DATA SCIENCE

SY. BSc.

UNIT I

TOPICS TO BE COVERED:

- What is data?
- What is data science?
- Example

Introduction

What is Data?

- Data is a collection of facts, such as numbers, words, measurements, observations or just descriptions of things.

- Data is different types of information that usually is formatted in a particular manner.

What is data science ?

Data science is a deep study of the massive amount of data , which involves extracting meaningful insights from raw, structured, and unstructured data.

Data science uses the most powerful hardware, programming systems, and most efficient algorithms to solve the data related problems.

Data science is all about:

- ✔ Asking the correct questions and analyzing the raw data.
- ✔ Modeling the data using various complex and efficient algorithms.
- ✔ Visualizing the data to get a better perspective.
- ✔ Understanding the data to make better decisions and finding the final result.

Example:

- Let suppose we want to travel from station A to station B by car. Now, we need to take some decisions such as which route will be the best route to reach faster at the location, in which route there will be no traffic jam, and which will be cost-effective. All these decision factors will act as input data, and we will get an appropriate answer from these decisions, so this analysis of data is called the data analysis, which is a part of data science.

IPython

- ✔ IPython stands for Interactive Python.
- ✔ IPython was originally developed by Fernando Perez in 2001 as an enhanced Python interpreter.
- ✔ It is an enhanced interactive environment for Python with many functionalities compared to the standard Python shell.
- ✔ IPython offers more features compared to the standard Python.

Features of IPython

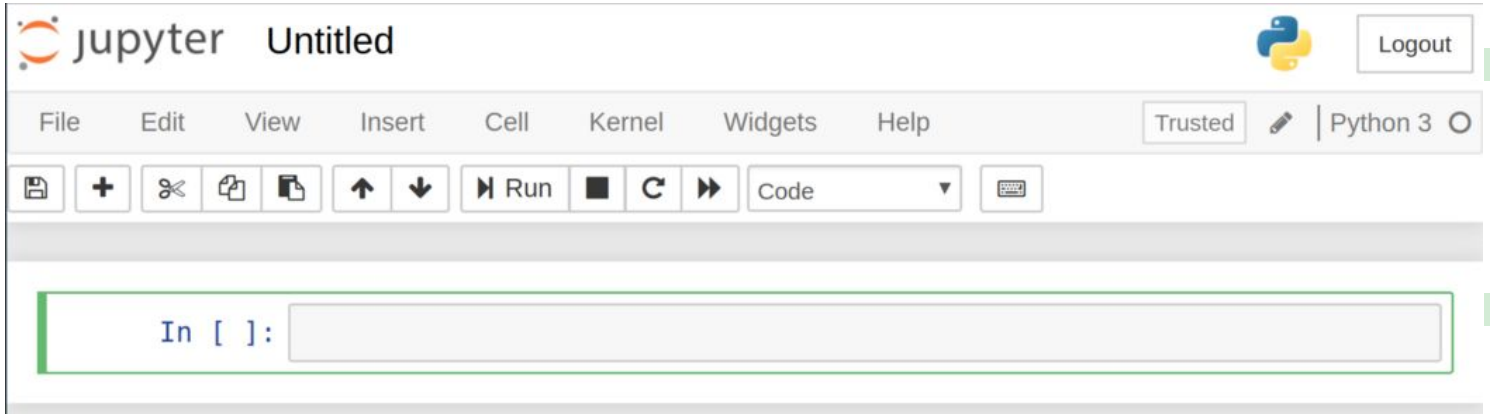
- Offers a powerful interactive Python shell.
- Syntax highlighting.
- Stores the history of interactions.
- Tab completion of keywords, variables and function names.

Jupyter

- A Jupyter Notebook is a powerful tool for interactively developing and presenting Data Science projects.
- It is a *server-client application* that allows you to edit your code through a web browser.

- Jupyter Notebooks integrate your code and its output into a single document.

- That document will contain the text, mathematical equations, and visualisations that the code produces directly in the same page.



Need of IPython and Jupyter in Data Science:

- Python is Open-source means it is free.
- A programming langauge that can be used in multiple domains.
- Calculations processing does not require too much time and its syntax is intuitive allowing for complex quantitative computations.
- Jupyter is a *server-client application* that allows you to edit your code through a web browser.
- Jupyter installation always comes with an installed Python kernel, and the

Exploratory Data Analysis (EDA)

- Exploratory data analysis is a simple classification technique usually done by visual methods.

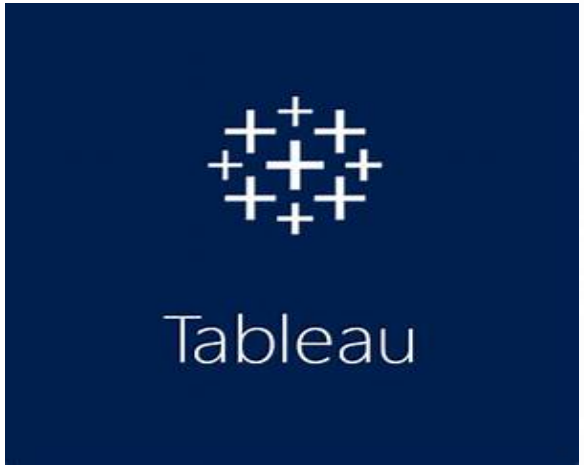
- *Exploratory data analysis (EDA) is a task of analyzing data using simple tools from statistics, simple plotting tools.*

What is the need of EDA?

- In the growing market, the size of data is also growing.
- It becomes harder for companies to make decision without proper analyzing it.
- Every machine learning problem solving starts with EDA.
- With the use of charts and certain graphs, one can make sense out of the data and check whether there is any relationship or not.
- Once **Exploratory Data Analysis** is complete and insights are drawn, its feature can be used for supervised and unsupervised machine learning modelling.
- Various plots are used to determine any conclusions. This helps the company to make a firm and profitable decisions.

How can we perform EDA?

- There are a lot of tools where one can perform EDA.



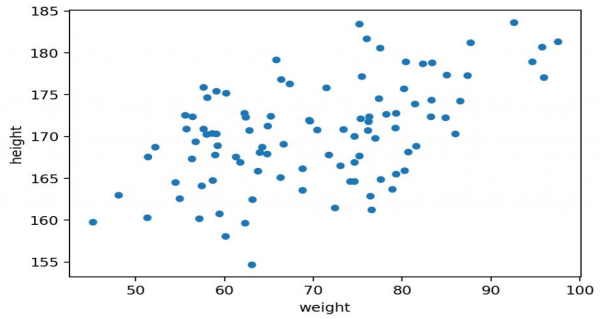
- Programming languages to perform EDA .

- EDA is considered to be the art part for **data scientist.**

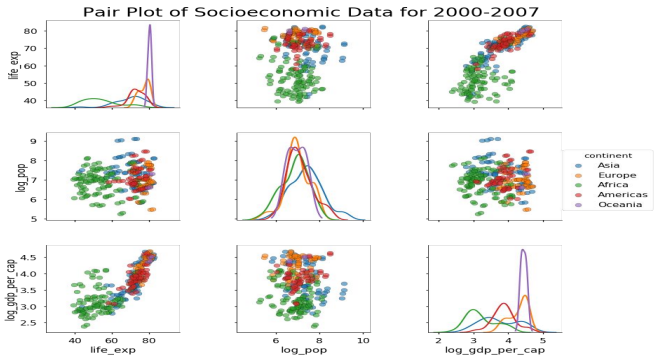
- The more creative we become with data more insights we can visualize.

graphs used while performing EDA:

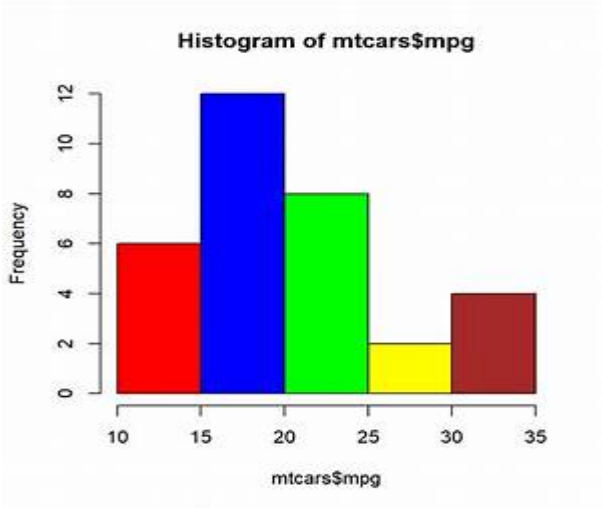
- **Scatter Plot**



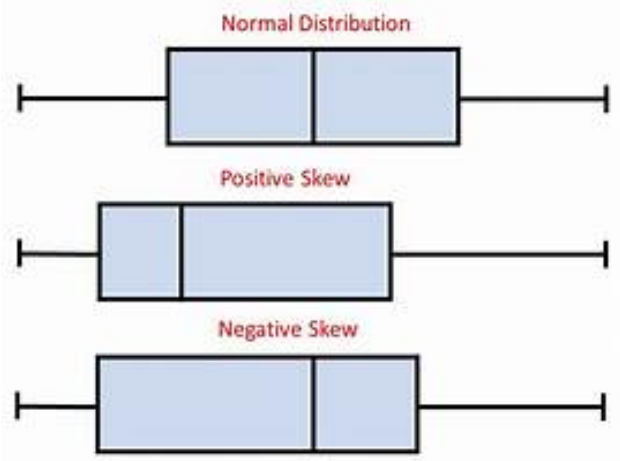
- **Pair plots**



- **Histogram**

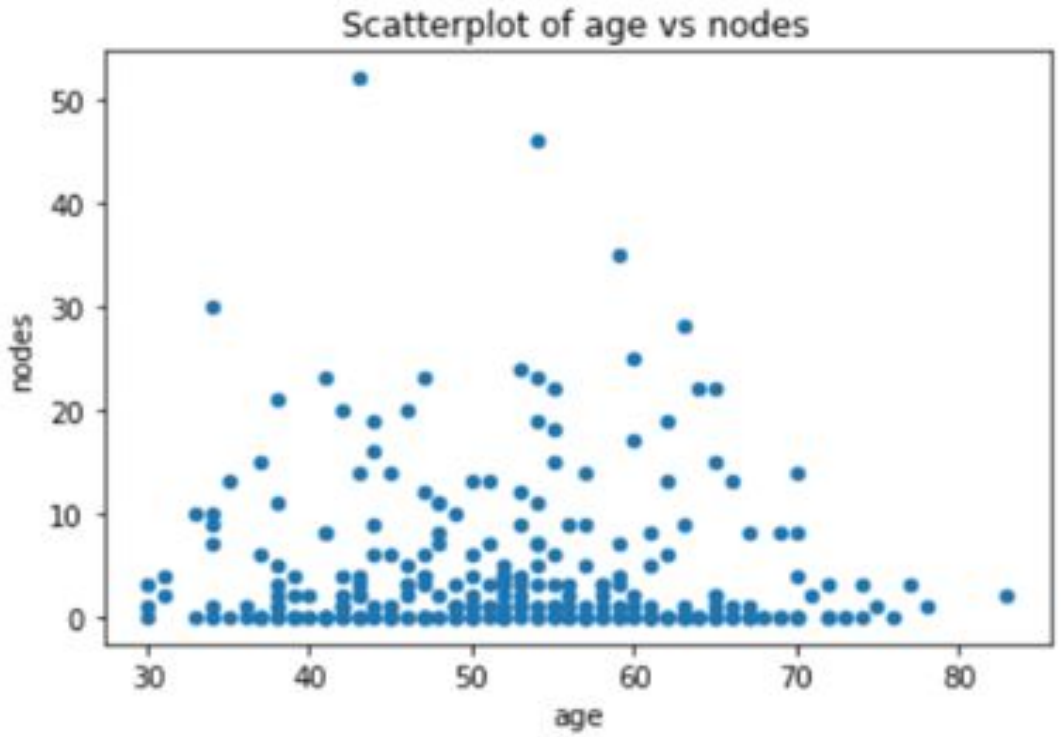


- **Box plots**



- **Scatter plot**: It is a type of plot which will be in a scatter format. It is mainly between 2 features.

```
In [5]:
1  haberdata.plot(kind='scatter',x='age',y='nodes')
2  plt.title('Scatterplot of age vs nodes')
3  plt.show()
```

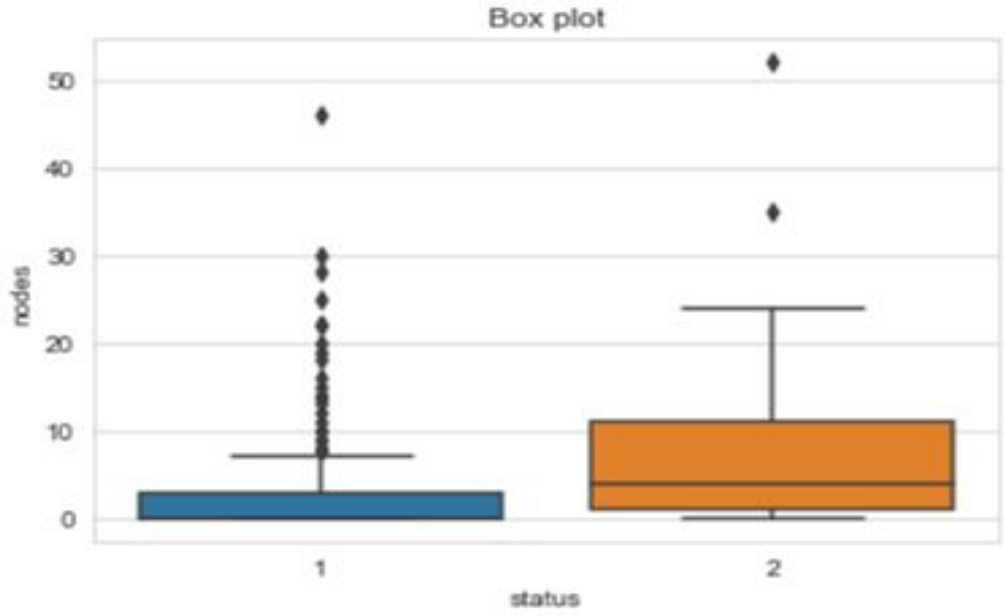


Pair plots: Used to see the behavior of all the features present in the dataset also we get to see the PDF representation.

```
In [7]:
1  sns.set_style('whitegrid')
2  sns.pairplot(haberdata,hue='status',height=3)
3  plt.show()
```

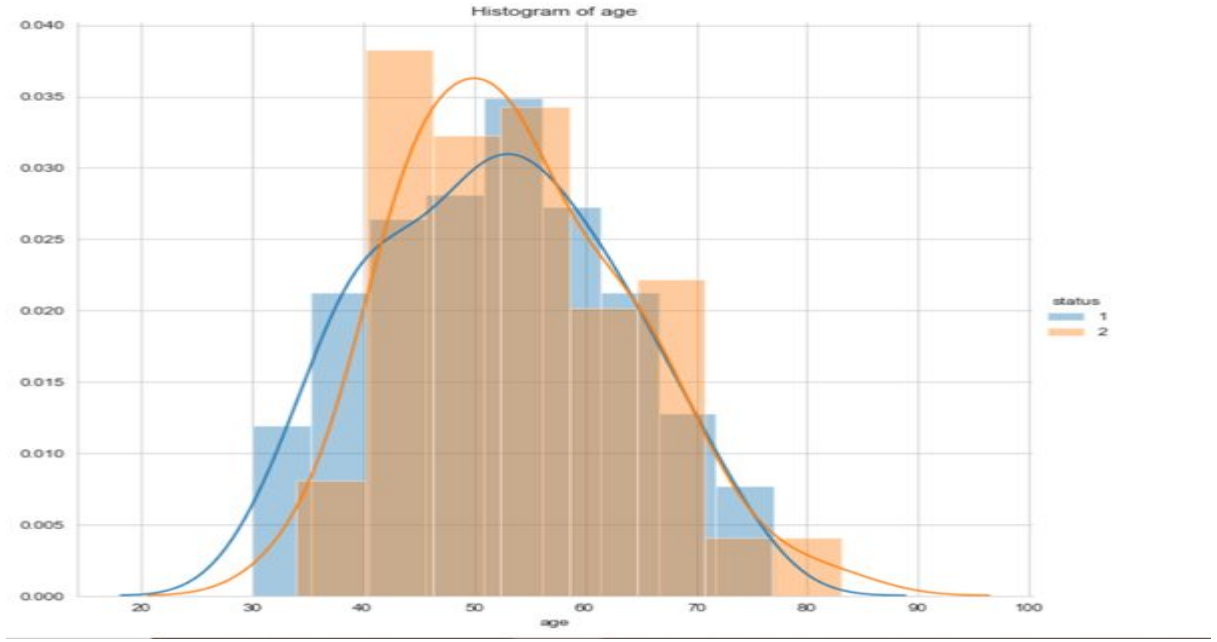
Box-Plots: Box plots tell us the percentile plotting.

In [16]:
```
#Box plot
sns.boxplot(x='status',y='nodes',data=haberdata)
plt.legend
plt.title('Box plot')
plt.show
```



Histogram: Histogram plots are used to depict the distribution of any continuous variable.

```
In [9]:
1  #PDF for age
2  sns.FacetGrid(haberdata,hue='status', height=8)\
3  .map(sns.distplot,'age')\
4  .add_legend()
5  plt.title('Histogram of age')
6
```



For performing EDA, we need to import some libraries.

- NumPy, Pandas, Matplotlib and seaborn.

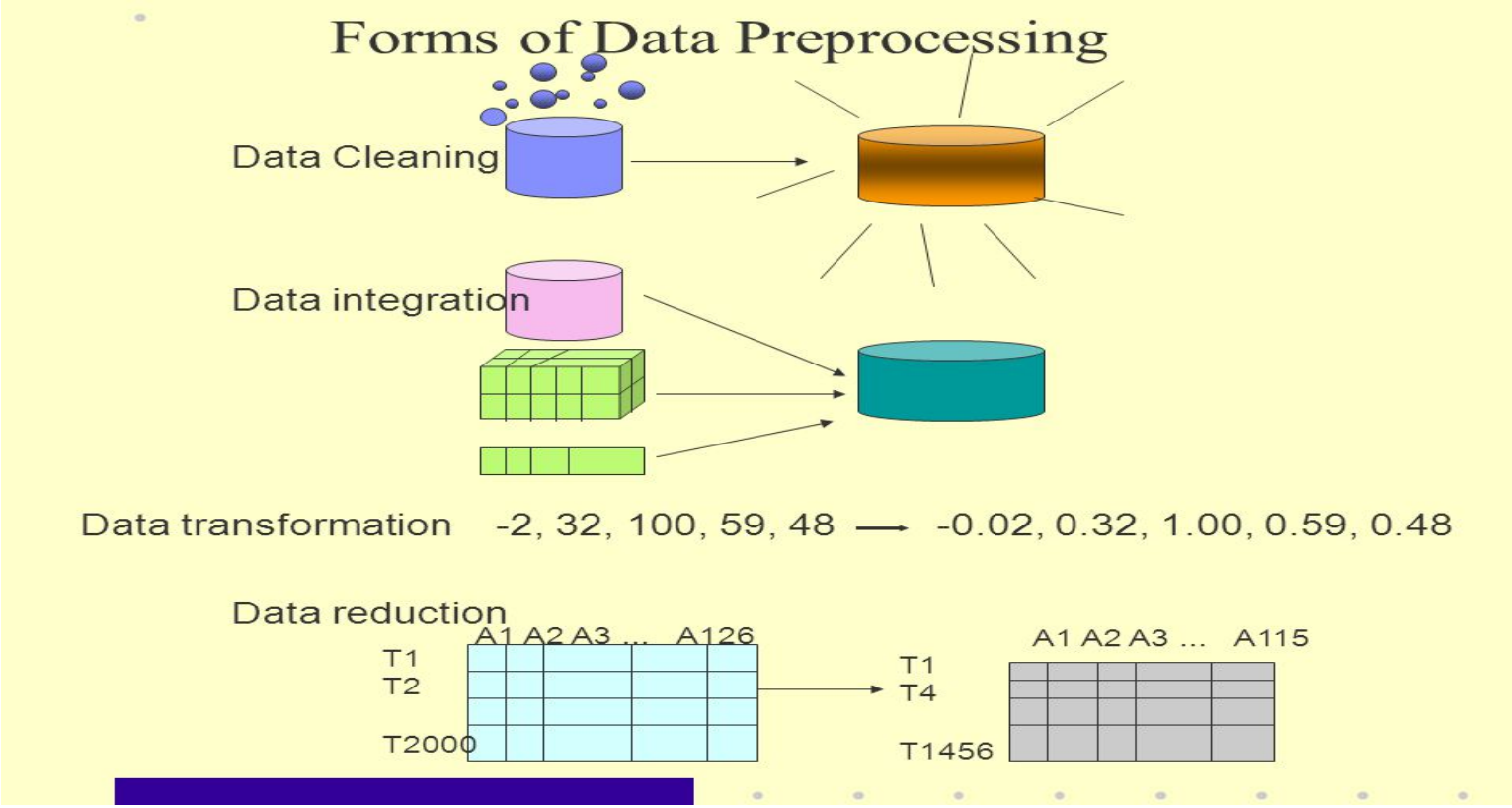
```
In [1]:
1 import pandas as pd
2 import seaborn as sns
3 import matplotlib.pyplot as plt
4 import numpy as np
5 haberdata=pd.read_csv('haberman.csv')
6 #haberdata
```

```
In [3]:
1 print(haberdata.head())
```

	age	year	nodes	status
0	30	64	1	1
1	30	62	3	1
2	30	65	0	1
3	31	59	2	1
4	31	65	4	1

Pre-processing data

- Data preprocessing is a data mining technique to turn the raw data gathered from diverse sources into cleaner information that's more suitable for work.

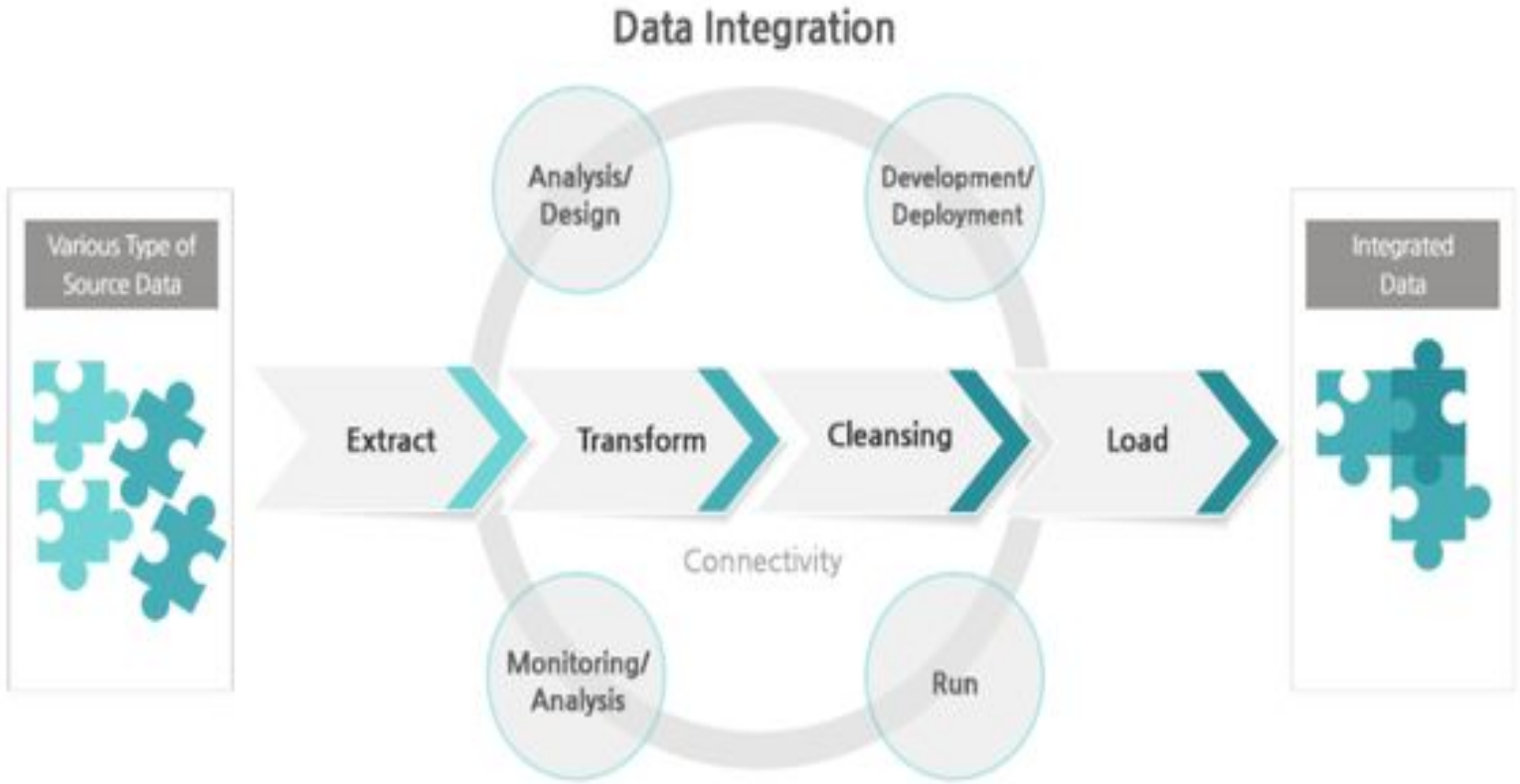


Forms of Data pre-processing

- **Data cleaning:**
  Data Cleaning means the process of identifying the incorrect, incomplete, inaccurate, irrelevant or missing part of the data and then modifying, replacing or deleting them according to the necessity.

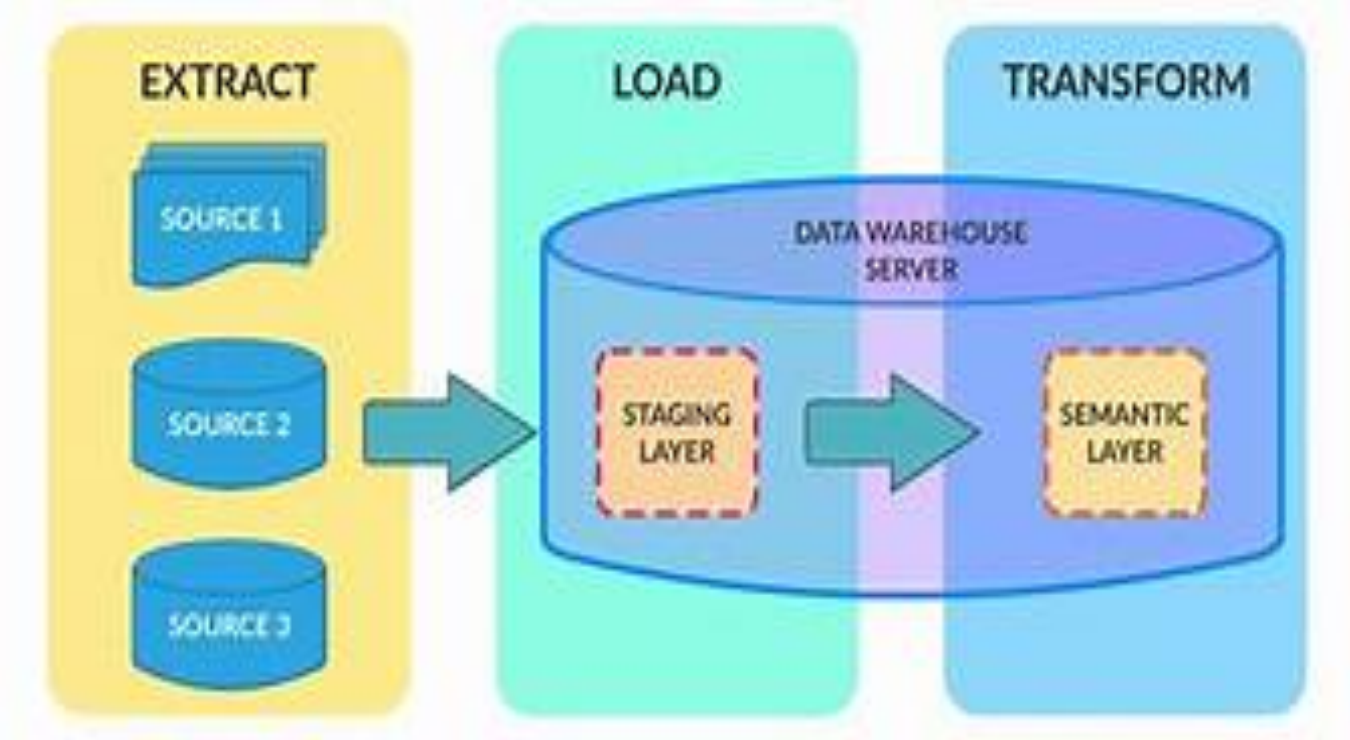
Data Integration

- **ETL** is used to integrate the data with the help of three steps **Extract, Transform, and Load**, and it is used to blend the data from multiple sources.



Extract:

- **Extract** is the process of fetching (reading) the information from the database. At this stage, data is collected from multiple or different types of sources.

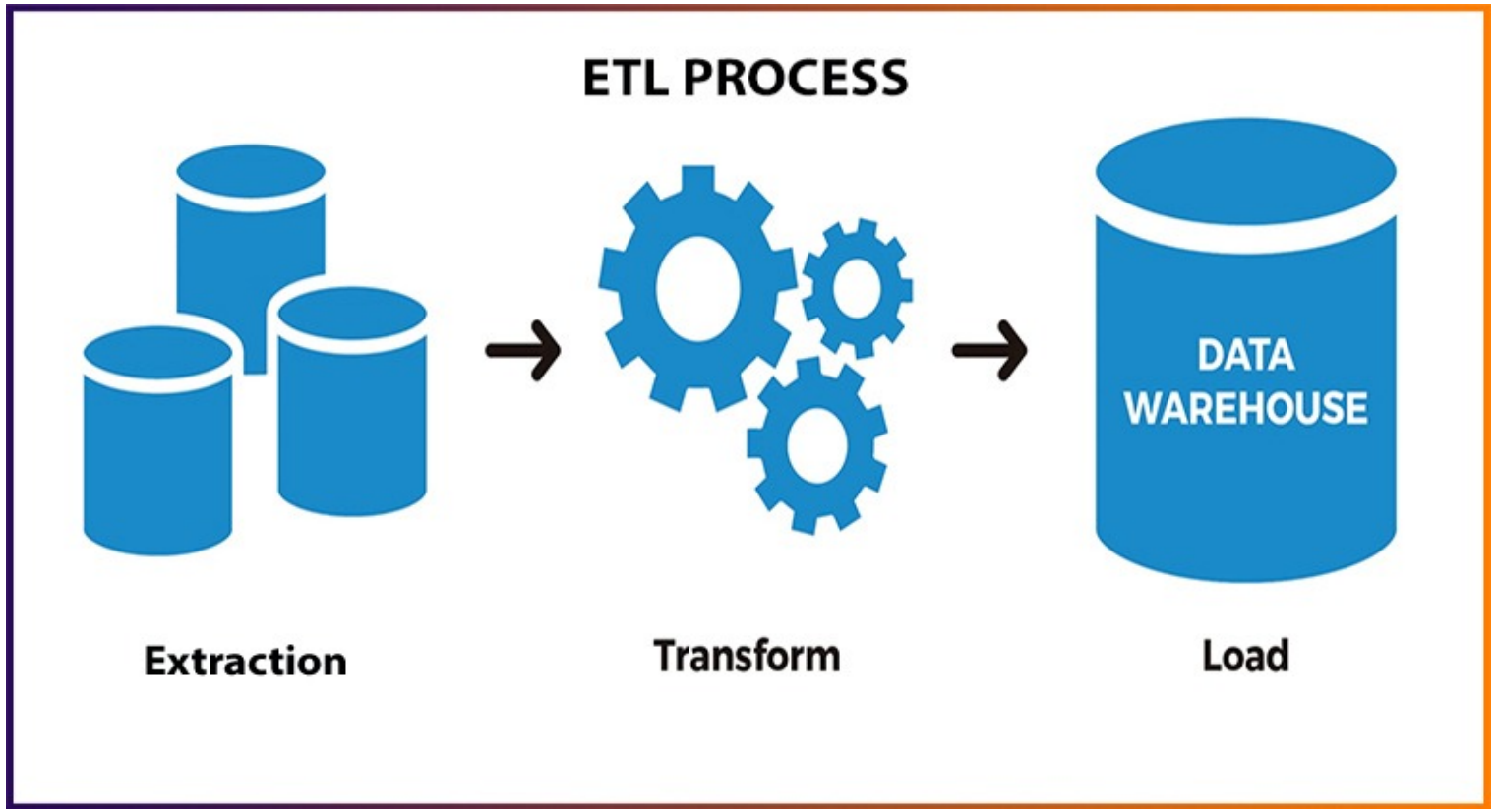


Transform:

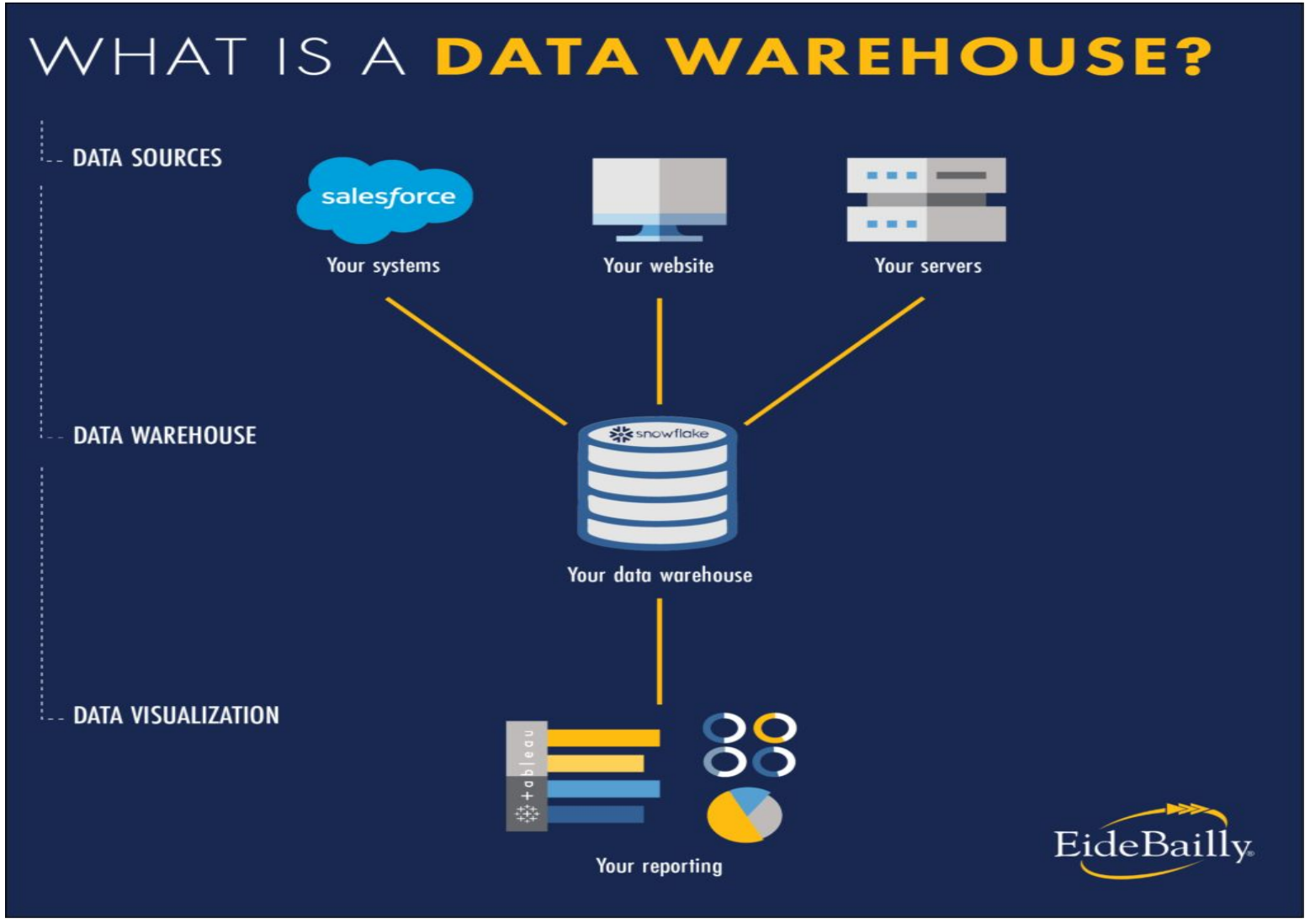
- ***Transform*** is the process of converting the extracted data from its previous form into the required form. Data can be placed into another database. Transformation can occur by using rules or lookup tables or by combining the data with other data.

Load:

- Load is the process of writing the data into the target database.



Data Warehouse:



Regular Expression in python

- A **Regular Expressions (RegEx)** is a special sequence of characters that uses a search pattern to find a string or set of strings.

- It can detect the presence or absence of a text by matching with a particular pattern, and also can split a pattern into one or more sub-patterns.

Metacharacter's

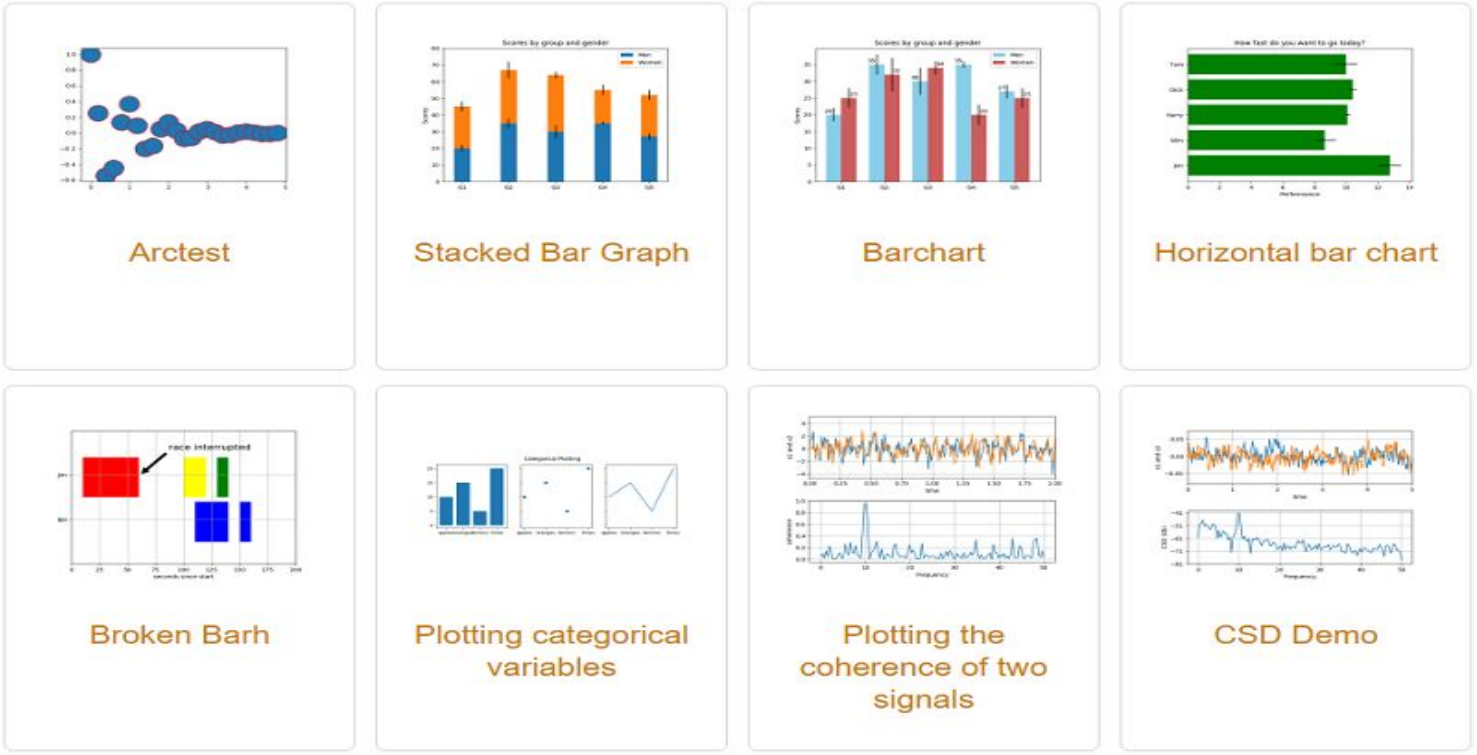
Metacharacter	Description	Description
\	whole numbers(0-9) (single digit)	\d = 7, \d\d=77
[a-z] or [0-9]	character set	geek[sy] = geeky geek[sy] != geeki
\w	alphanumeric character	\w\w\w\w = geek
*	0 or more characters	s* = _,s,ss,sss,ssss…..
+	1 or more characters	s+ = s,ss,sss,ssss…..
?	0 or 1 character	s? = _ or s
{m}	occurs “m” times	sd{3} = sddd

{m,n}	min “m” and max “n” times	sd{2,3}=sdd or sddd
\W	symbols	\W = %
[a-z] or [0-9]	character set	geek[sy] = geeky geek[sy] != geeki

Basic and Specialized visualization tools in Python

- An overview of the best Python data visualization tools, libraries, and software solutions.

- Matplotlib



- Matplotlib is one of the most popular and oldest data visualization tools using Python. It is a quite powerful but also a complex visualization tool.

- Matplotlib is a Python 2D plotting library that provides publication quality figures in a variety of hardcopy formats and interactive environments across many platforms.

- Matplotlib is used in Python scripts, the Python and IPython shells, the Jupyter notebook, web application servers, and four graphical user interface toolkits.

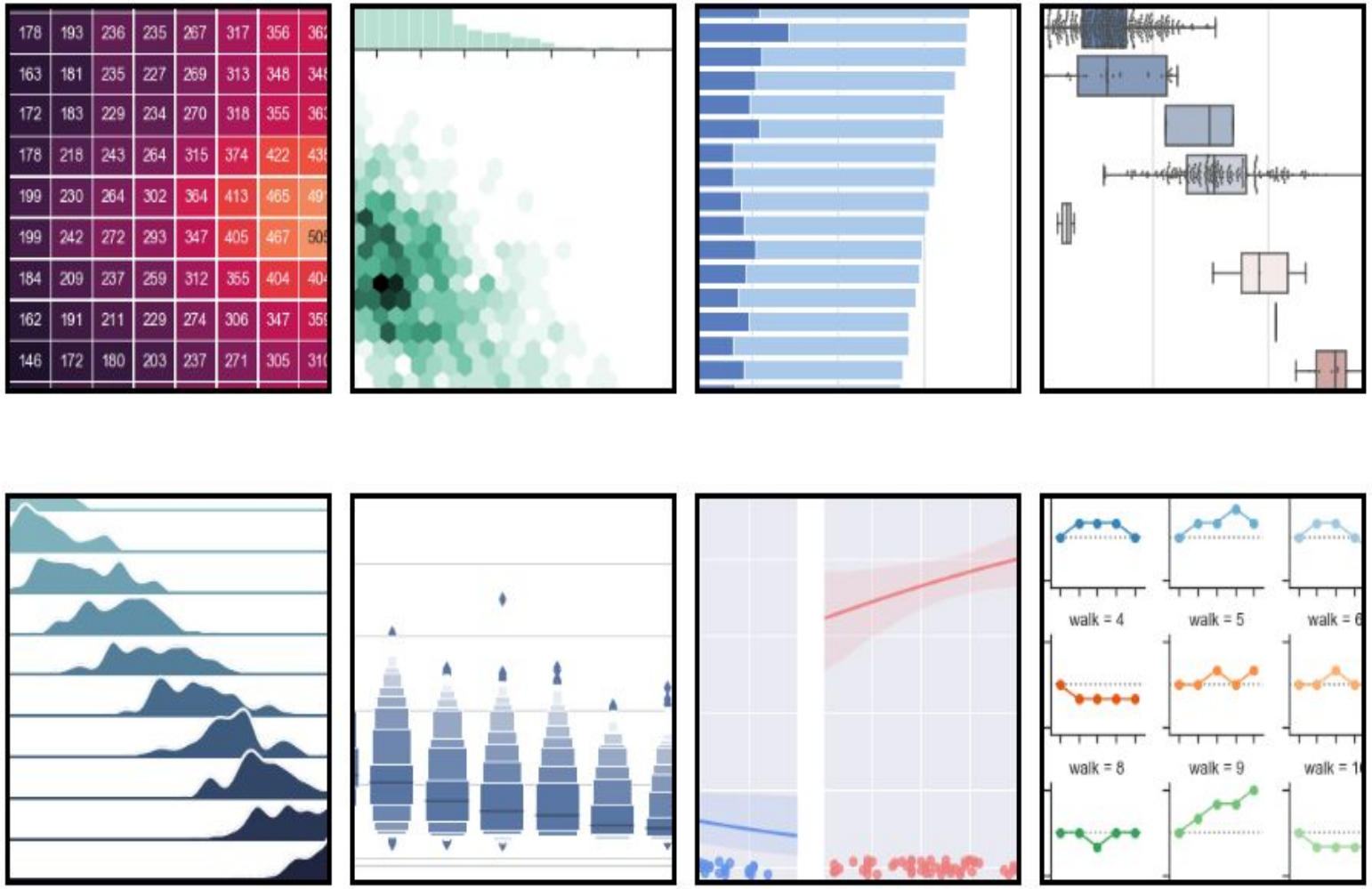
Key features and benefits:

- You can generate and use plots, bar charts, pie charts, 3D plotting, error charts, histograms, power spectra, scatter plot, etc.

- An object-oriented interface and a set of functions familiar to MATLAB to control your line styles, axes properties, font properties, etc.

- A great examples gallery and a list of plotting commands to help you learn how to do a particular kind of plot.

Seaborn


178 193 236 235 267 317 356
163 181 235 227 269 313 348
172 183 229 234 270 318 355
178 218 243 264 315 374 422
199 230 264 302 364 413 465
199 242 272 293 347 405 467
184 209 237 259 312 355 404
162 191 211 229 274 306 347
146 172 180 203 237 271 305
walk = 4
walk = 5
walk = 8
walk = 9

- Seaborn is also one of the very popular Python visualization tools and is based on Matplotlib. Seaborn is thin wrappers over Matplotlib.

- It is a good software program for those who want a high-level interface for creating beautiful, attractive, and informative statistical types of graphs and charts.

- Seaborn is able to build default data visualizations in a more visually appealing way. One of the best Seaborn's benefits is that it can make complex and complicated plots simpler to build.

- **Key features and benefits:**

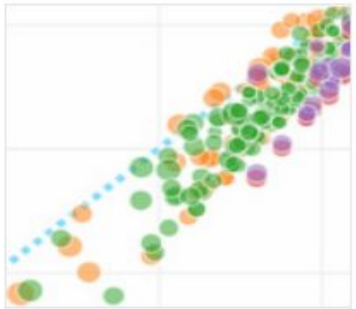
- A higher-level Python visualization library based on the Matplotlib library.

- A great range of settings for processing graphs and charts.

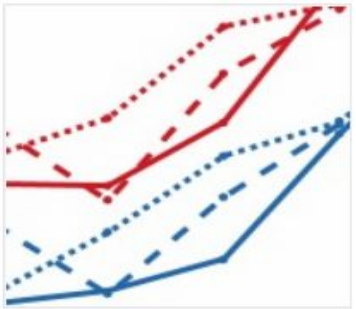
- Easily displays distributions and data relationships.

- Can show information from matrices and Data Frames.

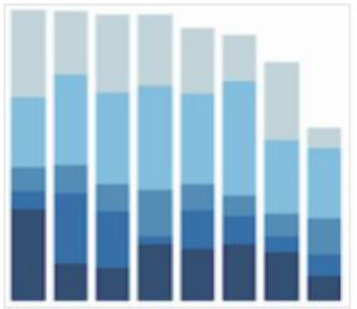
Plotly



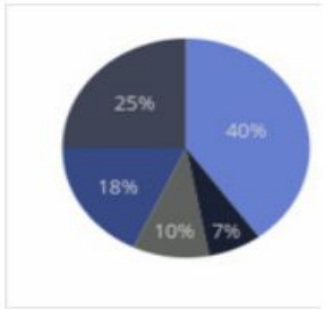
Scatter Plots



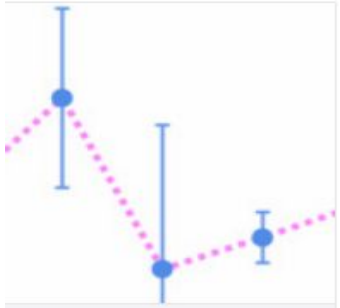
Line Charts



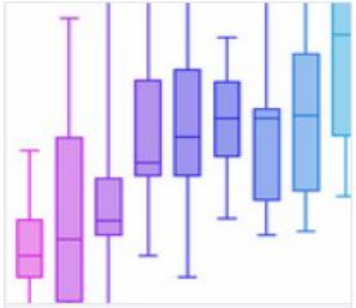
Bar Charts



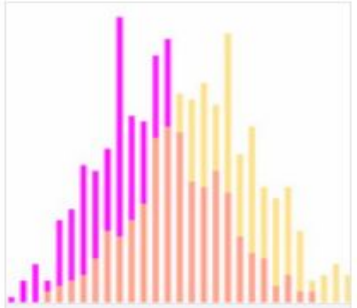
Pie Charts



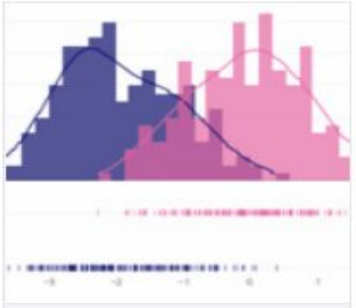
Error Bars



Box Plots



Histograms



Distplots

- When it comes to interactive Python visualization tools, Plotly has a top place here.

- Plotly's Python free and open source graphing library help you create interactive, publication-quality graphs easily online.

- Plotly has it all – 3D data visualization, line plots, bar charts, error bars, scatter plots, area charts, box plots, multiple-axes, histograms, heatmaps, subplots, polar charts, and bubble charts.

Key features and benefits:

- A comprehensive range of rich and interactive graphs.
- The interactive plots allow you to seamlessly explore the data by panning, selecting, zooming on the graphing surface.
- Ternary plots and 3D charts.
- Supports for “multiple linked views” and animation.

Bokeh

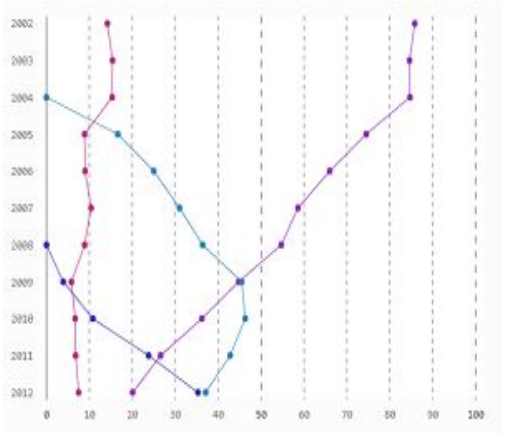

Time Spreads for Sprint Medalists (by Year)
Time (seconds)
United States
United Kingdom
Japan
China
Germany
India
Italy
Australia
Brazil
France
Taiwan
Spain
Categorical Heatmap
MPG by # Cylinders
Fruit Counts by Year
2015
2016
2017
Apples
Pears
Nectarines
Plums
Grapes
Strawberries

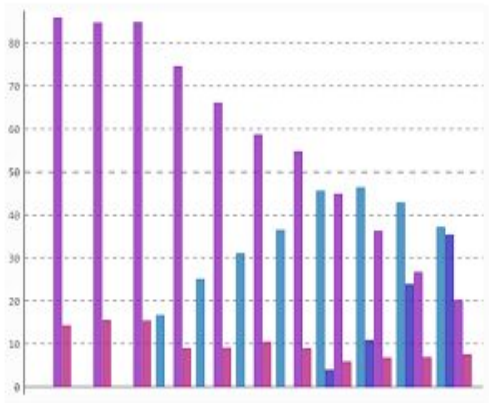
- Bokeh also is an interactive Python visualization library tool that provides elegant and versatile graphics.
- It is able to extend the capability with high-performance interactivity and scalability over very big data sets.
- Bokeh allows you to easily build interactive plots, dashboards or data applications.
- Bokeh also allows you to create network graph visualizations and geographical data such as Google Maps, GeoJSON, Tile Rendering.

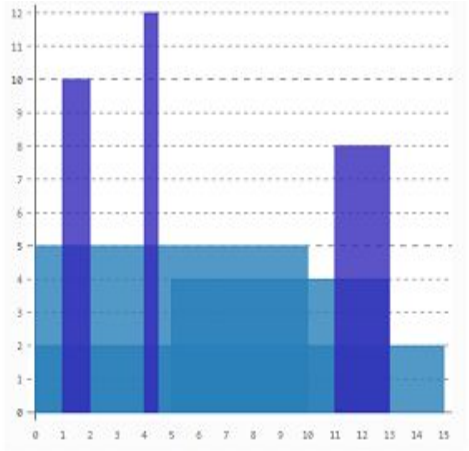
Key features and benefits:

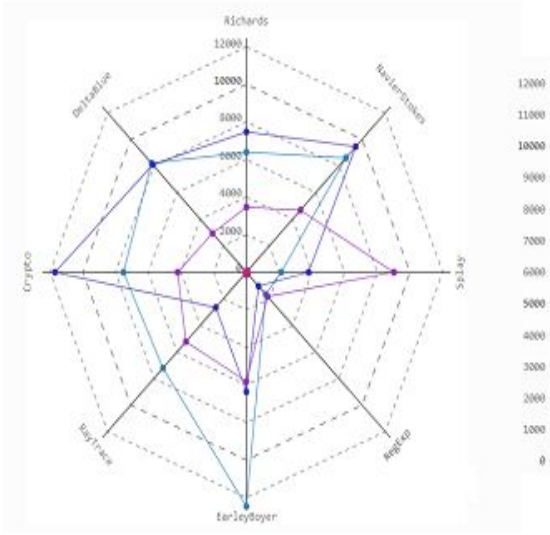
- Able to combine multiple plots and widgets.
- Handles categori
- cal data with different techniques such as bar charts and categorical heatmaps.
- Uses interactive tools such as pan, zoom, and select, on your plots.
- Working with geographical data.

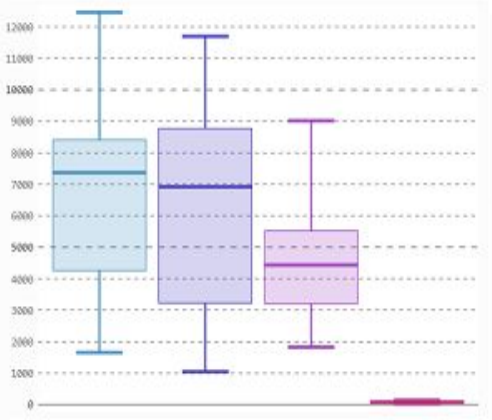
Pygal

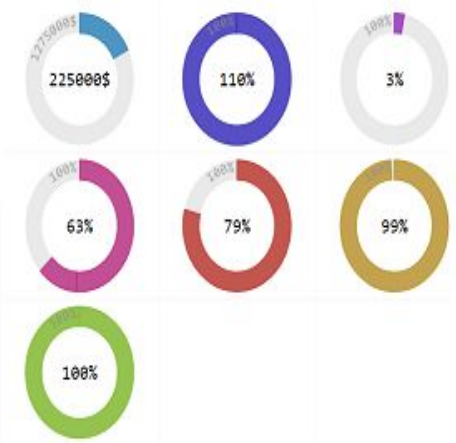







Richards
Deltablue
Raytrace
Splay
Earleyboyer




225000$
110%
3%
63%
79%
99%
100%

- Pygal concentrate on allowing you to create SVGs. SVG formatting is integrated greatly with Django and Flask.

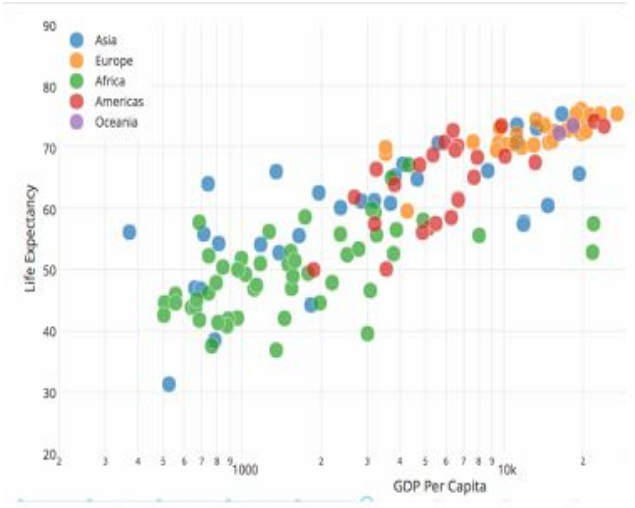
- you can easily create a wide variety of graph and charts such as line graph, bar chart, histogram, XY, pie charts, box plot.

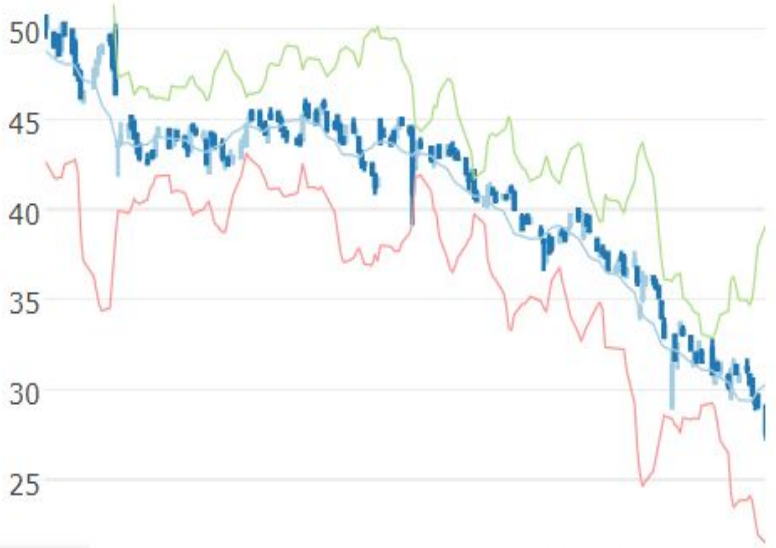
- Pygal is ideal for smaller datasets although it can handle big data sets too.

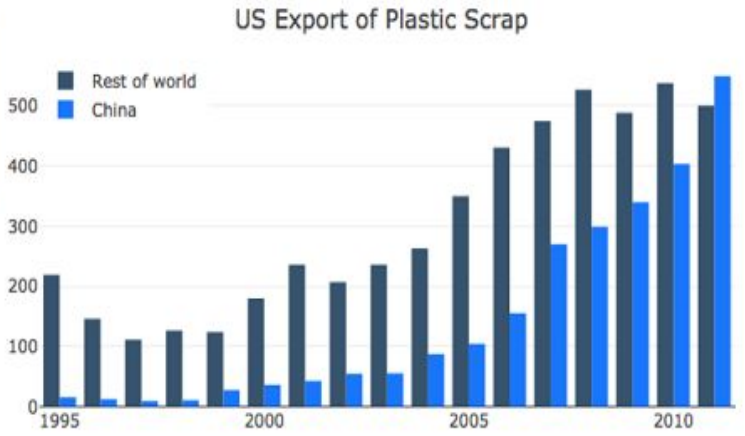
Key features and benefits:

- Three ways to style the charts – built-in styles, parametric styles, and custom styles.
- Provides good-looking interactive data visualizations.
- Pygal also supports an HTML table export.
- A lot of options for charts configuration as sizing, titles, labels, legend, axis, interpolations.

Dash


Asia
Europe
Africa
Americas
Oceania
Life Expectancy
GDP Per Capita
1000
10k




US Export of Plastic Scrap
Rest of world
China
1995
2000
2005
2010

- Dash is a Python framework for building web applications.
- It is perfect for creating data visualization apps with highly custom user interfaces in Python.
- Dash is written on Flask, Plotly.js, and React.js. Dash apps are rendered in the web browser and also mobile-ready.
- Dash is an open source library.

Key features and benefits:

- Dash apps are built and published on the Web, so you have the full power of CSS.
- Open source – you can run the awesome Dash on your desktop for free.
- Provides modern UI elements like sliders, dropdowns, and graphs to your analytical Python code.

Introduction to Linear Algebra:

- Linear algebra is the study of linear combinations. It is the study of vector spaces, lines and planes, and some mappings that are required to perform the linear transformations.

- The linear regression equation can be expressed in the following form:

$$Y=a_1x_1 + a_2x_2.....+a_nx_n + b$$

a1, a2, a3, ... an are the coefficients.
b is the parameter of the model.

x1, x2, x3, ... xn are the features.
y is the target variable.

Application of Linear Algebra :

- **Game Theory**

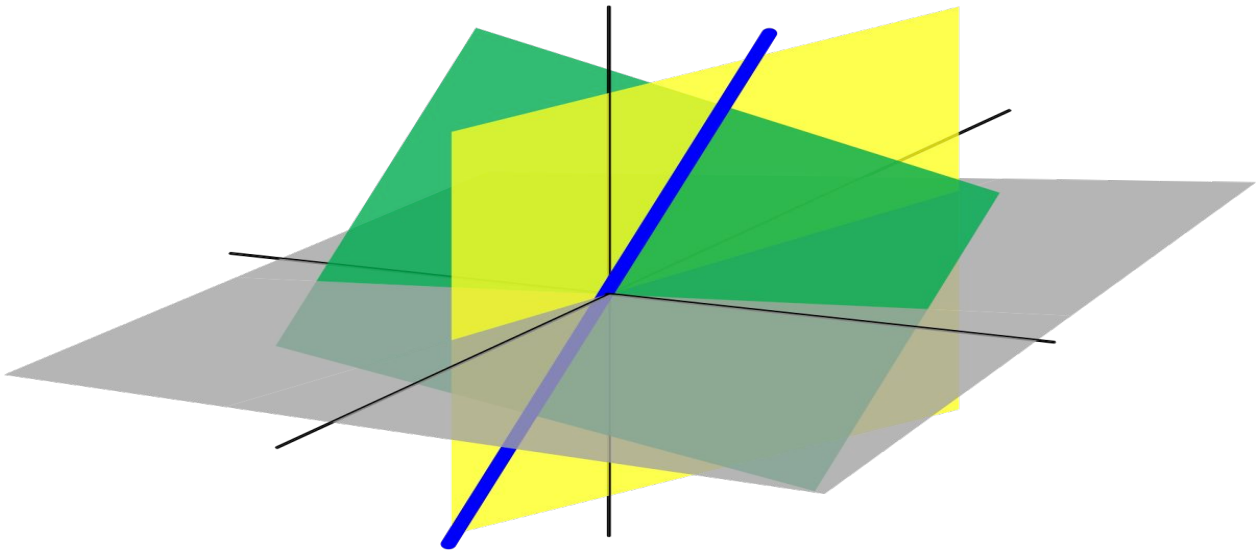
It is one of the applications of linear algebra, which is a mathematical study that describes the number of possible options. The players make these options during the game playing. As per the psychologists, the social interaction theory is used to consider the player's options against other players in the competition.

linear algebra, concept is also used in:

- **Ranking in Search Engines** – One of the most important applications of linear algebra is in the creation of Google. The most complicated ranking algorithm is created with the help of linear algebra.
- **Signal Analysis** – It is massively used in encoding, analyzing and manipulating the signals that can be either audio, video or images etc.
- **Linear Programming** – Optimization is an important application of linear algebra which is widely used in the field of linear programming.
- **Error-Correcting Codes** – It is used in coding theory. If encoded data is tampered with a little bit and with the help of linear algebra it should be recovered. One such important error-correcting code is called hamming code
- **Prediction** – Predictions of some objects should be found using linear models which are developed using linear algebra.
- **Facial Recognition-** An automated facial recognition technology that uses linear algebraic expression is called principal component analysis.
- **Graphics-** An important part of graphics is projecting a 3-dimensional scene on a 2-dimensional screen which is handled only by linear maps which are explained by linear

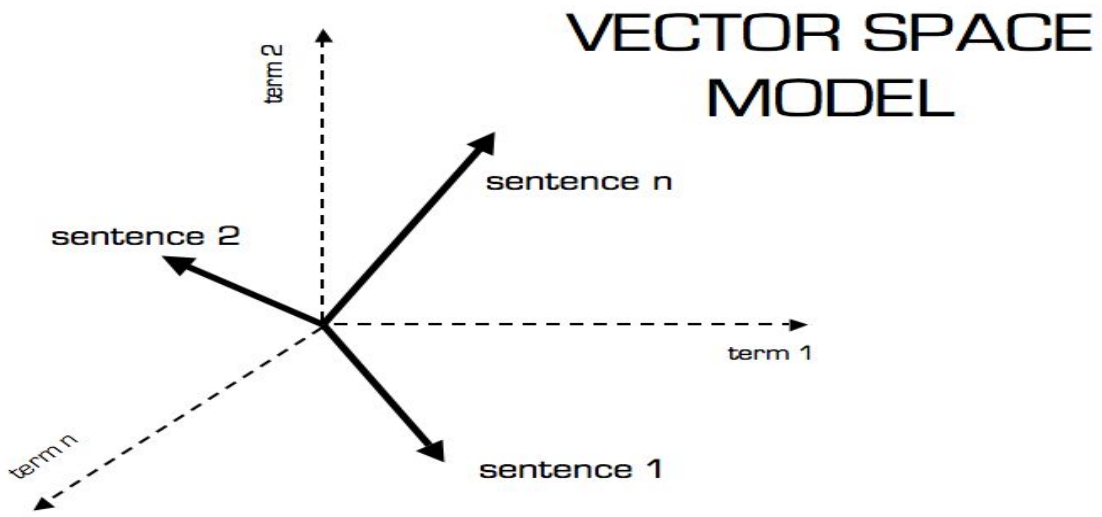
Main concepts which are the prerequisite to linear algebra are explained in detail :

- **VECTOR:** A vector is a quantity that has two independent properties called magnitude and direction.



Vector space :

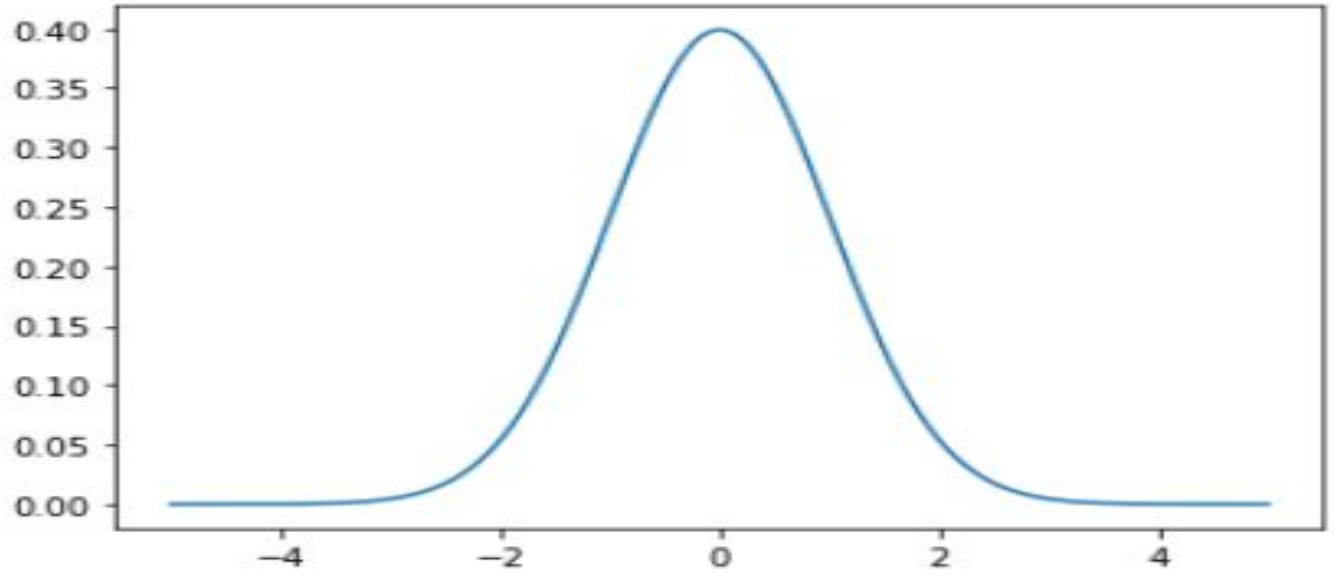
A vector space consists of a set of objects called vectors, which can be added together and multiplied by the numbers called scalars.



Gaussian Distribution Training and Testing Data

- **What is normal or Gaussian distribution?**

When we plot a dataset such as a histogram, the shape of that charted plot is what we call its distribution. The most commonly observed shape of continuous values is the bell curve, which is also called the Gaussian or no



It is named after the German mathematician, Carl Friedrich Gauss. Some common example datasets that **follow Gaussian distribution are:**

- Body temperature
- People's Heights
- Car mileage
- IQ scores

The Gaussian Function:

Fit the data to the Gaussian function.

Goal is to find the values of A and B that best fit our data.

```
#Define the Gaussian function

def gauss(x, H, A, x0, sigma):
    return H + A * np.exp(-(x - x0) ** 2 / (2 * sigma ** 2))
```

```
from __future__ import print_function
import numpy as np
import matplotlib.pyplot as plt
from scipy.optimize import curve_fit
xdata = [ -10.0, -9.0, -8.0, -7.0, -6.0, -5.0, -4.0, -3.0, -2.0, -1.0, 0.0, 1.0, 2.0, 3.0, 4.0, 5.0,
    6.0, 7.0, 8.0, 9.0, 10.0]
ydata = [1.2, 4.2, 6.7, 8.3, 10.6, 11.7, 13.5, 14.5, 15.7, 16.1, 16.6, 16.0, 15.4, 14.4, 14.2,
    12.7, 10.3, 8.6, 6.1, 3.9, 2.1]

# Recast xdata and ydata into numpy arrays so we can use their handy features
xdata = np.asarray(xdata)
ydata = np.asarray(ydata)
plt.plot(xdata, ydata, 'o')

# Define the Gaussian function
def Gauss(x, A, B):
    y = A*np.exp(-1*B*x**2)
    return y
parameters, covariance = curve_fit(Gauss, xdata, ydata)

fit_A = parameters[0]
fit_B = parameters[1]

fit_y = Gauss(xdata, fit_A, fit_B)
plt.plot(xdata, ydata, 'o', label='data')
plt.plot(xdata, fit_y, '-', label='fit')
plt.legend()
```