Data Exploration

Types of Data

Numerical Data

Categorical Data

Ex:

- Salary
- Age
- Weight

Ex:

- Male / Female
- Yes / No
- Rating

Numerical Data

Weight

Continuous Data

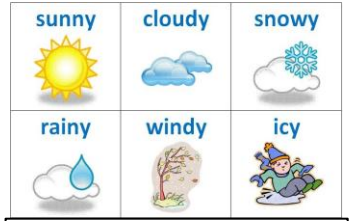
Data Changes with No Limit

Discrete Data

Data within some numerical Range

SAT Score

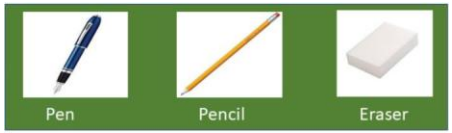
Categorical:



Categorical Data

Nominal Data

Ordinal Data



Categories without mathematical weightage

Data with mathematical weightage

4

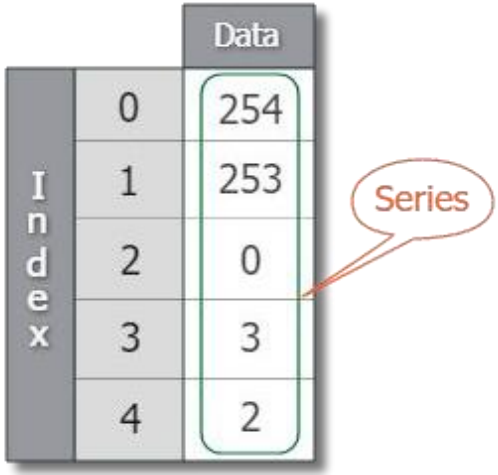
An open-source Python Library

Used for high-performance data manipulation and analysis

Data Types in Pandas

Series

- 1-Dimensional Homogeneous Data



DataFrame

- 2-Dimensional Heterogeneous Data

	Name	Symbol	Shares
0	Microsoft Corporation	MSFT	100
1	Google, LLC	GOOG	50
2	Tesla, Inc.	TSLA	150
3	Apple Inc.	AAPL	200
4	Netflix, Inc.	NFLX	80

Loading .csv File in Python Program

titanic Dataset

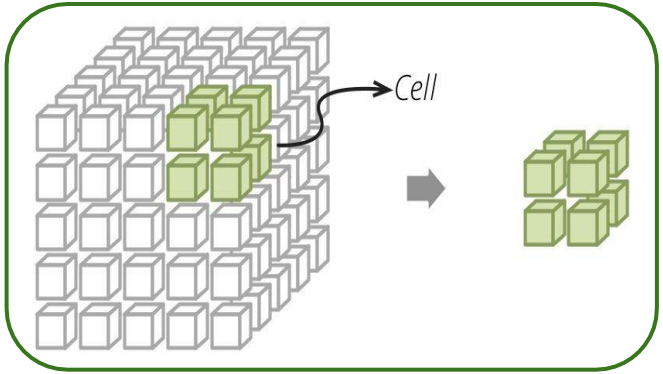
Python Code

```
import pandas as pd
Data = pd.read_csv("titanic.csv")
Data.head()
```

Data Accessing Methods in Pandas

1. Indexing

2. Slicing



3. Filtering

Indexing Method

Accessing single column

df[“<column_Name >”]

Python Code:

```
import pandas as pd
Data = pd.read_csv("titanic.csv")
print(Data["Name"])
```

Accessing Multiple columns

df[[“<column1>”,“<column2>,..”]]

Python Code:

```
import pandas as pd
Data = pd.read_csv("titanic.csv")
Data[["PassengerId","Name"]]
```

Indexing Method

⟹ **.loc function**

Access Rows

df.loc[“row1”]

Ex:

```
import pandas as pd
dat = pd.read_csv("titanic.csv")
dat.head()
dat.loc[1]
```

Access Rows & Columns

df.loc[[“row1”,“row2”],[“col1”,“col2”]]

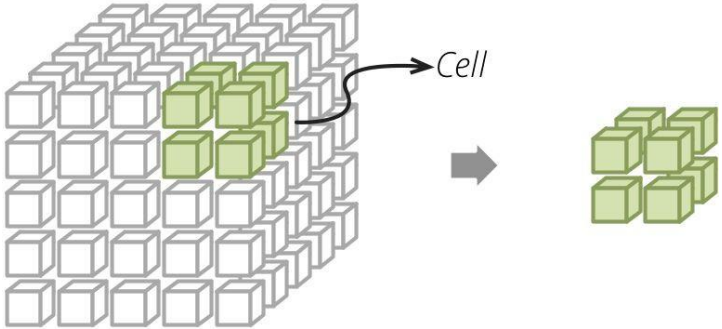
Ex:

```
import pandas as pd
dat = pd.read_csv("titanic.csv")
dat.head()
dat.loc[[1,2,3,4,5],["Name","Age"]]
```

Slicing Method

.iloc function ➔ Access Rows & Columns using index range.

df.iloc[<row_range>,<col_range>]



Python Code

```
import pandas as pd
Data = pd.read_csv("titanic.csv")
Data.iloc[0:2,0:3]
```

	PassengerId	Survived	Pclass
0	892	0	3
1	893	1	3

Filtering

➔ **Filter the required data based on Logic.**

Python Code

```
import pandas as pd
Data = pd.read_csv("titanic.csv")
Data[Data["Survived"]>0]
```

	PassengerId	Survived	Pclass	Name	Sex	Age
1	893	1	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0
4	896	1	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	female	22.0
6	898	1	3	Connolly, Miss. Kate	female	30.0
8	900	1	3	Abrahim, Mrs. Joseph (Sophie Halaut Easu)	female	18.0
12	904	1	1	Snyder, Mrs. John Pillsbury (Nelle Stevenson)	female	23.0
...	...	...	...	...	...	...
409	1301	1	3	Peacock, Miss. Treasteall	female	3.0
410	1302	1	3	Naughton, Miss. Hannah	female	NaN
411	1303	1	1	Minahan, Mrs. William Edward (Lillian E Thorpe)	female	37.0
412	1304	1	3	Henriksson, Miss. Jenny Lovisa	female	28.0
414	1306	1	1	Oliva y Ocana, Dona. Fermina	female	39.0

Data Exploration Techniques

1. Identifying Unique Elements → `dat["PassengerId"].unique()`
2. Value Count → `dat["Sex"].value_counts()`
3. Null Value Check → `dat.isnull().sum()`
4. Drop Feature → `dat.drop("Sex",axis = 1)`
5. Feature Mean → `dat["Age"].mean()`
6. Feature Median → `dat["Age"].median()`
7. Feature Mode → `dat["Sex"].mode()`

Map Function in Pandas:

Syntax:

df[“<column_name”] = df[“<column_name”].map(<function_name>)

Python Code:

```
import numpy as np
import pandas as pd
data=pd.read_csv('employee.csv')
data.head()
```

```
def function(x):
    return x/10

data['DailyRate'] = data['DailyRate'].map(function)
data['DailyRate']
```

Apply Function in Pandas

Syntax:

df[[“<col1”, “<col2”]] = df[[“<col1”, “<col2”]].apply(<function_name>)

Python Code:

```
import numpy as np
import pandas as pd
data=pd.read_csv('employee.csv')
print(data.head())

def function(x):
    return x/10

data[['DailyRate','MonthlyRate']] = data[['DailyRate','MonthlyRate']].apply(function)
data[['DailyRate','MonthlyRate']]
```


Data Cleaning

Data Cleaning Methods

- **1. Dropping** → Deleting the Rows or columns
- **2. Imputing** → Imputing the missing value by statistical Method (Mean, Median & Mode)

Dropping Method

➔ **Deleting the Rows or columns**

	column_a	column_b	column_c	column_d	column_e
0	1.0	1.2	a	True	1
1	2.0	1.4	NaN	True	2
3	4.0	6.2	d	None	4
5	NaN	1.1	NaN	True	5
6	6.0	4.3	d	False	NaN

When should we delete?

In Row,
Missing values > 30%
Delete Row

In Column,
Missing values > 30%
Delete Column

Practical Implementation using Titanic Dataset:


```
import pandas as pd

data = pd.read_csv("titanic.csv")
print(data.shape)
data.head()
```

```
data.isnull().sum(axis = 1).sort_values(ascending = False)
```

```
data.isnull().sum()
```

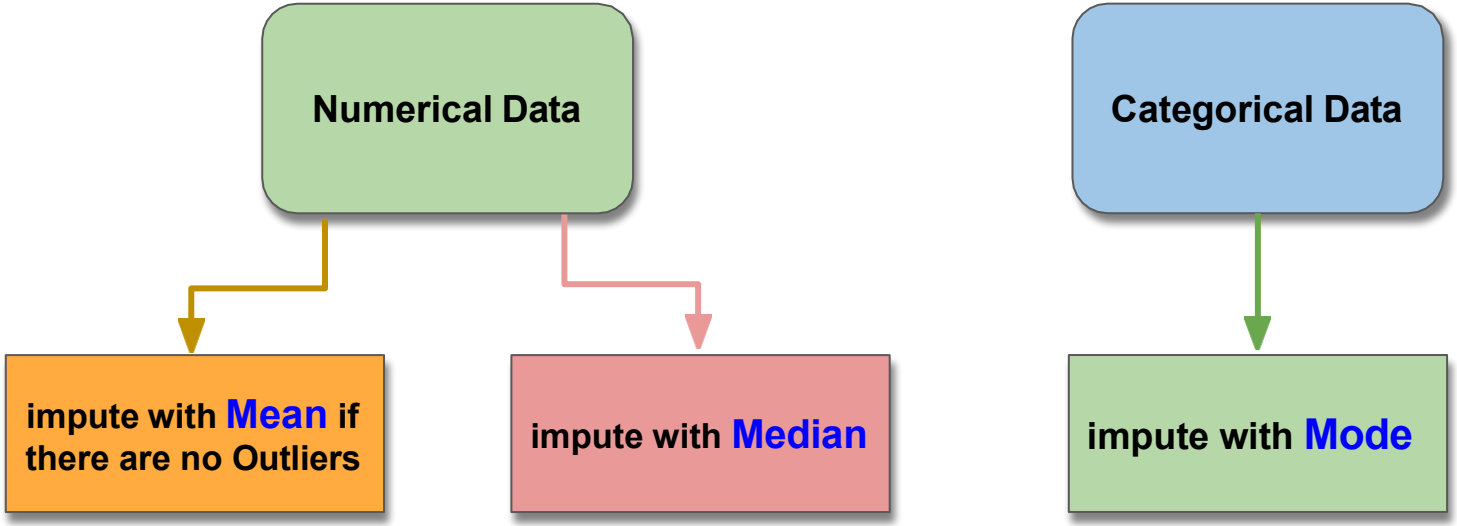
```
print(data.isnull().sum()/data.shape[0]*100)
```

```
data = data.drop(["Cabin"],axis = 1)
```

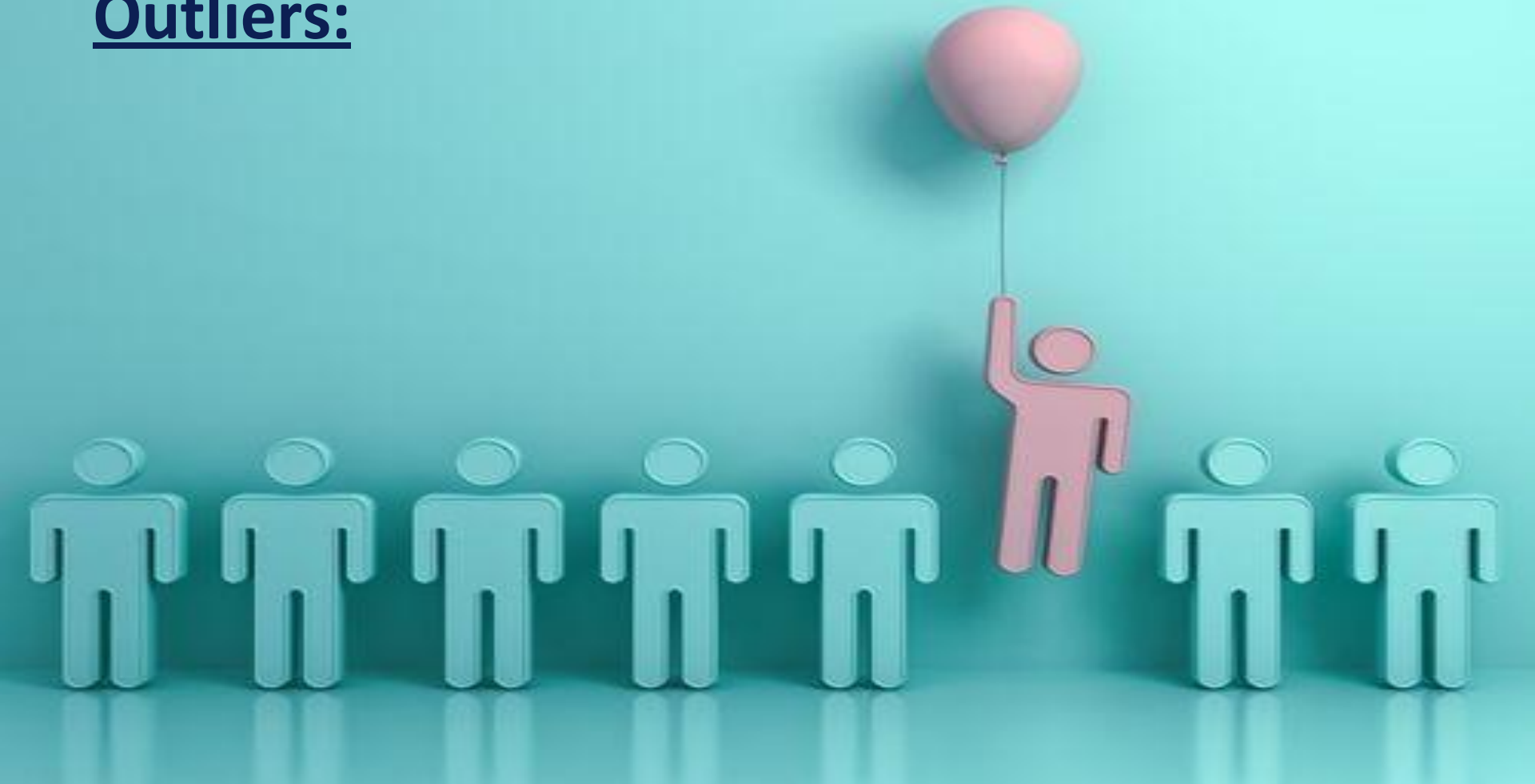
```
data.head()
```

Imputing Method

➔ Imputing the missing value by statistical Method (Mean, Median & Mode)



Outliers:

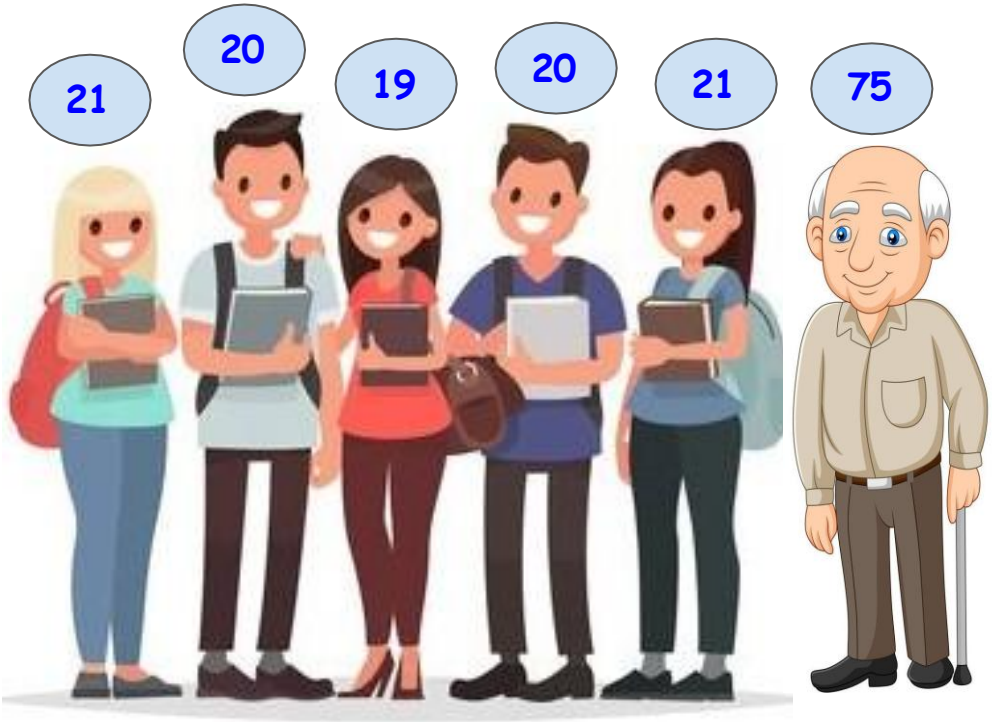


Mean = 20.2

Median = 20

No Outliers

Mean = 29.3



Median = 19.5

Outlier - 75

Practical Implementation using Bigmart Dataset:

bigmart.csv

```
import pandas as pd
data = pd.read_csv("bigmart.csv")
print(data.shape)
data.head()
```

```
data.isnull().sum()
```

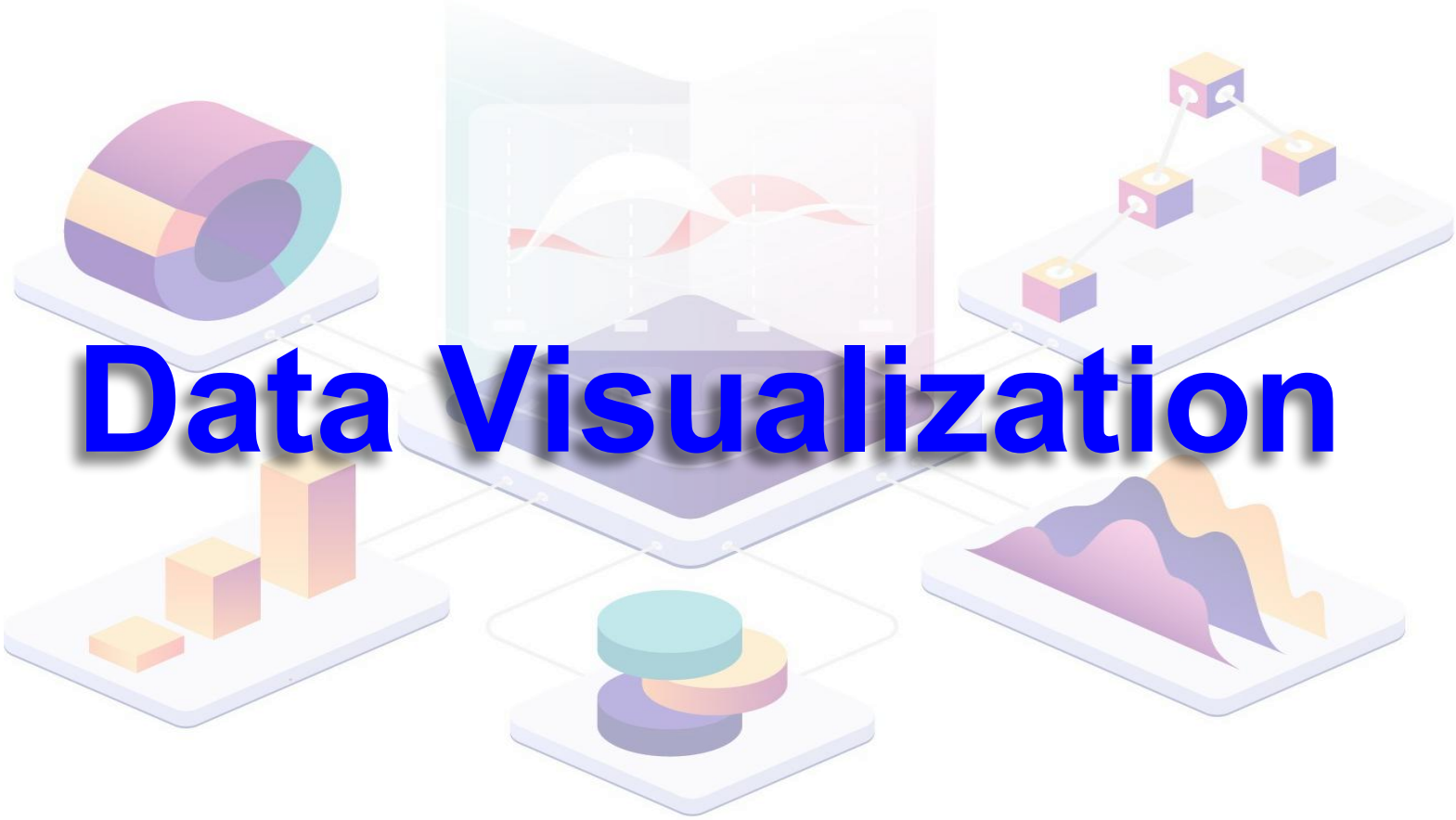
```
data.info()
```

```
data["Item_Weight"] = data["Item_Weight"].fillna(data["Item_Weight"].mean())
```

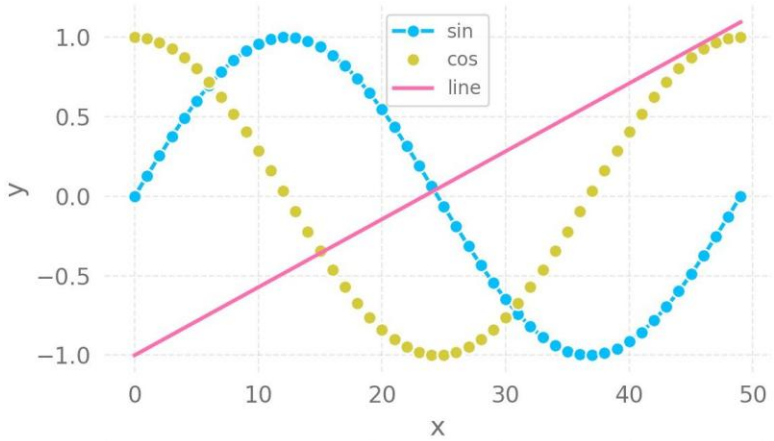
```
data["Outlet_Size"] = data["Outlet_Size"].fillna(data["Outlet_Size"].mode()[0])
```

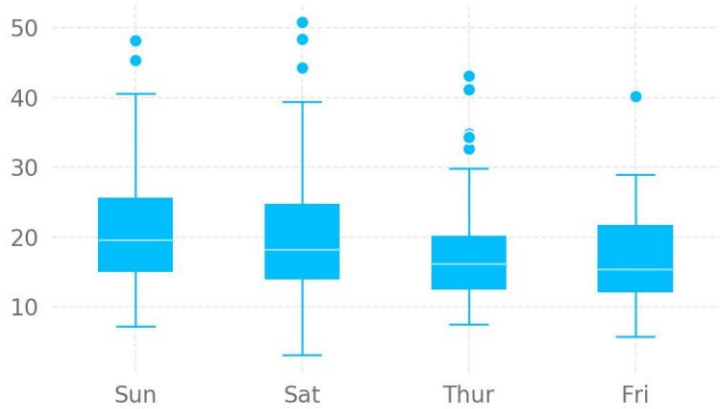
```
data.info()
```

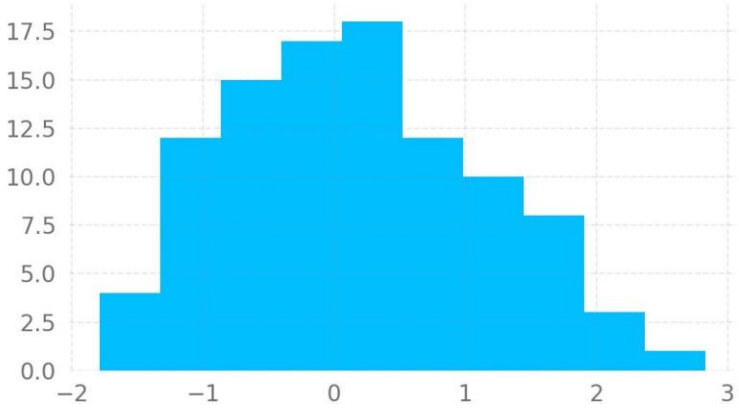
Data Visualization

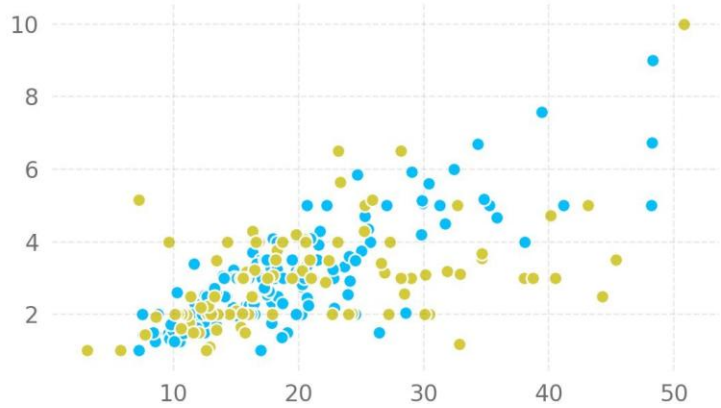


Matplotlib

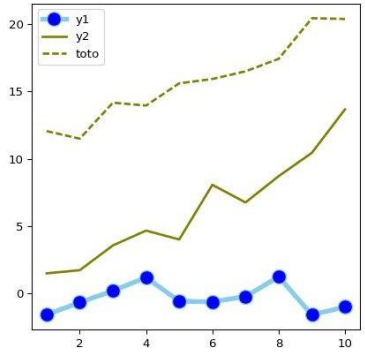

sin
cos
line
1.0
0.5
0.0
−0.5
−1.0
y
0
10
20
30
40
50
x


50
40
30
20
10
Sun
Sat
Thur
Fri


17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
−2
−1
0
1
2
3


10
8
6
4
2
10
20
30
40
50

Matplotlib plot types:



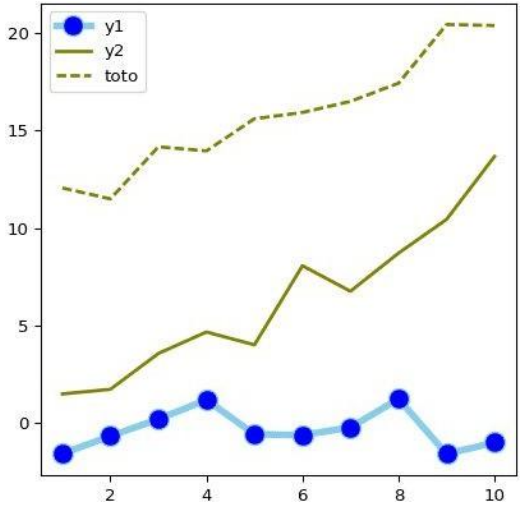
Line Plot:

.plot(): Used to Plot the Line graph

```
plt.plot(x_data, y_data)
```

x-axis Data

y-axis Data

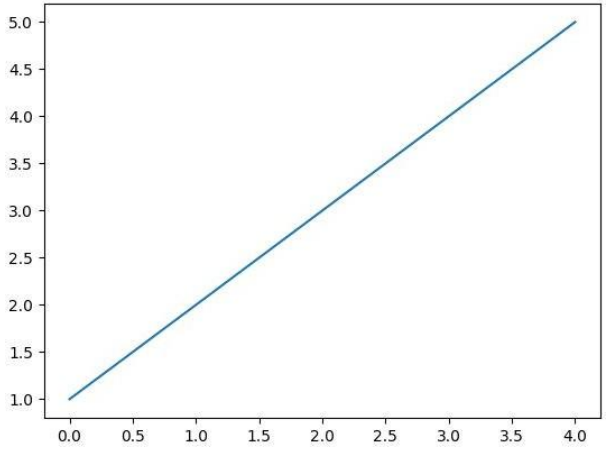


.show(): Used to display the graph

Ex: `plt.show()`

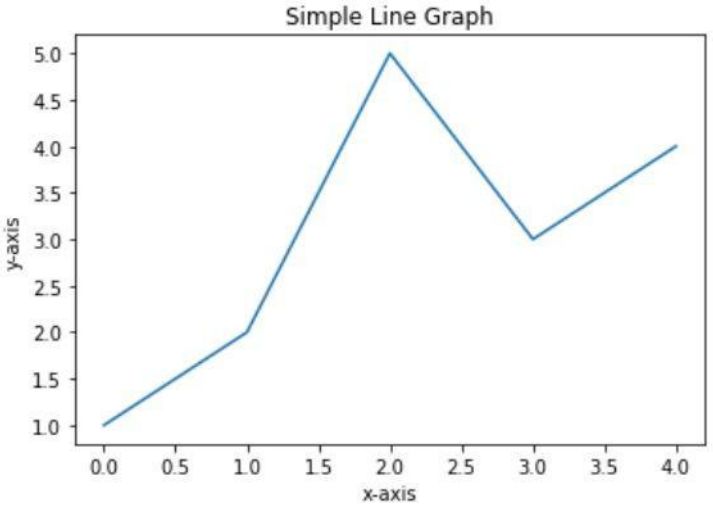
Ex1: Simple Line Plot

```
import numpy as np
a = np.array([1,2,3,4,5])
plt.plot(a)
plt.show()
```



Ex2 -Plot with x, y labels:

```
import numpy as np
a = np.array([1,2,5,3,4])
plt.plot(a)
plt.title("Simple Line Graph")
plt.xlabel("x-axis")
plt.ylabel("y-axis")
plt.show()
```



Types of Data Analysis

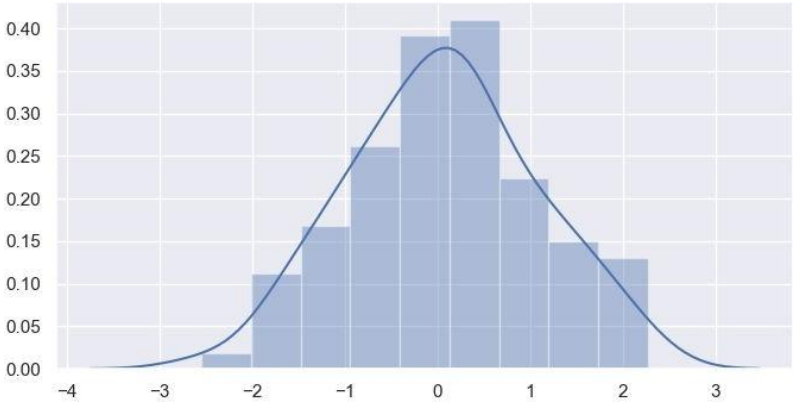
1. Univariate Analysis → Analysis of one variable at a time.
2. Bivariate Analysis → Analysis of Two variable at a time.
3. Multivariate Analysis → Analyzing Multiple Features at a Time

Univariate Analysis

➔ **Analyzing Numerical Data**

distplot( ):

```
sns.distplot(<data>)
```



Python Code:

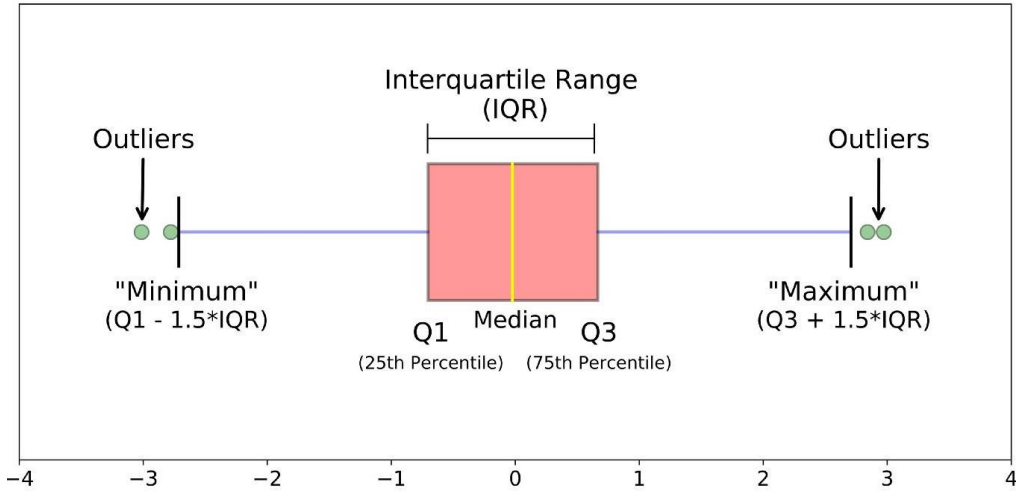
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
dat = pd.read_csv("employee.csv")
dat.head()
sns.distplot(dat["Age"])
```

Univariate Analysis

➔ **Analyzing Numerical Data**

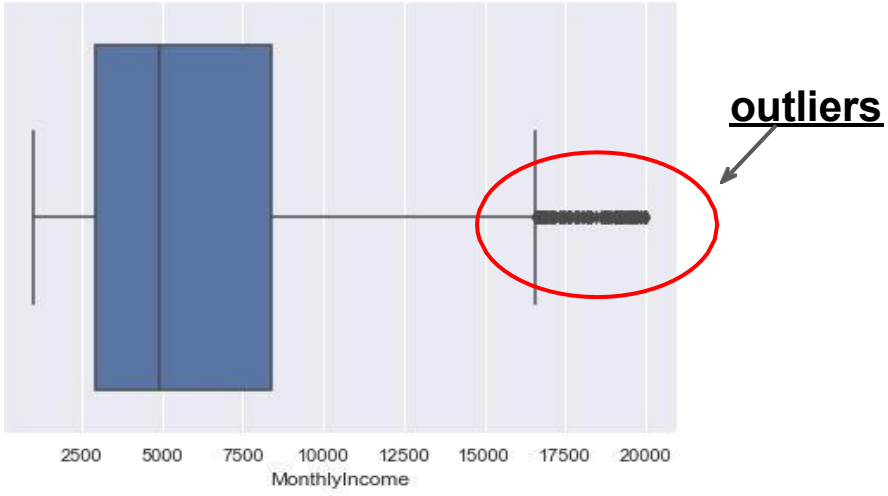
boxplot( ):

```
sns.boxplot(x = <data>)
```



Python Code:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
dat = pd.read_csv("employee.csv")
dat.head()
sns.boxplot(x=dat["MonthlyIncome"])
```



Univariate Analysis

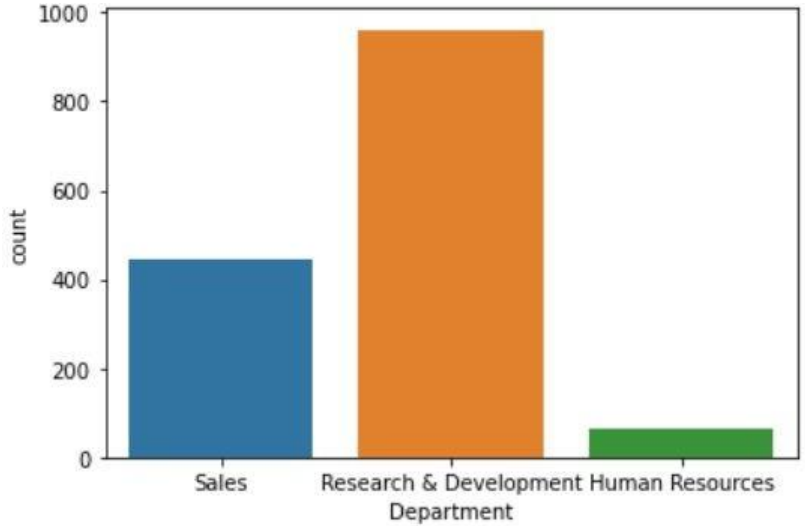
➔ **Analyzing Categorical Data**

countplot( ):

```
sns.countplot(data = <dataset>, x = <col name>)
```

Python Code:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
dat = pd.read_csv("employee.csv")
dat.head()
sns.countplot(data = dat,x = "Department")
```



Bivariate Analysis

➔ **Categorical v/s Numerical**

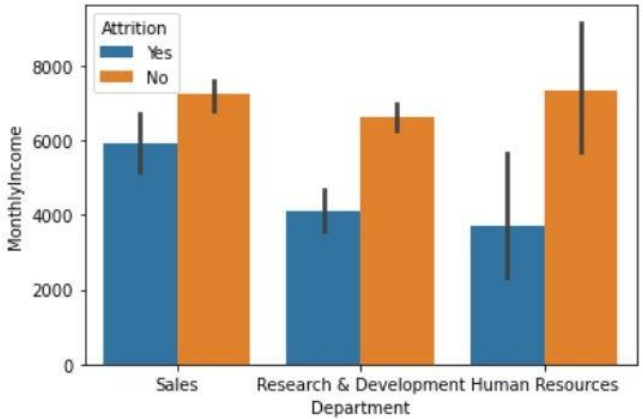
barplot( ):

```
sns.barplot(data=<dataset>, x=<column>, y=<column>, hue=<column>)
```

Python Code:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
dat = pd.read_csv("employee.csv")
dat.head()
```

```
sns.barplot(data=dat,x="Department",y = "MonthlyIncome",hue = "Attrition")
```



Bivariate Analysis

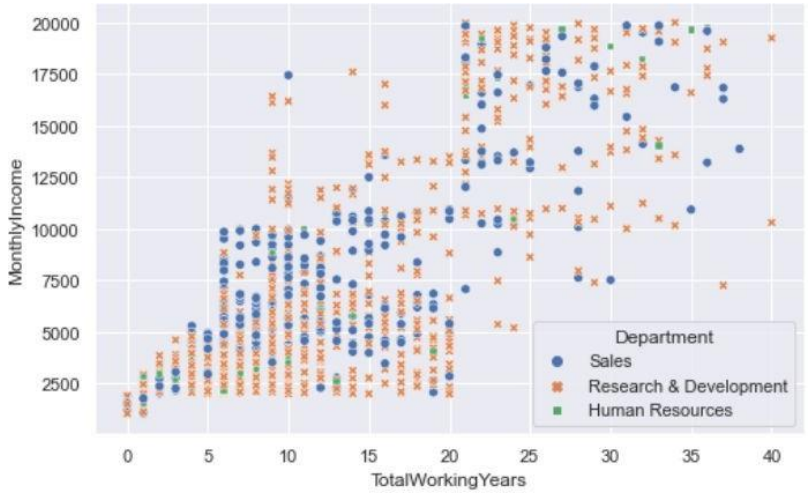
➔ **Numerical v/s Numerical**

scatterplot( ):

```
sns.scatterplot(data=<dataset>, x=<column>, y=<column>, hue=<column>, style=<column>)
```

Python Code:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
dat = pd.read_csv("employee.csv")
dat.head()
plt.rcParams["figure.figsize"] = (8,5)
sns.set_theme()
```




```
sns.scatterplot(data = dat, x="TotalWorkingYears",y="MonthlyIncome", hue="Department",style="Department")
```

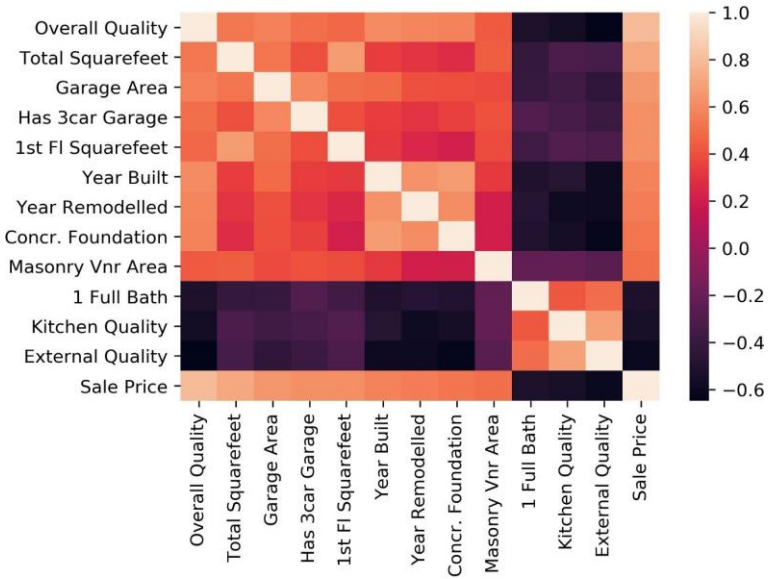
Multivariate Analysis

Heatmap

➔ It will helpful to visualize the correlation between Different Features

➔ -1 means strong negative correlation & +1 means strong positive correlation

heatmap( ): `sns.heatmap(data.corr(), annot=<True/False>, fmt="0.1f")`



Python Code:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
dat = pd.read_csv("employee.csv")
dat.head()
plt.rcParams["figure.figsize"] = (16,8)
sns.heatmap(dat.corr(),annot = True,fmt = "0.1f")
```