GL BAJAJ
Group of Institutions
Mathura
FIND YOUR SPARK
Approved by AICTE, CoA & Affiliated to AKTU

Data Analytics

3^{rd} year

Notes for unit -5

TOPICS COVERED IN UNIT 5

Frame work and Visualization

- MapReduce
- HADOOP
- Pig
- Hive
- Hbase
- MapR Sharding
- NoSQL Database
- S3
- Hadoop distributed file system

Visualization

- **Visualization –**
  - **Visual data analysis techniques**
  - **Interaction techniques**
  - **System and application**

Introduction to R

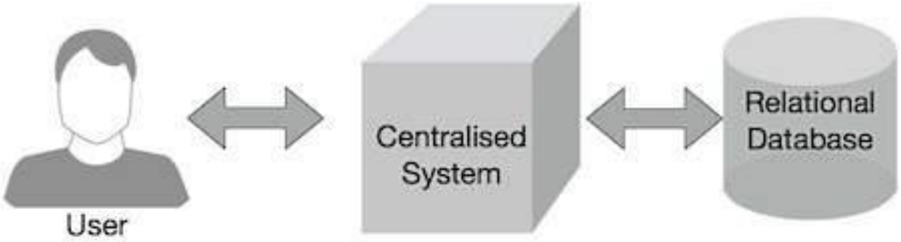
MapReduce is a programming model for writing applications that can process Big Data in parallel on multiple nodes. MapReduce provides analytical capabilities for analyzing huge volumes of complex data.

What is Big Data?

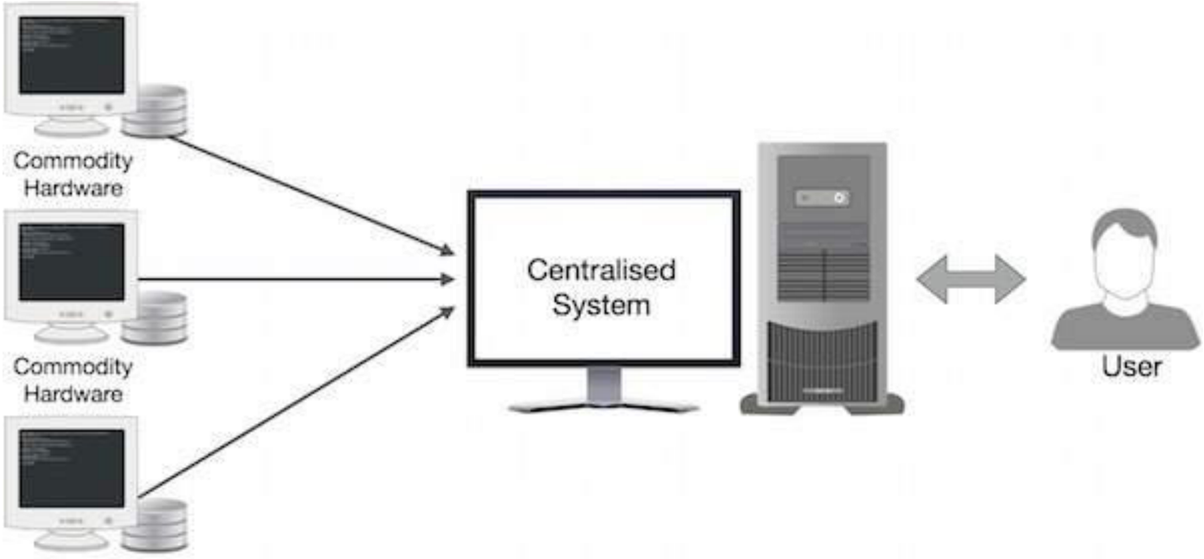
Big Data is a collection of large datasets that cannot be processed using traditional computing techniques. For example, the volume of data Facebook or Youtube need require it to collect and manage on a daily basis, can fall under the category of Big Data. However, Big Data is not only about scale and volume, it also involves one or more of the following aspects − Velocity, Variety, Volume, and Complexity.

Why MapReduce?

Traditional Enterprise Systems normally have a centralized server to store and process data. The following illustration depicts a schematic view of a traditional enterprise system. Traditional model is certainly not suitable to process huge volumes of scalable data and cannot be accommodated by standard database servers. Moreover, the centralized system creates too much of a bottleneck while processing multiple files simultaneously.



Google solved this bottleneck issue using an algorithm called MapReduce. MapReduce divides a task into small parts and assigns them to many computers. Later, the results are collected at one place and integrated to form the result dataset.



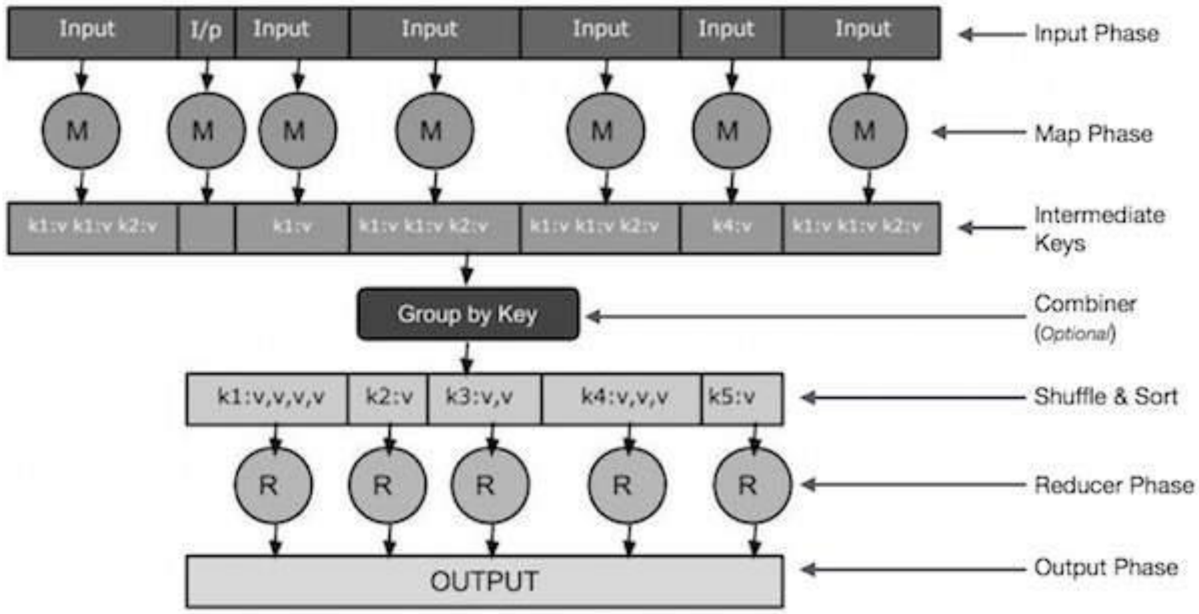
How MapReduce Works?

The MapReduce algorithm contains two important tasks, namely Map and Reduce.

- The Map task takes a set of data and converts it into another set of data, where individual elements are broken down into tuples (key-value pairs).
- The Reduce task takes the output from the Map as an input and combines those data tuples (key-value pairs) into a smaller set of tuples.

The reduce task is always performed after the map job.

Let us now take a close look at each of the phases and try to understand their significance.

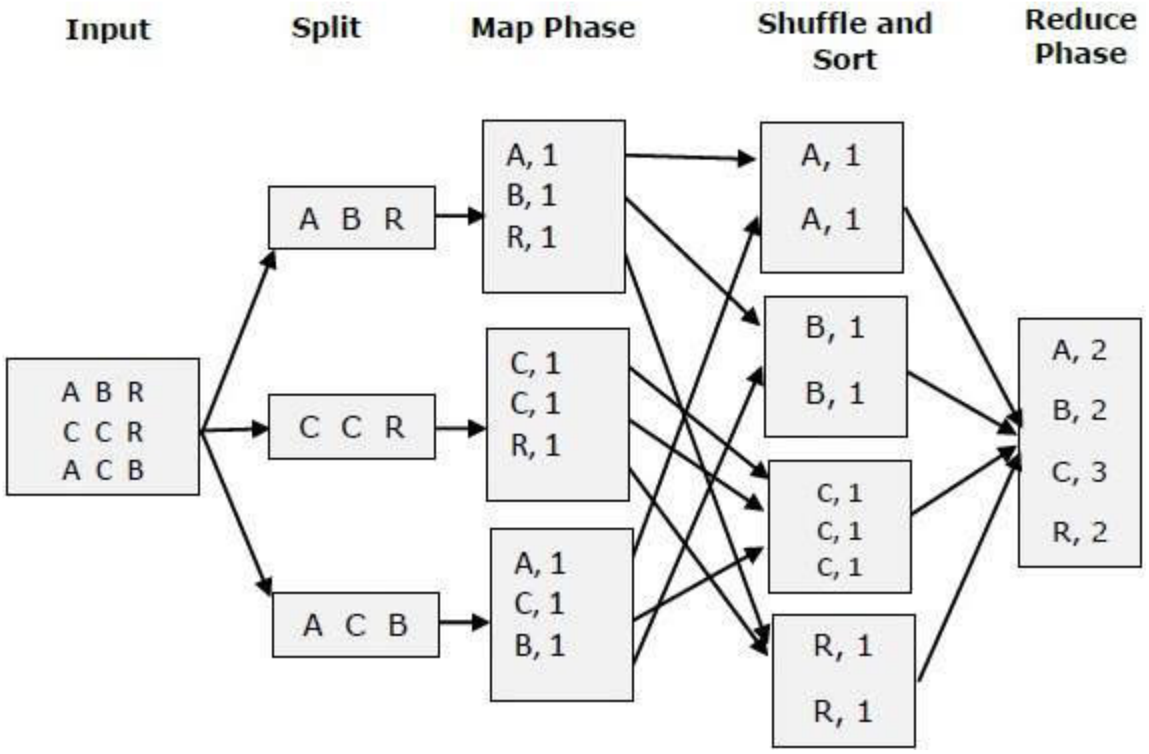


- **Input Phase** − Here we have a Record Reader that translates each record in an input file and sends the parsed data to the mapper in the form of key-value pairs.
- **Map** − Map is a user-defined function, which takes a series of key-value pairs and processes each one of them to generate zero or more key-value pairs.
- **Intermediate Keys** − They key-value pairs generated by the mapper are known as intermediate keys.
- **Combiner** − A combiner is a type of local Reducer that groups similar data from the map phase into identifiable sets. It takes the intermediate keys from the mapper as input and applies a user-defined code to aggregate the values in a small scope of one mapper. It is not a part of the main MapReduce algorithm; it is optional.
- **Shuffle and Sort** − The Reducer task starts with the Shuffle and Sort step. It downloads the grouped key-value pairs onto the local machine, where the Reducer is running. The individual key-value pairs are sorted by key into a larger data list. The data list groups the equivalent

keys together so that their values can be iterated easily in the Reducer task.

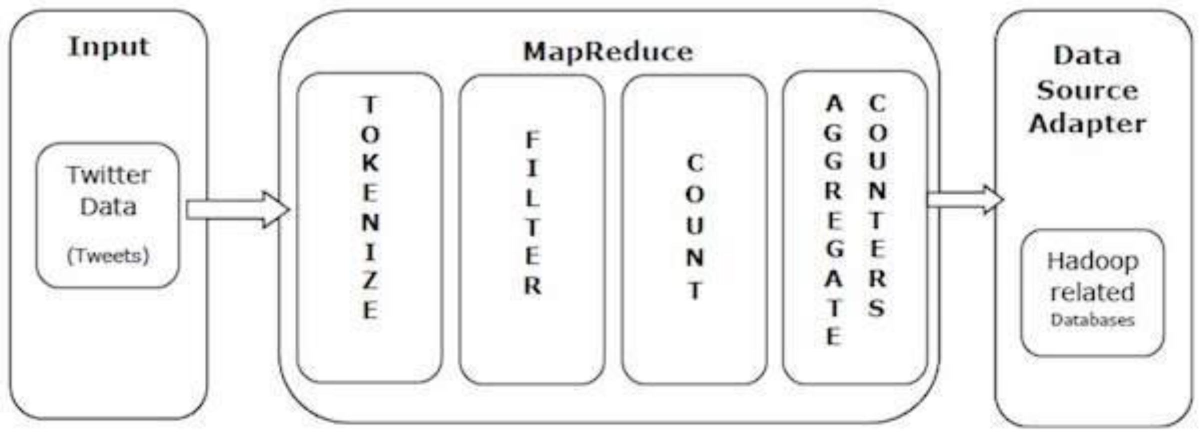
- **Reducer** − The Reducer takes the grouped key-value paired data as input and runs a Reducer function on each one of them. Here, the data can be aggregated, filtered, and combined in a number of ways, and it requires a wide range of processing. Once the execution is over, it gives zero or more key-value pairs to the final step.
- **Output Phase** − In the output phase, we have an output formatter that translates the final key-value pairs from the Reducer function and writes them onto a file using a record writer.

Let us try to understand the two tasks Map &f Reduce with the help of a small diagram −



MapReduce-Example

Let us take a real-world example to comprehend the power of MapReduce. Twitter receives around 500 million tweets per day, which is nearly 3000 tweets per second. The following illustration shows how Tweeter manages its tweets with the help of MapReduce.



As shown in the illustration, the MapReduce algorithm performs the following actions −

- **Tokenize** − Tokenizes the tweets into maps of tokens and writes them as key-value pairs.
- **Filter** − Filters unwanted words from the maps of tokens and writes the filtered maps as key-value pairs.
- **Count** − Generates a token counter per word.
- **Aggregate Counters** − Prepares an aggregate of similar counter values into small manageable units.

Hadoop

Hadoop is an open-source framework written in Java that uses lots of other analytical tools to improve its data analytics operations. The article demonstrates the most widely and essential analytics tools that Hadoop can use to improve its reliability and processing to generate new insight into data. Hadoop is used for some advanced level of analytics, which includes Machine Learning and data mining.

There is a wide range of analytical tools available in the market that help Hadoop deal with the astronomical size data efficiently. Let us discuss some of the most famous and widely used tools one by one. Below are the top 10 Hadoop analytics tools for big data.

1. Apache Spark

Apache spark in an open-source processing engine that is designed for ease of analytics operations. It is a cluster computing platform that is designed to be fast and made for general purpose uses. Spark is designed to cover various batch applications, Machine Learning, streaming data processing, and interactive queries.

Features of Spark:

- In memory processing
- Tight Integration Of component
- Easy and In-expensive
- The powerful processing engine makes it so fast

- Spark Streaming has high level library for streaming process

2. Map Reduce

MapReduce is just like an Algorithm or a data structure that is based on the YARN framework. The primary feature of MapReduce is to perform the distributed processing in parallel in a Hadoop cluster, which Makes Hadoop working so fast Because when we are dealing with Big Data, serial processing is no more of any use.

Features of Map-Reduce:

- Scalable
- Fault Tolerance
- Paraller Processing
- Tunable Replication
- Load Balancing

3. Apache Hive

Apache Hive is a Data warehousing tool that is built on top of the Hadoop, and Data Warehousing is nothing but storing the data at a fixed location generated from various sources. Hive is one of the best tools used for data analysis on Hadoop. The one who is having knowledge of SQL can comfortably use Apache Hive. The query language of high is known as HQL or HIVEQL.

Features of Hive:

- Queries are similar to SQL queries.

- Hive has different storage type HBase, ORC, Plain text, etc.
- Hive has in-built function for data-mining and other works.
- Hive operates on compressed data that is present inside Hadoop Ecosystem.

4. Apache Impala

Apache Impala is an open-source SQL engine designed for Hadoop. Impala overcomes the speed-related issue in Apache Hive with its faster-processing speed. Apache Impala uses similar kinds of SQL syntax, ODBC driver, and user interface as that of Apache Hive. Apache Impala can easily be integrated with Hadoop for data analytics purposes.

Features of Impala:

- Easy-Integration
- Scalability
- Security
- In Memory data processing

5. Apache Mahout

The name *Mahout* is taken from the Hindi word **Mahavat** which means the elephant rider. Apache Mahout runs the algorithm on the top of Hadoop, so it is named Mahout. Mahout is mainly used for implementing various Machine Learning algorithms on our Hadoop like classification, Collaborative filtering, Recommendation. Apache Mahout can implement the Machine algorithms without integration on Hadoop.

Features of Mahout:

- Used for Machine Learning Application
- Mahout has Vector and Matrix libraries
- Ability to analyze large datasets quickly

6. Apache Pig

This Pig was Initially developed by Yahoo to get ease in programming. Apache Pig has the capability to process an extensive dataset as it works on top of the Hadoop. Apache pig is used for analyzing more massive datasets by representing them as dataflow. Apache Pig also raises the level of abstraction for processing enormous datasets. Pig Latin is the scripting language that the developer uses for working on the Pig framework that runs on Pig runtime.

Features of Pig:

- Easy To Programme
- Rich set of operators
- Ability to handle various kind of data
- Extensibility

7. HBase

HBase is nothing but a non-relational, NoSQL distributed, and column-oriented database. HBase consists of various tables where each table has multiple numbers of data rows. These rows will have multiple numbers of column family’s, and this column family will have columns that contain key-value pairs.

HBase works on the top of HDFS(Hadoop Distributed File System). We use HBase for searching small size data from the more massive datasets.

Features of HBase:

- HBase has Linear and Modular Scalability
- JAVA API can easily be used for client access
- Block cache for real time data queries

8. Apache Sqoop

Sqoop is a command-line tool that is developed by Apache. The primary purpose of Apache Sqoop is to import structured data i.e., RDBMS(Relational database management System) like MySQL, SQL Server, Oracle to our HDFS(Hadoop Distributed File System). Sqoop can also export the data from our HDFS to RDBMS.

Features of Sqoop:

- Sqoop can Import Data To Hive or HBase
- Connecting to database server
- Controlling parallelism

9. Tableau

Tableau is a data visualization software that can be used for data analytics and business intelligence. It provides a variety of interactive visualization to showcase the insights of the data and can translate the queries to visualization and can also import all ranges and sizes of data. Tableau offers rapid analysis and processing, so it Generates useful visualizing charts on interactive dashboards and worksheets.

Features of Tableu:

- Tableau supports Bar chart, Histogram, Pie chart, Motion chart, Bullet chart, Gantt chart and so many
- Secure and Robust
- Interactive Dashboard and worksheets

10. Apache Storm

Apache Storm is a free open source distributed real-time computation system build using Programming languages like Clojure and java. It can be used with many programming languages. Apache Storm is used for the Streaming process, which is very faster. We use Daemons like Nimbus, Zookeeper, and Supervisor in Apache Storm. Apache Storm can be used for real-time processing, online Machine learning, and many more. Companies like Yahoo, Spotify, Twitter, and so many uses Apache Storm.

Features of Storm:

- Easily operatable
- each node can process millions of tuples in one second
- Scalable and Fault Tolerance

HDFS

Hadoop File System was developed using distributed file system design. It is run on commodity hardware. Unlike other distributed systems, HDFS is highly faulttolerant and designed using low-cost hardware.

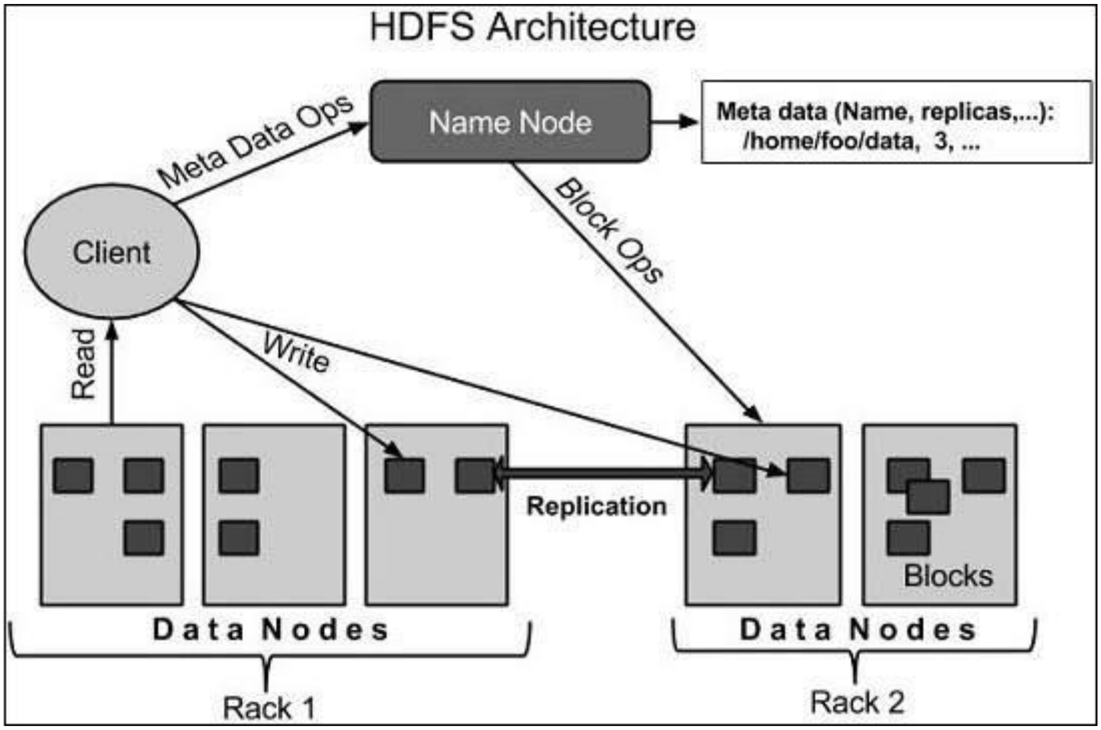
HDFS holds very large amount of data and provides easier access. To store such huge data, the files are stored across multiple machines. These files are stored in redundant fashion to rescue the system from possible data losses in case of failure. HDFS also makes applications available to parallel processing.

Features of HDFS

- It is suitable for the distributed storage and processing.
- Hadoop provides a command interface to interact with HDFS.
- The built-in servers of namenode and datanode help users to easily check the status of cluster.
- Streaming access to file system data.
- HDFS provides file permissions and authentication.

HDFS Architecture

Given below is the architecture of a Hadoop File System.



HDFS follows the master-slave architecture and it has the following elements.

Namenode

The namenode is the commodity hardware that contains the GNU/Linux operating system and the namenode software. It is a software that can be run on commodity hardware. The system having the namenode acts as the master server and it does the following tasks −

- Manages the file system namespace.
- Regulates client’s access to files.
- It also executes file system operations such as renaming, closing, and opening files and directories.

Datanode

The datanode is a commodity hardware having the GNU/Linux operating system and datanode software. For every node (Commodity hardware/System) in a cluster, there will be a datanode. These nodes manage the data storage of their system.

- Datanodes perform read-write operations on the file systems, as per client request.
- They also perform operations such as block creation, deletion, and replication according to the instructions of the namenode.

Block

Generally the user data is stored in the files of HDFS. The file in a file system will be divided into one or more segments and/or stored in individual data nodes. These file segments are called as blocks. In other words, the minimum amount of data that HDFS can read or write is called a Block. The default block size is 64MB, but it can be increased as per the need to change in HDFS configuration.

Goals of HDFS

Fault detection and recovery − Since HDFS includes a large number of commodity hardware, failure of components is frequent. Therefore HDFS should have mechanisms for quick and automatic fault detection and recovery.

Huge datasets − HDFS should have hundreds of nodes per cluster to manage the applications having huge datasets.

Hardware at data – A requested task can be done efficiently, when the computation takes place near the data. Especially where huge datasets are involved, it reduces the network traffic and increases the throughput.

Data visualization provides an important suite of tools for identifying a qualitative understanding. This can be helpful when we try to explore the dataset and extract some information to know about a dataset and can help with **identifying patterns, corrupt data, outliers**, and much more.

If we have a little domain knowledge, then data visualizations can be used to express and identify key relationships in plots and charts that are more helpful to yourself and stakeholders than measures of association or significance.

Table of Contents

1. What is Data Visualization?

2. Benefits of Good Data Visualization

3. Different Types of Analysis for Data Visualization

4. Univariate Analysis Techniques for Data Visualization

- Distribution Plot
- Box and Whisker Plot
- Violin Plot

5. Bivariate Analysis Techniques for Data Visualization

- Line Plot
- Bar Plot
- Scatter Plot

What is Data Visualization?

Data visualization is defined as a **graphical representation** that contains the **information** and the **data**.

By using visual elements like **charts**, **graphs**, and **maps**, data visualization techniques provide an accessible way to see and **understand trends, outliers, and patterns in data**.

In modern days we have a lot of data in our hands i.e, in the world of **Big Data**, data visualization tools, and technologies are crucial to analyze massive amounts of information and make data-driven decisions.

It is used in many areas such as:

- To model **complex events**.
- Visualize phenomenons that cannot be observed directly, such as **weather patterns**, **medical conditions**, or **mathematical relationships**.

Benefits of Good Data Visualization

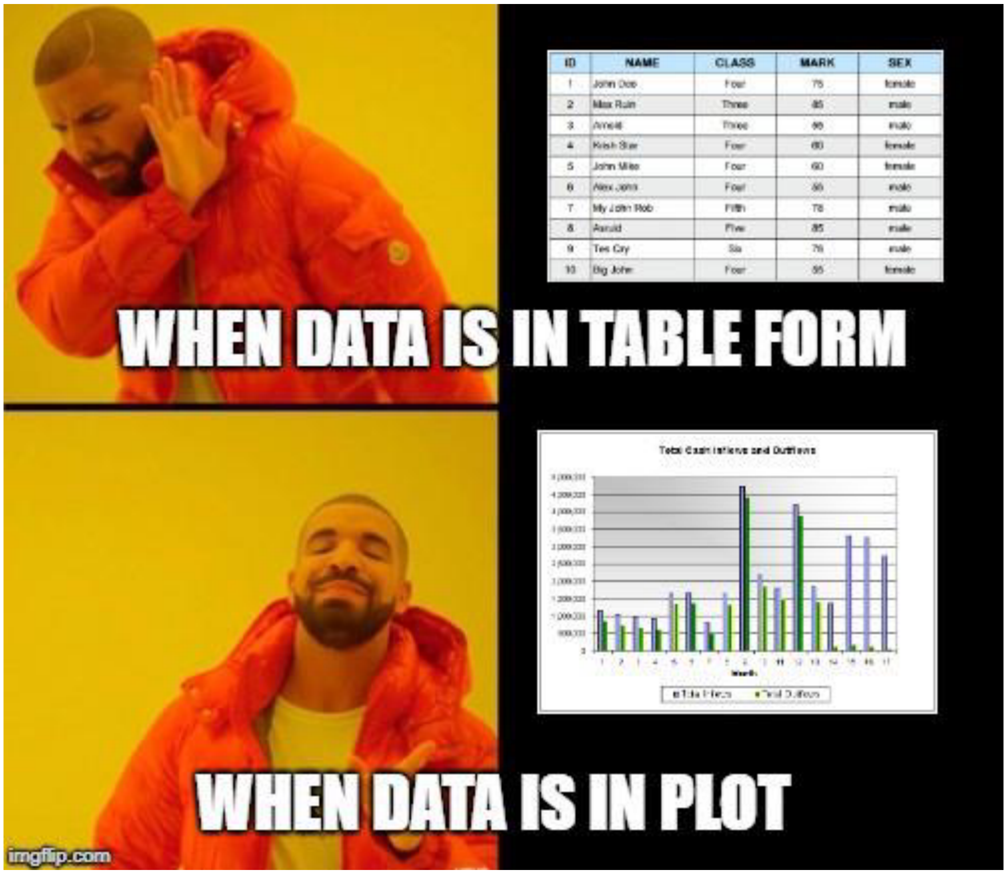
Since our eyes can capture the colors and patterns, therefore, we can quickly identify the red portion from blue, square from the circle, our culture is visual, including everything from art and advertisements to TV and movies.

So, Data visualization is another technique of visual art that grabs our interest and keeps our main focus on the message captured with the help of eyes.

Whenever we visualize a chart, we quickly identify the trends and outliers present in the dataset.

The basic uses of the Data Visualization technique are as follows:

- It is a powerful technique to explore the data with **presentable** and **interpretable** results.
- In the **data mining process**, it acts as a primary step in the pre-processing portion.
- It supports the **data cleaning process** by finding incorrect data and corrupted or missing values.
- It also helps to **construct and select variables**, which means we have to determine which variable to include and discard in the analysis.
- In the process of **Data Reduction**, it also plays a crucial role while combining the categories.



ID	NAME	CLASS	MARK	SEX
1	John Deo	Four	75	female
2	Max Ruin	Three	85	male
3	Arnold	Three	55	male
4	Krish Star	Four	60	female
5	John Mike	Four	60	female
6	Alex John	Four	55	male
7	My John Rob	Fifth	78	male
8	Asruid	Five	85	male
9	Tes Qry	Six	78	male
10	Big John	Four	55	female

Mainly, there are three different types of analysis for Data Visualization:

Univariate Analysis: In the univariate analysis, we will be using a single feature to analyze almost all of its properties.

Bivariate Analysis: When we compare the data between exactly 2 features then it is known as bivariate analysis.

Multivariate Analysis: In the multivariate analysis, we will be comparing more than 2 variables.

different Data Visualization techniques:

Univariate Analysis Techniques for Data Visualization

1. Distribution Plot

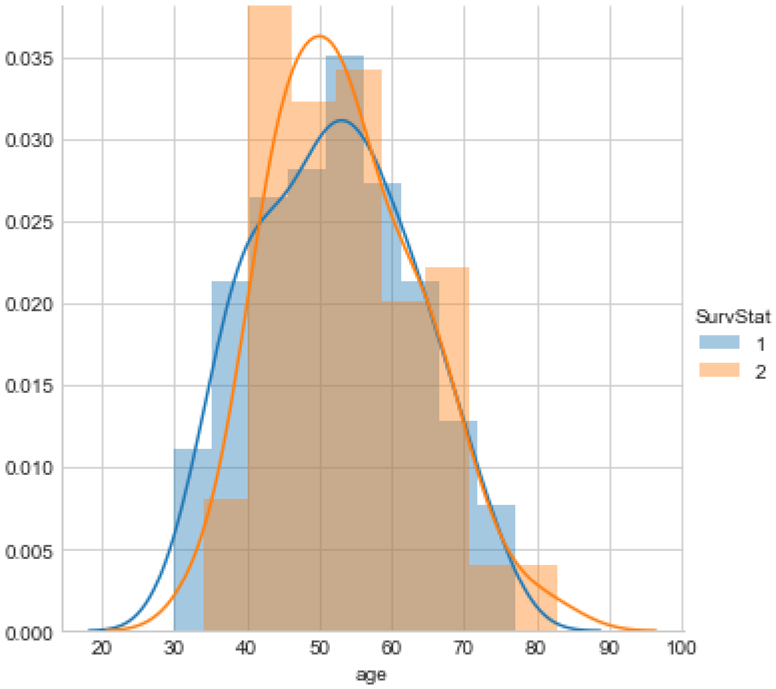
- It is one of the best univariate plots to know about the distribution of data.
- When we want to analyze the impact on the target variable(output) with respect to an independent variable(input), we use distribution plots a lot.
- This plot gives us a combination of both probability density functions(pdf) and histogram in a single plot.

Implementation:

- The distribution plot is present in the **Seaborn** package.

The code snippet is as follows:

```
sns.FacetGrid(hb,hue='SurvStat',size=5).map(sns.distplot,'age').add_legend()
```



Some conclusions inferred from the above distribution plot:

From the above distribution plot we can conclude the following observations:

- We have observed that we created a distribution plot on the feature **‘Age’**(input variable) and we used different colors for the **Survival status**(output variable) as it is the class to be predicted.

- There is a huge overlapping area between the PDFs for different combinations.
- In this plot, the sharp block-like structures are called histograms, and the smoothed curve is known as the Probability density function(PDF).

NOTE:

The Probability density function(PDF) of a curve can help us to capture the underlying distribution of that feature which is one major takeaway from Data visualization or Exploratory Data Analysis(EDA).

2. Box and Whisker Plot

- This plot can be used to obtain more **statistical details** about the data.
- The straight lines at the maximum and minimum are also called **whiskers**.
- Points that lie outside the whiskers will be considered as an outlier.
- The box plot also gives us a description of the **25th, 50th,75th quartiles**.
- With the help of a box plot, we can also determine the **Interquartile range(IQR)** where maximum details of the data will be present. Therefore, it can also give us a clear idea about the outliers in the dataset.

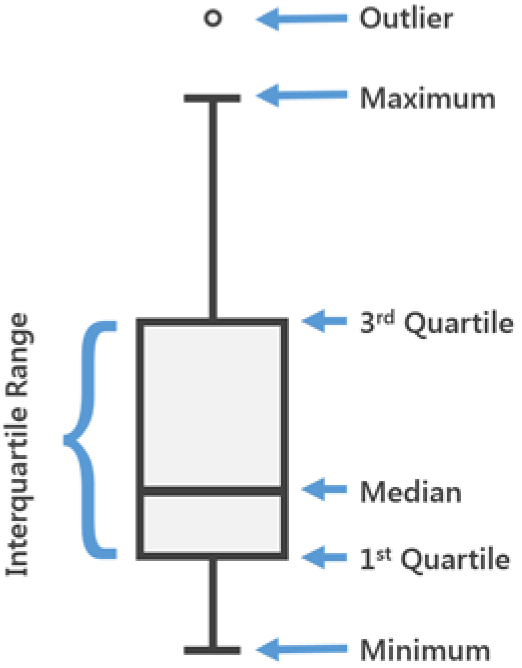


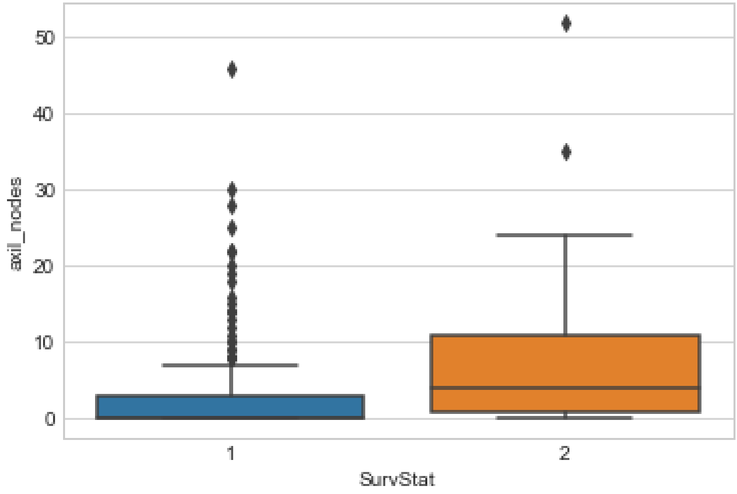
Fig. General Diagram for a Box-plot

Implementation:

- Boxplot is available in the **Seaborn** library.
- Here x is considered as the dependent variable and y is considered as the independent variable. These box plots come under **univariate analysis**, which means that we are exploring data only with one variable.
- Here we are trying to check the impact of a feature named **“axil_nodes”** on the class named **“Survival status”** and not between any two independent features.

The code snippet is as follows:

```
sns.boxplot(x='SurvStat',y='axil_nodes',data=hb)
```



Some conclusions inferred from the above box plot:

From the above box and whisker plot we can conclude the following observations:

- How much data is present in the 1st quartile and how many points are outliers etc.
- For class 1, we can see that it is very little or no data is present between the median and the 1st quartile.
- There are more outliers for class 1 in the feature named **axil_nodes**.

NOTE:

We can get details about outliers that will help us to well prepare the data before feeding it to a model since outliers influence a lot of Machine learning models.

3. Violin Plot

- The violin plots can be considered as a combination of Box plot at the middle and distribution plots**(Kernel Density Estimation)** on both sides of the data.
- This can give us the description of the distribution of the dataset like whether the distribution is **multimodal**, **Skewness**, etc.
- It also gives us useful information like a **95% confidence interval**.

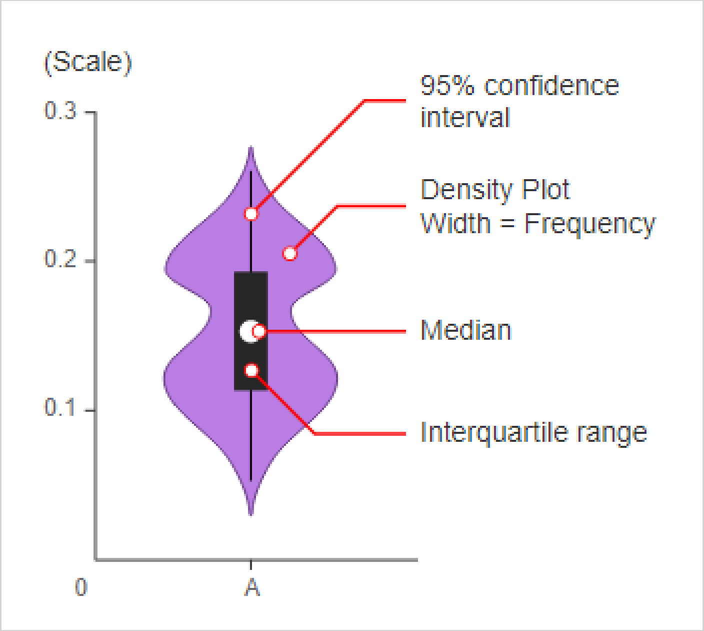


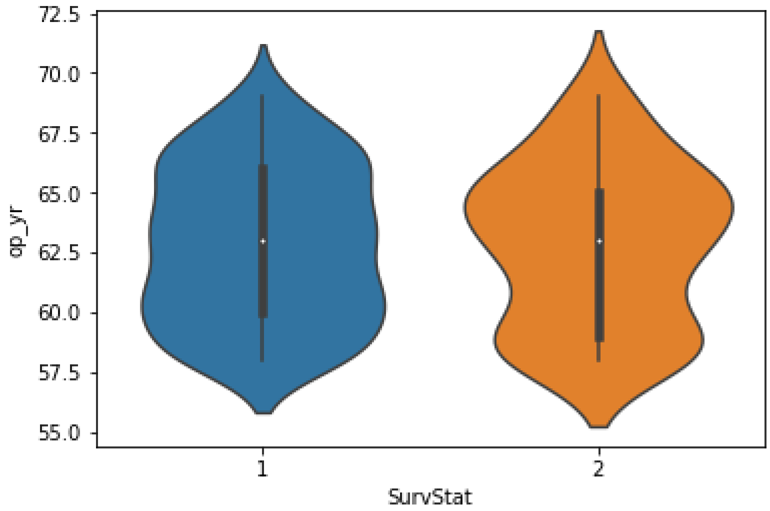
Fig. General Diagram for a Violin-plot

Implementation:

- The Violin plot is present in the **Seaborn** package.

The code snippet is as follows:

```
sns.violinplot(x='SurvStat',y='op_yr',data=hb,size=6)
```



Some conclusions inferred from the above violin plot:

From the above violin plot we can conclude the following observations:

- The median of both classes is close to 63.

- The maximum number of persons with class 2 has an **op_yr** value of 65 whereas, for persons in class1, the maximum value is around 60.
- Also, the 3rd quartile to median has a lesser number of data points than the median to the 1st quartile.

Bivariate Analysis Techniques for Data Visualization

1. Line Plot

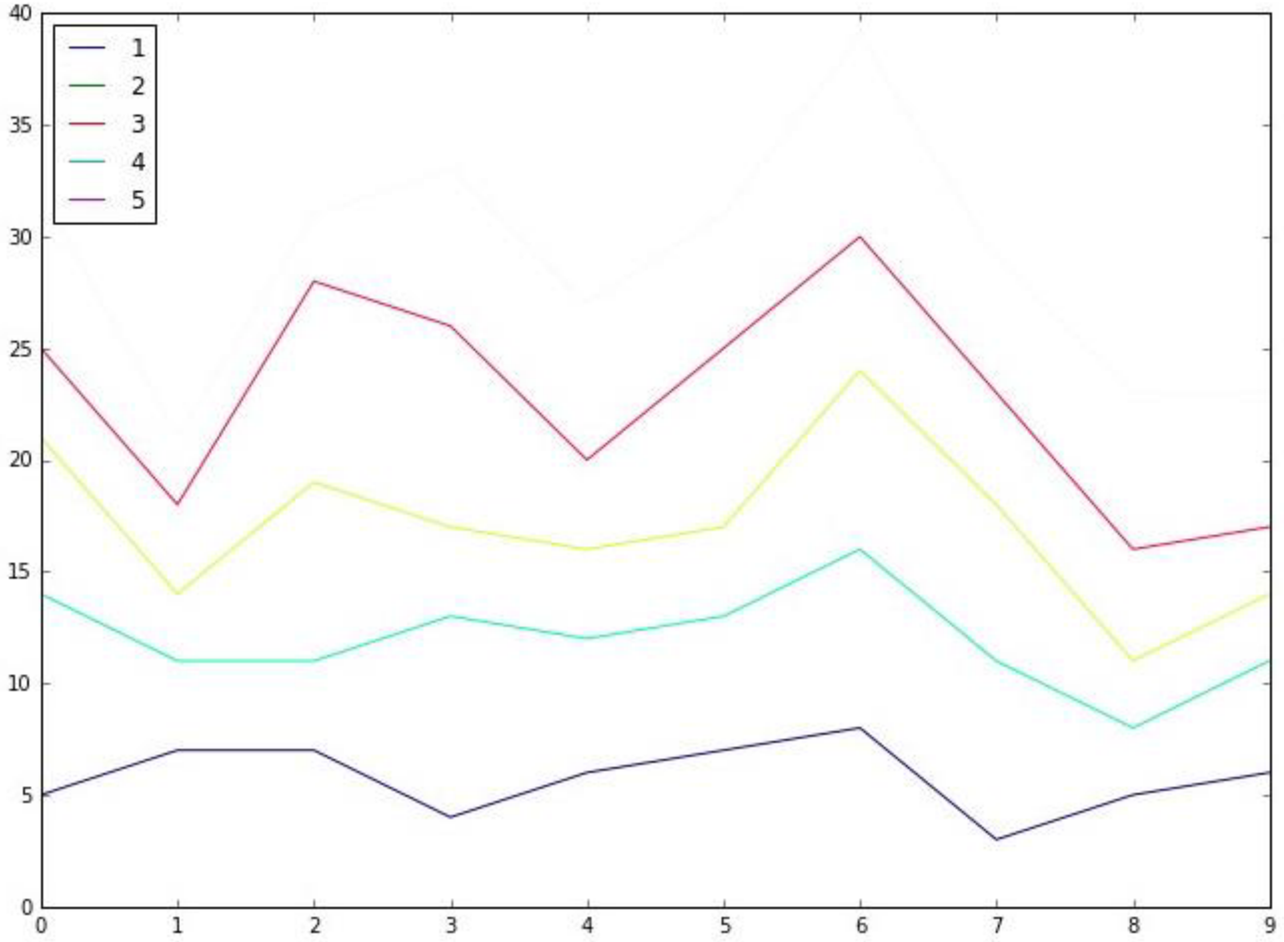
- This is the plot that you can see in the nook and corners of any sort of analysis between 2 variables.
- The line plots are nothing but the values on a series of data points will be connected with straight lines.
- The plot may seem very simple but it has more applications not only in machine learning but in many other areas.

Implementation:

- The line plot is present in the **Matplotlib** package.

The code snippet is as follows:

```
plt.plot(x,y)
```



Some conclusions inferred from the above line plot:

From the above line plot we can conclude the following observations:

- These are used right from performing distribution Comparison using **Q-Q plots** to CV tuning using the **elbow method**.

- Used to analyze the performance of a model using the **ROC- AUC curve**.

2. Bar Plot

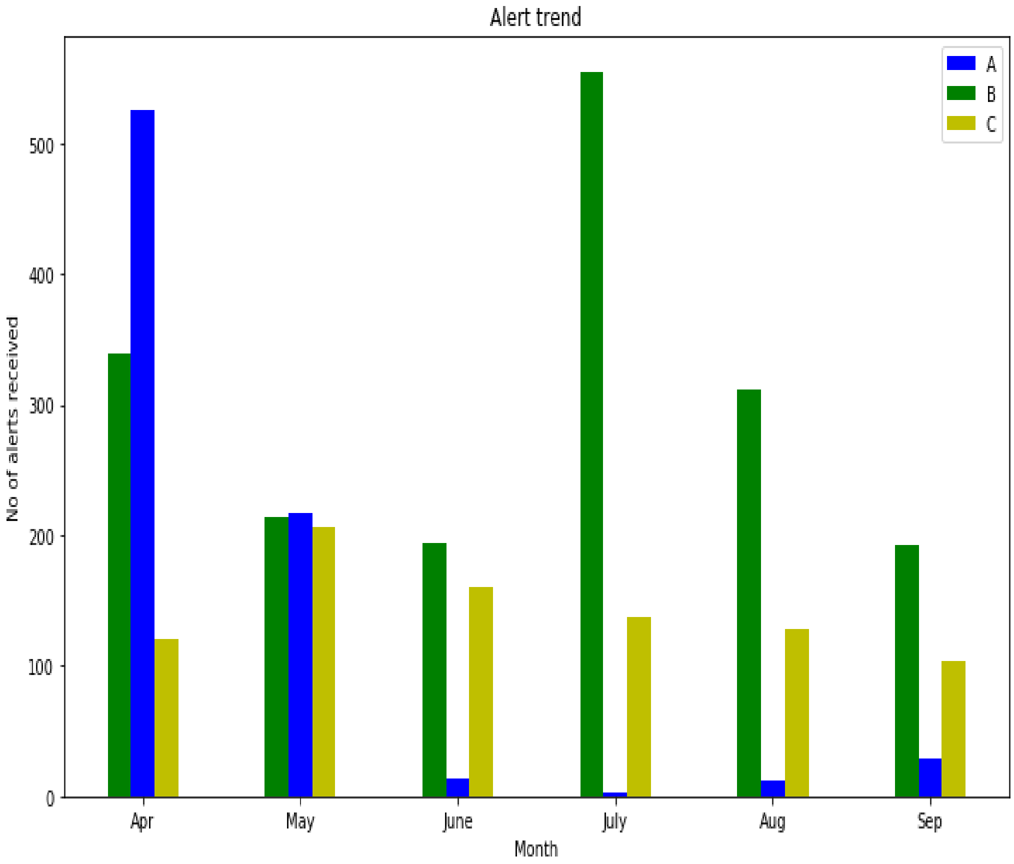
- This is one of the widely used plots, that we would have seen multiple times not just in data analysis, but we use this plot also wherever there is a trend analysis in many fields.
- Though it may seem simple it is powerful in analyzing data like **sales figures every week, revenue from a product**, **Number of visitors to a site on each day of a week**, etc.

Implementation:

- The bar plot is present in the **Matplotlib** package.

The code snippet is as follows:

```
plt.bar(x,y)
```



Some conclusions inferred from the above bar plot:

From the above bar plot we can conclude the following observations:

- We can visualize the data in a cool plot and can convey the details straight forward to others.
- This plot may be simple and clear but it’s not much frequently used in Data science applications.

3. Scatter Plot

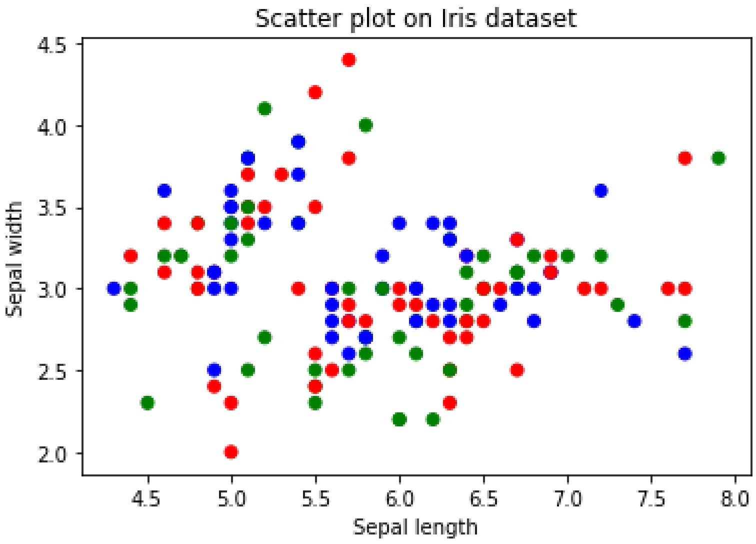
- It is one of the most commonly used plots used for visualizing simple data in Machine learning and Data Science.
- This plot describes us as a representation, where each point in the entire dataset is present with respect to any 2 to 3 features(Columns).
- Scatter plots are available in both 2-D as well as in 3-D. The 2-D scatter plot is the common one, where we will primarily try to find the patterns, clusters, and separability of the data.

Implementation:

- The scatter plot is present in the **Matplotlib** package.

The code snippet is as follows:

```
plt.scatter(x,y)
```



Some conclusions inferred from the above Scatter plot:

From the above Scatter plot we can conclude the following observations:

- The colors are assigned to different data points based on how they were present in the dataset **i.e, target column representation.**
- We can color the data points as per their class label given in the dataset.

Introduction to R

- R is a platform-independent programming language. This means that whichever operating system you use, your R program will work just fine.
- R is very easy to learn and understand. If you have a good understanding of statistics, R programming will make your tasks easier.
- R libraries provide one of the best and most insightful data visualizations.

- R programming is one of the most popular programming languages for data science and machine learning.
- R can easily be integrated with various other programming languages such as C and C++.
- R is a free language; anyone can download and use it without having to purchase a license. It is also open-source.
- The demand for R is growing at a very fast rate and it is currently a trend in the industry.
- R has a huge community of users and extensive community support to help you with the learning process.

Features of R

Some important features of R are as follows:

- It is a free and open-source programming language issued under GNU (General Public License).
- It has cross-platform interoperability which means that it has distributions running on Windows, Linux, and Mac. R code can easily be ported from one platform to another.
- It uses an interpreter instead of a compiler, which makes the development of code easier.

- It effectively associates different databases, and it does well in bringing in information from Microsoft Excel, as well as, Microsoft Access, MySQL, SQLite, Oracle, etc.
- It is a flexible language that bridges the gap between Software Development and Data Analysis.
- It provides a wide variety of packages with a diversity of codes, functions, and features tailored for data analysis, statistical modeling, visualization, Machine Learning, and importing and manipulating data.
- It integrates various powerful tools to communicate reports in different forms like CSV, XML, HTML, and pdf, and also through interactive websites, with the help of R packages.

Steps to perform Data Analysis in R

- **Import**: The first step is to import data into the R environment. It means that you take the data stored in files, databases, HTML tables, etc., and load it into an R data frame to perform data analysis on it.
- **Transform:** In this step, first, we make our data tidy by making each column a variable, and each row an observation. Once we have tidy data, we narrow down on

it to find observations of our interest, create new variables that are functions of existing variables, and find summary statistics of the observations.

- **Visualization:** It is used to make our data more understandable by representing data in graphical form. Visualization makes it easy to recognize patterns, find trends, and exceptions in our data. It enables us to convey information and results in a quick and visual way.
- **Model:** Models are complementary tools for visualization. These are fundamentally mathematical or computational tools used to answer questions related to our observations.
- **Communication:** In this last step of data analysis, we focus on communicating the results from visualization and modeling with others.R provides the ease to produce well-designed print-quality plots for sharing worldwide.

A lot of programmers choose R over Python these days. Here’s why:

- Even novices can start doing data analysis quickly on R, which was designed specifically keeping statisticians in mind.

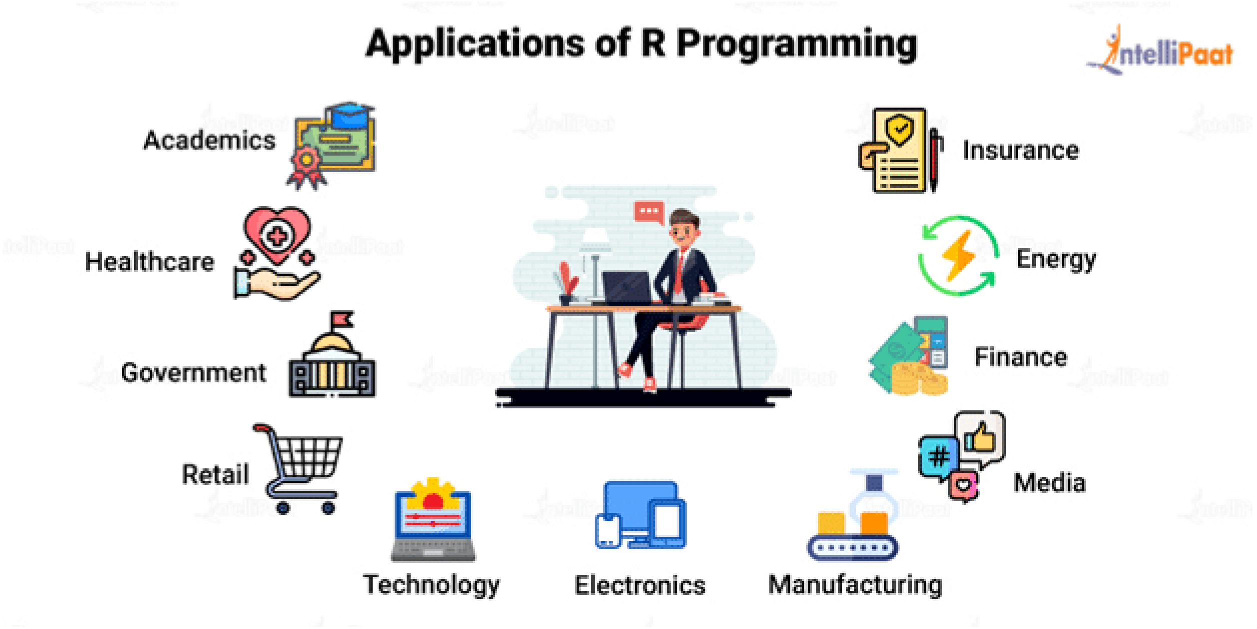
- R is better suited, as compared to Python, when it comes to statistical learning. R programming has exceptional libraries for exploring and experimenting with data.
- With amazing graphics, R is perfect for data visualization.

Now let us learn about some of the applications of R programming.

Applications of R Programming

- R is very widely used for data science. In addition to giving us an environment for statistical design, R programming also gives us many libraries for data science. Some of them are:
  - Dplyr
  - Ggplot2
  - Shiny
  - Lubridate
  - Knitr
  - Quanteda.*dictionaries*
  - RCrawler
  - Caret
  - RMarkdown
  - Leaflet
  - Janitor

- R also helps in importing and cleaning data and quantitative analysis.
- R has applications in a wide range of industries such as academics, healthcare, government, insurance, energy, finance, retail, media, manufacturing, technology, and electronics.



Visualization before analysis

Data visualization is basically representing the raw data in a visual format such as a **bar chart, pie chart, histogram, scatterplot**, etc. This is extremely important in this age of Big Data because it is very difficult to understand such large amounts of data without context. We can analyze Big Data using Data Analytics to obtain useful conclusions, but it’s best if those conclusions are presented in a format that humans can easily understand. And that’s where Data Visualization comes in!

urrently, Data Visualization has four major uses from an industrial standpoint. Let’s check them out:

1. Understand data quickly

Businesses can understand large amounts of data much more quickly and efficiently using Data Visualization. After all, it’s much easier to analyze and understand data if its in a visual form like a bar graph or pie chart rather than in a textual form like spreadsheets. Understanding data quickly also means that businesses can take decisions based on that data much more quickly as well.

2. Identify relationships and patterns

It is much easier to identify the relationships and patterns in the data when it is presented visually. Of course, there are some patterns that are obvious and immediately found, but there may be some hidden links and patterns in the data that you never thought were there. These are not visible when the data is in textual form and only becomes obvious when it is visually presented.

3. Pinpoint emerging trends

Businesses can obviously find out current trends in the data but it is sometimes possible to even estimate future trends using Data Visualization. This gives a huge edge to companies in the market that actually use Data Visualization as they can move ahead of their competitors by analyzing future market trends.

4. Communicate the story to others

It is not only enough that only data analysts and other technicians in the business understand the data. It is equally important to showcase the data analysis and results obtained to other people in the company such as the shareholders. In such a situation, Data Visualization is extremely helpful because it condenses the data into a form everybody can understand.

1. Bar Chart

Bar charts organize the data into rectangular bars that can easily be used to compare data sets. You should create a bar chart if you want to compare two or more data values of a similar kind and if you don’t have too many data groups to display. However, bar charts show discrete data so it might not be a good idea to use it if you want continuous data.

2. Line Chart

Line Charts visualize data in the form of a line that is very useful in understanding trends and patterns. It’s best to use the Line Chart if you want to show data relative to a continuous variable like time. Different colored lines for different variables in the data make it very easy to understand a line chart.

3. Scatterplot

Scatterplots are used to understand the relationship between two variables in the data. You can also find the outliers in your data or understand the overall distribution by plotting a scatterplot. If the data moves from lower left to upper right, there might be a positive correlation between the two variables if the data move in the opposite direction, there might be a negative correlation.

4. Sparkline

Sparklines are the best Data Visualization if you want to show general trends in a speedier manner. Sparkline is also useful if you want to show a particular data variable changing with time. It can paint an approximate picture which is very easy to understand but only if the readers can understand the Sparkline as well.

5. Pie Chart

Pie Charts are best if you want to compare some parts of the whole in the data. They can easily give an idea of the number of different parts on the whole but they are not very precise unless you add numerical values to each part of the pie chart representing each individual share on the whole.

6. Gauge

A Gauge is used to compare a value on a single scale. This value is usually specified as the current value and the total possible value with the gauge indicating your progress in green and the rest of the part in red. It is not a good idea to use a gauge if you want to display more than one value simultaneously.

7. Area Chart

Area Charts are similar to Line Charts in that they visualize data in the form of a line that is very useful in understanding trends and patterns. However, the area under the line in an are chart is also colored. This can be used with multiple variables in the data to demonstrate the relative differences between the variables.

8. Geographical Map

A Geographical Map visualizes the data on top of a map of a geographical location. It is best to use this map if the data has geography as an important part with different shades of the same color representing different data meanings on the map. However, if you want to show precise points of data, then a Geographical Map is not the best idea.

9. Heat Map

Heat maps provide the relationship between two variables in the data along with rating information between these variables. This rating information is normally displayed using various shades of the same color with light to dark demonstrating an increase in rating.

10. Histogram

Histograms are a cross between bar charts and line charts. They organize the data into rectangular bars across a continuous time interval. This is different than bar charts as they can be across discrete intervals. You should use Histograms to show the distribution of data over time or to compare two variables in the data over time.

THE END