1 = (2, 1)$
- $x_2 = (3, 5)$
- $x_3 = (4, 3)$

- $x_4 = (5, 6)$
- $x_5 = (6, 7)$
- $x_6 = (7, 8)$

Step-02:

Calculate the mean vector (μ).

Mean vector (μ)

= ((2 + 3 + 4 + 5 + 6 + 7) / 6, (1 + 5 + 3 + 6 + 7 + 8) / 6)

= (4.5, 5)

Thus,

$$\text{Mean vector } (\mu) = \begin{bmatrix} 4.5 \\ 5 \end{bmatrix}$$

Step-03:

Subtract mean vector (μ) from the given feature vectors.

- $x_1 - \mu$, y1 - μ = (2 – 4.5, 1 – 5) = (-2.5, -4)
- $x_2 - \mu$, y2 - μ = (3 – 4.5, 5 – 5) = (-1.5, 0)
- $x_3 - \mu$, y3 - μ = (4 – 4.5, 3 – 5) = (-0.5, -2)
- $x_4 - \mu$, y4 - μ = (5 – 4.5, 6 – 5) = (0.5, 1)
- $x_5 - \mu$, y5 - μ = (6 – 4.5, 7 – 5) = (1.5, 2)
- $x_6 - \mu$, y6 - μ = (7 – 4.5, 8 – 5) = (2.5, 3)

Feature vectors (x_i) after subtracting mean vector (μ) are-

$$\begin{bmatrix} -2.5 \\ -4 \end{bmatrix} \begin{bmatrix} -1.5 \\ 0 \end{bmatrix} \begin{bmatrix} -0.5 \\ -2 \end{bmatrix} \begin{bmatrix} 0.5 \\ 1 \end{bmatrix} \begin{bmatrix} 1.5 \\ 2 \end{bmatrix} \begin{bmatrix} 2.5 \\ 3 \end{bmatrix}$$

Step-04:

Calculate the covariance matrix.

$$\text{Covariance Matrix} = \frac{\sum (x_i - \mu)(x_i - \mu)^t}{n}$$

Hence,

$$m_1 = (x_1 - \mu)(x_1 - \mu)^t = \begin{bmatrix} -2.5 \\ -4 \end{bmatrix} \begin{bmatrix} -2.5 & -4 \end{bmatrix} = \begin{bmatrix} 6.25 & 10 \\ 10 & 16 \end{bmatrix}$$

$$m_2 = (x_2 - \mu)(x_2 - \mu)^t = \begin{bmatrix} -1.5 \\ 0 \end{bmatrix} \begin{bmatrix} -1.5 & 0 \end{bmatrix} = \begin{bmatrix} 2.25 & 0 \\ 0 & 0 \end{bmatrix}$$

$$m_3 = (x_3 - \mu)(x_3 - \mu)^t = \begin{bmatrix} -0.5 \\ -2 \end{bmatrix} \begin{bmatrix} -0.5 & -2 \end{bmatrix} = \begin{bmatrix} 0.25 & 1 \\ 1 & 4 \end{bmatrix}$$

$$m_4 = (x_4 - \mu)(x_4 - \mu)^t = \begin{bmatrix} 0.5 \\ 1 \end{bmatrix} \begin{bmatrix} 0.5 & 1 \end{bmatrix} = \begin{bmatrix} 0.25 & 0.5 \\ 0.5 & 1 \end{bmatrix}$$

$$m_5 = (x_5 - \mu)(x_5 - \mu)^t = \begin{bmatrix} 1.5 \\ 2 \end{bmatrix} \begin{bmatrix} 1.5 & 2 \end{bmatrix} = \begin{bmatrix} 2.25 & 3 \\ 3 & 4 \end{bmatrix}$$

$$m_6 = (x_6 - \mu)(x_6 - \mu)^t = \begin{bmatrix} 2.5 \\ 3 \end{bmatrix} \begin{bmatrix} 2.5 & 3 \end{bmatrix} = \begin{bmatrix} 6.25 & 7.5 \\ 7.5 & 9 \end{bmatrix}$$

Now,

Covariance matrix = $(m_1 + m_2 + m_3 + m_4 + m_5 + m_6) / 6$

On adding the above matrices and dividing by 6, we get-

$$\text{Covariance Matrix} = \frac{1}{6} \begin{bmatrix} 17.5 & 22 \\ 22 & 34 \end{bmatrix}$$

$$\text{Covariance Matrix} = \begin{bmatrix} 2.92 & 3.67 \\ 3.67 & 5.67 \end{bmatrix}$$

Step-05:

Calculate the eigen values and eigen vectors of the covariance matrix.

λ is an eigen value for a matrix M if it is a solution of the characteristic equation

|M – λI| = 0.

So, we have-

$$\begin{vmatrix} 2.92 & 3.67 \\ 3.67 & 5.67 \end{vmatrix} - \begin{vmatrix} \lambda & 0 \\ 0 & \lambda \end{vmatrix} = 0$$

$$\begin{vmatrix} 2.92 - \lambda & 3.67 \\ 3.67 & 5.67 - \lambda \end{vmatrix} = 0$$

From here,

$(2.92 – \lambda)(5.67 – \lambda) – (3.67 \times 3.67) = 0$

$16.56 – 2.92\lambda – 5.67\lambda + \lambda^2 – 13.47 = 0$

$\lambda^2 – 8.59\lambda + 3.09 = 0$

Solving this quadratic equation, we get $\lambda = 8.22, 0.38$

Thus, two eigen values are $\lambda_1 = 8.22$ and $\lambda_2 = 0.38$.

Clearly, the second eigen value is very small compared to the first eigen value.

So, the second eigen vector can be left out.

Eigen vector corresponding to the greatest eigen value is the principal component for the given data set.

So. we find the eigen vector corresponding to eigen value λ_1.

We use the following equation to find the eigen vector-

MX = λX

where-

- M = Covariance Matrix

- X = Eigen vector
- λ = Eigen value

Substituting the values in the above equation, we get-

$$\begin{bmatrix} 2.92 & 3.67 \\ 3.67 & 5.67 \end{bmatrix} \begin{bmatrix} X1 \\ X2 \end{bmatrix} = 8.22 \begin{bmatrix} X1 \\ X2 \end{bmatrix}$$

Solving these, we get-

$2.92X_1 + 3.67X_2 = 8.22X_1$

$3.67X_1 + 5.67X_2 = 8.22X_2$

On simplification, we get-

$5.3X_1 = 3.67X_2$ ……(1)

$3.67X_1 = 2.55X_2$ ……(2)

From (1) and (2), $\mathbf{X_1 = 0.69X_2}$

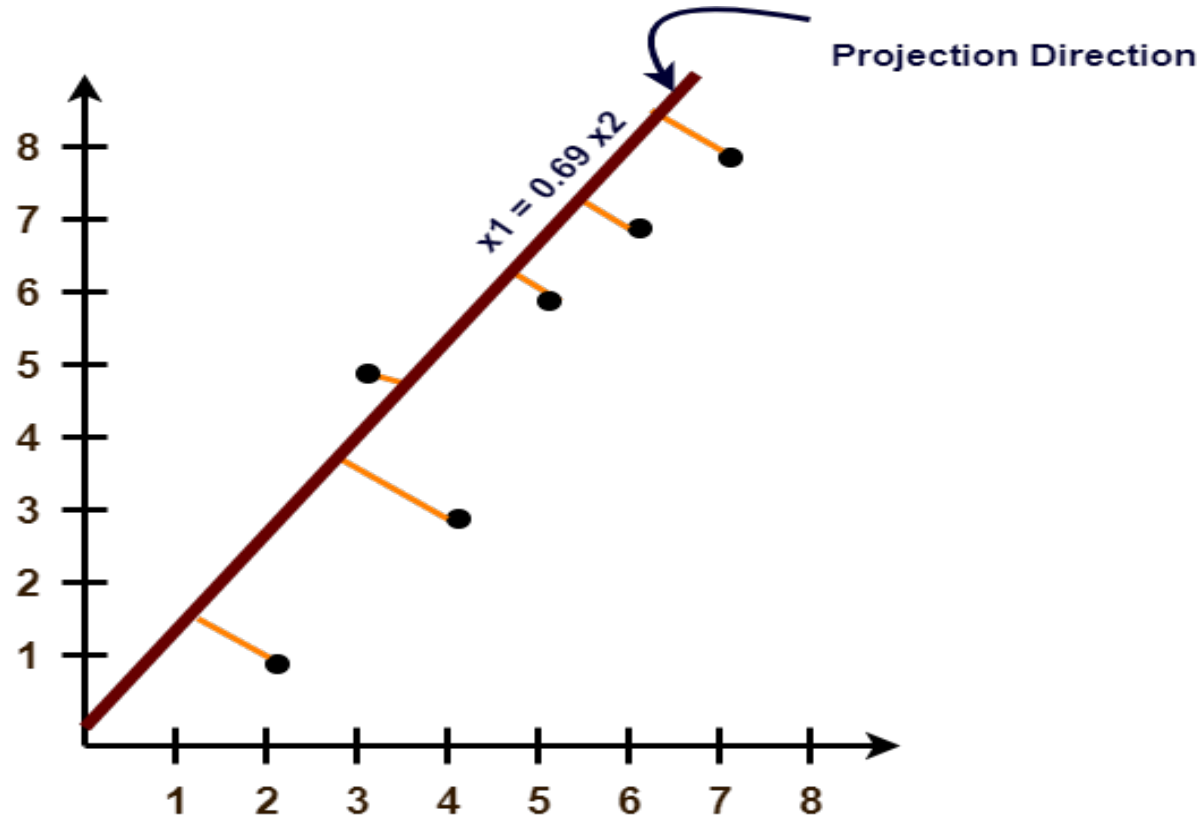
From (2), the eigen vector is-

$$\text{Eigen Vector :} \quad \begin{bmatrix} X1 \\ X2 \end{bmatrix} = \begin{bmatrix} 2.55 \\ 3.67 \end{bmatrix}$$

Thus, principal component for the given data set is-

$$\text{Principal Component :} \quad \begin{bmatrix} X1 \\ X2 \end{bmatrix} = \begin{bmatrix} 2.55 \\ 3.67 \end{bmatrix}$$

Lastly, we project the data points onto the new subspace as-



PCA Example using Python

```
#Importing required libraries
import numpy as np
```

1. Subtract the mean of each variable

Subtract the mean of each variable from the dataset so that the dataset should be centered on the origin. Doing this proves to be very helpful when calculating the covariance matrix.

```
#Generate a dummy dataset.
X = np.random.randint(10,50,100).reshape(20,5)
# mean Centering the data
X_meaned = X - np.mean(X , axis = 0)
```

Data generated by the above code have dimensions (20,5) i.e. 20 examples and 5 variables for each example. we calculated the mean of each variable and subtracted that from every row of the respective column

2. Calculate the Covariance Matrix

Calculate the Covariance Matrix of the mean-centered data. The covariance matrix is a square matrix denoting the covariance of the elements with each other. The covariance of an element with itself is nothing but just its Variance.

So the diagonal elements of a covariance matrix are just the variance of the elements.

```
# calculating the covariance matrix of the mean-centered data.
cov_mat = np.cov(X_meaned , rowvar = False)
```

We can find easily calculate covariance Matrix using numpy.cov( ) method. The default value for rowvar is set to True, remember to set it to False to get the covariance matrix in the required dimensions.

3. Compute the Eigenvalues and Eigenvectors

Now, compute the Eigenvalues and Eigenvectors for the calculated Covariance matrix. The Eigenvectors of the Covariance matrix we get are Orthogonal to each other and each vector represents a principal axis.

A Higher Eigenvalue corresponds to a higher variability. Hence the principal axis with the higher Eigenvalue will be an axis capturing higher variability in the data.
Orthogonal means the vectors are mutually perpendicular to each other.

```
#Calculating Eigenvalues and Eigenvectors of the covariance matrix
eigen_values , eigen_vectors = np.linalg.eigh(cov_mat)
```

NumPy linalg.eigh( ) method returns the eigenvalues and eigenvectors of a complex Hermitian or a real symmetric matrix.

4. Sort Eigenvalues in descending order

Sort the Eigenvalues in the descending order along with their corresponding Eigenvector.

Remember each column in the Eigen vector-matrix corresponds to a principal component, so arranging them in descending order of their Eigenvalue will automatically arrange the principal component in descending order of their variability.

Hence the first column in our rearranged Eigen vector-matrix will be a principal component that captures the highest variability.

5. Select a subset from the rearranged Eigenvalue matrix

Select a subset from the rearranged Eigenvalue matrix as per our need i.e. number_comp = 2. This means we selected the first two principal components.

```
# select the first n eigenvectors, n is desired dimension of our final reduced data.

n_components = 2 #you can select any number of components.
eigenvector_subset = sorted_eigenvectors[:,0:n_components]
```

n_components = 2 means our final data should be reduced to just 2 variables. if we change it to 3 then we get our data reduced to 3 variables.

6. **Transform the data**

Finally, transform the data by having a dot product between the Transpose of the Eigenvector subset and the Transpose of the mean-centered data. By transposing the outcome of the dot product, the result we get is the data reduced to lower dimensions from higher dimensions.

```
#Transform the data
X_reduced = np.dot(eigenvector_subset.transpose(),X_meaned.transpose()).transpose()
```

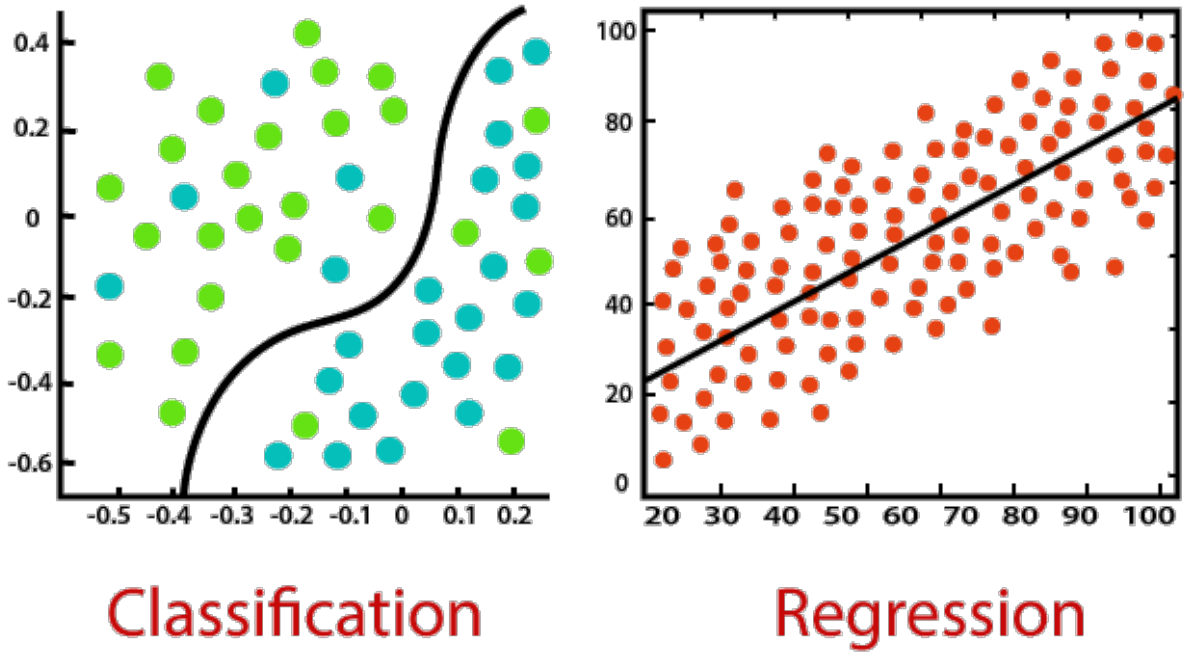
The final dimensions of X_reduced will be (20, 2) and originally the data was of higher dimensions (20, 5).

4.5. REGRESSION AND CLASSIFICATION IN MACHINE LEARNING

Regression and Classification algorithms are Supervised Learning algorithms. Both the algorithms are used for prediction in Machine learning and work with the labelled datasets.

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

Example :



4.5.1. REGRESSION

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

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

Example: Suppose we want to do weather forecasting, so for this, we will use the Regression algorithm. In weather prediction, the model is trained on the past data, and once the training is completed, it can easily predict the weather for future days.

Types of Regression Algorithm:

- Simple Linear Regression
- Multiple Linear Regression
- Polynomial Regression
- Logistic Regression

Regression Analysis in Machine learning

Regression analysis is a statistical method to model the relationship between a dependent (target) and independent (predictor) variables with one or more independent variables. More specifically, Regression analysis helps us to understand how the value of the dependent variable is changing corresponding to an independent variable when other independent variables are held fixed. It predicts continuous/real values such as **temperature, age, salary, price,** etc.

We can understand the concept of regression analysis using the below example:

Example: Suppose there is a marketing company A, who does various advertisement every year and get sales on that. The below list shows the advertisement made by the company in the last 5 years and the corresponding sales:

Advertisement	Sales
$90	$1000
$120	$1300
$150	$1800
$100	$1200
$130	$1380
$200	??

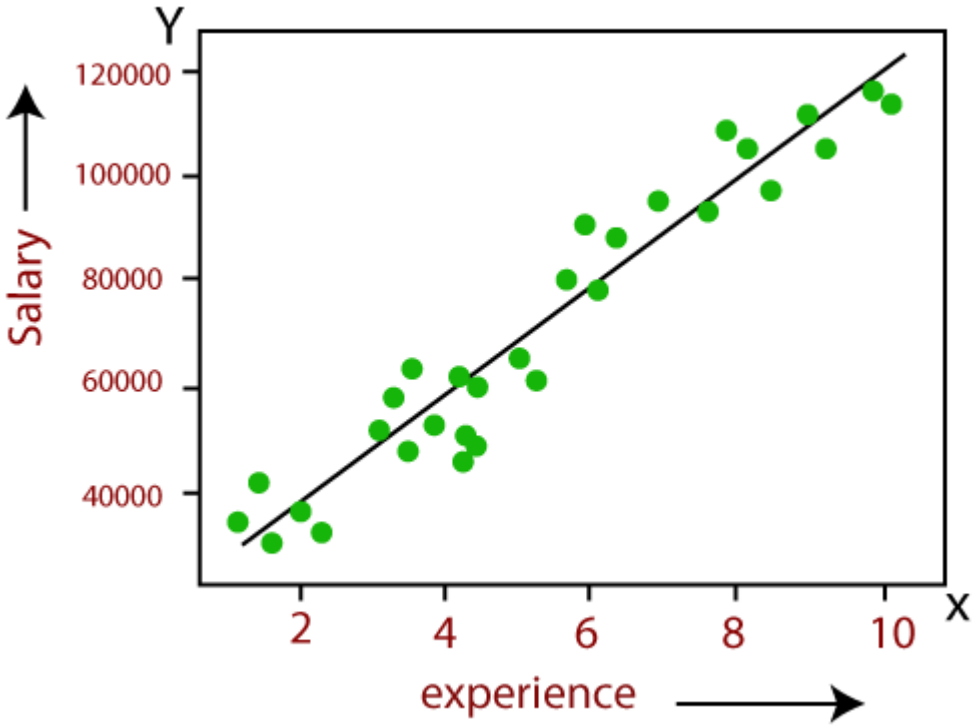
Now, the company wants to do the advertisement of $200 in the year 2019 **and wants to know the prediction about the sales for this year**. So to solve such type of prediction problems in machine learning, we need regression analysis.

Terminologies Related to the Regression Analysis:

- **Dependent Variable:** The main factor in Regression analysis which we want to predict or understand is called the dependent variable. It is also called **target variable**.
- **Independent Variable:** The factors which affect the dependent variables or which are used to predict the values of the dependent variables are called independent variable, also called as a **predictor**.
- **Outliers:** Outlier is an observation which contains either very low value or very high value in comparison to other observed values. An outlier may hamper the result, so it should be avoided.
- **Multicollinearity:** If the independent variables are highly correlated with each other than other variables, then such condition is called Multicollinearity. It should not be present in the dataset, because it creates problem while ranking the most affecting variable.
- **Underfitting and Overfitting:** If our algorithm works well with the training dataset but not well with test dataset, then such problem is called **Overfitting**. And if our algorithm does not perform well even with training dataset, then such problem is called **underfitting**.

Linear Regression:

- Linear regression is a statistical regression method which is used for predictive analysis.
- It is one of the very simple and easy algorithms which works on regression and shows the relationship between the continuous variables.
- It is used for solving the regression problem in machine learning.
- Linear regression shows the linear relationship between the independent variable (X-axis) and the dependent variable (Y-axis), hence called linear regression.
- If there is only one input variable (x), then such linear regression is called **simple linear regression**.
- And if there is more than one input variable, then such linear regression is called **multiple linear regression**.
- The relationship between variables in the linear regression model can be explained using the below image.
- Here we are predicting the salary of an employee on the basis of **the year of experience**.



Below is the mathematical equation for Linear regression:

$$Y = aX + b$$

Here,

Y = dependent variables (target variables)

X = Independent variables (predictor variables)

a and **b** are the linear coefficients

Types of Linear Regression :

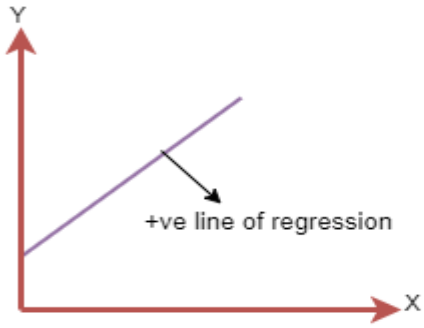
Linear regression can be further divided into two types of the algorithm:

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

Linear Regression Line

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

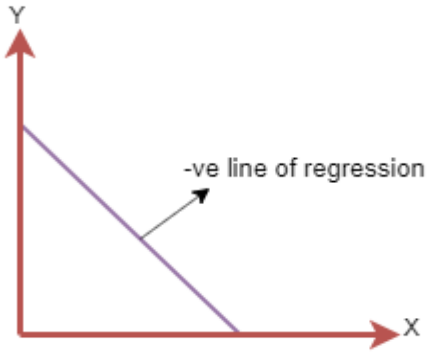
- **Positive Linear Relationship:**
  If the dependent variable increases on the Y-axis and independent variable increases on X-axis, then such a relationship is termed as a Positive linear relationship.



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

- **Negative Linear Relationship:**

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



The line of equation will be: $Y= -a_0+a_1x$

Example of Linear Regression:
Consider the following table with x,y:

SUBJECT	AGE X	GLUCOSE LEVEL Y
1	43	99
2	21	65
3	25	79
4	42	75
5	57	87
6	59	81
Σ	247	486

Step 1: Calculate the following : xy, X^2, Y^2

SUBJECT	AGE X	GLUCOSE LEVEL Y	XY	X^2	Y^2
1	43	99	4257	1849	9801
2	21	65	1365	441	4225
3	25	79	1975	625	6241
4	42	75	3150	1764	5625
5	57	87	4959	3249	7569
6	59	81	4779	3481	6561
Σ	**247**	**486**	**20485**	**11409**	**40022**

From the above table,

Σx = 247,

Σy = 486,

Σxy = 20485,

Σx2 = 11409,

Σy2 = 40022.

n is the sample size (6, in our case).

Step 2 : Use the following equations to find a and b.

$$a = \frac{(\Sigma y)(\Sigma x^2) - (\Sigma x)(\Sigma x y)}{n(\Sigma x^2) - (\Sigma x)^2}$$

$$b = \frac{n(\Sigma x y) - (\Sigma x)(\Sigma y)}{n(\Sigma x^2) - (\Sigma x)^2}$$

Find a:

= ((486 × 11,409) – ((247 × 20,485)) / 6 (11,409) – 247^2)

= 484979 / 7445

= **65.14**

Find b:

= (6(20,485) – (247 × 486)) / (6 (11409) – 247^2)

= (122,910 – 120,042) / 68,454 – 247^2

= 2,868 / 7,445

= **0.385225**

Step 3 : *Insert the values into the equation*.

y' = a + bx
y' = 65.14 + .385225x

Now, if suppose age is 34 then we can find the glucose level by substituting in the above equation:
X=34
y' = 65.14 + .385225x
y' = 65.14 + .385225 x 34
y' = 65.14 + 13.09765
y' = 78.23765
Thus the glucose level for the person with age 34 is approximately 78.

Linear regression using Python

There are five basic steps when you're implementing linear regression:

1. Import the packages and classes you need.
2. Provide data to work with and eventually do appropriate transformations.
3. Create a regression model and fit it with existing data.
4. Check the results of model fitting to know whether the model is satisfactory.
5. Apply the model for predictions.

Step 1: Import packages and classes

The first step is to import the package numpy and the class LinearRegression from sklearn.linear_model:

```
import numpy as np
from sklearn.linear_model import LinearRegression
```

Step 2: Provide data

The inputs (regressors, x) and output (predictor, y) should be arrays (the instances of the class `numpy.ndarray`) or similar objects. This is the simplest way of providing data for regression:

```
x = np.array([5, 15, 25, 35, 45, 55]).reshape((-1, 1))
y = np.array([5, 20, 14, 32, 22, 38])
```

The `.reshape()` on `x` variable is used because this array is required to be **two-dimensional**, or to be more precise, to have **one column and as many rows as necessary**. That's exactly what the argument `(-1, 1)` of `.reshape()` specifies.

Step 3: Create a model and fit it

The next step is to create a linear regression model and fit it using the existing data.

Let's create an instance of the class `LinearRegression`, which will represent the regression model:

```
model = LinearRegression()
```

This statement creates the variable `model` as the instance of `LinearRegression`. You can provide several optional parameters to `LinearRegression`:

- **`fit_intercept`** is a boolean (`True` by default) that decides whether to calculate the intercept b_0 (`True`) or consider it equal to zero (`False`).

- **`normalize`** is a Boolean (`False` by default) that decides whether to normalize the input variables (`True`) or not (`False`).
- **`copy_X`** is a Boolean (`True` by default) that decides whether to copy (`True`) or overwrite the input variables (`False`).
- **`n_jobs`** is an integer or `None` (default) and represents the number of jobs used in parallel computation. `None` usually means one job and `-1` to use all processors.

This example uses the default values of all parameters.

With `.fit()`, you calculate the optimal values of the weights b_0 and b_1, using the existing input and output (`x` and `y`) as the arguments. In other words, `.fit()` **fits the model**. It returns `self`, which is the variable `model` itself. That's why you can replace the last two statements with this one:

```
model = LinearRegression().fit(x, y)
```

Step 4: Get results

Once you have your model fitted, you can get the results to check whether the model works satisfactorily and interpret it.

You can obtain the coefficient of determination (R^2) with .score() called on model:

```
r_sq = model.score(x, y)
print('coefficient of determination:', r_sq)
coefficient of determination: 0.715875613747954
```

When you're applying .score(), the arguments are also the predictor x and regressor y, and the return value is R^2.

The attributes of model are .intercept_, which represents the coefficient, b_0 and .coef_, which represents b_1:

```
print('intercept:', model.intercept_)
intercept: 5.633333333333329
print('slope:', model.coef_)
slope: [0.54]
```

The code above illustrates how to get b_0 and b_1. You can notice that .intercept_ is a scalar, while .coef_ is an array.

The value $b_0 = 5.63$ (approximately) illustrates that your model predicts the response 5.63 when x is zero. The value $b_1 = 0.54$ means that the predicted response rises by 0.54 when x is increased by one.

Step 5: Predict response

Once there is a satisfactory model, you can use it for predictions with either existing or new data.

To obtain the predicted response, use .predict():

```
y_pred = model.predict(x)
print('predicted response:', y_pred)
predicted response:
[ 8.33333333 13.73333333 19.13333333 24.53333333 29.93333333 35.33333333]
```

When applying .predict(), you pass the regressor as the argument and get the corresponding predicted response.

4.5.2. CLASSIFICATION

Classification is a process of finding a function which helps in dividing the dataset into classes based on different parameters. In Classification, a computer program is trained on the training dataset and based on that training, it categorizes the data into different classes.

The task of the classification algorithm is to find the mapping function to map the input(x) to the discrete output(y).

Example: The best example to understand the Classification problem is Email Spam Detection. The model is trained on the basis of millions of emails on different parameters, and whenever it receives a new email, it identifies whether the email is spam or not. If the email is spam, then it is moved to the Spam folder.

Types of ML Classification Algorithms:

Classification Algorithms can be further divided into the following types:

- K-Nearest Neighbours
- Support Vector Machines
- Kernel SVM
- Naïve Bayes
- Decision Tree Classification
- Random Forest Classification

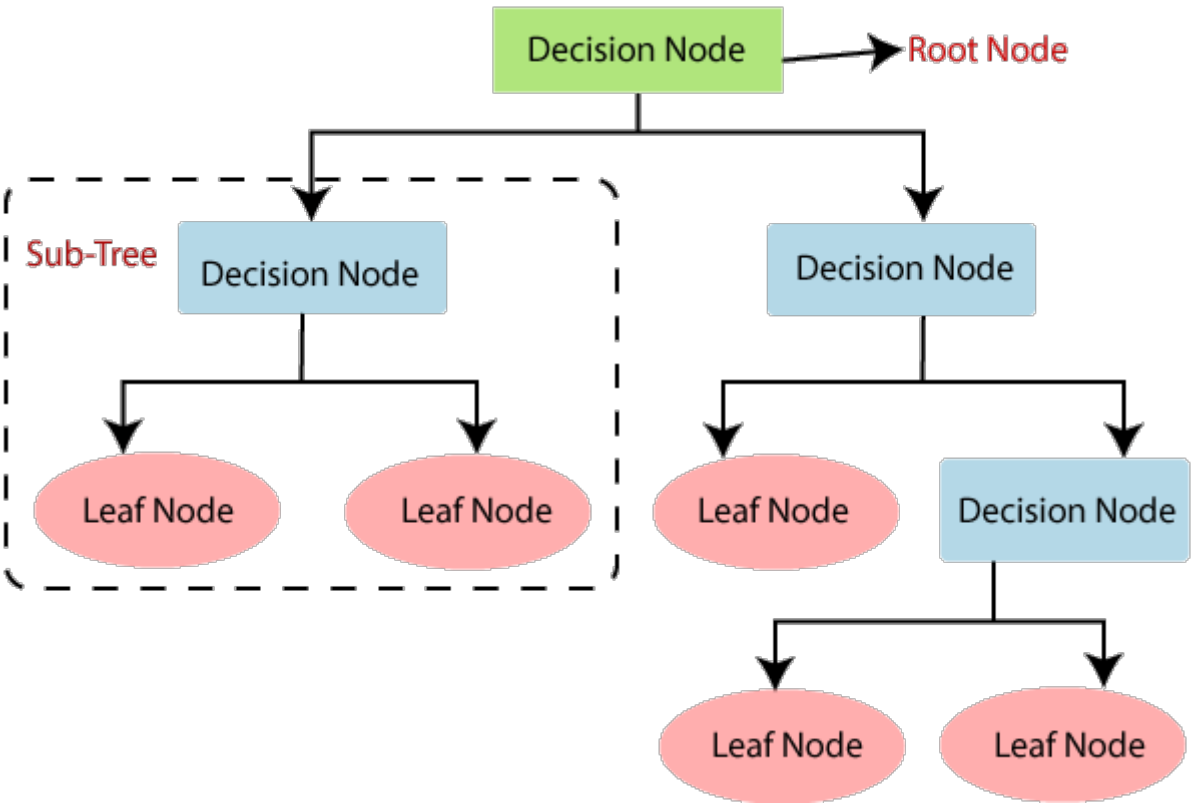
Difference between Regression and Classification :

Regression Algorithm	Classification Algorithm
In Regression, the output variable must be of continuous nature or real value.	In Classification, the output variable must be a discrete value.
The task of the regression algorithm is to map the input value (x) with the continuous output variable(y).	The task of the classification algorithm is to map the input value(x) with the discrete output variable(y).
Regression Algorithms are used with continuous data.	Classification Algorithms are used with discrete data.
In Regression, we try to find the best fit line, which can predict the output more accurately.	In Classification, we try to find the decision boundary, which can divide the dataset into different classes.
Regression algorithms can be used to solve the regression problems such as Weather Prediction, House price prediction, etc.	Classification Algorithms can be used to solve classification problems such as Identification of spam emails, Speech Recognition, Identification of cancer cells, etc.
The regression Algorithm can be further divided into Linear and Non-linear Regression.	The Classification algorithms can be divided into Binary Classifier and Multi-class Classifier.

DECISION TREE CLASSIFICATION ALGORITHM

- 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.

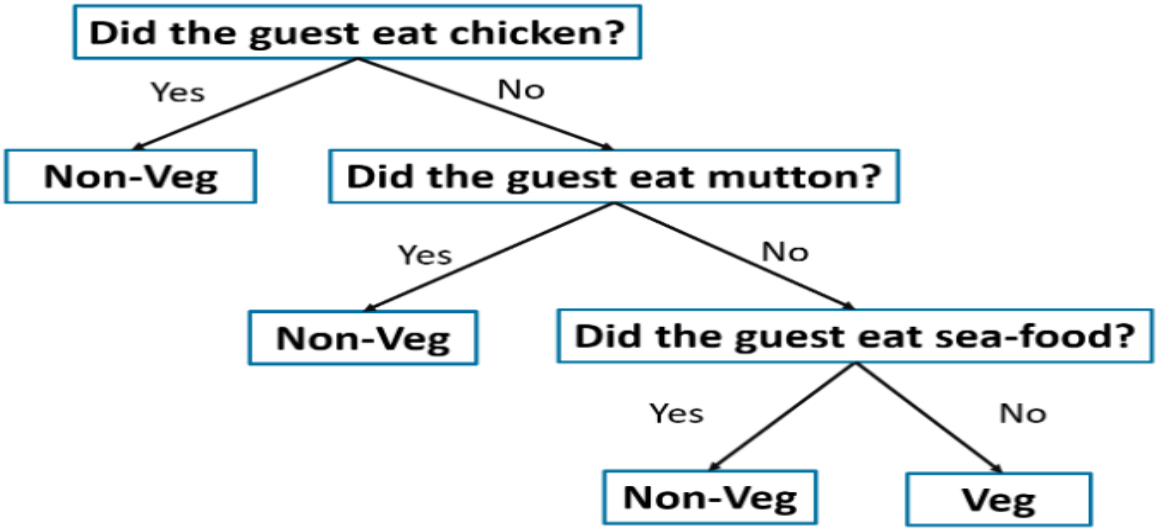


Examples:

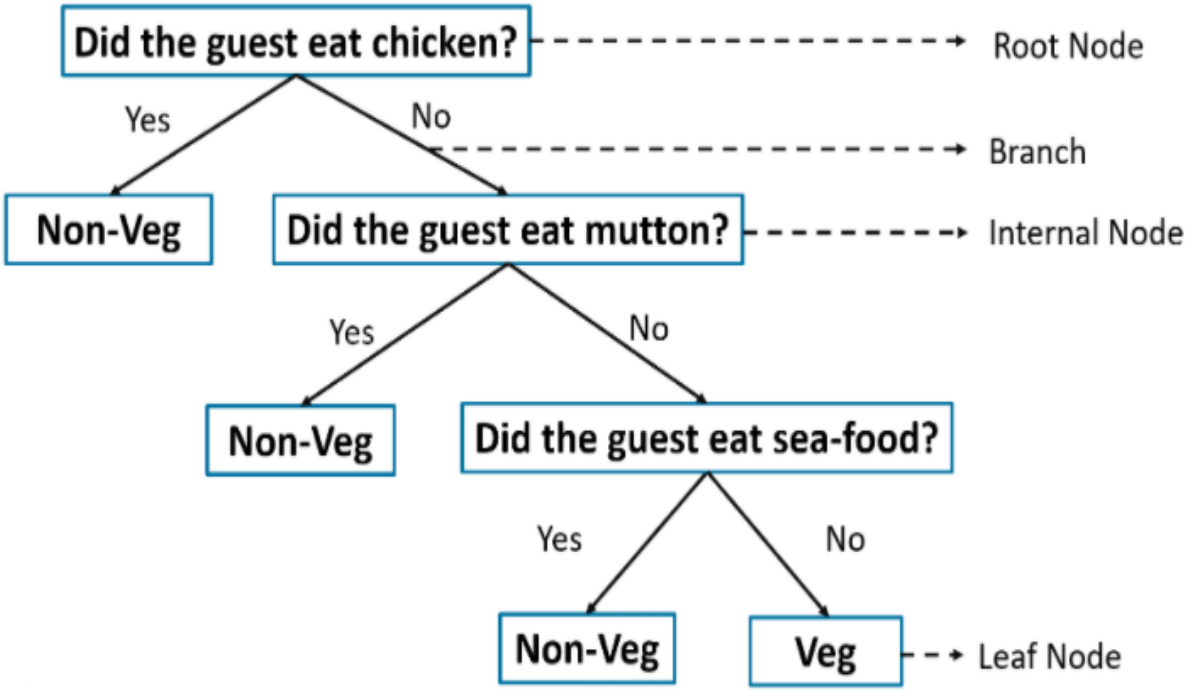
1. Predicting an email as spam or not spam
2. Predicting of a tumor is cancerous
3. Predicting a loan as a good or bad credit risk based on the factors in each of these.

Generally, a model is created with observed data also called **training data**. Then a set of **validation data** is used to *verify* and improve the model.

Suppose you hosted a huge party and you want to know how many of your guests were non-vegetarians. To solve this problem, a simple Decision Tree is used.



Structure of Decision Tree :



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:** The root node of the tree is called the parent node, and other nodes are called the child nodes.

Decision Tree Algorithm :

The Decision Tree Algorithm follows the below steps:

Step 1: Select the feature (predictor variable) that best classifies the data set into the desired classes and assign that feature to the root node.

Step 2: Traverse down from the root node, while making relevant decisions at each internal node such that each internal node best classifies the data.

Step 3: Route back to step 1 and repeat until you assign a class to the input data.

Types of Decision Tree Algorithm:

- **Decision stump**
  Used for generating a decision tree with just a single split hence also known as a one-level decision tree. It is known for its low predictive performance in most cases due to its simplicity.
- **M5**
  Known for its precise classification accuracy and its ability to work well to a boosted decision tree and small datasets with too much noise.
- **ID3(Iterative Dichotomiser 3)**
  One of the *core and widely used decision tree algorithms* uses a top-down, greedy search approach through the given dataset and selects the best attribute for classifying the given dataset
- **C4.5**
  Also known as the statistical classifier this type of decision tree is derived from its parent ID3. This generates decisions based on a bunch of predictors.
- **C5.0**
  Being the successor of the C4.5 it broadly has two models namely the basic tree and rule-based model, and its nodes can only predict categorical targets.
- **CHAID**
  Expanded as Chi-squared Automatic Interaction Detector, this algorithm basically studies the merging variables to justify the outcome on the dependant variable by structuring a predictive model
- **MARS**
  Expanded as multivariate adaptive regression splines, this algorithm creates a series of piecewise linear models which is used to model irregularities and interactions among variables, they are known for their ability to handle numerical data with greater efficiency.

- **Conditional Inference Trees**
  This is a type of decision tree that uses a conditional inference framework to recursively segregate the response variables, it's known for its flexibility and strong foundations.
- **CART**
  Expanded as Classification and Regression Trees, the values of the target variables are predicted if they are continuous else the necessary classes are identified if they are categorical.

Decision tree using id3 algorithm :

- ID3 or the Iterative Dichotomiser 3 algorithm is one of the most effective algorithms used to build a Decision Tree.
- It uses the concept of ***Entropy*** and ***Information Gain*** to generate a Decision Tree for a given set of data.

ID3 algorithm

- The ID3 algorithm follows the below workflow in order to build a Decision Tree:
  - Select **Best Attribute** (A)
  - Assign A as a decision variable for the root node.
  - For each value of A, build a descendant of the node.
  - Assign classification labels to the leaf node.
  - If data is correctly classified: Stop.
  - Else: Iterate over the tree.

1. Entropy

- Entropy measures the impurity or uncertainty present in the data. It is used to decide how a Decision Tree can split the data.
- If the sample is completely uniform then entropy is 0, if it's uniformly partitioned it is one. Higher the entropy more difficult it becomes to draw conclusions from that information.
- **Equation For Entropy:**

$$Entropy = -\sum p(x) \log p(x)$$

2. Information Gain (IG)

- Information Gain (IG) is the most significant measure used to build a Decision Tree.
- It indicates how much "information" a particular feature/ variable gives us about the final outcome.

- Information Gain is important because it used to choose the variable that best splits the data at each node of a Decision Tree.
- The variable with the highest IG is used to split the data at the root node.
- **Equation For Information Gain (IG):**

Example of Decision Tree using ID3 Algorithm:

At which days the children's will play Tennis?

Day	Outlook	Temp	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Step :1

Day	Outlook	Temp	Humidity	Wind	Play Tennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Attribute: Outlook

$Values\,(Outlook) = Sunny, Overcast, Rain$

$S = [9+, 5-]$ $\quad Entropy(S) = -\frac{9}{14}log_2\frac{9}{14} - \frac{5}{14}log_2\frac{5}{14} = 0.94$

$S_{Sunny} \leftarrow [2+, 3-]$ $\quad Entropy(S_{Sunny}) = -\frac{2}{5}log_2\frac{2}{5} - \frac{3}{5}log_2\frac{3}{5} = 0.971$

$S_{Overcast} \leftarrow [4+, 0-]$ $\quad Entropy(S_{Overcast}) = -\frac{4}{4}log_2\frac{4}{4} - \frac{0}{4}log_2\frac{0}{4} = 0$

$S_{Rain} \leftarrow [3+, 2-]$ $\quad Entropy(S_{Rain}) = -\frac{3}{5}log_2\frac{3}{5} - \frac{2}{5}log_2\frac{2}{5} = 0.971$

$$Gain\,(S, Outlook) = Entropy(S) - \sum_{v \in \{Sunny, Overcast, Rain\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$Gain(S, Outlook) = Entropy(S) - \frac{5}{14}Entropy(S_{Sunny}) - \frac{4}{14}Entropy(S_{Overcast}) - \frac{5}{14}Entropy(S_{Rain})$$

$$Gain(S, Outlook) = 0.94 - \frac{5}{14}0.971 - \frac{4}{14}0 - \frac{5}{14}0.971 = 0.2464$$

Step 2:

Day	Outlook	Temp	Humidity	Wind	Play Tennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Attribute: Temp

$Values\,(Temp) = Hot, Mild, Cool$

$S = [9+, 5-]$ $\quad Entropy(S) = -\frac{9}{14}log_2\frac{9}{14} - \frac{5}{14}log_2\frac{5}{14} = 0.94$

$S_{Hot} \leftarrow [2+, 2-]$ $\quad Entropy(S_{Hot}) = -\frac{2}{4}log_2\frac{2}{4} - \frac{2}{4}log_2\frac{2}{4} = 1.0$

$S_{Mild} \leftarrow [4+, 2-]$ $\quad Entropy(S_{Mild}) = -\frac{4}{6}log_2\frac{4}{6} - \frac{2}{6}log_2\frac{2}{6} = 0.91$

$S_{Cool} \leftarrow [3+, 1-]$ $\quad Entropy(S_{Cool}) = -\frac{3}{4}log_2\frac{3}{4} - \frac{1}{4}log_2\frac{1}{4} = 0.811$

$$Gain\,(S, Temp) = Entropy(S) - \sum_{v \in \{Hot, Mild, Cool\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$Gain(S, Temp) = Entropy(S) - \frac{4}{14}Entropy(S_{Hot}) - \frac{6}{14}Entropy(S_{Mild}) - \frac{4}{14}Entropy(S_{Cool})$$

$$Gain(S, Temp) = 0.94 - \frac{4}{14}1.0 - \frac{6}{14}0.9183 \text{ - } 4/14\ (0.8113)$$

$$= 0.0289$$

Step 3:

Day	Outlook	Temp	Humidity	Wind	Play Tennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Attribute: Humidity

$Values\ (Humidity) = High, Normal$

$S = [9+, 5-]$ $\quad Entropy(S) = -\frac{9}{14}log_2\frac{9}{14} - \frac{5}{14}log_2\frac{5}{14} = 0.94$

$S_{High} \leftarrow [3+, 4-]$ $\quad Entropy(S_{High}) = -\frac{3}{7}log_2\frac{3}{7} - \frac{4}{7}log_2\frac{4}{7} = 0.9852$

$S_{Normal} \leftarrow [6+, 1-]$ $\quad Entropy(S_{Normal}) = -\frac{6}{7}log_2\frac{6}{7} - \frac{1}{7}log_2\frac{1}{7} = 0.5916$

$$Gain\ (S, Humidity) = Entropy(S) - \sum_{v \in \{High, Normal\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$Gain(S, Humidity) = Entropy(S) - \frac{7}{14}Entropy(S_{High}) - \frac{7}{14}Entropy(S_{Normal})$$

$$Gain(S, Humidity) = 0.94 - \frac{7}{14}0.9852 - \frac{7}{14}0.5916 = 0.1516$$

Step 4:

Day	Outlook	Temp	Humidity	Wind	Play Tennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Attribute: Wind

$Values\ (Wind) = Strong, Weak$

$S = [9+, 5-]$ $\quad Entropy(S) = -\frac{9}{14}log_2\frac{9}{14} - \frac{5}{14}log_2\frac{5}{14} = 0.94$

$S_{Strong} \leftarrow [3+, 3-]$ $\quad Entropy(S_{Strong}) = 1.0$

$S_{Weak} \leftarrow [6+, 2-]$ $\quad Entropy(S_{Weak}) = -\frac{6}{8}log_2\frac{6}{8} - \frac{2}{8}log_2\frac{2}{8} = 0.8113$

$$Gain\ (S, Wind) = Entropy(S) - \sum_{v \in \{Strong, Weak\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$Gain(S, Wind) = Entropy(S) - \frac{6}{14}Entropy(S_{Strong}) - \frac{8}{14}Entropy(S_{Weak})$$

$$Gain(S, Wind) = 0.94 - \frac{6}{14}1.0 - \frac{8}{14}0.8113 = 0.0478$$

Step 5: Choose the maximum gain…

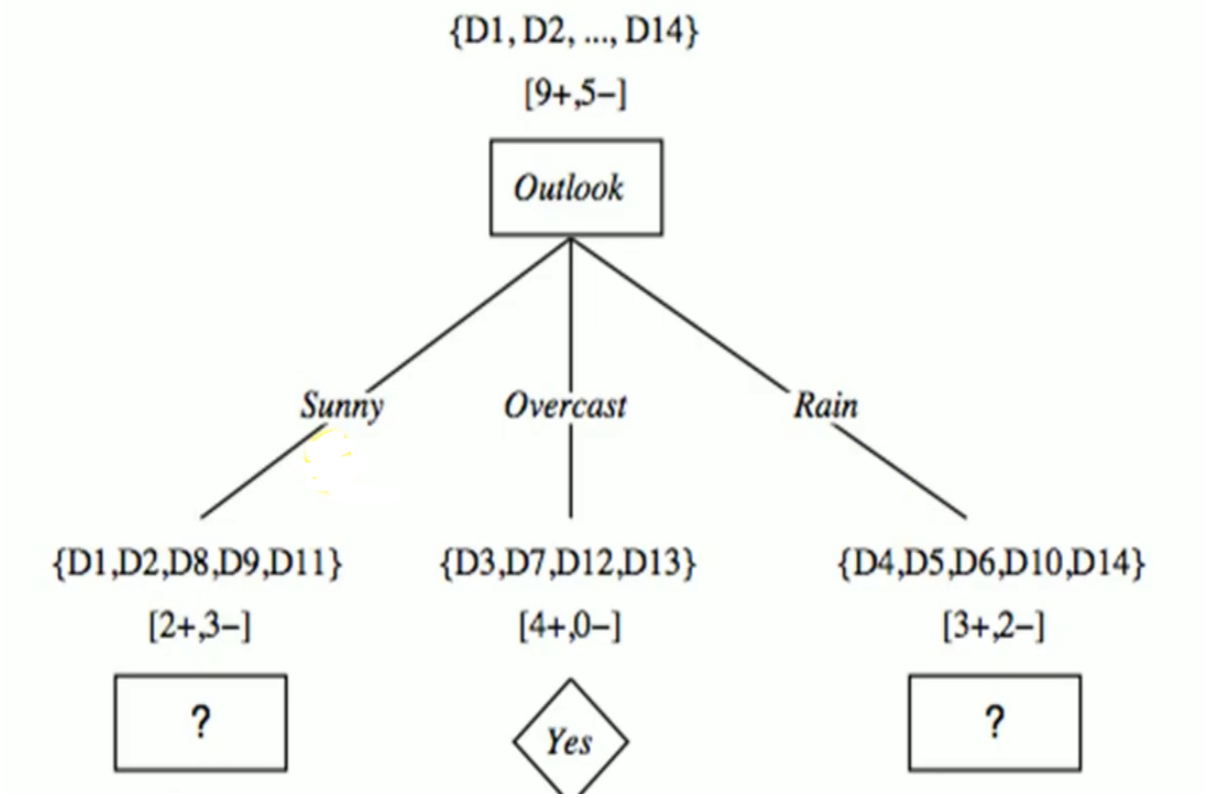
Day	Outlook	Temp	Humidity	Wind	Play Tennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

$$Gain(S, Outlook) = 0.2464$$

$$Gain(S, Temp) = 0.0289$$

$$Gain(S, Humidity) = 0.1516$$

$$Gain(S, Wind) = 0.0478$$



Step 6:

Day	Temp	Humidity	Wind	Play Tennis
D1	Hot	High	Weak	No
D2	Hot	High	Strong	No
D8	Mild	High	Weak	No
D9	Cool	Normal	Weak	Yes
D11	Mild	Normal	Strong	Yes

Attribute: Temp

$Values\ (Temp) = Hot, Mild, Cool$

$S_{Sunny} = [2+, 3-]$ $\quad Entropy(S_{Sunny}) = -\frac{2}{5}log_2\frac{2}{5} - \frac{3}{5}log_2\frac{3}{5} = 0.97$

$S_{Hot} \leftarrow [0+, 2-]$ $\quad Entropy(S_{Hot}) = 0.0$

$S_{Mild} \leftarrow [1+, 1-]$ $\quad Entropy(S_{Mild}) = 1.0$

$S_{Cool} \leftarrow [1+, 0-]$ $\quad Entropy(S_{Cool}) = 0.0$

$$Gain\left(S_{Sunny}, Temp\right) = Entropy(S) - \sum_{v \in \{Hot, Mild, Cool\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$Gain(S_{Sunny}, Temp)$$

$$= Entropy(S) - \frac{2}{5}Entropy(S_{Hot}) - \frac{2}{5}Entropy(S_{Mild})$$

$$-\frac{1}{5}Entropy(S_{Cool})$$

$$Gain(S_{sunny}, Temp) = 0.97 - \frac{2}{5}0.0 - \frac{2}{5}1 - \frac{1}{5}0.0 = 0.570$$

Step 7:

Day	Temp	Humidity	Wind	Play Tennis
Dl	Hot	High	Weak	No
D2	Hot	High	Strong	No
D8	Mild	High	Weak	No
D9	Cool	Normal	Weak	Yes
Dl1	Mild	Normal	Strong	Yes

Attribute: Humidity

$Values\ (Humidity) = High, Normal$

$S_{Sunny} = [2+, 3-]$ $\quad Entropy(S) = -\frac{2}{5}log_2\frac{2}{5} - \frac{3}{5}log_2\frac{3}{5} = 0.97$

$S_{high} \leftarrow [0+, 3-]$ $\quad Entropy(S_{High}) = 0.0$

$S_{Normal} \leftarrow [2+, 0-]$ $\quad Entropy(S_{Normal}) = 0.0$

$$Gain\left(S_{Sunny}, Humidity\right) = Entropy(S) - \sum_{v \in \{High, Normal\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$Gain(S_{Sunny}, Humidity) = Entropy(S) - \frac{3}{5}Entropy(S_{High}) - \frac{2}{5}Entropy(S_{Normal})$$

$$Gain(S_{sunny}, Humidity) = 0.97 - \frac{3}{5}0.0 - \frac{2}{5}0.0 = 0.97$$

Step 8:

Day	Temp	Humidity	Wind	Play Tennis
Dl	Hot	High	Weak	No
D2	Hot	High	Strong	No
D8	Mild	High	Weak	No
D9	Cool	Normal	Weak	Yes
Dl1	Mild	Normal	Strong	Yes

Attribute: Wind

$Values\ (Wind) = Strong, Weak$

$S_{Sunny} = [2+, 3-]$ $\quad Entropy(S) = -\frac{2}{5}log_2\frac{2}{5} - \frac{3}{5}log_2\frac{3}{5} = 0.97$

$S_{Strong} \leftarrow [1+, 1-]$ $\quad Entropy(S_{Strong}) = 1.0$

$S_{Weak} \leftarrow [1+, 2-]$ $\quad Entropy(S_{Weak}) = -\frac{1}{3}log_2\frac{1}{3} - \frac{2}{3}log_2\frac{2}{3} = 0.9183$

$$Gain\ (S_{Sunny}, Wind) = Entropy(S) - \sum_{v \in \{Strong, Weak\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$Gain(S_{Sunny}, Wind) = Entropy(S) - \frac{2}{5}Entropy(S_{Strong}) - \frac{3}{5}Entropy(S_{Weak})$$

$$Gain(S_{sunny}, Wind) = 0.97 - \frac{2}{5}1.0 - \frac{3}{5}0.918 = 0.0192$$

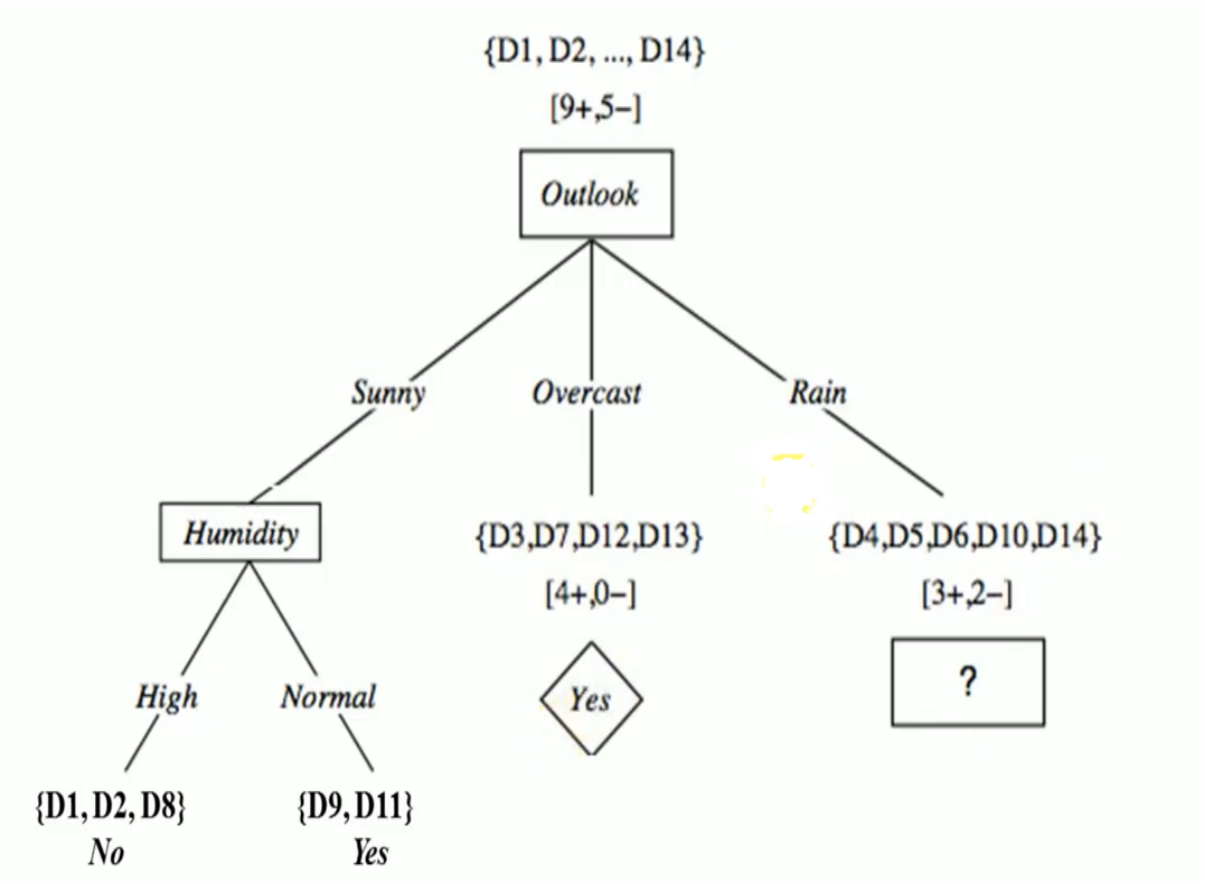
Step 9: Choose the maximum gain…

Day	Temp	Humidity	Wind	Play Tennis
D1	Hot	High	Weak	No
D2	Hot	High	Strong	No
D8	Mild	High	Weak	No
D9	Cool	Normal	Weak	Yes
D11	Mild	Normal	Strong	Yes

$$Gain(S_{sunny}, Temp) = 0.570$$

$$Gain(S_{sunny}, Humidity) = 0.97$$

$$Gain(S_{sunny}, Wind) = 0.0192$$



Step 10:

Day	Temp	Humidity	Wind	Play Tennis
D4	Mild	High	Weak	Yes
D5	Cool	Normal	Weak	Yes
D6	Cool	Normal	Strong	No
D10	Mild	Normal	Weak	Yes
D14	Mild	High	Strong	No

Attribute: Temp

$Values\ (Temp) = Hot, Mild, Cool$

$S_{Rain} = [3+, 2-]$ $\quad Entropy(S_{Sunny}) = -\frac{3}{5}log_2\frac{3}{5} - \frac{2}{5}log_2\frac{2}{5} = 0.97$

$S_{Hot} \leftarrow [0+, 0-]$ $\quad Entropy(S_{Hot}) = 0.0$

$S_{Mild} \leftarrow [2+, 1-]$ $\quad Entropy(S_{Mild}) = -\frac{2}{3}log_2\frac{2}{3} - \frac{1}{3}log_2\frac{1}{3} = 0.9183$

$S_{Cool} \leftarrow [1+, 1-]$ $\quad Entropy(S_{Cool}) = 1.0$

$$Gain\ (S_{Rain}, Temp) = Entropy(S) - \sum_{v \in \{Hot, Mild, Cool\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$Gain(S_{Rain}, Temp)$$

$$= Entropy(S) - \frac{0}{5}Entropy(S_{Hot}) - \frac{3}{5}Entropy(S_{Mild})$$

$$- \frac{2}{5}Entropy(S_{Cool})$$

$$Gain(S_{Rain}, Temp) = 0.97 - \frac{0}{5}0.0 - \frac{3}{5}0.918 - \frac{2}{5}1.0 = 0.0192$$

Step 11:

Day	Temp	Humidity	Wind	Play Tennis
D4	Mild	High	Weak	Yes
D5	Cool	Normal	Weak	Yes
D6	Cool	Normal	Strong	No
DI0	Mild	Normal	Weak	Yes
DI4	Mild	High	Strong	No

Attribute: Humidity

$Values\ (Humidity) = High, Normal$

$S_{Rain} = [3+, 2-]$ $\quad Entropy(S_{Sunny}) = -\frac{3}{5}log_2\frac{3}{5} - \frac{2}{5}log_2\frac{2}{5} = 0.97$

$S_{High} \leftarrow [1+, 1-]$ $\quad Entropy(S_{High}) = 1.0$

$S_{Normal} \leftarrow [2+, 1-]$ $\quad Entropy(S_{Normal}) = -\frac{2}{3}log_2\frac{2}{3} - \frac{1}{3}log_2\frac{1}{3} = 0.9183$

$$Gain\ (S_{Rain}, Humidity) = Entropy(S) - \sum_{v \in \{High, Normal\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$Gain(S_{Rain}, Humidity) = Entropy(S) - \frac{2}{5}Entropy(S_{High}) - \frac{3}{5}Entropy(S_{Normal})$$

$$Gain(S_{Rain}, Humidity) = 0.97 - \frac{2}{5}1.0 - \frac{3}{5}0.918 = 0.0192$$

Step 12:

Day	Temp	Humidity	Wind	Play Tennis
D4	Mild	High	Weak	Yes
D5	Cool	Normal	Weak	Yes
D6	Cool	Normal	Strong	No
DI0	Mild	Normal	Weak	Yes
DI4	Mild	High	Strong	No

Attribute: Wind

$Values\ (wind) = Strong, Weak$

$S_{Rain} = [3+, 2-]$ $\quad Entropy(S_{Sunny}) = -\frac{3}{5}log_2\frac{3}{5} - \frac{2}{5}log_2\frac{2}{5} = 0.97$

$S_{Strong} \leftarrow [0+, 2-]$ $\quad Entropy(S_{Strong}) = 0.0$

$S_{Weak} \leftarrow [3+, 0-]$ $\quad Entropy(S_{weak}) = 0.0$

$$Gain\ (S_{Rain}, Wind) = Entropy(S) - \sum_{v \in \{Strong, Weak\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$Gain(S_{Rain}, Wind) = Entropy(S) - \frac{2}{5}Entropy(S_{Strong}) - \frac{3}{5}Entropy(S_{Weak})$$

$$Gain(S_{Rain}, Wind) = 0.97 - \frac{2}{5}0.0 - \frac{3}{5}0.0 = 0.97$$

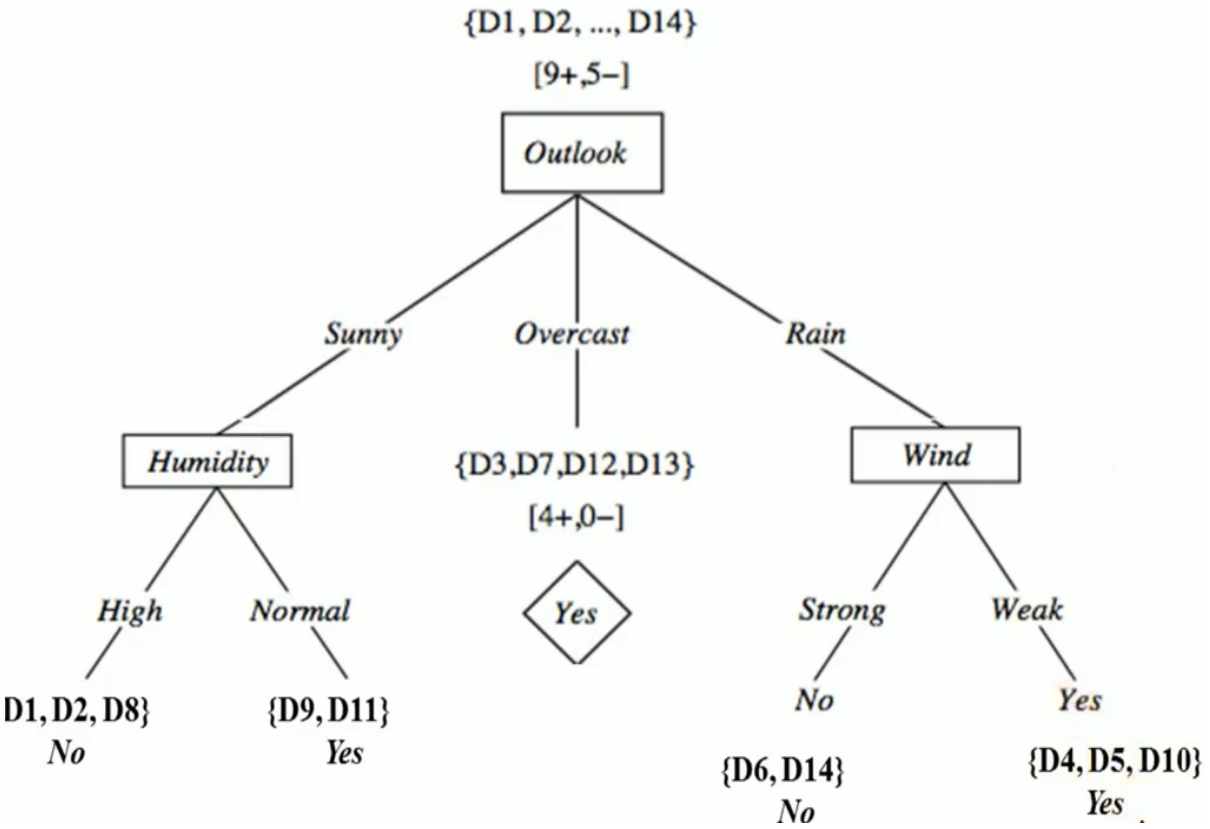
Step 13: Choose the maximum gain…

Day	Temp	Humidity	Wind	Play Tennis
D4	Mild	High	Weak	Yes
D5	Cool	Normal	Weak	Yes
D6	Cool	Normal	Strong	No
Dl0	Mild	Normal	Weak	Yes
Dl4	Mild	High	Strong	No

$$Gain(S_{Rain}, Temp) = 0.0192$$

$$Gain(S_{Rain}, Humidity) = 0.0192$$

$$Gain(S_{Rain}, Wind) = 0.97$$



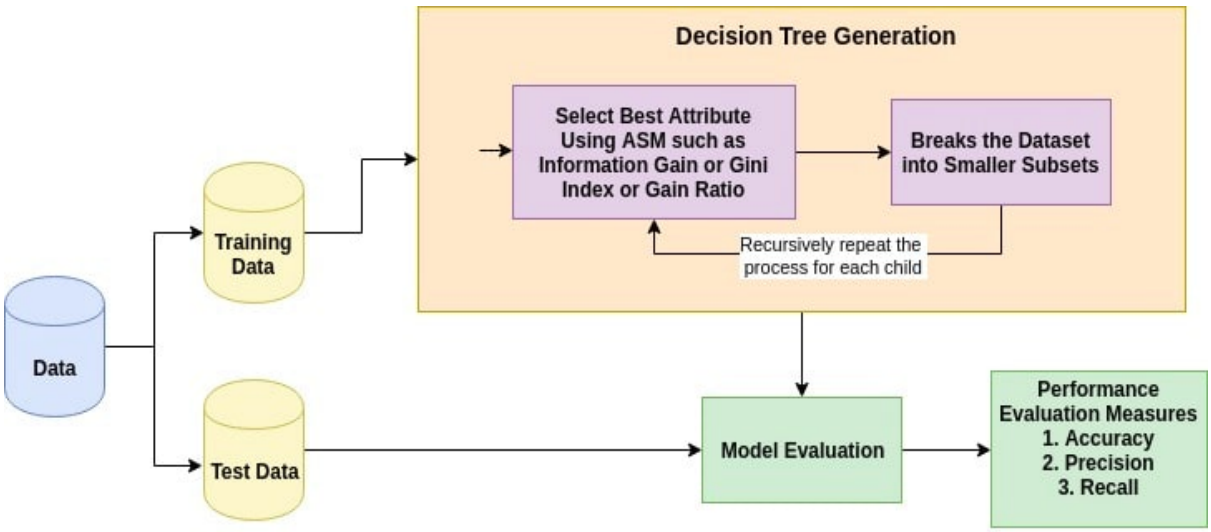
Advantages

- Easy to understand and interpret.
- Does not require Data normalization

- Doesn’t facilitate the need for scaling of data
- The preprocessing stage requires lesser effort compared to other major algorithms, hence in a way optimizes the given problem

Disadvantages

- Requires higher time to train the model
- It has a considerable high complexity and takes more time to process the data
- When decrease in user input parameter is very small it leads to the termination of the tree
- Calculations can get very complex at times

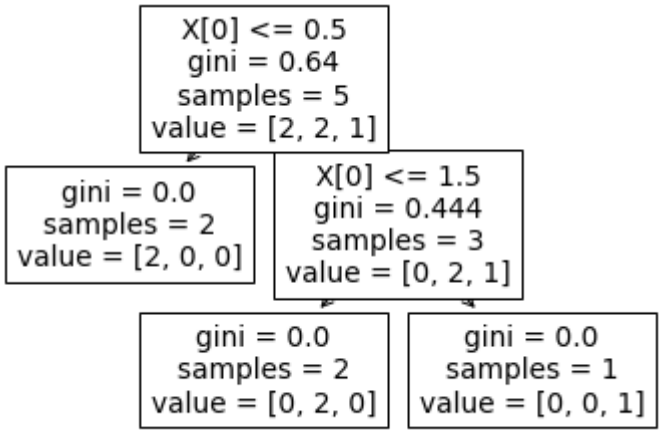


Decision Tree using Python : (Dummy Data)

```
from sklearn import tree
X = [[0, 0], [1, 1], [0, 1], [1, 0], [2, 2]]
Y = [0, 1, 0, 1, 2]
clf = tree.DecisionTreeClassifier()
clf = clf.fit(X, Y)
print(clf.predict([[2., 1.]]))
```

Now, plot using the built-in `plot_tree` in the `tree` module

```
tree.plot_tree(clf)
```



At the root node, we have 5 samples. It checks for the first value in X and if it's less than or equal to 0.5, it classifies the sample as **0.** If it's not, then it checks if the first value in X is less than or equal to 1.5, in which case it assigns the label **1** to it, otherwise **2.**

Note that the decision tree doesn't include any check on the second value in X. This is not an error as the second value is not needed in this case. If the decision tree is able to make all the classifications without the need for all the features, then it can ignore other features.

4.6. BAYESIAN NETWORK

Bayesian belief network is key computer technology for dealing with probabilistic events and to solve a problem which has uncertainty. We can define a Bayesian network as:

"A Bayesian network is a probabilistic graphical model which represents a set of variables and their conditional dependencies using a directed acyclic graph."

It is also called a **Bayes network, belief network, decision network**, or **Bayesian model**.

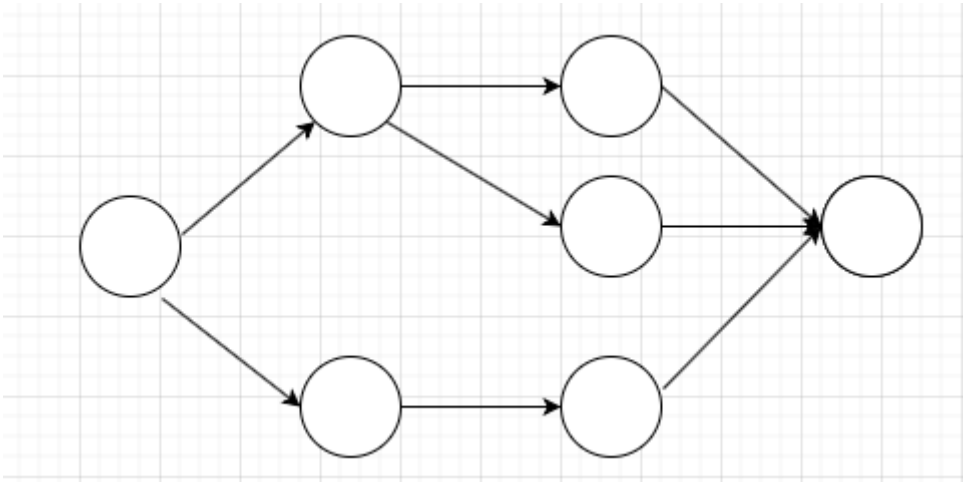
Bayesian networks are probabilistic, because these networks are built from a **probability distribution**, and also use probability theory for prediction and anomaly detection.

Real world applications are probabilistic in nature, and to represent the relationship between multiple events, we need a Bayesian network. It can also be used in various tasks including **prediction, anomaly detection, diagnostics, automated insight, reasoning, time series prediction**, and **decision making under uncertainty**.

What is a Bayesian Network?

A Bayesian network falls under the category of Probabilistic Graphical Modelling technique, which is used to calculate uncertainties by using the notion of probability.

They are used to model improbability using directed acyclic graphs.

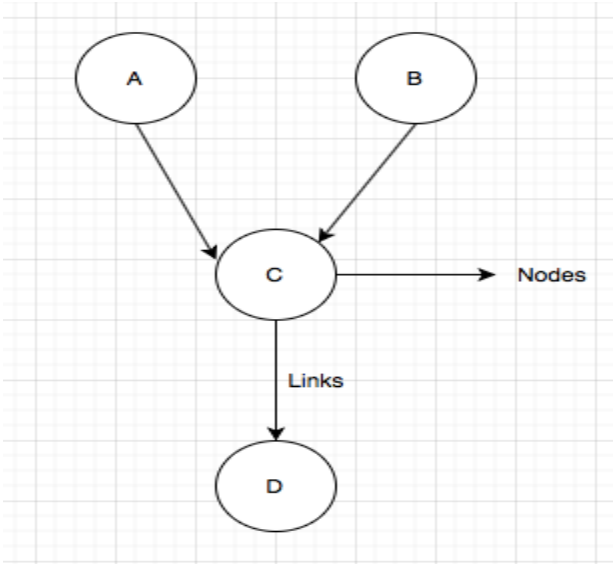


Bayesian Network can be used for building models from data and experts opinions, and it consists of two parts:

- Directed Acyclic Graph
- Table of conditional probabilities.

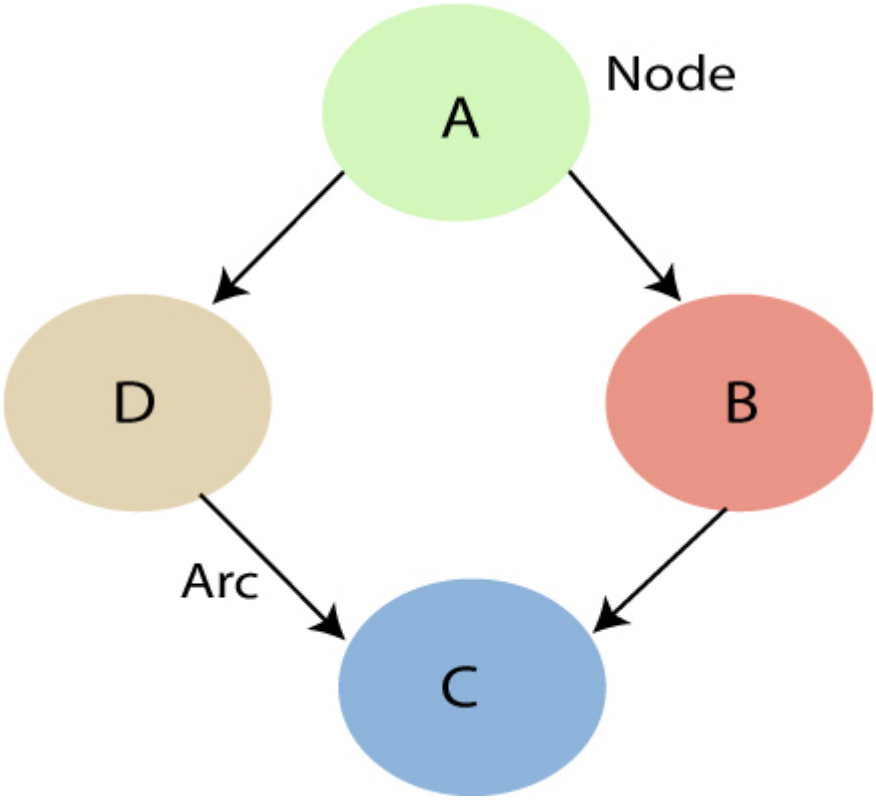
What is Directed Acyclic Graph?

It is used to represent the Bayesian Network. A directed acyclic graph contains nodes and links, where links denote the relationship between nodes.



The generalized form of Bayesian network that represents and solve decision problems under uncertain knowledge is known as an **Influence diagram**.

A Bayesian network graph is made up of nodes and Arcs (directed links), where:



Each **node** corresponds to the random variables, and a variable can be **continuous** or **discrete**.

Arc or directed arrows represent the causal relationship or conditional probabilities between random variables. These directed links or arrows connect the pair of nodes in the graph.

These links represent that one node directly influence the other node, and if there is no directed link that means that nodes are independent with each other

- In the above diagram, A, B, C, and D are random variables represented by the nodes of the network graph.
- If we are considering node B, which is connected with node A by a directed arrow, then node A is called the parent of Node B.
- Node C is independent of node A.

The Bayesian network has mainly two components:

- **Causal Component**
- **Actual numbers**

Each node in the Bayesian network has condition probability distribution $\mathbf{P(X_i \mid Parent(X_i))}$, which determines the effect of the parent on that node.

Bayesian network is based on *Joint probability distribution* and *conditional probability.*

Joint probability distribution:

If we have variables x1, x2, x3,....., xn, then the probabilities of a different combination of x1, x2, x3.. xn, are known as Joint probability distribution.

$\mathbf{P[x_1, x_2, x_3,....., x_n]}$, it can be written as the following way in terms of the joint probability distribution.

$\mathbf{= P[x_1 \mid x_2, x_3,....., x_n]P[x_2, x_3,....., x_n]}$

$\mathbf{= P[x_1 \mid x_2, x_3,....., x_n]P[x_2 \mid x_3,....., x_n]....P[x_{n-1} \mid x_n]P[x_n].}$

In general for each variable Xi, we can write the equation as:

$$P(X_i \mid X_{i-1},....., X_1) = P(X_i \mid Parents(X_i))$$

Example: Harry installed a new burglar alarm at his home to detect burglary. The alarm reliably responds at detecting a burglary but also responds for minor earthquakes. Harry has two neighbours David and Sophia, who have taken a responsibility to inform Harry at work when they hear the alarm. David always calls Harry when he hears the alarm, but sometimes he got confused with the phone ringing and calls at that time too. On the other hand, Sophia likes to listen to high music, so sometimes she misses to hear the alarm. Here we would like to compute the probability of Burglary Alarm.

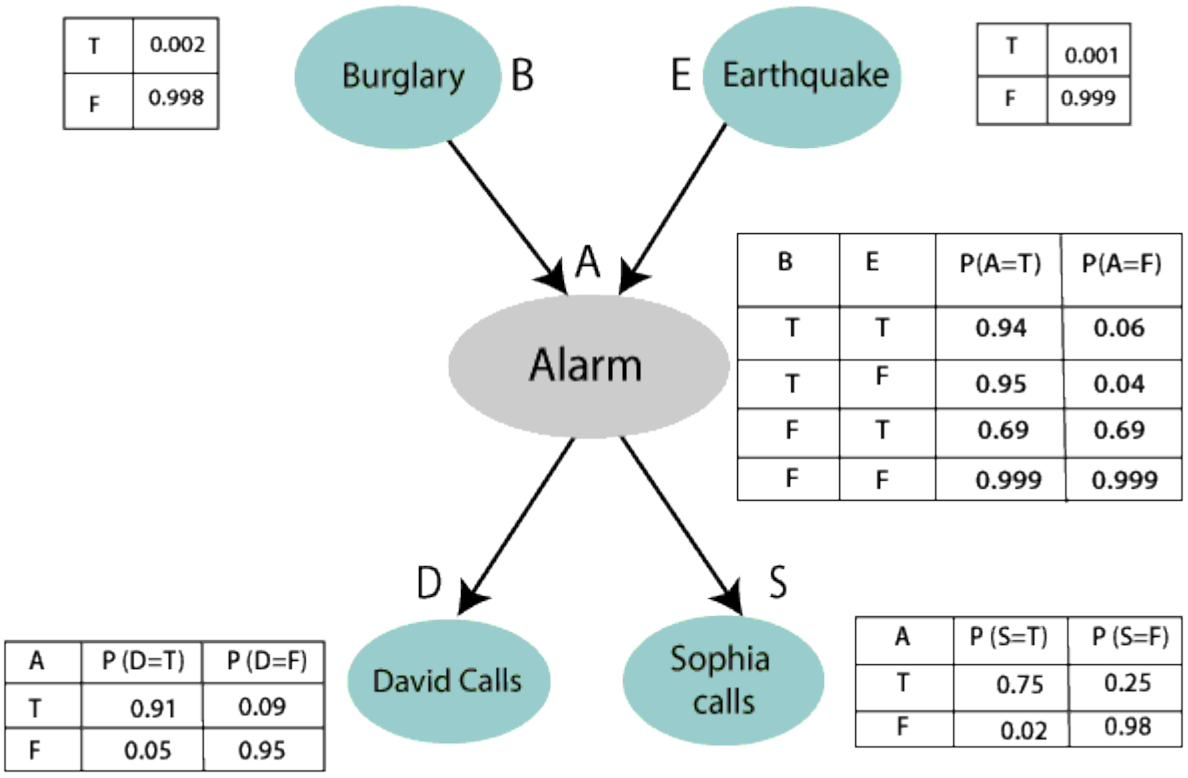
Problem:

Calculate the probability that alarm has sounded, but there is neither a burglary, nor an earthquake occurred, and David and Sophia both called the Harry.

Solution:

- The network structure is showing that burglary and earthquake is the parent node of the alarm and directly affecting the probability of alarm's going off, but David and Sophia's calls depend on alarm probability.
- The network is representing that our assumptions do not directly perceive the burglary and also do not notice the minor earthquake, and they also not confer before calling.
- The conditional distributions for each node are given as conditional probabilities table or CPT.
- Each row in the CPT must be sum to 1 because all the entries in the table represent an exhaustive set of cases for the variable.

- In CPT, a boolean variable with k boolean parents contains 2^K probabilities. Hence, if there are two parents, then CPT will contain 4 probability values



T	0.002
F	0.998

T	0.001
F	0.999

B	E	P(A=T)	P(A=F)
T	T	0.94	0.06
T	F	0.95	0.04
F	T	0.69	0.69
F	F	0.999	0.999

A	P (D=T)	P (D=F)
T	0.91	0.09
F	0.05	0.95

A	P (S=T)	P (S=F)
T	0.75	0.25
F	0.02	0.98

List of all events occurring in this network:

- Burglary (B)
- Earthquake(E)
- Alarm(A)
- David Calls(D)
- Sophia calls(S)

We can write the events of problem statement in the form of probability: **P[D, S, A, B, E]**, can rewrite the above probability statement using joint probability distribution:

P[D, S, A, B, E] = P[D | S, A, B, E]. P[S, A, B, E]

= P[D | S, A, B, E]. P[S | A, B, E]. P[A, B, E]

= P [D| A]. P [S| A, B, E]. P[A, B, E]

= P[D | A]. P[S | A]. P[A| B, E]. P[B, E]

= P[D | A]. P[S | A]. P[A| B, E]. P[B |E]. P[E]

Let's take the observed probability for the Burglary and earthquake component:

P(B= True) = 0.002, which is the probability of burglary.

P(B= False)= 0.998, which is the probability of no burglary.

P(E= True)= 0.001, which is the probability of a minor earthquake

P(E= False)= 0.999, Which is the probability that an earthquake not occurred.

We can provide the conditional probabilities as per the below tables:

Conditional probability table for Alarm A:

The Conditional probability of Alarm A depends on Burglar and earthquake:

B	**E**	**P(A= True)**	**P(A= False)**
True	True	0.94	0.06
True	False	0.95	0.04
False	True	0.31	0.69
False	False	0.001	0.999

Conditional probability table for David Calls:

The Conditional probability of David that he will call depends on the probability of Alarm.

A	**P(D= True)**	**P(D= False)**
True	0.91	0.09
False	0.05	0.95

Conditional probability table for Sophia Calls:

The Conditional probability of Sophia that she calls is depending on its Parent Node "Alarm."

A	**P(S= True)**	**P(S= False)**
True	0.75	0.25
False	0.02	0.98

From the formula of joint distribution, we can write the problem statement in the form of probability distribution:

P(S, D, A, ¬B, ¬E) = P (S|A) *P (D|A)*P (A|¬B ^ ¬E) *P (¬B) *P (¬E)

= 0.75* 0.91* 0.001* 0.998*0.999

= 0.00068045.

Hence, a Bayesian network can answer any query about the domain by using Joint distribution.

Bayesian Networks in Python

Bayesian Networks can be developed and used for inference in Python.

A popular library for this is called **PyMC** and provides a range of tools for Bayesian modelling, including graphical models like Bayesian Networks.

The most recent version of the library is called **PyMC3**, named for Python version 3, and was developed on top of the Theano mathematical computation library that offers fast automatic differentiation.

Installation with environment:

conda create -n BNLEARN python=3.6
conda activate BNLEARN
conda install -c ankurankan pgmpy

conda deactivate
conda activate BNLEARN

pip install bnlearn

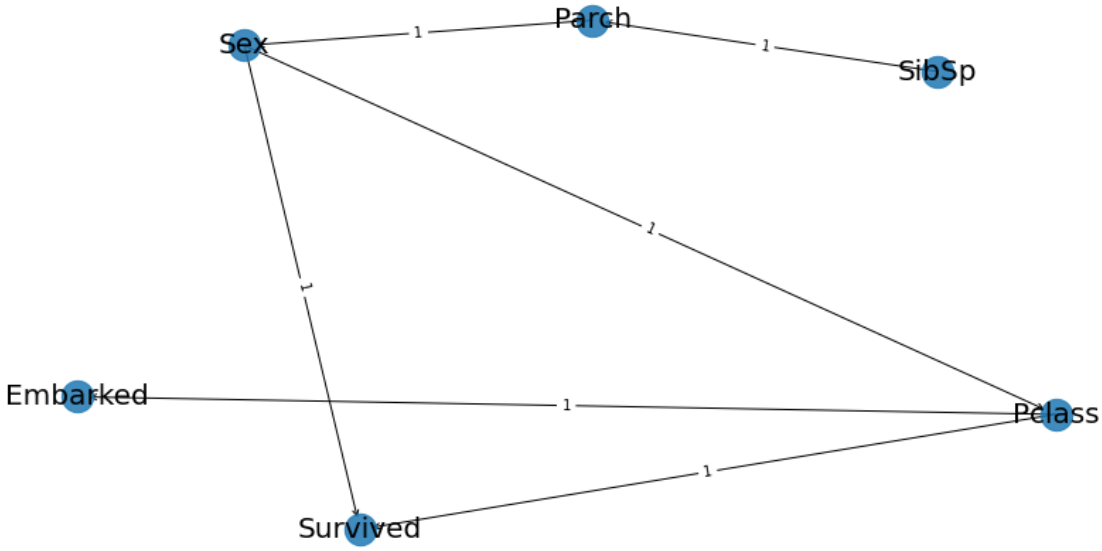
Load titanic dataset containing mixed variables
df_raw = bnlearn.import_example(data='titanic')

Pre-processing of the input dataset
dfhot, dfnum = bnlearn.df2onehot(df_raw)

Structure learning
DAG = bnlearn.structure_learning.fit(dfnum)

Plot

G = bnlearn.plot(DAG)



Parameter learning
model = bnlearn.parameter_learning.fit(DAG, df)

Print CPDs
bnlearn.print_CPD(model)

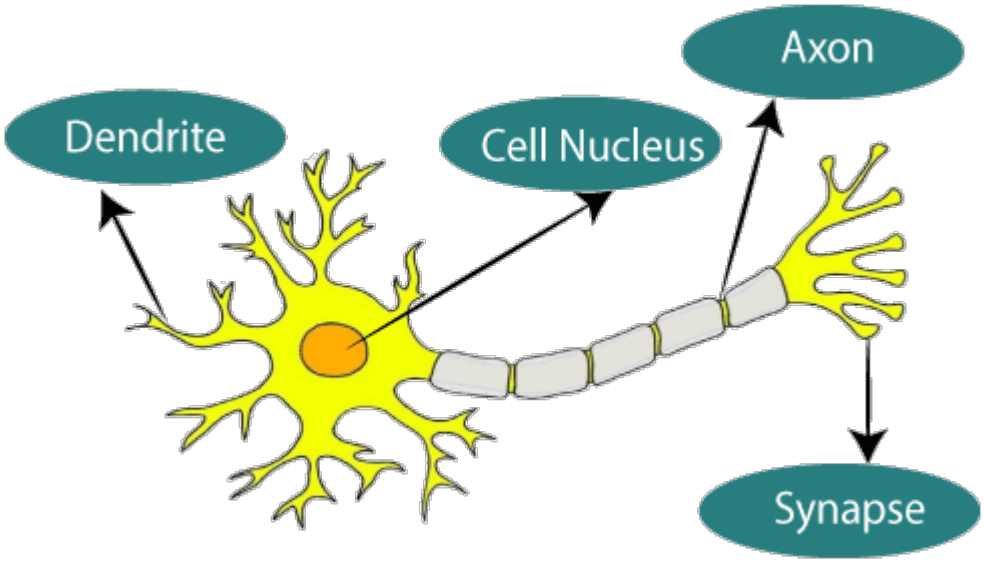
Make inference
q = bnlearn.inference.fit(model, variables=['Survived'], evidence={'Sex':0, 'Pclass':1})

print(q.values)
print(q.variables)

4.7. NEURAL NETWORK

Neural Networks is a computational learning system that uses a network of functions to understand and translate a data input of one form into a desired output, usually in another form. The concept of the artificial neural network was inspired by human biology and the way neurons of the human brain function together to understand inputs from human senses.

In simple words, Neural Networks are a set of algorithms that tries to recognize the patterns, relationships, and information from the data through the process which is inspired by and works like the human brain/biology.

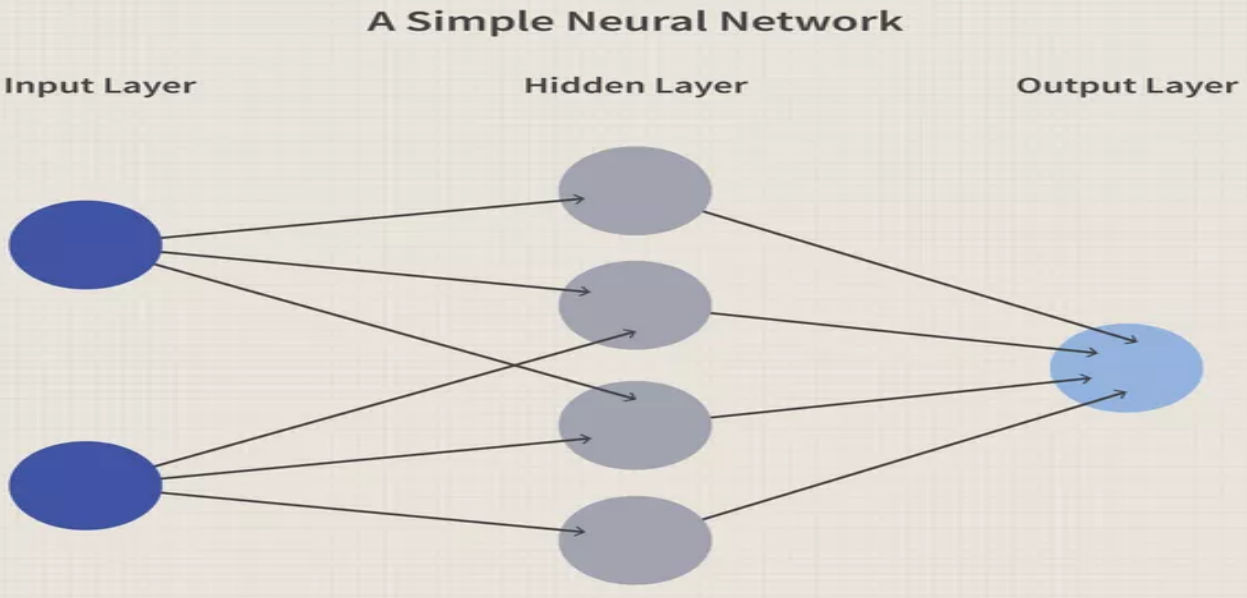


The figure illustrates the typical diagram of Biological Neural Network.

Components / Architecture of Neural Network

A simple neural network consists of three components :

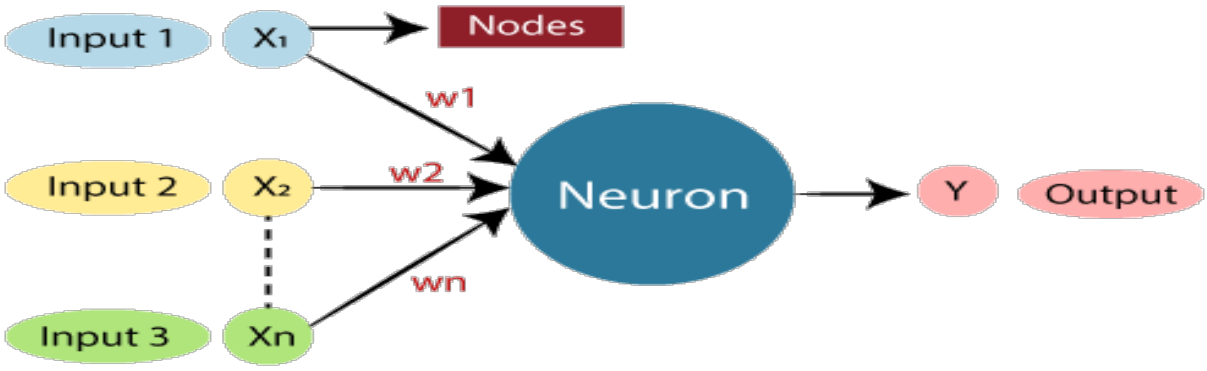
- Input layer
- Hidden layer
- Output layer



Input Layer: Also known as Input nodes are the inputs/information from the outside world is provided to the model to learn and derive conclusions from. Input nodes pass the information to the next layer i.e Hidden layer.

Hidden Layer: Hidden layer is the set of neurons where all the computations are performed on the input data. There can be any number of hidden layers in a neural network. The simplest network consists of a single hidden layer.

Output layer: The output layer is the output/conclusions of the model derived from all the computations performed. There can be single or multiple nodes in the output layer. If we have a binary classification problem the output node is 1 but in the case of multi-class classification, the output nodes can be more than 1.



The typical Artificial Neural Network looks something like the above figure.

Relationship between Biological neural network and artificial neural network:

Biological Neural Network	Artificial Neural Network (ANN)
Dendrites	Inputs
Cell nucleus	Nodes
Synapse	Weights
Axon	Output

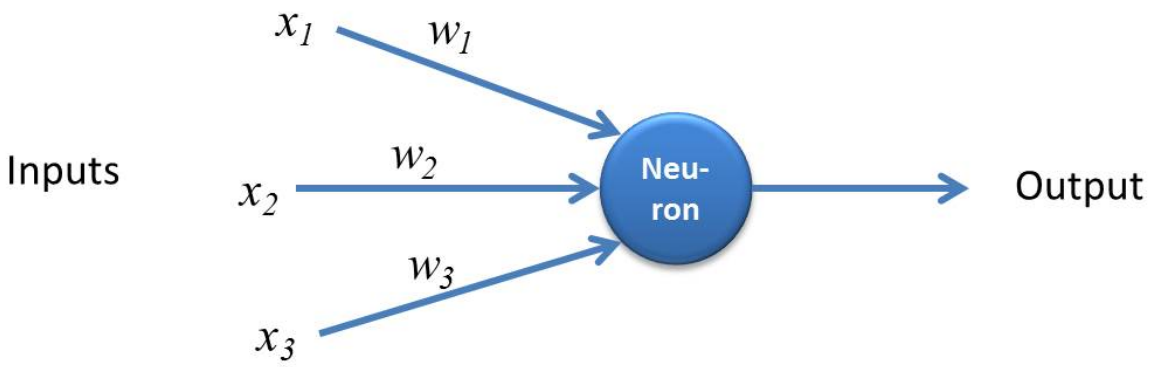
The artificial neural network takes input and computes the weighted sum of the inputs and includes a bias. This computation is represented in the form of a transfer function.

$$\sum_{i=1}^{n} Wi * Xi + b$$

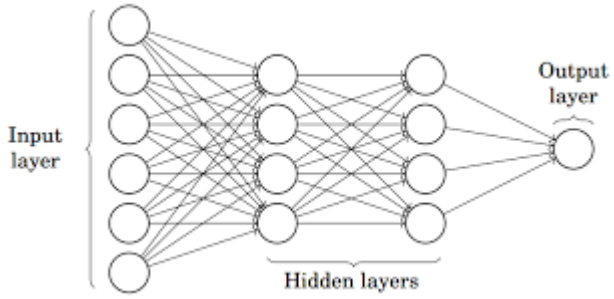
It determines weighted total is passed as an input to an activation function to produce the output. Activation functions choose whether a node should fire or not. Only those who are fired make it to the output layer. There are distinctive activation functions available that can be applied upon the sort of task we are performing.

Perceptron and Multi-Layer Perceptron

Perceptron is a simple form of Neural Network and consists of a single layer where all the mathematical computations are performed.



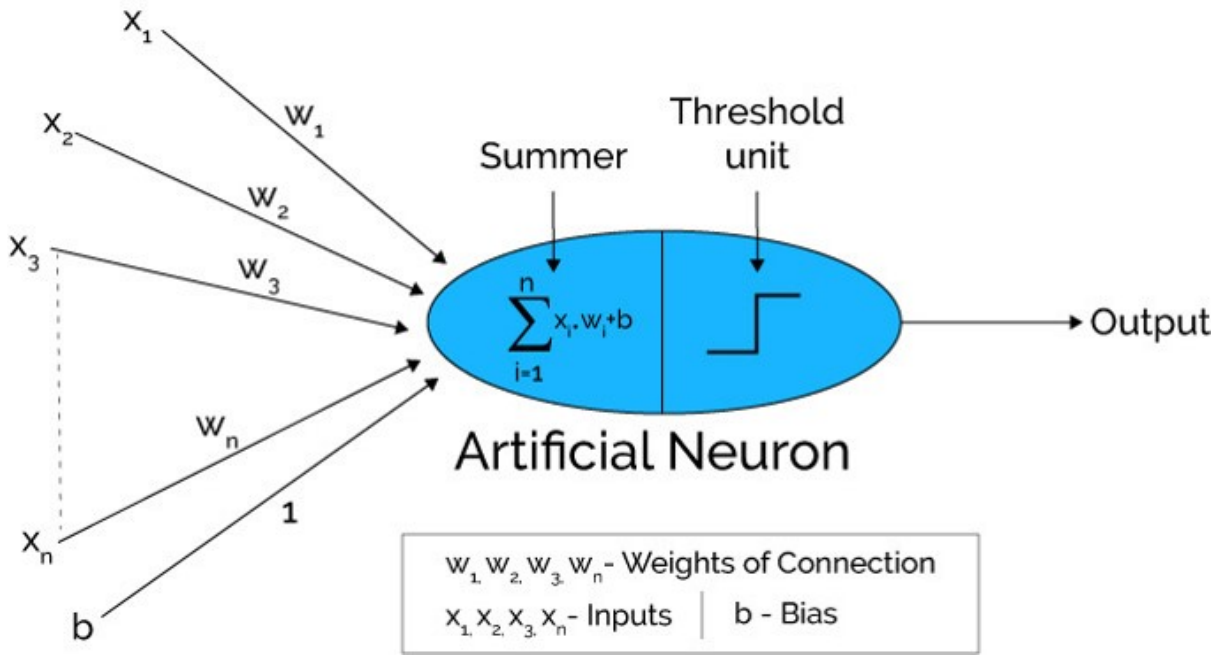
Whereas, **Multilayer Perceptron** also known as **Artificial Neural Networks** consists of more than one perception which is grouped together to form a multiple layer neural network.



In the above figure, The Artificial Neural Network consists of four layers interconnected with each other:

- An input layer, with 6 input nodes
- Hidden Layer 1, with 4 hidden nodes/4 perceptrons
- Hidden layer 2, with 4 hidden nodes
- Output layer with 1 output node

Step by Step Working of the Artificial Neural Network



1. In the first step, **Input units are passed i.e data is passed with some weights attached to it to the hidden layer**. We can have any number of hidden layers. In the above image inputs $x_1,x_2,x_3,\ldots x_n$ is passed.
2. Each hidden layer consists of neurons. All the inputs are connected to each neuron.
3. After passing on the inputs, **all the computation is performed in the hidden layer**.

Computation performed in hidden layers are done in two steps which are as follows :

- First of all, **all the inputs are multiplied by their weights**. Weight is the gradient or coefficient of each variable. It shows the strength of the particular input. After assigning the weights, a bias variable is added. **Bias** is a constant that helps the model to fit in the best way possible.

$$Z_1 = W_1*In_1 + W_2*In_2 + W_3*In_3 + W_4*In_4 + W_5*In_5 + b$$

W_1, W_2, W_3, W_4, W5 are the weights assigned to the inputs In_1, In_2, In_3, In_4, In_5, and b is the bias.

- Then in the second step, the **activation function is applied to the linear equation Z1.** The activation function is a non-linear transformation that is applied to the input before sending it to the next layer of neurons. The importance of the activation function is to inculcate nonlinearity in the model.

4. The whole process described in point 3 is performed in each hidden layer. After passing through every hidden layer, **we move to the last layer i.e our output layer which gives us the final output.**

The process explained above is known as **forwarding Propagation.**

5. After getting the predictions from the output layer, the **error is calculated i.e the difference between the actual and the predicted output.**

If the error is large, then the steps are taken to minimize the error and for the same purpose, **Back Propagation is performed.**

What is Back Propagation and How it works?

Back Propagation is the process of updating and finding the optimal values of weights or coefficients which helps the model to minimize the error i.e difference between the actual and predicted values.

The weights are updated with the help of optimizers. Optimizers are the methods/ mathematical formulations to change the attributes of neural networks i.e weights to minimize the error.

Activation Functions

Activation functions are attached to each neuron and are mathematical equations that determine whether a neuron should be activated or not based on whether the neuron's input is relevant for the model's prediction or not. The purpose of the activation function is to introduce the nonlinearity in the data.

Various Types of Activation Functions are :

- Sigmoid Activation Function
- TanH / Hyperbolic Tangent Activation Function
- Rectified Linear Unit Function (ReLU)
- Leaky ReLU
- Softmax

What Is a Convolutional Neural Network?
A convolutional neural network is one adapted for analysing and identifying visual data such as digital images or photographs.

What Is a Recurrent Neural Network?
A recurrent neural network is one adapted for analysing time series data, event history, or temporal ordering.

What Is a Deep Neural Network?
Also known as a deep learning network, is one that involves two or more processing layers.

Types of Artificial Neural Network:

There are various types of Artificial Neural Networks (ANN) depending upon the human brain neuron and network functions, an artificial neural network similarly performs tasks. For example, segmentation or classification.

- Feedback ANN:

  The feedback networks feed information back into itself and are well suited to solve optimization issues. The Internal system error corrections utilize feedback ANNs.

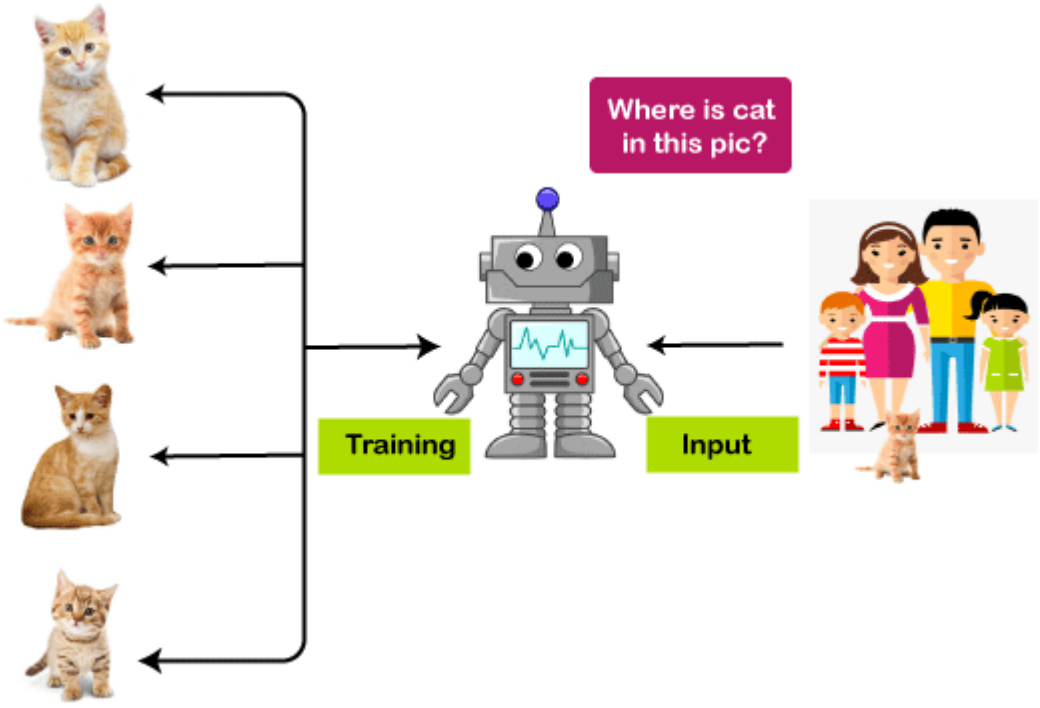
- Feed-Forward ANN:

A feed-forward network is a basic neural network comprising of an input layer, an output layer, and at least one layer of a neuron. Through assessment of its output by reviewing its input, the intensity of the network can be noticed based on group behaviour of the associated neurons, and the output is decided. The primary advantage of this network is that it figures out how to evaluate and recognize input patterns.

Importance of Neural Network:

- **Without Neural Network:**

Let's have a look at the example given below. Here we have a machine, such that we have trained it with four types of cats, as you can see in the image below. And once we are done with the training, we will provide a random image to that particular machine that has a cat. Since this cat is not similar to the cats through which we have trained our system, so without the neural network, our machine would not identify the cat in the picture. Basically, the machine will get confused in figuring out where the cat is.



- **With Neural Network:**

However, when we talk about the case with a neural network, even if we have not trained our machine with that particular cat. But still, it can identify certain features of a cat that we have trained on, and it can match those features with the cat that is there in that particular image and can also identify the cat.

Advantages of Artificial Neural Network

- **Parallel processing capability**
- **Storing data on the entire network**
- **Capability to work with incomplete knowledge**
- **Having a memory distribution**
- **Having fault tolerance**

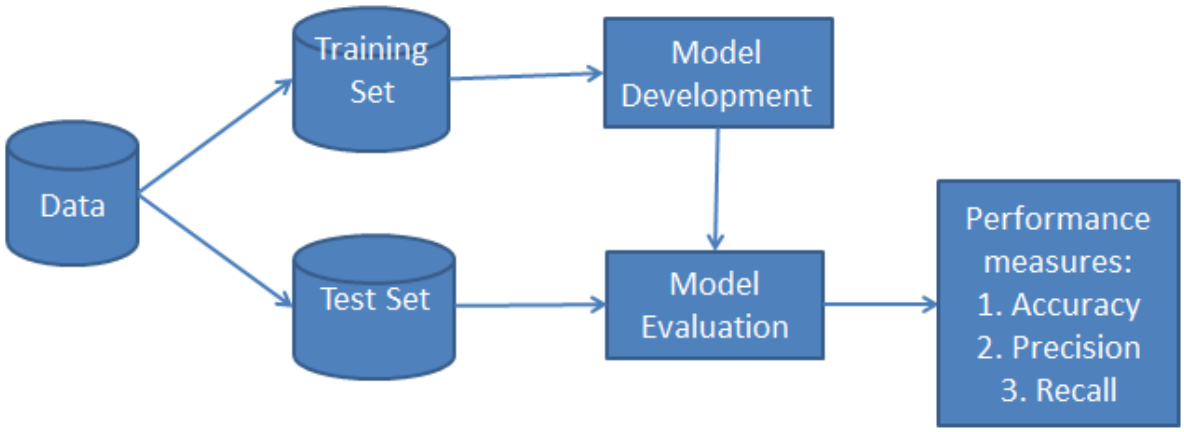
Disadvantages of Artificial Neural Network

- **Assurance of proper network structure**
- **Unrecognized behaviour of the network**
- **Hardware dependence**
- **Difficulty of showing the issue to the network**
- **The duration of the network is unknown**

4.8. TRAINING, TESTING AND EVALUATING MACHINE LEARNING MODELS

Introduction to Model Evaluation

In Machine Learning, our goal is to achieve a machine learning model that *generalizes* well on new unseen data or unknown data. There is always a problem of overfitting while training your machine learning model which is a central obstacle to take care of. While some of you may have a small dataset and training neural networks on small datasets doesn't generalize well and began to overfit. It's very important to measure the generalization power of your model so that it can perform well for what it was trained for avoiding overfitting case.



Training, Validation and Test Sets

While evaluating a model we always split our data into three sets:

- Training
- Validation
- Test set.

First train the model on the training data and evaluate the validation data and once the model is ready, test it one final time on the test data.

Training set – refers to a subset of a dataset used to build predictive models. It includes a set of input examples that will be used to train a model by adjusting the parameters of the set.

Validation set – is a subset of a dataset whose purpose is to assess the performance of the model built, during the training phase. It periodically evaluates a model and allows for fine-tuning of the parameters of the model.

Test set – this is also known as unseen data. It is the final evaluation that a model undergoes after the training phase. A test set is a subset of a dataset used to assess the possible future performance of a model. For example, if a model fits the training better than the test set, overfitting is likely present.

Overfitting– refers to when a model contains more parameters than can be accounted for by the dataset. Noisy data contributes to overfitting. The generalization of these models is unreliable since the model learns more than it is meant to from the dataset.

Common split percentages include:

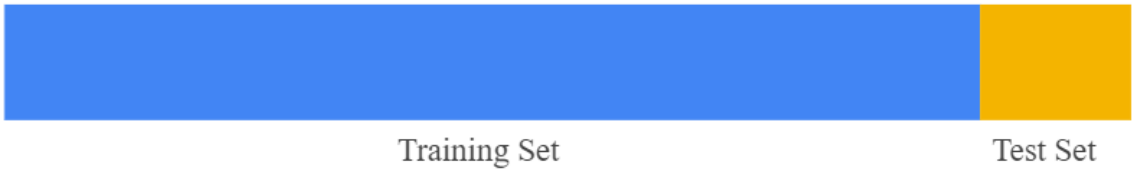
- Train: 80%, Test: 20%
- Train: 67%, Test: 33%
- Train: 50%, Test: 50%

Why not only training and test set, Why Validation?

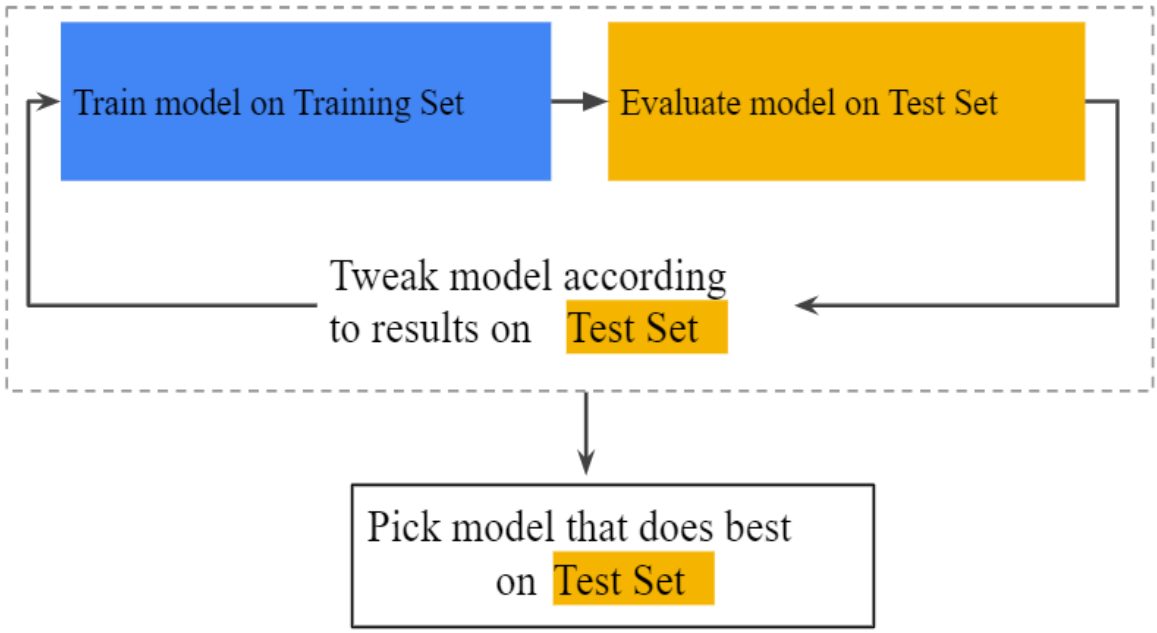
We can train on the training set and evaluate it on the test set and model will be ready, then why validation set?

Workflow without a validation set

We divide our dataset into two parts :



Then we follow this process :

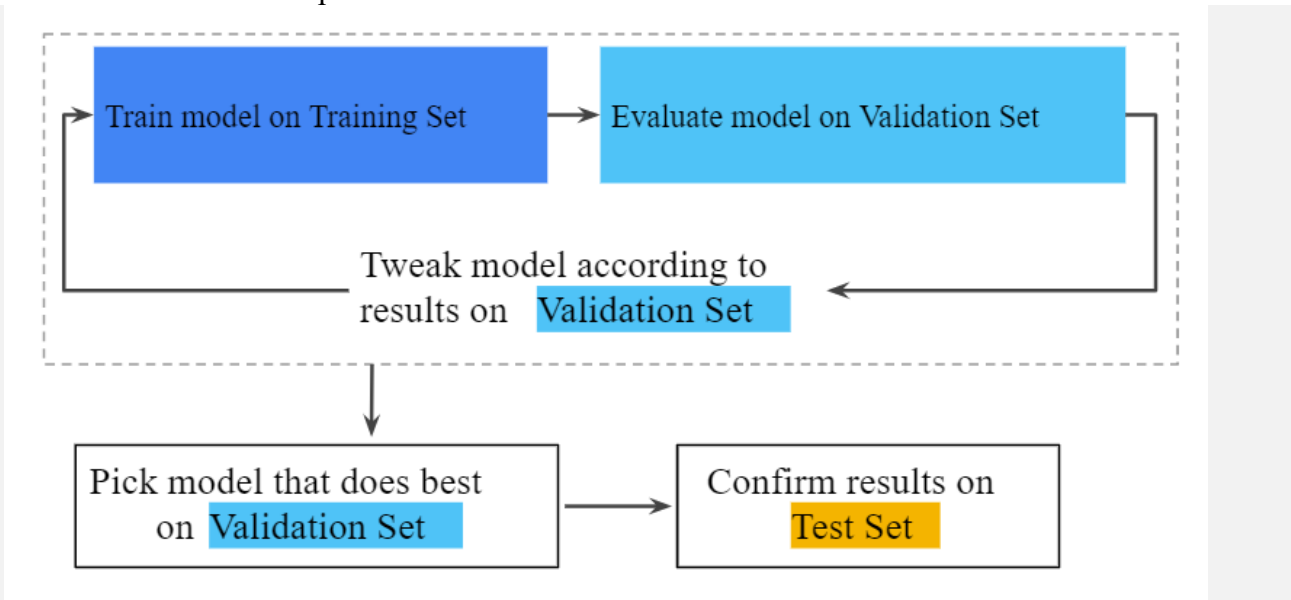


We train a model on a training set and then evaluate it on a test set and according to the result we tune our model parameters and then after the different iterative process on tuning model, we pick a model that does best on the test set.

Why the Validation set?

To reduce overfitting, a new set called validation set is used. In this approach, we divide our dataset into three sets: training, testing and validation sets.

And then we follow this process :



We train our model on the training set and this time we evaluate our model on the validation set and tune the parameters of the model according to the validation loss and accuracy and we repeat this process until we get the model that best fit on the validation set. And after choosing the best model we test or confirm our results on testing set to get the correct accuracy or how well our model is generalised.

Model evaluation techniques

The techniques to evaluate the performance of a model can be divided into two parts: cross-validation and holdout. Both these techniques make use of a test set to assess model performance.

1. Cross validation :

Cross-validation involves the use of a training dataset and an independent dataset. These two sets result from partitioning the original dataset. The sets are used to evaluate an algorithm.

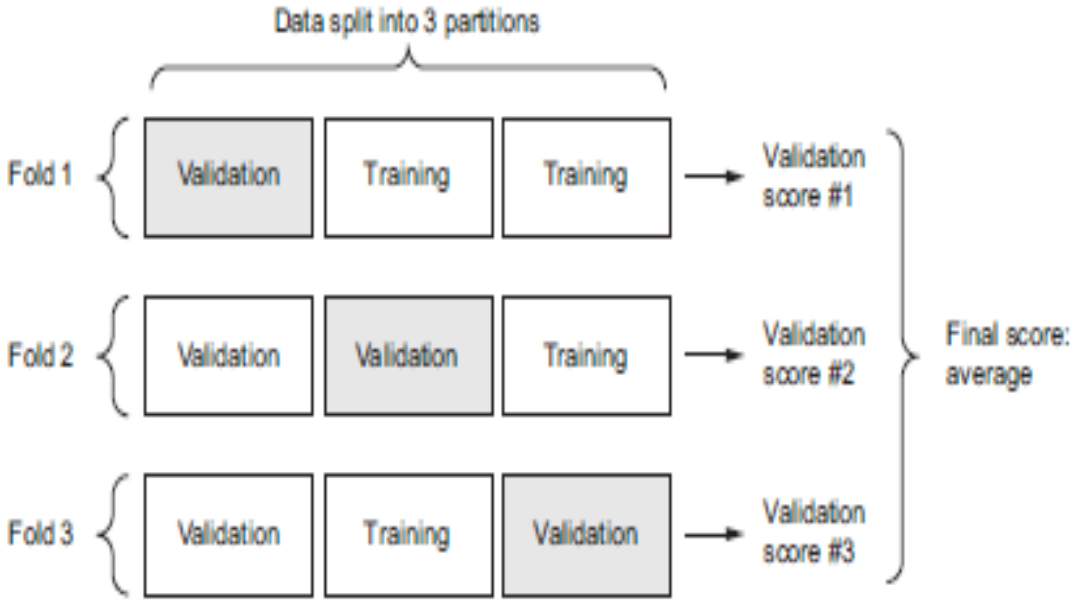
First, we split the dataset into groups of instances equal in size. These groups are called folds. The model to be evaluated is trained on all the folds except one. After training, we test the model on the fold that was excluded. This process is then repeated over and over again, depending on the number of folds.

If there are six folds, we will repeat the process six times. The reason for the repetition is that each fold gets to be excluded and act as the test set. Last, we measure the average performance across all folds to get an estimation of how effective the algorithm is on a problem.

A popular cross-validation technique is the k-fold cross-validation. It uses the same steps described above. The k, (is a user-specified number), stands for the number of folds. The value of k may vary based on the size of the dataset but as an example, let us use a scenario of 3-fold cross-validation.

The model will be trained and tested three times. Let's say the first-round trains on folds 1 and 2 . The testing will be on fold 3. For the second round, it may train on folds 1 and 3 and test on fold 2. For the last round, it may train on folds 2 and 3 and test on fold 1.

The interchange between training and test data makes this method very effective. However, compared to the holdout technique, cross-validation takes more time to run and uses more computational resources.



2. Holdout

It's important to get an unbiased estimate of model performance. This is exactly what the holdout technique offers. To get this unbiased estimate, we test a model on data different from the data we trained it on. This technique divides a dataset into three subsets: training, validation, and test sets.

As mentioned earlier, that the training set helps the model make predictions and that the test set assesses the performance of the model. The validation set also helps to assess the performance of the model by providing an environment to fine-tune the parameters of the model. From this, we select the best performing model.

The holdout method is ideal when dealing with a very large dataset, it prevents model overfitting, and incurs lower computational costs.

When a function fits too tightly to a set of data points, an error known as overfitting occurs. As a result, a model performs poorly on unseen data. To detect overfitting, we could first split our dataset into training and test sets. We then monitor the performance of the model on both training data and test data.

If our model offers superior performance on the training set when compared to the test set, there's a good chance overfitting is present. For instance, a model might offer 90% accuracy on the training set yet give 50% on the test set.

Model evaluation metrics

a. Metrics for classification problems

Predictions for classification problems yield four types of outcomes: true positives, true negatives, false positives, and false negatives.

1. **Classification accuracy**

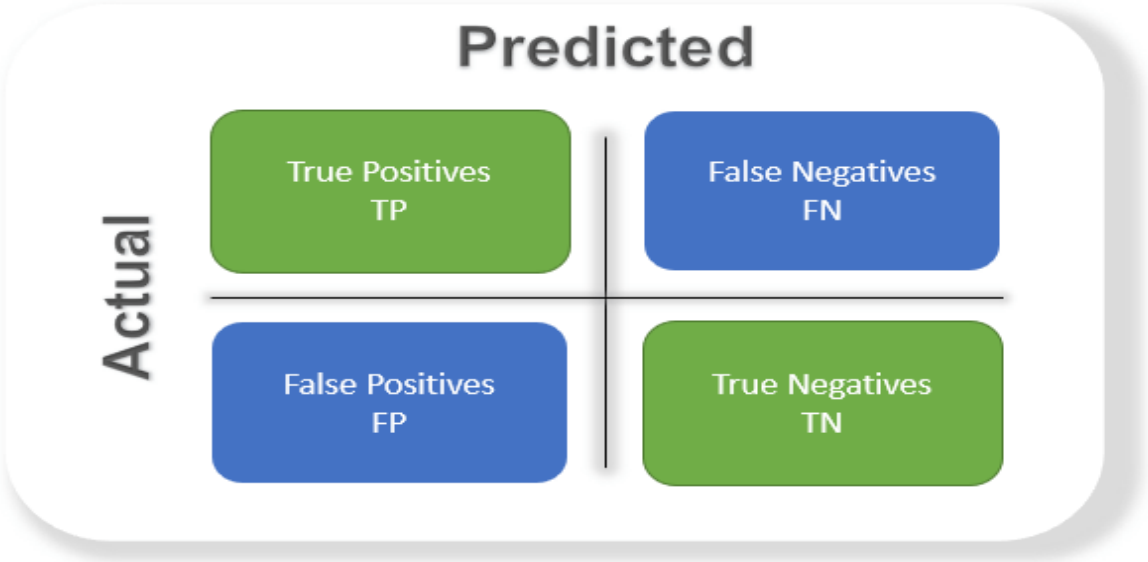
The most common evaluation metric for classification problems is accuracy. It's taken as the number of correct predictions against the total number of predictions made (or input samples).

Classification accuracy works best if the samples belonging to each class are equal in number. Consider a scenario with 97% samples from class X and 3% from class Y in a training set. A model can very easily achieve 97% training accuracy by predicting each training sample in class X.

Testing the same model on a test set with 55% samples of X and 45% samples of Y, the test accuracy is reduced to 55%. This is why classification accuracy is not a clear indicator of performance. It provides a false sense of attaining high levels of accuracy.

2. **Confusion matrix**

The confusion matrix forms the basis for the other types of classification metrics. It's a matrix that fully describes the performance of the model. A confusion matrix gives an in-depth breakdown of the correct and incorrect classifications of each class.



Confusion Matrix Terms:

- **True positives** are when you predict an observation belongs to a class and it actually does belong to that class.
- **True negatives** are when you predict an observation does not belong to a class and it actually does not belong to that class.
- **False positives** occur when you predict an observation belongs to a class when in reality it does not.
- **False negatives** occur when you predict an observation does not belong to a class when in fact it does.

The confusion matrix explained above is an example for the case of binary classification.

From this it's important to amplify true positives and true negatives. False positives and false negatives represent misclassification, that could be costly in real-world applications. Consider instances of misdiagnosis in a medical deployment of a model.

A model may wrongly predict that a healthy person has cancer. It may also classify someone who actually has cancer as cancer-free. Both these outcomes would have unpleasant consequences in terms of the well being of the patients after being diagnosed (or finding out about the misdiagnosis), treatment plans as well as expenses. Therefore it's important to minimize false negatives and false positives.

The green shapes in the image represent when the model makes the correct prediction. The blue ones represent scenarios where the model made the wrong predictions. The rows of the matrix represent the actual classes while the columns represent predicted classes.

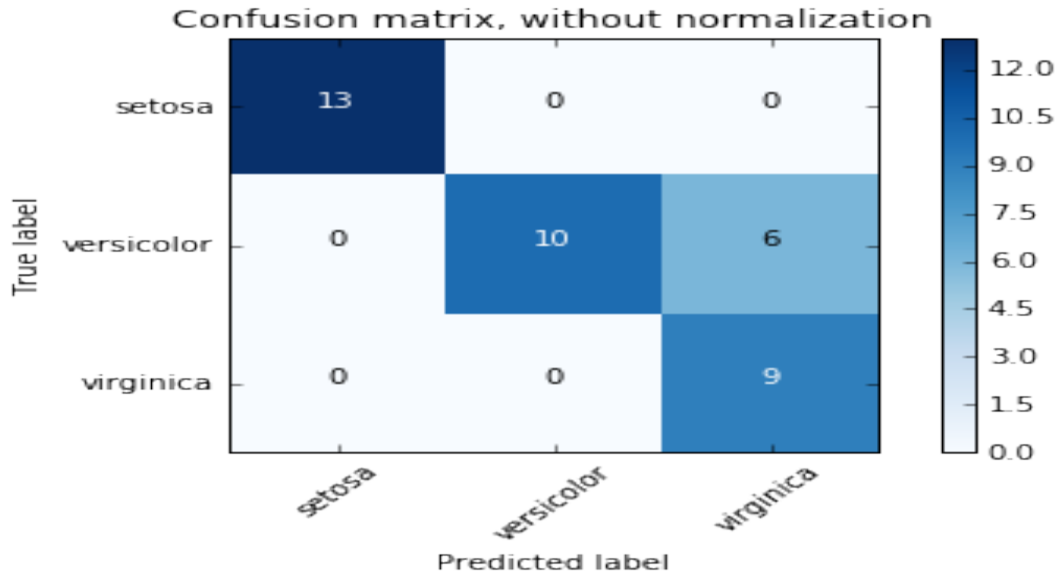
We can calculate accuracy from the confusion matrix. The accuracy is given by taking the average of the values in the "true" diagonal.

Accuracy = (True Positive + True Negative) / Total Sample

That translates to:

Accuracy = Total Number of Correct Predictions / Total Number of Observations

This can also extended to plot multi-class classification predictions like following example of classifying observations from the Iris flower dataset.



Accuracy

It is also useful to calculate the accuracy based on classifier prediction and actual value.

n=165	Predicted: NO	Predicted: YES	
Actual: NO	TN=50	FP=10	60
Actual: YES	FN=5	TP=100	105
	55	110	

Accuracy is a measure of how often, over all observations, the classifier is correct.

Accuracy is : ***(TP+TN)/total = (100+50)/(165) = 0.91.***

3. **Precision**

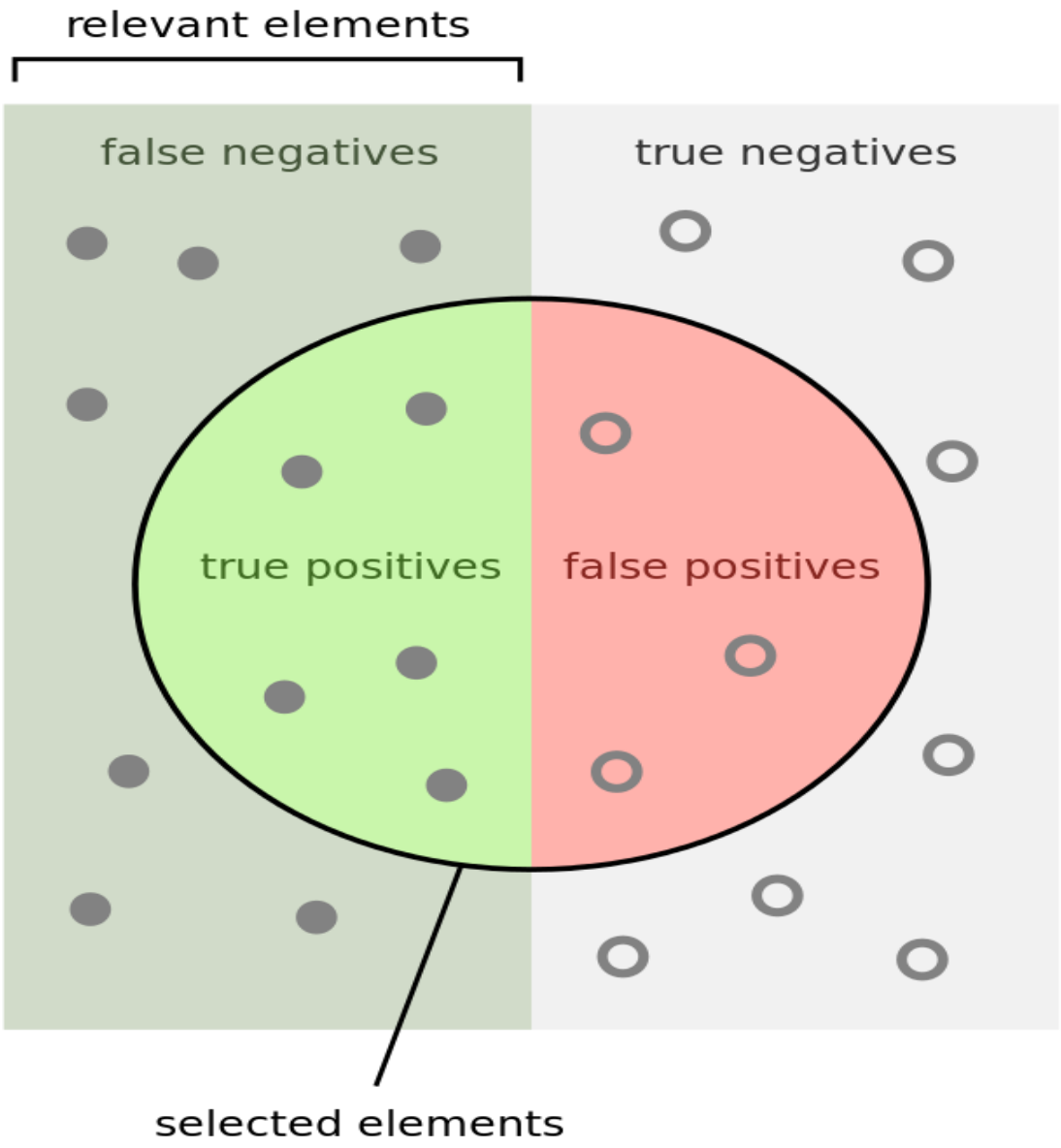
Precision refers to the number of true positives divided by the total positive results predicted by a classifier. That is, precision aims to understand what fraction of all positive predictions were actually correct.

Precision = True Positives / (True Positives + False Positives)

4. **Recall**

On the other hand, recall is the number of true positives divided by all the samples that should have been predicted as positive. Recall has the goal to perceive what fraction of actual positive predictions were identified accurately.

Recall = True Positives / (True Positives + False Negatives)



How many selected items are relevant?

$$\text{Precision} = \frac{\text{true positives}}{\text{selected elements}}$$

How many relevant items are selected?

$$\text{Recall} = \frac{\text{true positives}}{\text{relevant elements}}$$

5. **F-score**

F-score is a metric that incorporates both the precision and recall of a test to determine the score. F-score is also known as F-measure or F1 score.

In addition to robustness, the F-score shows us how precise a model is by letting us know how many correct classifications are made. The F-score ranges between 0 and 1. The higher the F-score, the greater the performance of the model.

$$F1 = 2 * \frac{1}{\frac{1}{precision} + \frac{1}{recall}}$$

b. Metrics for regression problems

Classification models deal with discrete data. The above metrics are ideal for classification tasks since they are concerned with whether a prediction is correct.

Regression models, on the other hand, deal with continuous data. Predictions are in a continuous range. This is the distinction between the metrics for classification and regression problems.

1. **Mean absolute error**

The mean absolute error represents the average of the absolute difference between the original and predicted values.

Mean absolute error provides the estimate of how far off the actual output the predictions were. However, since it's an absolute value, it does not indicate the direction of the error.

Mean absolute error is given by:

$$MeanAbsoluteError = \frac{1}{N}\sum_{j=1}^{N}|y_j - \hat{y}_j|$$

2. Mean squared error

The mean squared error is quite similar to the mean absolute error. However, mean squared error uses the average of the square of the difference between original and predicted values. Since this involves the squaring of the errors, larger errors are very notable.

Mean squared error is given by:

$$MeanSquaredError = \frac{1}{N}\sum_{j=1}^{N}(y_j - \hat{y}_j)^2$$

3. Root mean squared error

The root mean squared error (RMSE), computes the idealness of fit by calculating the square root of the average of squared differences between the predicted and actual values. It's a measure of the average error magnitude.

The root mean squared error is a form of normalized distance between the vectors of the observed and predicted values.

$$\mathbf{RMSD} = \sqrt{\frac{\sum_{i=1}^{N}(x_i - \hat{x}_i)^2}{N}}$$

$\mathbf{RMSD}$ = root-mean-square deviation
i = variable i
N = number of non-missing data points
x_i = actual observations time series
$\hat{x}_i$ = estimated time series

TEXT / REFERENCE BOOKS

1. Jiawei Han, Micheline Kamber and Jian Pei. Data Mining: Concepts and Techniques, 3rd Edition. ISBN 0123814790. 2011.
2. Andriy Burkov, The Hundred-Page Machine Learning Book, Publisher: Andriy Burkov, ISBN: 9781999579548, 1999579542, Edition1,2019
3. Andreas Muller, Introduction to Machine Learning with Python: A Guide for Data Scientists Paperback –1 January 2016

QUESTION BANK

Part-A			
Q.No	**Questions**	**Competence**	**BT Level**
1.	Define Machine Learning.	Remember	BTL 1
2.	List the types of Machine Learning? Explain with example	Understand	BTL 2
3.	List the types of Reinforcement Learning.	Understand	BTL 2
4.	Define curse of Dimensionality.	Remember	BTL 1
5.	Define Entropy and Information Gain in Decision Tree algorithm	Remember	BTL 1
6.	Differentiate between Regression and Classification Algorithm.	Understand	BTL 2
7.	Illustrate Outlier, Overfitting and Underfitting?	Understand	BTL 2
8.	List the approaches of dimensionality reduction?	Understand	BTL 2
9.	Define PCA.	Remember	BTL 1
10.	Interpret Perceptron and Multi-Layer Perceptron?	Understand	BTL 2
11.	List the types of Activation functions?	Understand	BTL 2
12.	List the common splitting percentages of training and testing?	Understand	BTL 2
13.	Define feature selection	Understand	BTL 2
14.	Mention the two components of Bayesian Network.	Understand	BTL 2
15.	Define Neural Network and mention its components	Understand	BTL 2
16.	Differentiate between Precision and Recall.	Analysis	BTL 4
17.	List the different methods in feature selection?	Understand	BTL 2
18.	How F-score is calculated?	Analysis	BTL 4
19.	List the types of ANN	Apply	BTL 3
20.	Define Confusion Matrix?	Understand	BTL 2

PART B			
Q.No	**Questions**	**Competence**	**BT Level**
1.	Explain the three types of Machine Learning.	Analysis	BTL 4
2.	Compute the principal component of following data : (2, 1), (3, 5), (4, 3), (5, 6), (6, 7), (7, 8).	Apply	BTL 3
3.	Explain Linear Regression with example.	Analysis	BTL 4
4.	Explain the following terms in Bayesian Network: 1. DAG 2. Components of Bayesian network 3. Joint Probability Distribution	Analysis	BTL 4
5.	Discuss about Decision tree? Explain its structure and steps involved in ID3 algorithm.	Analysis	BTL 4
6.	Explain the working of neural network with its architecture	Analysis	BTL 4
7.	Explain the need of Validation set with its working flow?	Analysis	BTL 4
8.	Explain the following Evaluation Metrics: 1. Classification Accuracy 2. Confusion Matrix 3. Mean absolute error 4. Mean squared error	Analysis	BTL 4
9.	Explain the methods to test and validate the machine learning model?	Analysis	BTL 4
10.	Explain about the Bayesian network with its building models?	Analysis	BTL 4

SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

SCHOOL OF COMPUTING
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

UNIT – V – Introduction to Python– SCSA3016

Data Structures-Functions-Numpy-Matplotlib-Pandas-Problems based on Computational Complexity-Simple Case Studies based on Python(Binary Search, Common elements in list),Hash tables, Dictionary

5.1. INTRODUCTION TO DATA SCIENCE WITH PYTHON

The main focus of businesses using big data was on building frameworks that can store a large amount of data. Then, frameworks like Hadoop were created, which helped in storing massive amounts of data.With the problem of storage solved, the focus then shifted to processing the data that is stored. This is where data science came in as the future for processing and analysing data. Now, data science has become an integral part of all the businesses that deal with large amounts of data. Companies today hire data scientists and professionals who take the data and turn it into a meaningful resource.

What is Data Science? Data science is all about finding and exploring data in the real world and using that knowledge to solve business problems. Some examples of data science are:

- Customer Prediction - System can be trained based on customer behavior patterns to predict the likelihood of a customer buying a product
- Service Planning - Restaurants can predict how many customers will visit on the weekend and plan their food inventory to handle the demand

Why Python? When it comes to data science, we need some sort of programming language or tool, like Python. Although there are other tools for data science, like R and SAS, we will focus on Python and how it is beneficial for data science in this article.

- Python as a programming language has become very popular in recent times. It has been used in data science, IoT, AI, and other technologies, which has added to its popularity.
- Python is used as a programming language for data science because it contains costly tools from a mathematical or statistical perspective. It is one of the significant reasons why data scientists around the world use Python. If you track the trends over the past few years, you will notice that Python has become the programming language of choice, particularly for data science.
- There are several other reasons why Python is one of the most used programming languages for data science, including:

    - Speed - Python is relatively faster than other programming languages
    - Availability - There are a significant number of packages available that other users have developed, which can be reused
    - Design goal - The syntax roles in Python are intuitive and easy to understand, thereby helping in building applications with a readable codebase.

Python has been used worldwide for different fields such as making websites, artificial intelligence and much more. But to make all of this possible, **data** plays a very important role which means that this data should be stored efficiently and the access to it must be timely. So how do you

achieve this? We use something called Data Structures. With that being said, let us go through the topics we will cover in **Data Structures** in Python.

- What is a Data Structure?
- Types of Data Structures in Python
- Built-in Data Structures
    - List
    - Dictionary
    - Tuple
    - Sets
- User-Defined Data Structures
    - Arrays vs. List
    - Stack
    - Queue
    - Trees
    - Linked Lists
    - Graphs
    - HashMaps

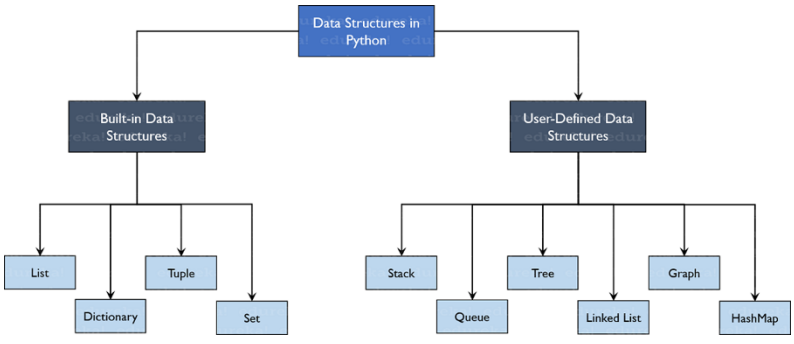
5.2. DATA STRUCTURE

Organizing, **managing** and **storing** data is important as it enables easier access and efficient modifications. Data Structures allows you to organize your data in such a way that enables you to store collections of data, relate them and perform operations on them accordingly.

Types of Data Structures in Python

Python has **implicit** support for Data Structures which enable you to store and access data. These structures are called List, Dictionary, Tuple and Set.

Python allows its users to create their own Data Structures enabling them to have **full control** over their functionality. The most prominent Data Structures are Stack, Queue, Tree, Linked List and so on which are also available to you in other programming languages. So now that you know what are the types available to you, why don't we move ahead to the Data Structures and implement them using Python.



5.2.1. Built-in Data Structures

As the name suggests, these Data Structures are built-in with Python which makes programming easier and helps programmers use them to obtain solutions faster.

Lists
Lists are used to store data of different data types in a sequential manner. There are addresses assigned to every element of the list, which is called as Index. The index value starts from 0 and goes on until the last element called the **positive index**. There is also **negative indexing** which starts from -1 enabling you to access elements from the last to first.

Dictionary
Dictionaries are used to store **key-value** pairs. To understand better, think of a phone directory where hundreds and thousands of names and their corresponding numbers have been added. Now the constant values here are Name and the Phone Numbers which are called as the keys. And the various names and phone numbers are the values that have been fed to the keys. If you access the values of the keys, you will obtain all the names and phone numbers. So that is what a key-value pair is. And in Python, this structure is stored using Dictionaries. Let us understand this better with an example program.

Tuple
Tuples are the same as lists are with the exception that the data once entered into the tuple cannot be changed no matter what. The only exception is when the data inside the tuple is mutable, only then the tuple data can be changed. The example program will help you understand better.

Sets
Sets are a collection of unordered elements that are unique. Meaning that even if the data is repeated more than one time, it would be entered into the set only once. It resembles the sets that you have learnt in arithmetic. The operations also are the same as is with the arithmetic sets. An example program would help you understand better.

5.2.3. User-Defined Data Structures

Arrays vs. Lists
Arrays and lists are the same structure with one difference. Lists allow heterogeneous data element storage whereas Arrays allow only homogenous elements to be stored within them.

Stack
Stacks are linear Data Structures which are based on the principle of Last-In-First-Out (LIFO) where data which is entered last will be the first to get accessed. It is built using the array structure and has operations namely, pushing (adding) elements, popping (deleting) elements and accessing elements only from one point in the stack called as the TOP. This TOP is the pointer to the current position of

the stack. Stacks are prominently used in applications such as Recursive Programming, reversing words, undo mechanisms in word editors and so forth.

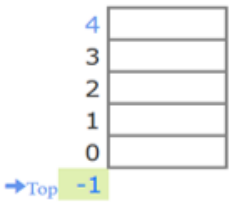


Figure 5.1: Stack

Queue

A queue is also a linear data structure which is based on the principle of First-In-First-Out (FIFO) where the data entered first will be accessed first. It is built using the array structure and has operations which can be performed from both ends of the Queue, that is, head-tail or front-back. Operations such as adding and deleting elements are called En-Queue and De-Queue and accessing the elements can be performed. Queues are used as Network Buffers for traffic congestion management, used in Operating Systems for Job Scheduling and many more.

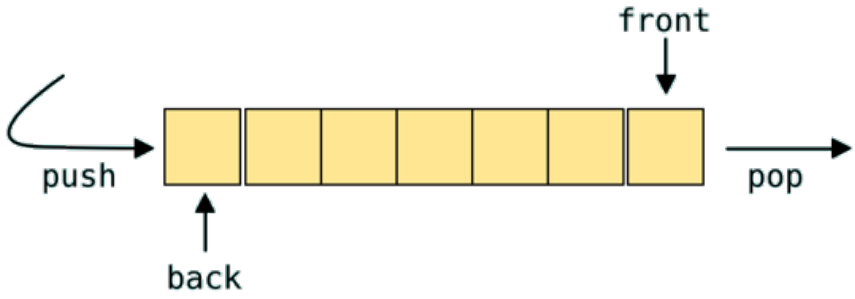


Figure 5.2: Queue

Tree

Trees are non-linear Data Structures which have root and nodes. The root is the node from where the data originates and the nodes are the other data points that are available to us. The node that precedes is the parent and the node after is called the child. There are levels a tree has to show the depth of information. The last nodes are called the leaves. Trees create a hierarchy which can be used in a lot of real-world applications such as the HTML pages use trees to distinguish which tag comes under which block. It is also efficient in searching purposes and much more.

Linked List

Linked lists are linear Data Structures which are not stored consequently but are linked with each other using pointers. The node of a linked list is composed of data and a pointer called next. These structures are most widely used in image viewing applications, music player applications and so forth.

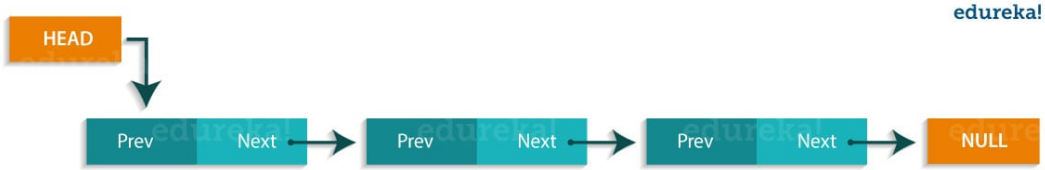


Figure 5.3: LinkedList

Graph

Graphs are used to store data collection of points called vertices (nodes) and edges (edges). Graphs can be called as the most accurate representation of a real-world map. They are used to find the various cost-to-distance between the various data points called as the nodes and hence find the least path. Many applications such as Google Maps, Uber, and many more use Graphs to find the least distance and increase profits in the best ways.

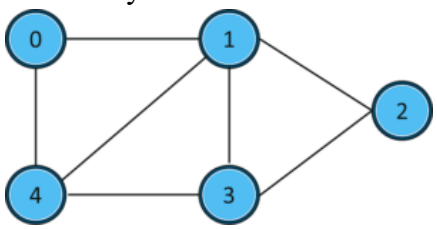


Figure 5.4: HashMaps

HashMaps are the same as what dictionaries are in Python. They can be used to implement applications such as phonebooks, populate data according to the lists and much more.

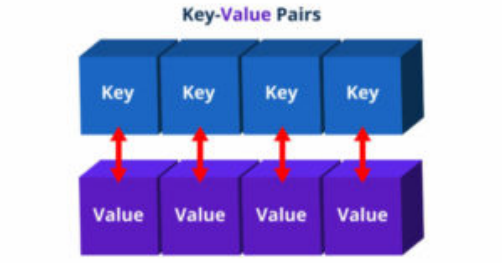


Figure 5.5: HashMaps

That wraps up all the prominent Data Structures in Python. I hope you have understood built-in as well as the user-defined Data Structures that we have in Python and why they are important.

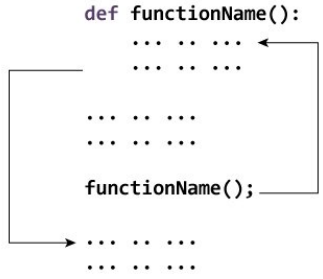
5.3. FUNCTION IN PYTHON

- A function is a block of code which only runs when it is called.
- You can pass data, known as parameters, into a function.
- A function can return data as a result.
- Python Functions is a block of related statements designed to perform a computational, logical, or evaluative task. The idea is to put some commonly or repeatedly done tasks together and make a function so that instead of writing the same code again and again for different inputs, we can do the function calls to reuse code contained in it over and over again.
- Functions can be both built-in or user-defined. It helps the program to be concise, non-repetitive, and organized.

Syntax

```
def functionname( parameters ):
   "function_docstring"
   function_suite
   return [expression]
```

How Function works in Python?



Types of Functions

Basically, we can divide functions into the following two types:

1. **Built-in functions** - Functions that are built into Python.
2. **User-defined functions** - Functions defined by the users themselves.

Creating a Function

In Python a function is defined using the def keyword:

Example

```
def my_function():
  print("Hello from a function")
```

Calling a Function

To call a function, use the function name followed by parenthesis:

Example

```
def my_function():
  print("Hello from a function")
my_function()
```

Function Arguments

You can call a function by using the following types of formal arguments −

1. Required arguments

2. Keyword arguments
3. Default arguments
4. Variable-length arguments

Arguments of a Function

Arguments are the values passed inside the parenthesis of the function. A function can have any number of arguments separated by a comma.

Example: Python Function with arguments
In this example, we will create a simple function to check whether the number passed as an argument to the function is even or odd.

```
# A simple Python function to check
# whether x is even or odd
def evenOdd(x):
    if (x % 2 == 0):
        print("even")
    else:
        print("odd")
 # Driver code to call the function
evenOdd(2)
evenOdd(3)
```

Output
even
odd

Types of Arguments
Python supports various types of arguments that can be passed at the time of the function call.
Required arguments
Required arguments are the arguments passed to a function in correct positional order. Here, the number of arguments in the function call should match exactly with the function definition.

```
# Function definition is here
def printme( str ):
   "This prints a passed string into this function"
   print str
   return;

# Now you can call printme function
printme()
```

When the above code is executed, it produces the following result −

```
Traceback (most recent call last):
  File "test.py", line 11, in <module>
    printme();
TypeError: printme() takes exactly 1 argument (0 given)
```

Keyword arguments

Keyword arguments are related to the function calls. When you use keyword arguments in a function call, the caller identifies the arguments by the parameter name.

```
# Function definition is here
def printinfo( name, age ):
  "This prints a passed info into this function"
  print "Name: ", name
  print "Age ", age
  return;

# Now you can call printinfo function
printinfo( age=50, name="miki" )
```

When the above code is executed, it produces the following result −

```
Name:  miki
Age  50
```

Default arguments

A default argument is a parameter that assumes a default value if a value is not provided in the function call for that argument. The following example illustrates Default arguments.

```
# Python program to demonstrate
# default arguments


def myFun(x, y=50):
    print("x: ", x)
    print("y: ", y)


# Driver code (We call myFun() with only
# argument)
myFun(10)
```

Output

```
('x: ', 10)
('y: ', 50)
```

5.4. PYTHON LIBRARIES FOR DATA ANALYSIS

Python is a simple programming language to learn, and there is some basic stuff that you can do with it, like adding, printing statements, and so on. However, if you want to perform data analysis, you need to import specific libraries. Some examples include:

- Pandas - Used for structured data operations
- NumPy - A powerful library that helps you create n-dimensional arrays
- SciPy - Provides scientific capabilities, like linear algebra and Fourier transform
- Matplotlib - Primarily used for visualization purposes
- Scikit-learn - Used to perform all machine learning activities

In addition to these, there are other libraries as well, like:

- Networks & I graph
- TensorFlow
- BeautifulSoup
- OS

5.5. NumPy

- **NumPy**, which stands for Numerical Python, is a library consisting of multidimensional array objects and a collection of routines for processing those arrays. Using NumPy, mathematical and logical operations on arrays can be performed.
- NumPy is a Python package. It stands for 'Numerical Python'. It is a library consisting of multidimensional array objects and a collection of routines for processing of array.
- **Numeric**, the ancestor of NumPy, was developed by Jim Hugunin. Another package Numarray was also developed, having some additional functionalities. In 2005, Travis Oliphant created NumPy package by incorporating the features of Numarray into Numeric package. There are many contributors to this open-source project.
- NumPy – A Replacement for MatLab
- NumPy is often used along with packages like **SciPy** (Scientific Python) and **Matplotlib** (plotting library). This combination is widely used as a replacement for MatLab, a popular platform for technical computing. However, Python alternative to MatLab is now seen as a more modern and complete programming language.
- It is open-source, which is an added advantage of NumPy.
- The most important object defined in NumPy is an N-dimensional array type called **ndarray**. It describes the collection of items of the same type. Items in the collection can be accessed using a zero-based index.
- Every item in a ndarray takes the same size as the block in the memory. Each element in ndarray is an object of the data-type object (called **dtype**).
- Any item extracted from ndarray object (by slicing) is represented by a Python object of one of array scalar types.
- NumPy is the fundamental package for scientific computing with Python. It contains:

- Powerful N-dimensional array objects
- Tools for integrating C/C++, and Fortran code
- It has useful linear algebra, Fourier transform, and random number capabilities

- **Operations using NumPy**
  - Using NumPy, a developer can perform the following operations −
    - Mathematical and logical operations on arrays.
    - Fourier transforms and routines for shape manipulation.
    - Operations related to linear algebra. NumPy has in-built functions for linear algebra and random number generation.
- An instance of ndarray class can be constructed by different array creation routines described later in the tutorial. The basic ndarray is created using an array function in NumPy as follows

numpy.array

- It creates a ndarray from any object exposing an array interface, or from any method that returns an array.

numpy.array(object, dtype = None, copy = True, order = None, subok = False, ndmin = 0)

- The **ndarray** object consists of a contiguous one-dimensional segment of computer memory, combined with an indexing scheme that maps each item to a location in the memory block. The memory block holds the elements in row-major order (C style) or a column-major order (FORTRAN or MatLab style).

The above constructor takes the following parameters −

Sr.No.	Parameter & Description
1	**object** Any object exposing the array interface method returns an array or any (nested) sequence.
2 3	**dtype** The desired data type of array, optional**copy**Optional. By default (true), the object is copied
4	**order** C (row-major) or F (column-major) or A (any) (default)
5	**subok** By default, returned array forced to be a base class array. If true, sub-classes passed through
6	**ndmin** Specifies minimum dimensions of the resultant array

Example 1

```
import numpy as np
a = np.array([1,2,3])
print(a)
```

The output is as follows –

[1, 2, 3]

The **ndarray** object consists of a contiguous one-dimensional segment of computer memory, combined with an indexing scheme that maps each item to a location in the memory block.

5.5.1. NumPy – Data Types

bool_
Boolean (True or False) stored as a byte

int_
Default integer type (same as C long; normally either int64 or int32)

intc
Identical to C int (normally int32 or int64)

intp
An integer used for indexing (same as C ssize_t; normally either int32 or int64)

int8
Byte (-128 to 127)

int16
Integer (-32768 to 32767)

float_
Shorthand for float64

float64
Double precision float: sign bit, 11 bits exponent, 52 bits mantissa

float64
Double precision float: sign bit, 11 bits exponent, 52 bits mantissa

complex_
Shorthand for complex128

complex64

Complex number, represented by two 32-bit floats (real and imaginary components)

complex128
Complex number, represented by two 64-bit floats (real and imaginary components)
NumPy numerical types are instances of dtype (data-type) objects, each having unique characteristics. The dtypes are available as np.bool_, np.float32, etc.

Data Type Objects (dtype)
A data type object describes the interpretation of a fixed block of memory corresponding to an array, depending on the following aspects −

- Type of data (integer, float or Python object)
- Size of data
- Byte order (little-endian or big-endian)
- In case of structured type, the names of fields, data type of each field and part of the memory block taken by each field.
- If the data type is a subarray, its shape and data type

The byte order is decided by prefixing '<' or '>' to the data type. '<' means that encoding is little-endian (least significant is stored in smallest address). '>' means that encoding is big-endian (a most significant byte is stored in smallest address).

A dtype object is constructed using the following syntax −

numpy.dtype(object, align, copy)

The parameters are −

- **Object** − To be converted to data type object
- **Align** − If true, adds padding to the field to make it similar to C-struct
- **Copy** − Makes a new copy of dtype object. If false, the result is a reference to builtin data type object

Example 1

```
# using array-scalar type
import numpy as np
dt = np.dtype(np.int32)
print(dt)
```

The output is as follows − int32

5.5.2. ndarray.shape
This array attribute returns a tuple consisting of array dimensions. It can also be used to resize the array.

Example 1

```
import numpy as np
```

```
a = np.array([[1,2,3],[4,5,6]])
print (a.shape)
```

The output is as follows −(2, 3)

Example 2

```
# this resizes the ndarray
import numpy as np
a = np.array([[1,2,3],[4,5,6]])
a.shape = (3,2)
print(a)
```

The output is as follows -[[1, 2][3, 4] [5, 6]]

5.5.3. ndarray.ndim

This array attribute returns the number of array dimensions.

Example 1

```
# an array of evenly spaced numbers
import numpy as np
a = np.arange(24)
print(a)
```

The output is as follows –

```
[0 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16 17 18 19 20 21 22 23]
```

Example 2

```
# this is one dimensional array
import numpy as np
a = np.arange(24)
a.ndim
# now reshape it
b = a.reshape(2,4,3)
print(b)
# b is having three dimensions
```

The output is as follows –

```
[[[ 0,  1,  2]
  [ 3,  4,  5]
  [ 6,  7,  8]
  [ 9, 10, 11]]
  [[12, 13, 14]
   [15, 16, 17]
   [18, 19, 20]
```

[21, 22, 23]]]

5.5.4. numpy.itemsize

This array attribute returns the length of each element of array in bytes.

Example 1

```
# dtype of array is int8 (1 byte)
import numpy as np
x = np.array([1,2,3,4,5], dtype = np.int8)
print (x.itemsize)
```

5.5.5. numpy.flags

The ndarray object has the following attributes. Its current values are returned by this function.

Sr.No.	Attribute & Description
1	**C_CONTIGUOUS (C)**The data is in a single, C-style contiguous segment
2	**F_CONTIGUOUS (F)**The data is in a single, Fortran-style contiguous segment
3	**OWNDATA (O)**The array owns the memory it uses or borrows it from another object
4	**WRITEABLE (W)**The data area can be written to. Setting this to False locks the data, making it read-only
5	**ALIGNED (A)**The data and all elements are aligned appropriately for the hardware
6	**UPDATEIFCOPY (U)**This array is a copy of some other array. When this array is deallocated, the base array will be updated with the contents of this array

Example

The following example shows the current values of flags.

```
import numpy as np
x = np.array([1,2,3,4,5])
print(x.flags)
```

The output is as follows −

```
C_CONTIGUOUS : True
F_CONTIGUOUS : True
OWNDATA : True
WRITEABLE : True
```

ALIGNED : True
UPDATEIFCOPY : False

5.5.6. NumPy – Array Creation Routines

A new **ndarray** object can be constructed by any of the following array creation routines or using a low-level ndarray constructor.

numpy.empty

It creates an uninitialized array of specified shape and dtype. It uses the following constructor −

numpy.empty(shape, dtype = float, order = 'C')

The constructor takes the following parameters.

Sr.No.	Parameter & Description
1	**Shape:** Shape of an empty array in int or tuple of int
2	**Dtype:** Desired output data type. Optional
3	**Order:** 'C' for C-style row-major array, 'F' for FORTRAN style column-

Example

The following code shows an example of an empty array.

```
import numpy as np
x = np.empty([3,2], dtype = int)
print(x)
```

The output is as follows −[[22649312 1701344351]
[1818321759 1885959276] [16779776 156368896]]

5.5.7. numpy.zeros

Returns a new array of specified size, filled with zeros.
numpy.zeros(shape, dtype = float, order = 'C')

Example 1

```
# array of five ones. Default dtype is float
import numpy as np
x = np.ones(5)
print(x)
```

The output is as follows −

[1. 1. 1. 1. 1.]

5.5.8. NumPy – Indexing & Slicing

Contents of ndarray object can be accessed and modified by indexing or slicing, just like Python's in-built container objects. items in ndarray object follows zero-based index. Three types of indexing methods are available − **field access, basic slicing** and **advanced indexing**.

- Basic slicing is an extension of Python's basic concept of slicing to n dimensions. A Python slice object is constructed by giving **start, stop**, and **step** parameters to the built-in **slice** function. This slice object is passed to the array to extract a part of array.

Example 1

```
import numpy as np
a = np.arange(10)
s = slice(2,7,2)
print (a[s])
```

Its output is as follows −
[2 4 6]

In the above example, an **ndarray** object is prepared by **arange()** function. Then a slice object is defined with start, stop, and step values 2, 7, and 2 respectively. When this slice object is passed to the ndarray, a part of it starting with index 2 up to 7 with a step of 2 is sliced.

5.5.9. NumPy – Advanced Indexing

It is possible to make a selection from ndarray that is a non-tuple sequence, ndarray object of integer or Boolean data type, or a tuple with at least one item being a sequence object. Advanced indexing always returns a copy of the data. As against this, the slicing only presents a view.

There are two types of advanced indexing − **Integer** and **Boolean**.

Integer Indexing

- This mechanism helps in selecting any arbitrary item in an array based on its N-dimensional index. Each integer array represents the number of indexes into that dimension. When the index consists of as many integer arrays as the dimensions of the target ndarray, it becomes straightforward.

- In the following example, one element of the specified column from each row of ndarray object is selected. Hence, the row index contains all row numbers, and the column index specifies the element to be selected.

Example 1

```
import numpy as np
x = np.array([[1, 2], [3, 4], [5, 6]])
y = x[[0,1,2], [0,1,0]]
print(y)
```

Its output would be as follows −

```
[1  4  5]
```

- The selection includes elements at (0,0), (1,1) and (2,0) from the first array.
- In the following example, elements placed at corners of a 4X3 array are selected. The row indices of selection are [0, 0] and [3,3] whereas the column indices are [0,2] and [0,2].
- Advanced and basic indexing can be combined by using one slice (:) or ellipsis (…) with an index array. The following example uses a slice for the advanced index for column. The result is the same when a slice is used for both. But advanced index results in copy and may have different memory layout.

Boolean Array Indexing

This type of advanced indexing is used when the resultant object is meant to be the result of Boolean operations, such as comparison operators.

Example 1

In this example, items greater than 5 are returned as a result of Boolean indexing.

```
import numpy as np
x = np.array([[ 0,  1,  2],[ 3,  4,  5],[ 6,  7,  8],[ 9, 10, 11]])
print ('Our array is:')
print(x)
print '\n'
# Now we will print the items greater than 5
print ('The items greater than 5 are:')
print (x[x > 5])
```

The output of this program would be −

Our array is:
[[0 1 2]
 [3 4 5]
 [6 7 8]
 [9 10 11]]
The items greater than 5 are:
[6 7 8 9 10 11]

5.5.10. NumPy – Broadcasting

The term **broadcasting** refers to the ability of NumPy to treat arrays of different shapes during arithmetic operations. Arithmetic operations on arrays are usually done on corresponding elements. If two arrays are of exactly the same shape, then these operations are smoothly performed.

Example 1

```
import numpy as np
a = np.array([1,2,3,4])
b = np.array([10,20,30,40])
c = a * b
print(c)
```

Its output is as follows −[10 40 90 160]

- If the dimensions of the two arrays are dissimilar, element-to-element operations are not possible. However, operations on arrays of non-similar shapes is still possible in NumPy, because of the broadcasting capability. The smaller array is **broadcast** to the size of the larger array so that they have compatible shapes.

5.5.11. NumPy – Iterating Over Array

NumPy package contains an iterator object **numpy.nditer**. It is an efficient multidimensional iterator object using which it is possible to iterate over an array. Each element of an array is visited using Python's standard Iterator interface.
Let us create a 3X4 array using arrange() function and iterate over it using **nditer**.

5.5.12. NumPy – Array Manipulation

Several routines are available in NumPy package for manipulation of elements in ndarray object. They can be classified into the following types −

Changing Shape

Sr.No.	Shape & Description

1	reshape:Gives a new shape to an array without changing its data
2	flat:A 1-D iterator over the array
3	flatten:Returns a copy of the array collapsed into one dimension
4	ravel:Returns a contiguous flattened array

Transpose Operations

Sr.No.	Operation & Description
1	transpose:Permutes the dimensions of an array
2	ndarray.T:Same as self.transpose()
3	rollaxis:Rolls the specified axis backwards
4	swapaxes:Interchanges the two axes of an array

Changing Dimensions

Sr.No.	Dimension & Description
1	broadcast:Produces an object that mimics broadcasting
2	broadcast_to:Broadcasts an array to a new shape
3	expand_dims:Expands the shape of an array
4	squeeze:Removes single-dimensional entries from the shape of an array

Joining Arrays

Sr.No.	Array & Description
1	concatenate:Joins a sequence of arrays along an existing axis
2	stack:Joins a sequence of arrays along a new axis
3	hstack:Stacks arrays in sequence horizontally (column wise)
4	vstack:Stacks arrays in sequence vertically (row wise)

Splitting Arrays

Sr.No.	Array & Description
1	split:Splits an array into multiple sub-arrays
2	hsplit:Splits an array into multiple sub-arrays horizontally (column-wise)
3	vsplit:Splits an array into multiple sub-arrays vertically (row-wise)

Adding / Removing Elements

Sr.No.	Element & Description
1	resizeReturns a new array with the specified shape
2	appendAppends the values to the end of an array
3	insertInserts the values along the given axis before the given indices
4	deleteReturns a new array with sub-arrays along an axis deleted
5	uniqueFinds the unique elements of an array

NumPy – Binary Operators

Following are the functions for bitwise operations available in NumPy package.

Sr.No.	Operation & Description
1	bitwise_and:Computes bitwise AND operation of array elements
2	bitwise_or:Computes bitwise OR operation of array elements
3	invert:Computes bitwise NOT
4	right_shift:Shifts bits of binary representation to the right

5.5.13. NumPy – Mathematical Functions

Quite understandably, NumPy contains a large number of various mathematical operations. NumPy provides standard trigonometric functions, functions for arithmetic operations, handling complex numbers, etc.

Trigonometric Functions

NumPy has standard trigonometric functions which return trigonometric ratios for a given angle in radians.

Example

```
import numpy as np
a = np.array([0,30,45,60,90])
print (‘Sine of different angles:’ )
# Convert to radians by multiplying with pi/180
Print(np.sin(a*np.pi/180))
print (‘\n’)
print(‘Cosine values for angles in array:’_
print(np.cos(a*np.pi/180))
print (‘\n’ )
print(‘Tangent values for given angles:’ )
print (np.tan(a*np.pi/180) )
```

Here is its output −

```
Sine of different angles:
[ 0.         0.5        0.70710678  0.8660254   1.        ]
Cosine values for angles in array:
[  1.00000000e+00   8.66025404e-01   7.07106781e-01   5.00000000e-01
   6.12323400e-17]
Tangent values for given angles:
[  0.00000000e+00   5.77350269e-01   1.00000000e+00   1.73205081e+00
```

1.63312394e+16]

arcsin, arcos, and **arctan** functions return the trigonometric inverse of sin, cos, and tan of the given angle. The result of these functions can be verified by **numpy.degrees() function** by converting radians to degrees.

Functions for Rounding

numpy.around()

This is a function that returns the value rounded to the desired precision. The function takes the following parameters.

numpy.around(a,decimals)

Where,

Sr.No.	Parameter & Description
1	**a:** Input data
2	**decimals:** The number of decimals to round to. Default is 0. If negative, the integer is rounded to position to the left of the decimal point

5.5.14. NumPy – Statistical Functions

NumPy has quite a few useful statistical functions for finding minimum, maximum, percentile standard deviation and variance, etc. from the given elements in the array. The functions are explained as follows −

numpy.amin() and numpy.amax()numpy.amin() and numpy.amax()

These functions return the minimum and the maximum from the elements in the given array along the specified axis.

Example

```
import numpy as np
a = np.array([[3,7,5],[8,4,3],[2,4,9]])
print ‘Our array is:’
print(a)
print (‘\n’ )
print ‘Applying amin() function:’
print(np.amin(a,1))
print (‘\n’ )
print ‘Applying amin() function again:’
print (np.amin(a,0))
print (‘\n’ )
```

```
print ‘Applying amax() function:’
print (np.amax(a) )
print (‘\n’)
print ‘Applying amax() function again:’
print(np.amax(a, axis = 0))
```

It will produce the following output −

```
Our array is:
[[3 7 5]
[8 4 3]
[2 4 9]]
Applying amin() function:
[3 3 2]
Applying amin() function again:
[2 4 3]
Applying amax() function:
9
Applying amax() function again:
[8 7 9]
```

5.5.15. numpy.ptp()

The **numpy.ptp()** function returns the range (maximum-minimum) of values along an axis.

```
import numpy as np
a = np.array([[3,7,5],[8,4,3],[2,4,9]])
print (‘Our array is:’)
print(a)
print (‘\n’)
print ‘Applying ptp() function:’
print np.ptp(a)
print (‘\n’)
print ‘Applying ptp() function along axis 1:’
print np.ptp(a, axis = 1)
print (‘\n’)
print(‘Applying ptp() function along axis 0:’)
print(np.ptp(a, axis = 0) )
numpy.percentile()
```

Percentile (or a centile) is a measure used in statistics indicating the value below which a given percentage of observations in a group of observations fall. The function **numpy.percentile()** takes the following arguments.

Where,

Sr.No.	Argument & Description
1	**a** Input array
2	**q** The percentile to compute must be between 0-100
3	**axis** The axis along which the percentile is to be calculated

A variety of sorting related functions are available in NumPy. These sorting functions implement different sorting algorithms, each of them characterized by the speed of execution, worst-case performance, the workspace required and the stability of algorithms. Following table shows the comparison of three sorting algorithms.

kind	speed	worst case	work space	stable
'quicksort'	1	O(n^2)	0	no
'mergesort'	2	O(n*log(n))	~n/2	yes
'heapsort'	3	O(n*log(n))	0	no

5.5.16. numpy.sort()

The sort() function returns a sorted copy of the input array. It has the following parameters −

numpy.sort(a, axis, kind, order)

Where,

Sr.No.	Parameter & Description
1	**a**Array to be sorted
2	**axis**The axis along which the array is to be sorted. If none, the array is flattened, sorting on the last axis
3	**kind**Default is quicksort
4	**order**If the array contains fields, the order of fields to be sorted

5.5.17. NumPy – Byte Swapping

We have seen that the data stored in the memory of a computer depends on which architecture the CPU uses. It may be little-endian (least significant is stored in the smallest address) or big-endian (most significant byte in the smallest address).

numpy.ndarray.byteswap()

The **numpy.ndarray.byteswap()** function toggles between the two representations: bigendian and little-endian.

5.5.18. NumPy – Copies & Views

While executing the functions, some of them return a copy of the input array, while some return the view. When the contents are physically stored in another location, it is called **Copy**. If on the other hand, a different view of the same memory content is provided, we call it as **View**.

5.5.19. No Copy

Simple assignments do not make the copy of array object. Instead, it uses the same id() of the original array to access it. The **id()** returns a universal identifier of Python object, similar to the pointer in C.

Furthermore, any changes in either gets reflected in the other. For example, the changing shape of one will change the shape of the other too.

5.5.20. View or Shallow Copy

NumPy has **ndarray.view()** method which is a new array object that looks at the same data of the original array. Unlike the earlier case, change in dimensions of the new array doesn't change dimensions of the original.

5.5.21. NumPy – Matrix Library

NumPy package contains a Matrix library **numpy.matlib**. This module has functions that return matrices instead of ndarray objects.

5.5.22. matlib.empty()

The **matlib.empty()** function returns a new matrix without initializing the entries. The function takes the following parameters.

numpy.matlib.empty(shape, dtype, order)

Where,

Sr.No.	Parameter & Description
1	**shapeint** or tuple of **int** defining the shape of the new matrix
2	**Dtype** Optional. Data type of the output
3	**order** C or F

Example

```
import numpy.matlib
import numpy as np
print(np.matlib.empty((2,2)))
# filled with random data
```

It will produce the following output −

```
[[ 2.12199579e-314,   4.24399158e-314]
 [ 4.24399158e-314,   2.12199579e-314]]
```

numpy.matlib.eye()

This function returns a matrix with 1 along the diagonal elements and the zeros elsewhere. The function takes the following parameters.

numpy.matlib.eye(n, M,k, dtype)

Where,

Sr.No.	Parameter & Description
1	**n** The number of rows in the resulting matrix
2	**M** The number of columns, defaults to n
3	**k** Index of diagonal
4	**dtype** Data type of the output

Example

```
import numpy.matlib
import numpy as np
print np.matlib.eye(n = 3, M = 4, k = 0, dtype = float)
```

It will produce the following output −

```
[[ 1.  0.  0.  0.]
 [ 0.  1.  0.  0.]
 [ 0.  0.  1.  0.]]
```

5.5.23. Numpy- Linear Algebra

NumPy package contains **numpy.linalg** module that provides all the functionality required for linear algebra. Some of the important functions in this module are described in the following table.

Sr.No.	Function & Description
1	dot Dot product of the two arrays
2	vdot Dot product of the two vectors
3	inner Inner product of the two arrays
4	matmul Matrix product of the two arrays
5	determinant Computes the determinant of the array
6	solve Solves the linear matrix equation
7	inv Finds the multiplicative inverse of the matrix

Addition and Subtraction

```
# importing numpy for matrix operations
import numpy

# initializing matrices
x = numpy.array([[1, 2], [4, 5]])
y = numpy.array([[7, 8], [9, 10]])

# using add() to add matrices
print ("The element wise addition of matrix is : ")
print (numpy.add(x,y))

# using subtract() to subtract matrices
print ("The element wise subtraction of matrix is : ")
print  (numpy.subtract(x,y))
```

Multiplication and Dot Product

```
# importing numpy for matrix operations
import numpy

# initializing matrices
x = numpy.array([[1, 2], [4, 5]])
y = numpy.array([[7, 8], [9, 10]])

# using multiply() to multiply matrices element wise
print ("The element wise multiplication of matrix is : ")
print (numpy.multiply(x,y))

# using dot() to multiply matrices
print ("The product of matrices is : ")
print (numpy.dot(x,y))
```

Transpose

“T” :- This argument is used to transpose the specified matrix.

```
# importing numpy for matrix operations
import numpy

# initializing matrices
x = numpy.array([[1, 2], [4, 5]])
```

```
# using "T" to transpose the matrix
print ("The transpose of given matrix is : ")
print (x.T)
```

```
The transpose of given matrix is :
[[1 4]
 [2 5]]
```

Determinant

Given the following Matrices:

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$$

$$det(A) = -2$$

$$B = \begin{bmatrix} -1 & 3 & 0 \\ 2 & 1 & -5 \\ 1 & 4 & -2 \end{bmatrix}$$

$$det(B) = -21$$

Python Solution:

```
-2.0000000000000004
-21.00000000000001
```

```
import numpy as np
import numpy.linalg as la

A = np.array([[1, 2],
              [3, 4]])

Adet = la.det(A)

print(Adet)

B = np.array([[-1, 3, 0],
              [2, 1, -5],
              [1, 4, -2]])

Bdet = la.det(B)

print(Bdet)
```

Inverse

```
# Import required package
import numpy as np
import numpy.linalg as la

# Taking a 3 * 3 matrix
A = np.array([[6, 1, 1],
              [4, -2, 5],
              [2, 8, 7]])

# Calculating the inverse of the matrix
print(la.inv(A))
```

Output:

```
[[ 0.17647059 -0.00326797 -0.02287582]
 [ 0.05882353 -0.13071895  0.08496732]
 [-0.11764706  0.1503268   0.05228758]]
```

Solve

Example-1

```
import numpy as np
m_list = [[4, 3], [-5, 9]]
A = np.array(m_list)
#To find the inverse of a matrix, the matrix is passed to the linalg.inv() method of the Numpy module
inv_A = np.linalg.inv(A)
print(inv_A)
#find the dot product between the inverse of matrix A, and the matrix B.
B = np.array([20, 26])
X = np.linalg.inv(A).dot(B)
print(X)
```

Output:

```
[2. 4.]
```

Here, 2 and 4 are the respective values for the unknowns x and y in *Equation 1*.

Example 2

```
A = np.array([[4, 3, 2], [-2, 2, 3], [3, -5, 2]])
B = np.array([25, -10, -4])
X = np.linalg.inv(A).dot(B)
print(X)
```

Output:

```
[ 5.  3. -2.]
```

np.arange

```
import  numpy as np
#create an array
arr = np.arange(1,10).reshape(3,3)
#finding the Eigenvalue and Eigenvectors of arr
np.linalg.eig(arr)
```

5.6. Matplotlib

- Matplotlib is an amazing visualization library in Python for 2D plots of arrays. Matplotlib is a multi-platform data visualization library built on NumPy arrays and designed to work with the broader SciPy stack. It was introduced by John Hunter in the year 2002.
- One of the greatest benefits of visualization is that it allows us visual access to huge amounts of data in easily digestible visuals. Matplotlib consists of several plots like line, bar, scatter, histogram etc.

Installation :

Windows, Linux and macOS distributions have matplotlib and most of its dependencies as wheel packages. Run the following command to install matplotlib package :

python -mpip install -U matplotlib

Importing matplotlib :

from matplotlib import pyplot as plt
or
import matplotlib.pyplot as plt

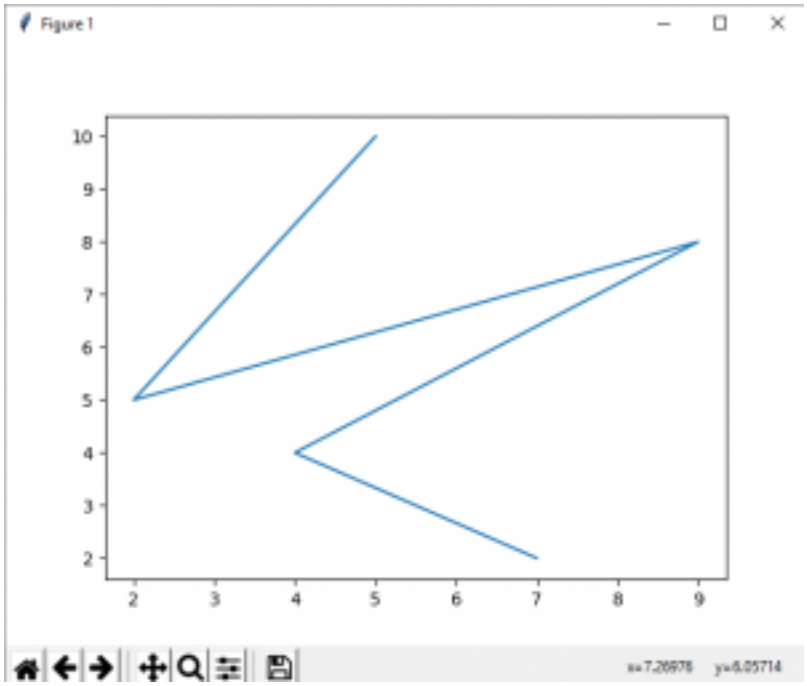
Basic plots in Matplotlib :

Matplotlib comes with a wide variety of plots. Plots helps to understand trends, patterns, and to make correlations. They're typically instruments for reasoning about quantitative information. Some of the sample plots are covered here.

1. Line plot :

```
# importing matplotlib module
from matplotlib import pyplot as plt

# x-axis values
x = [5, 2, 9, 4, 7]

# Y-axis values
y = [10, 5, 8, 4, 2]

# Function to plot
plt.plot(x,y)

# function to show the plot
plt.show()
```

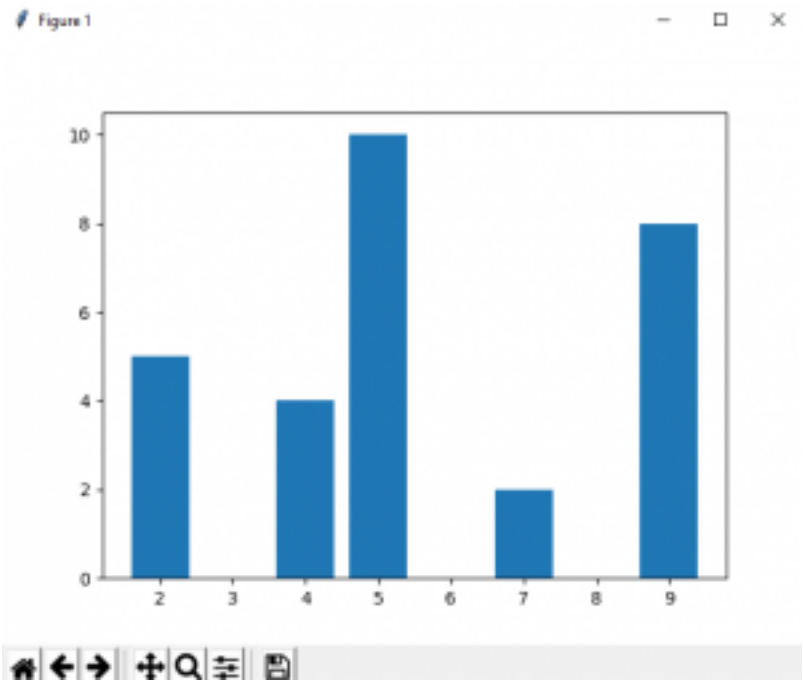
Output:



2. Bar plot :

```
# importing matplotlib module
from matplotlib import pyplot as plt

# x-axis values
x = [5, 2, 9, 4, 7]

# Y-axis values
y = [10, 5, 8, 4, 2]

# Function to plot the bar
plt.bar(x,y)

# function to show the plot
plt.show()
```

Output:



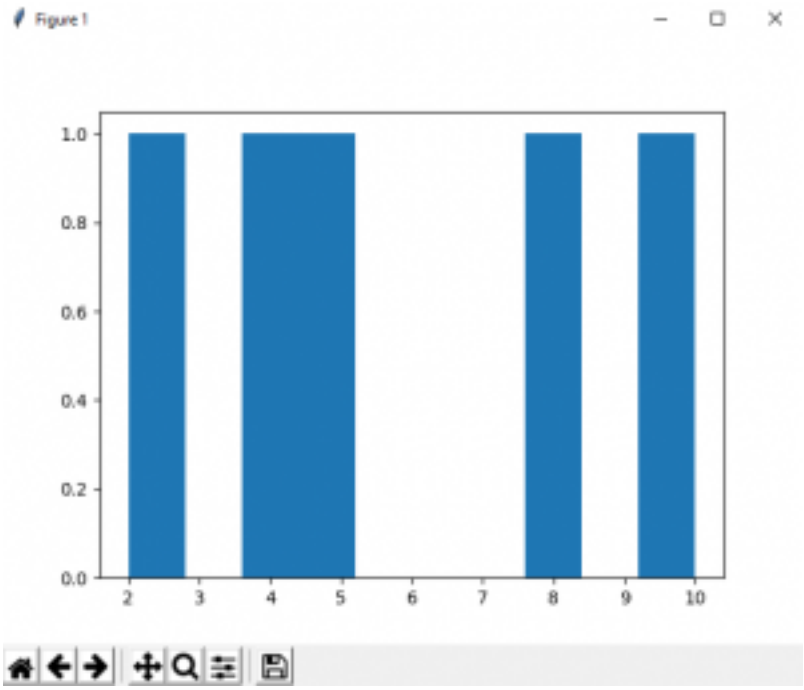
3. Histogram :

A histogram is a graph showing frequency distributions.

It is a graph showing the number of observations within each given interval.

```
# importing matplotlib module
from matplotlib import pyplot as plt

# Y-axis values
y = [10, 5, 8, 4, 2]

# Function to plot histogram
plt.hist(y)

# Function to show the plot
plt.show()
```

Output:

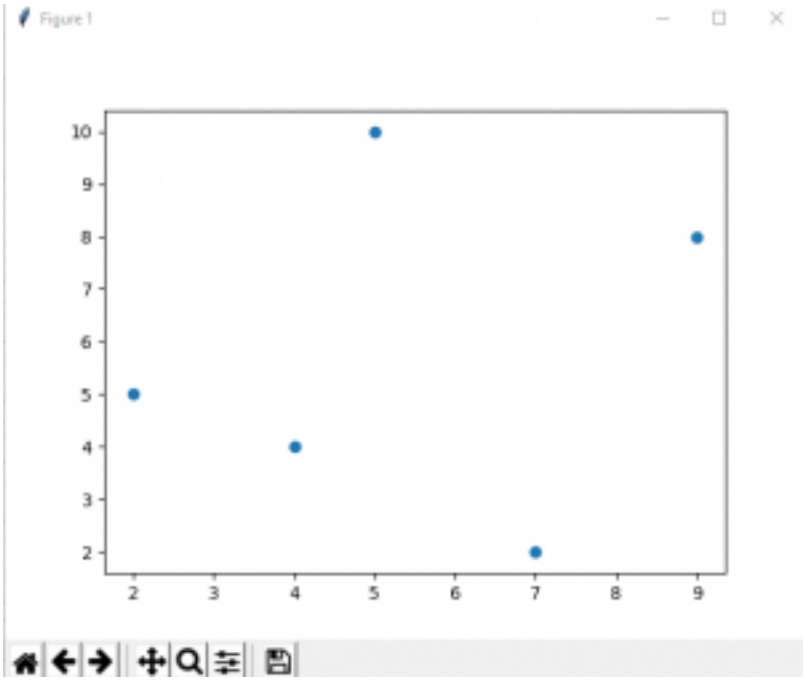


4. **Scatter Plot :**
   With Pyplot, you can use the scatter() function to draw a scatter plot.

   The scatter() function plots one dot for each observation. It needs two arrays of the same length, one for the values of the x-axis, and one for values on the y-axis:

```
# importing matplotlib module
from matplotlib import pyplot as plt

# x-axis values
x = [5, 2, 9, 4, 7]

# Y-axis values
y = [10, 5, 8, 4, 2]

# Function to plot scatter
plt.scatter(x, y)

# function to show the plot
plt.show()
```

Output:



Creating Pie Charts
With Pyplot, you can use the pie() function to draw pie charts:

```
import matplotlib.pyplot as plt
import numpy as np

y = np.array([35, 25, 25, 15])

plt.pie(y)
plt.show()
```

5.7. Pandas

- **Pandas** is an open-source library that is built on top of NumPy library. It is a Python package that offers various data structures and operations for manipulating numerical data and time series. It is mainly popular for importing and analyzing data much easier. Pandas is fast and it has high-performance & productivity for users.

What is Python Pandas?

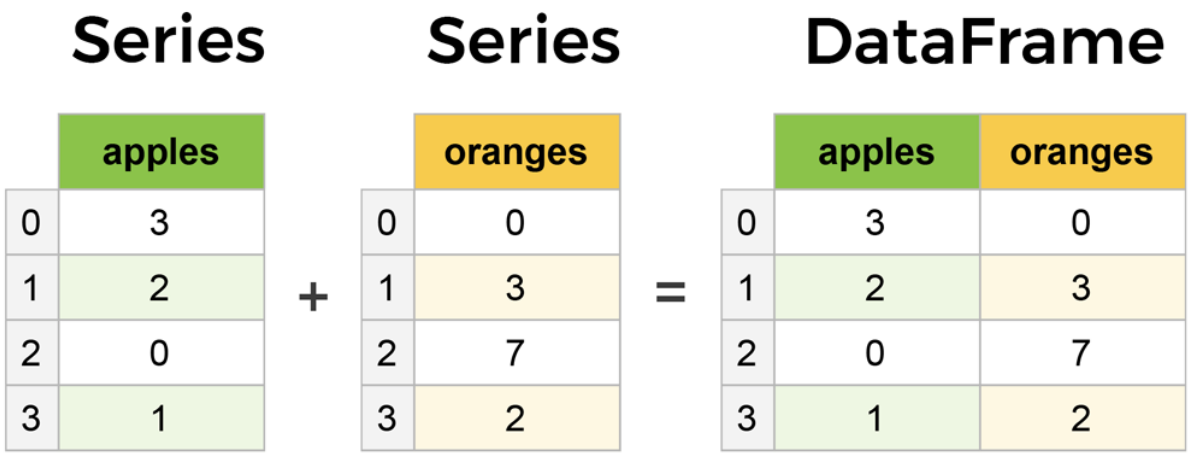
Pandas is used for data manipulation, analysis and cleaning. Python pandas is well suited for different kinds of data, such as:

- Tabular data with heterogeneously-typed columns
- Ordered and unordered time series data
- Arbitrary matrix data with row & column labels
- Unlabelled data
- Any other form of observational or statistical data sets

Python Pandas Data Structure

The primary two components of pandas are the Series and DataFrame.

A Series is essentially a column, and a DataFrame is a multi-dimensional table made up of a collection of Series.



5.7.1. Series

Create a simple Pandas Series from a list:

```
import pandas as pd

a = [1, 7, 2]

myvar = pd.Series(a)
```

```
print(myvar)
```

5.7.2. What is a DataFrame?

A Pandas DataFrame is a 2 dimensional data structure, like a 2 dimensional array, or a table with rows and columns.

It is a widely used data structure of pandas and works with a two-dimensional array with labeled axes (rows and columns). DataFrame is defined as a standard way to store data and has two different indexes, i.e., row index and column index. It consists of the following properties:

- The columns can be heterogeneous types like int, bool, and so on.
- It can be seen as a dictionary of Series structure where both the rows and columns are indexed. It is denoted as "columns" in case of columns and "index" in case of rows.

Example
Create a simple Pandas DataFrame:

```
import pandas as pd

data = {
  "calories": [420, 380, 390],
  "duration": [50, 40, 45]
}

#load data into a DataFrame object:
df = pd.DataFrame(data)

print(df)
```

Result

```
   calories  duration
0       420        50
1       380        40
2       390        45
```

Create a DataFrame using List:

We can easily create a DataFrame in Pandas using list.

```
import pandas as pd
# a list of strings
x = ['Python', 'Pandas']
```

```
# Calling DataFrame constructor on list
df = pd.DataFrame(x)
print(df)
```

Output

```
   0
0  Python
1  Pandas
```

Create an empty DataFrame

The below code shows how to create an empty DataFrame in Pandas:

```
# importing the pandas library
import pandas as pd
df = pd.DataFrame()
print (df)
```

Output

```
Empty DataFrame
Columns: []
Index: []
```

Create a DataFrame from Dict of ndarrays/ Lists

```
# importing the pandas library
import pandas as pd
info = {'ID' :[101, 102, 103],'Department' :['B.Sc','B.Tech','M.Tech',]}
df = pd.DataFrame(info)
print (df)
```

Output

```
    ID     Department
0   101       B.Sc
1   102       B.Tech
2   103       M.Tech
```

Create a DataFrame from Dict of Series:

```
# importing the pandas library
import pandas as pd

info = {'one' : pd.Series([1, 2, 3, 4, 5, 6], index=['a', 'b', 'c', 'd', 'e', 'f']),
        'two' : pd.Series([1, 2, 3, 4, 5, 6, 7, 8], index=['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h'

d1 = pd.DataFrame(info)
print (d1)
```

Output

	one	two
a	1.0	1
b	2.0	2
c	3.0	3
d	4.0	4
e	5.0	5
f	6.0	6
g	NaN	7
h	NaN	8

Column Selection

We can select any column from the DataFrame. Here is the code that demonstrates how to select a column from the DataFrame.

```
# importing the pandas library
import pandas as pd
info = {'one' : pd.Series([1, 2, 3, 4, 5, 6], index=['a', 'b', 'c', 'd', 'e', 'f']),
        'two' : pd.Series([1, 2, 3, 4, 5, 6, 7, 8], index=['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h'}
d1 = pd.DataFrame(info)
print (d1 ['one'])
```

Output

```
a    1.0
b    2.0
c    3.0
d    4.0
e    5.0
f    6.0
g    NaN
h    NaN
Name: one, dtype: float64
```

Column Addition

We can also add any new column to an existing DataFrame. The below code demonstrates how to add any new column to an existing DataFrame:

```
# importing the pandas library
import pandas as pd
info = {'one' : pd.Series([1, 2, 3, 4, 5], index=['a', 'b', 'c', 'd', 'e']),
  'two' : pd.Series([1, 2, 3, 4, 5, 6], index=['a', 'b', 'c', 'd', 'e', 'f'])}
df = pd.DataFrame(info)
# Add a new column to an existing DataFrame object
print ("Add new column by passing series")
df['three']=pd.Series([20,40,60],index=['a','b','c'])
print (df)

print ("Add new column using existing DataFrame columns")
df['four']=df['one']+df['three']

print (df)
```

Output

```
Add new column by passing series
   one  two  three
a  1.0   1   20.0
b  2.0   2   40.0
c  3.0   3   60.0
d  4.0   4    NaN
e  5.0   5    NaN
f  NaN   6    NaN

Add new column using existing DataFrame columns
   one  two  three  four
a  1.0   1   20.0  21.0
b  2.0   2   40.0  42.0
c  3.0   3   60.0  63.0
d  4.0   4    NaN   NaN
e  5.0   5    NaN   NaN
f  NaN   6    NaN   NaN
```

Column Deletion:

We can also delete any column from the existing DataFrame. This code helps to demonstrate how the column can be deleted from an existing DataFrame:

```
# importing the pandas library
import pandas as pd
```

```
info = {'one' : pd.Series([1, 2], index= ['a', 'b']),
   'two' : pd.Series([1, 2, 3], index=['a', 'b', 'c'])}

df = pd.DataFrame(info)
print ("The DataFrame:")
print (df)

# using del function
print ("Delete the first column:")
del df['one']
print (df)
# using pop function
print ("Delete the another column:")
df.pop('two')
print (df)
```

Output

```
The DataFrame:
     one    two
a     1.0     1
b     2.0     2
c     NaN     3

Delete the first column:
    two
a     1
b     2
c     3

Delete the another column:
Empty DataFrame
Columns: []
Index: [a, b, c]
```

Row Selection, Addition, and Deletion

Row Selection:

We can easily select, add, or delete any row at anytime. First of all, we will understand the row selection. Let's see how we can select a row using different ways that are as follows:

Selection by Label:

We can select any row by passing the row label to a **loc** function.

```
# importing the pandas library
import pandas as pd

info = {'one' : pd.Series([1, 2, 3, 4, 5], index=['a', 'b', 'c', 'd', 'e']),
  'two' : pd.Series([1, 2, 3, 4, 5, 6], index=['a', 'b', 'c', 'd', 'e', 'f'])}

df = pd.DataFrame(info)
print (df.loc['b'])
```

Output

```
one    2.0
two    2.0
Name: b, dtype: float64
```

Selection by integer location:

The rows can also be selected by passing the integer location to an **iloc**function.

```
# importing the pandas library
import pandas as pd
info = {'one' : pd.Series([1, 2, 3, 4, 5], index=['a', 'b', 'c', 'd', 'e']),
  'two' : pd.Series([1, 2, 3, 4, 5, 6], index=['a', 'b', 'c', 'd', 'e', 'f'])}
df = pd.DataFrame(info)
print (df.iloc[3])
```

Output

```
one    4.0
two    4.0
Name: d, dtype: float64
```

Slice Rows

It is another method to select multiple rows using **':'** operator.

```
# importing the pandas library
import pandas as pd
info = {'one' : pd.Series([1, 2, 3, 4, 5], index=['a', 'b', 'c', 'd', 'e']),
  'two' : pd.Series([1, 2, 3, 4, 5, 6], index=['a', 'b', 'c', 'd', 'e', 'f'])}
df = pd.DataFrame(info)
print (df[2:5])
```

Output

```
   one  two
c  3.0   3
d  4.0   4
e  5.0   5
```

Addition of rows:

We can easily add new rows to the DataFrame using **append** function. It add the new rows at the end.

```
# importing the pandas library
import pandas as pd
d = pd.DataFrame([[7, 8], [9, 10]], columns = ['x','y'])
d2 = pd.DataFrame([[11, 12], [13, 14]], columns = ['x','y'])
d = d.append(d2)
print (d)
```

Output

```
   x   y
0  7   8
1  9   10
0  11  12
1  13  14
```

Deletion of rows:

We can delete or drop any rows from a DataFrame using the **index** label. If in case, the label is duplicate then multiple rows will be deleted.

```
# importing the pandas library
import pandas as pd

a_info = pd.DataFrame([[4, 5], [6, 7]], columns = ['x','y'])
b_info = pd.DataFrame([[8, 9], [10, 11]], columns = ['x','y'])
a_info = a_info.append(b_info)
# Drop rows with label 0
a_info = a_info.drop(0)
```

Output

```
x   y
1   6   7
1   10  11
```

5.7.3. Pandas Read CSV

A simple way to store big data sets is to use CSV files (comma separated files).

CSV files contains plain text and is a well know format that can be read by everyone including Pandas.

Load the CSV into a DataFrame:

```
import pandas as pd

df = pd.read_csv('data.csv')

print(df.to_string())
```

Print the DataFrame without the to_string() method:

```
import pandas as pd

df = pd.read_csv('data.csv')

print(df)
```

Viewing the Data

One of the most used method for getting a quick overview of the DataFrame, is the head() method.

The head() method returns the headers and a specified number of rows, starting from the top.

Example

```
import pandas as pd

df = pd.read_csv('data.csv')

print(df.head(10))
```

There is also a tail() method for viewing the *last* rows of the DataFrame.

The tail() method returns the headers and a specified number of rows, starting from the bottom.

Example
Print the last 5 rows of the DataFrame:

```
print(df.tail())
```

Info About the Data

The DataFrames object has a method called info(), that gives you more information about the data set.

Example: Print information about the data:

```
print(df.info())
```

5.8. PROBLEM BASED ON COMPUTATIONAL COMPLEXITY

Computational Complexity

- Computational complexity is a field from computer science which analyzes algorithms based on the amount resources required for running it. The amount of required resources varies based on the input size, so the complexity is generally expressed as a function of n, where n is the size of the input.
- It is important to note that when analyzing an algorithm we can consider the time complexity and space complexity. The space complexity is basically the amount of memory space required to solve a problem in relation to the input size. Even though the space complexity is important when analyzing an algorithm, in this story we will focus only on the time complexity.

Time Complexity in Python Now-a-days, for one problem we can write the solution in n number of ways, but, how can we decide which type is better. We can use different types of algorithms to solve one problem. We need to compare these algorithms and have to choose the best one to solve the problem.

What is Time Complexity?

The amount of time it takes to run the program and perform the functions in it is known as **Time Complexity**. By using Time Complexity we can determine whether the program is efficient or we have to use another algorithm which take less time compared to the other one. Reducing Time Complexity of an algorithm is often difficult in Data Science, rather than difficult we can say its a bigger challenge.

We will tell the time complexity of a program by calculating the time taken to run the algorithm in the **worst-case** scenario.

When analyzing the time complexity of an algorithm we may find three cases: **best-case, average-case and worst-case**.

Example: Suppose we have the following unsorted list [1, 5, 3, 9, 2, 4, 6, 7, 8] and we need to find the index of a value in this list using linear search.

best-case: this is the complexity of solving the problem for the best input. In our example, the best case would be to search for the value 1. Since this is the first value of the list, it would be found in the first iteration.

average-case: this is the average complexity of solving the problem. This complexity is defined with respect to the distribution of the values in the input data. Maybe this is not the best example but, based on our sample, we could say that the average-case would be when we're searching for some value in the "middle" of the list, for example, the value 2.

worst-case: this is the complexity of solving the problem for the worst input of size n. In our example, the worst-case would be to search for the value 8, which is the last element from the list.

To quantify the Time Complexity, **Big-O** notation is used.

5.8.1. Big-O Notation

Big-O notation, sometimes called "asymptotic notation", **is a mathematical notation that describes the limiting behavior of a function** when the argument tends towards a particular value or infinity**.**

Big-O notation is used to classify algorithms according to how their run time or space requirements grow as the input size grows. The letter O is used because the growth rate of a function is also referred to as the **order of the function** or **order of the program**. We will always refer order of the function in its worst-case.

A list of some common asymptotic notations is mentioned below,

Complexity Class	**Name**
O(1)	constant
O(logn)	logarithmic
O(n)	linear
O(n log n)	Linear logarithmic
$O(n^2)$	quadratic
$O(n^3)$	cubic
O(2n)	exponential

Example 1: Linear Search- O(n)

- A **linear search** is the most basic kind of search that is performed. A linear or sequential search, is done when you inspect each item in a list one by one from one end to the other to find a match for what you are searching for.
- Let's see the example I stated above once again to understand the Linear Search
- We have a list which consists of integers and we have to check whether the number given by the user is present in that list or not.

  l = [1,2,3,6,4,9,10,12]
  k = 12

The simple code for this is

```
l = [1,2,3,6,4,9,10,12]
k = 12
for i in range(0, len(l)):
  if l[i] == k:
    print("Yes")
    break
```

- Here, the worst-case for this algorithm is to check the number which is present in the last element of the given list. So, if we go by the above program, first it'll start with index 0 and check whether that element in the list is equal to k or not, i.e, one operation and we have to check for every element in the list for worst-case scenario.
- The Time Complexity of the above program is **O(n)**.

Example 2: O(n^2)

The complexity of an algorithm is said to be quadratic when the steps required to execute an algorithm are a quadratic function of the number of items in the input. Quadratic complexity is denoted as O(n^2). Take a look at the following example to see a function with quadratic complexity:

```
def quadratic_algo(items):
    for item in items:
        for item2 in items:
            print(item, ' ' ,item)

quadratic_algo([4, 5, 6, 8])
```

In this example, we have an outer loop that iterates through all the items in the input list and then a nested inner loop, which again iterates through all the items in the input list. The total number of steps performed is n * n, where n is the number of items in the input array.

Example 3: Merge Sort - O(n*logn).

```
def mergeSort(alist):
    print("Splitting ",alist)
    if len(alist)>1:
        mid = len(alist)//2
        lefthalf = alist[:mid]
        righthalf = alist[mid:]

        mergeSort(lefthalf)
        mergeSort(righthalf)

        i=0
        j=0
        k=0
        while i < len(lefthalf) and j < len(righthalf):
            if lefthalf[i] <= righthalf[j]:
                alist[k]=lefthalf[i]
                i=i+1
            else:
                alist[k]=righthalf[j]
                j=j+1
            k=k+1

        while i < len(lefthalf):
            alist[k]=lefthalf[i]
            i=i+1
            k=k+1

        while j < len(righthalf):
            alist[k]=righthalf[j]
            j=j+1
            k=k+1

alist = input('Enter the list of numbers: ').split()
alist = [int(x)for x in alist]
mergeSort(alist)
print('Sorted list: ', end='')
print(alist)
```

Output:

```
Enter the list of numbers: 56 48 10 2 40

Sorted list: [2, 10, 40, 48, 56]
```

The Time Complexity for the above program is **O(n*logn)**.

Example 4: Insertion sort- $O(n^2)$

```
def insertionSort(nlist):
  for index in range(1,len(nlist)):
    currentvalue = nlist[index]
    position = index
    while position>0 and nlist[position-1]>currentvalue:
       nlist[position]=nlist[position-1]
       position = position-1
    nlist[position]=currentvalue

nlist = input('Enter the list of numbers: ').split()
nlist = [int(x)for x in nlist]
insertionSort(nlist)
print('Sorted list: ', end='')
print(nlist)
```

Output:
Enter the list of numbers: 4 5 6 3 1
Sorted list: [1, 3, 4, 5, 6]

The time complexity of insertion sort is $O(n^2)$

5.9. Simple Case Studies based on Python

Binary Search

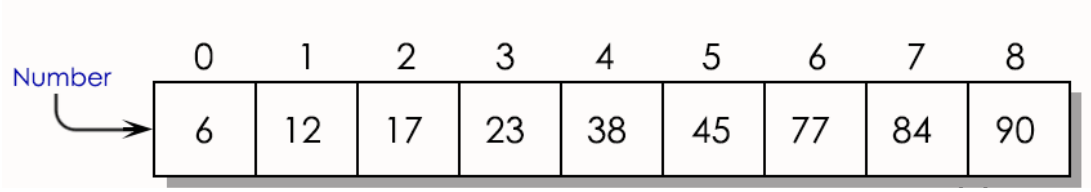
- Binary search is a searching algorithm that works efficiently with a sorted list.
- If a list is already sorted, then the search for an element in the list can be made faster by using 'divide and conquer' technique.
- The list is divided into two halves separated by the middle element.
- The binary search follows the following steps:

  Step 1: The middle element is tested for the required element. If found, then its position is reported else the following test is made.

  Step 2: If search element 'val'< 'middle' element, search the left half of the list, else search the right half of the list.

  Step 3: Repeat step 1 and 2 on the selected half until the entry is found otherwise report failure.

This search is called binary because in each iteration, the given list is divided into two parts. Then the search becomes limited to half the size of the list to be searched.

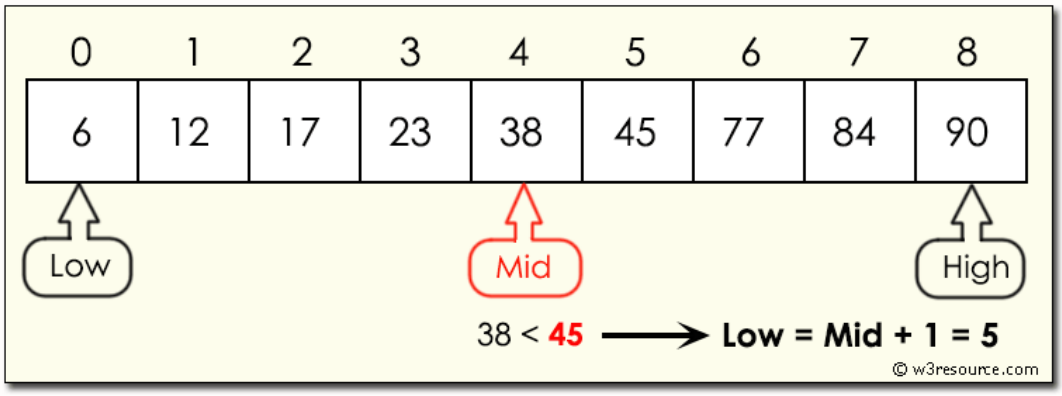


Step 1:

	Low	High	Mid
#1	0	8	4

Search (45)

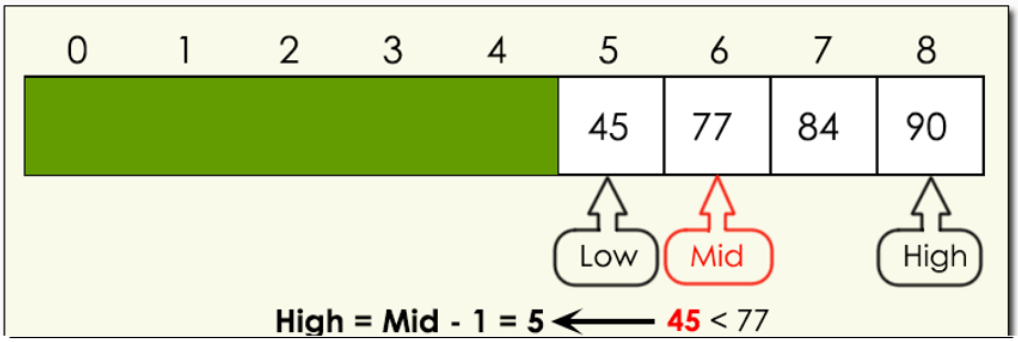
$$mid = \left[\frac{low + high}{2} \right]$$

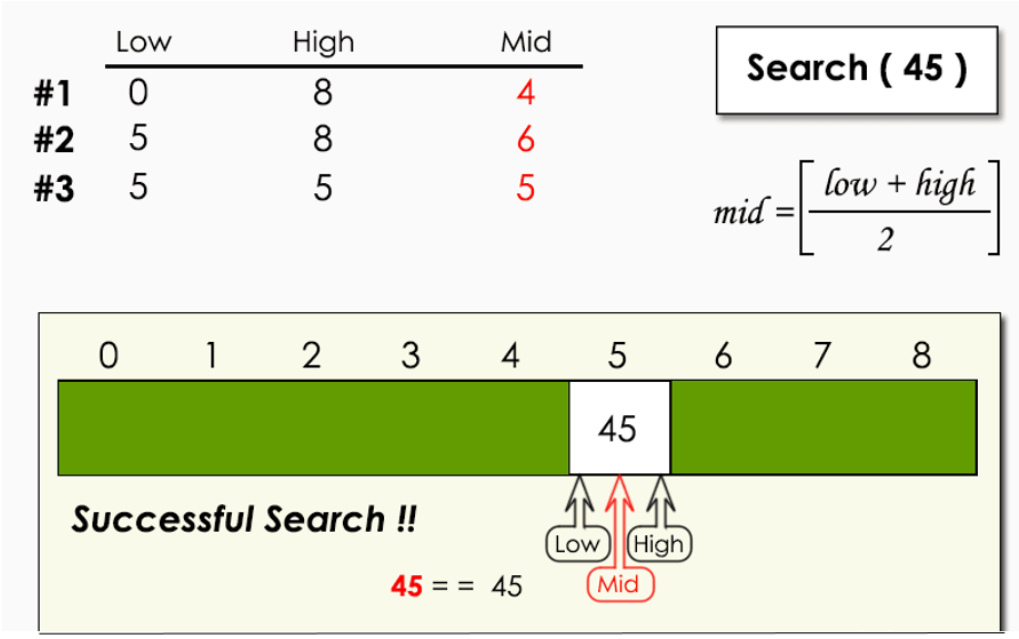


	Low	High	Mid
#1	0	8	4
#2	5	8	6

Search (45)

$$mid = \left[\frac{low + high}{2} \right]$$





 # Iterative Binary Search Function method Python Implementation
 # It returns index of n in given list1 if present,
 # else returns -1

```
def binary_search(list1, n):
    low = 0
    high = len(list1) - 1
    mid = 0

    while low <= high:
        # for get integer result
        mid = (high + low) // 2

        # Check if n is present at mid
        if list1[mid] < n:
            low = mid + 1

        # If n is greater, compare to the right of mid
        elif list1[mid] > n:
            high = mid - 1

        # If n is smaller, compared to the left of mid
        else:
            return mid

            # element was not present in the list, return -1
    return -1
```

```
# Initial list1
list1 = input('Enter the list of numbers: ').split()
list1 = [int(x)for x in list1]
n=int(input('enter the search element'))


# Function call
result = binary_search(list1, n)

if result != -1:
   print("Element is present at index", str(result))
else:
   print("Element is not present in list1")
```

Output:

Enter the list of numbers: 10 3 2 13 5 6
enter the search element 2
2
Element is present at index 2

Binary Search Complexity

Time Complexities

- **Best case complexity**: O(1)
- **Average case complexity**: O(log n)
- **Worst case complexity**: O(log n)

Space Complexity

The space complexity of the binary search is O(1).

Binary Search Applications

- In libraries of Java, .Net, C++ STL
- While debugging, the binary search is used to pinpoint the place where the error happens.

5.10. Common elements in list

Given two lists, print all the common elements of two lists.
Input : list1 = [1, 2, 3, 4, 5]

list2 = [5, 6, 7, 8, 9]

Output : {5}

Explanation: The common elements of

both the lists are 3 and 4

Input : list1 = [1, 2, 3, 4, 5]

list2 = [6, 7, 8, 9]

Output : No common elements

Explanation: They do not have any

elements in common in between them

Method 1:Using Set's & property

Convert the lists to sets and then print **set1&set2**. set1&set2 returns the common elements set, where set1 is the list1 and set2 is the list2.
Below is the Python3 implementation of the above approach:

```
# Python program to find the common elements
# in two lists
def common_member(a, b):
    a_set = set(a)
    b_set = set(b)

    if (a_set & b_set):
        print(a_set & b_set)
    else:
        print("No common elements")
a = [1, 2, 3, 4, 5]
b = [5, 6, 7, 8, 9]
common_member(a, b)

a = [1, 2, 3, 4, 5]
b = [6, 7, 8, 9]
common_member(a, b)
```

Output:

{5}

No common elements

Method 2:Using Set's intersection property

Convert the list to set by conversion. Use the intersection function to check if both sets have any elements in common. If they have many elements in common, then print the intersection of both sets.
Below is the Python3 implementation of the above approach:

```
# Python program to find common elements in
# both sets using intersection function in
# sets
# function
def common_member(a, b):
    a_set = set(a)
    b_set = set(b)

    # check length
    if len(a_set.intersection(b_set)) > 0:
        return(a_set.intersection(b_set))
    else:
        return("no common elements")
a = [1, 2, 3, 4, 5]
b = [5, 6, 7, 8, 9]
print(common_member(a, b))

a =[1, 2, 3, 4, 5]
b =[6, 7, 8, 9]
print(common_member(a, b))
```

Output:

{5}

No common elements

5.11. Hash Table

The Hash table data structure stores elements in key-value pairs where

- **Key**- unique integer that is used for indexing the values
- **Value** - data that are associated with keys.

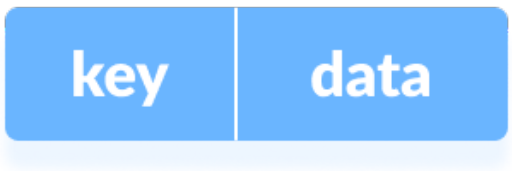


Figure: Key and Value in Hash table

- A hash table is a form of list where elements are accessed by a keyword rather than an index number.

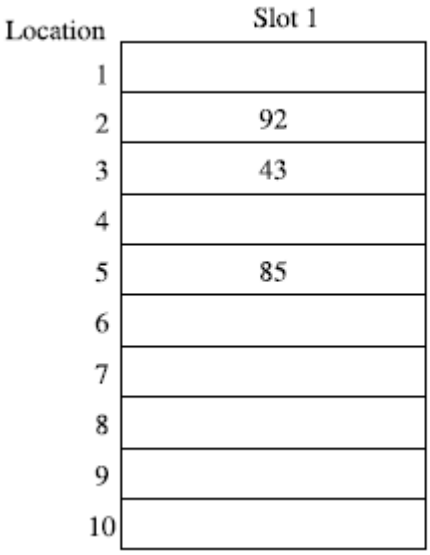


Figure: An ideal hash table

In Python, the Dictionary data types represent the implementation of hash tables. The Keys in the dictionary satisfy the following requirements.

- The keys of the dictionary are hashable i.e. the are generated by hashing function which generates unique result for each unique value supplied to the hash function.
- The order of data elements in a dictionary is not fixed.

5.12. DICTIONARY

- Python dictionary is an unordered collection of items. Each item of a dictionary has a key/value pair.
- Dictionaries are optimized to retrieve values when the key is known.

Creating Python Dictionary

- Creating a dictionary is as simple as placing items inside curly braces {} separated by commas.
- An item has a key and a corresponding value that is expressed as a pair (key: value).
- While the values can be of any data type and can repeat, keys must be of immutable type (string, number or tuple with immutable elements) and must be unique.

```
# empty dictionary
my_dict = {}

# dictionary with integer keys
my_dict = {1: 'apple', 2: 'ball'}

# dictionary with mixed keys
my_dict = {'name': 'John', 1: [2, 4, 3]}
```

```
# using dict()
my_dict = dict({1:'apple', 2:'ball'})

# from sequence having each item as a pair
my_dict = dict([(1,'apple'), (2,'ball')])
```

As you can see from above, we can also create a dictionary using the built-in dict()function.

5.12.1. Accessing Elements from Dictionary

- While indexing is used with other data types to access values, a dictionary uses keys. Keys can be used either inside square brackets [] or with the get() method.
- If we use the square brackets [], KeyError is raised in case a key is not found in the dictionary. On the other hand, the get() method returns None if the key is not found.

```
# get vs [] for retrieving elements
my_dict = {'name': 'Jack', 'age': 26}

# Output: Jack
print(my_dict['name'])

# Output: 26
print(my_dict.get('age'))

# Trying to access keys which doesn't exist throws error
# Output None
print(my_dict.get('address'))

# KeyError
print(my_dict['address'])
```

Output

```
Jack
26
None
Traceback (most recent call last):
  File "<string>", line 15, in <module>
    print(my_dict['address'])
KeyError: 'address'
```

5.12.2. Changing and Adding Dictionary elements

- Dictionaries are mutable. We can add new items or change the value of existing items using an assignment operator.

- If the key is already present, then the existing value gets updated. In case the key is not present, a new (**key: value**) pair is added to the dictionary.

```
# Changing and adding Dictionary Elements
my_dict = {'name': 'Jack', 'age': 26}

# update value
my_dict['age'] = 27

#Output: {'age': 27, 'name': 'Jack'}
print(my_dict)

# add item
my_dict['address'] = 'Downtown'

# Output: {'address': 'Downtown', 'age': 27, 'name': 'Jack'}
print(my_dict)
```

Output

```
{'name': 'Jack', 'age': 27}
{'name': 'Jack', 'age': 27, 'address': 'Downtown'}
```

5.12.3. Removing elements from Dictionary

- We can remove a particular item in a dictionary by using the pop() method. This method removes an item with the provided key and returns the value.
- The popitem() method can be used to remove and return an arbitrary (key, value)item pair from the dictionary. All the items can be removed at once, using the clear() method.
- We can also use the del keyword to remove individual items or the entire dictionary itself.

```
# Removing elements from a dictionary

# create a dictionary
squares = {1: 1, 2: 4, 3: 9, 4: 16, 5: 25}

# remove a particular item, returns its value
# Output: 16
print(squares.pop(4))

# Output: {1: 1, 2: 4, 3: 9, 5: 25}
print(squares)

# remove an arbitrary item, return (key,value)
# Output: (5, 25)
print(squares.popitem())

# Output: {1: 1, 2: 4, 3: 9}
print(squares)
```

```
# remove all items
squares.clear()

# Output: {}
print(squares)

# delete the dictionary itself
del squares

# Throws Error
print(squares)
```

Output

```
16
{1: 1, 2: 4, 3: 9, 5: 25}
(5, 25)
{1: 1, 2: 4, 3: 9}
{}
Traceback (most recent call last):
  File "<string>", line 30, in <module>
    print(squares)
NameError: name 'squares' is not defined
```

5.12.4. Python Dictionary Methods

Methods that are available with a dictionary are tabulated below. Some of them have already been used in the above examples.

Method	Description
clear()	Removes all items from the dictionary.
copy()	Returns a shallow copy of the dictionary.
fromkeys(seq[, v])	Returns a new dictionary with keys from seq and value equal to v (defaults to None).
get(key[,d])	Returns the value of the key. If the key does not exist, returns d (defaults to None).
items()	Return a new object of the dictionary's items in (key, value) format.
keys()	Returns a new object of the dictionary's keys.
pop(key[,d])	Removes the item with the key and returns its value or d if key is not found. If d is not provided and the key is not found, it raises KeyError.
popitem()	Removes and returns an arbitrary item (key, value). Raises KeyError if the dictionary is empty.
setdefault(key[,d])	Returns the corresponding value if the key is in the dictionary. If not, inserts the key with a value of d and returns d (defaults to None).
update([other])	Updates the dictionary with the key/value pairs from other, overwriting existing keys.
values()	Returns a new object of the dictionary's values

Here are a few example use cases of these methods.

```
# Dictionary Methods
marks = {}.fromkeys(['Math', 'English', 'Science'], 0)

# Output: {'English': 0, 'Math': 0, 'Science': 0}
print(marks)

for item in marks.items():
   print(item)

# Output: ['English', 'Math', 'Science']
print(list(sorted(marks.keys())))
```

Output

```
{'Math': 0, 'English': 0, 'Science': 0}
('Math', 0)
('English', 0)
('Science', 0)
['English', 'Math', 'Science']
```

5.12.5. Python Dictionary Comprehension

- Dictionary comprehension is an elegant and concise way to create a new dictionary from an iterable in Python.
- Dictionary comprehension consists of an expression pair (**key: value**) followed by a for statement inside curly braces `{}`.
- Here is an example to make a dictionary with each item being a pair of a number and its square.

```
# Dictionary Comprehension
squares = {x: x*x for x in range(6)}

print(squares)
```

Output

```
{0: 0, 1: 1, 2: 4, 3: 9, 4: 16, 5: 25}
```

```
This code is equivalent to
squares = {}
for x in range(6):
   squares[x] = x*x
print(squares)
```

Output

```
{0: 0, 1: 1, 2: 4, 3: 9, 4: 16, 5: 25}
```

A dictionary comprehension can optionally contain more for or if statements.
An optional if statement can filter out items to form the new dictionary.

Here are some examples to make a dictionary with only odd items.

```
# Dictionary Comprehension with if conditional
odd_squares = {x: x*x for x in range(11) if x % 2 == 1}

print(odd_squares)
```

Output

```
{1: 1, 3: 9, 5: 25, 7: 49, 9: 81}
```

To learn more dictionary comprehensions, visit Python Dictionary Comprehension.

5.12.6. Other Dictionary Operations

Dictionary Membership Test

We can test if a `key` is in a dictionary or not using the keyword `in`. Notice that the membership test is only for the keys and not for the values.

```
# Membership Test for Dictionary Keys
squares = {1: 1, 3: 9, 5: 25, 7: 49, 9: 81}

# Output: True
print(1 in squares)

# Output: True
print(2 not in squares)

# membership tests for key only not value
# Output: False
print(49 in squares)
```

Output

```
True
True
False
```

Iterating Through a Dictionary

We can iterate through each key in a dictionary using a for loop.

```
# Iterating through a Dictionary
squares = {1: 1, 3: 9, 5: 25, 7: 49, 9: 81}
for i in squares:
    print(squares[i])
```

Output

```
1
9
25
49
81
```

Dictionary Built-in Functions

Built-in functions like all(), any(), len(), cmp(), sorted(), etc. are commonly used with dictionaries to perform different tasks.

Function	Description
all()	Return True if all keys of the dictionary are True (or if the dictionary is empty).
any()	Return True if any key of the dictionary is true. If the dictionary is empty, return False.
len()	Return the length (the number of items) in the dictionary.
cmp()	Compares items of two dictionaries. (Not available in Python 3)
sorted()	Return a new sorted list of keys in the dictionary.

Here are some examples that use built-in functions to work with a dictionary.

```
# Dictionary Built-in Functions
squares = {0: 0, 1: 1, 3: 9, 5: 25, 7: 49, 9: 81}

# Output: False
print(all(squares))

# Output: True
print(any(squares))

# Output: 6
print(len(squares))

# Output: [0, 1, 3, 5, 7, 9]
print(sorted(squares))
```

Output

```
False
True
6
[0, 1, 3, 5, 7, 9]
```

TEXT / REFERENCE BOOKS

1. William McKinney, Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython, 2nd Edition Paperback – 27 October 2017
2. Luca Massaron John Paul Mueller, Python for Data Science For Dummies, 2nd Paperback – 1 January 2019

QUESTION BANK

Part-A			
Q.No	**Questions**	**Competence**	**BT Level**
1.	Define Data structure	Remember	BTL 1
2.	List the types of built-in data structures	Understand	BTL 2
3.	Define function and write the syntax of function?	Remember	BTL 1
4.	Define function call	Remember	BTL 1
5.	Write a python program to find even or odd using function	Apply	BTL 3
6.	Define pandas	Remember	BTL 1
7.	Interpret Dataframe	Understand	BTL 2
8.	Elaborate numpy	Understand	BTL 2
9.	How to load data in pandas?	Understand	BTL 2
10.	Write about the libraries used for pre-processing and array operations?	Understand	BTL 2
11.	List various plots using matplotlib	Understand	BTL 2
12.	Define matplotlib and How to import matplotlib	Understand	BTL 2
13.	Interpret dictionary	Understand	BTL 2
14.	State the use of hash table	Understand	BTL 2
15.	Define computational complexity	Understand	BTL 2
16.	Elaborate time complexity	Understand	BTL 2
17.	Write the time complexity of Linear search with an example program.	Analysis	BTL 4
18.	State the operations using Numpy	Understand	BTL 2
19.	Define series	Remember	BTL 1

20.	How to remove an element from dictionary?	Analysis	BTL 4
PART B			
Q.No	**Questions**	**Competence**	**BT Level**
1.	Explain about the python function with an example	Analysis	BTL 4
2.	Explain the various operations performed using numpy	Analysis	BTL 4
3.	Explain about Dataframe in pandas	Analysis	BTL 4
4.	Explain about NumPy – Array Manipulation	Analysis	BTL 4
5.	Write a python code to create 1D and 2D array using numpy	Analysis	BTL 4
6.	Explain about the operations performed using Numpy-Linear Algebra with an example program	Analysis	BTL 4
7.	Explain about the basic plots in Matplotlib	Analysis	BTL 4
8.	Explain about python pandas data structure with an example	Analysis	BTL 4
9.	Discuss about the different problems based on computational complexity	Analysis	BTL 4
10.	Write the python program to find the binary search	Apply	BTL 3
11.	Write the python program to find common elements in python list using two different methods	Apply	BTL 3
12.	Write a python code to add and remove elements from dictionary?	Apply	BTL 3