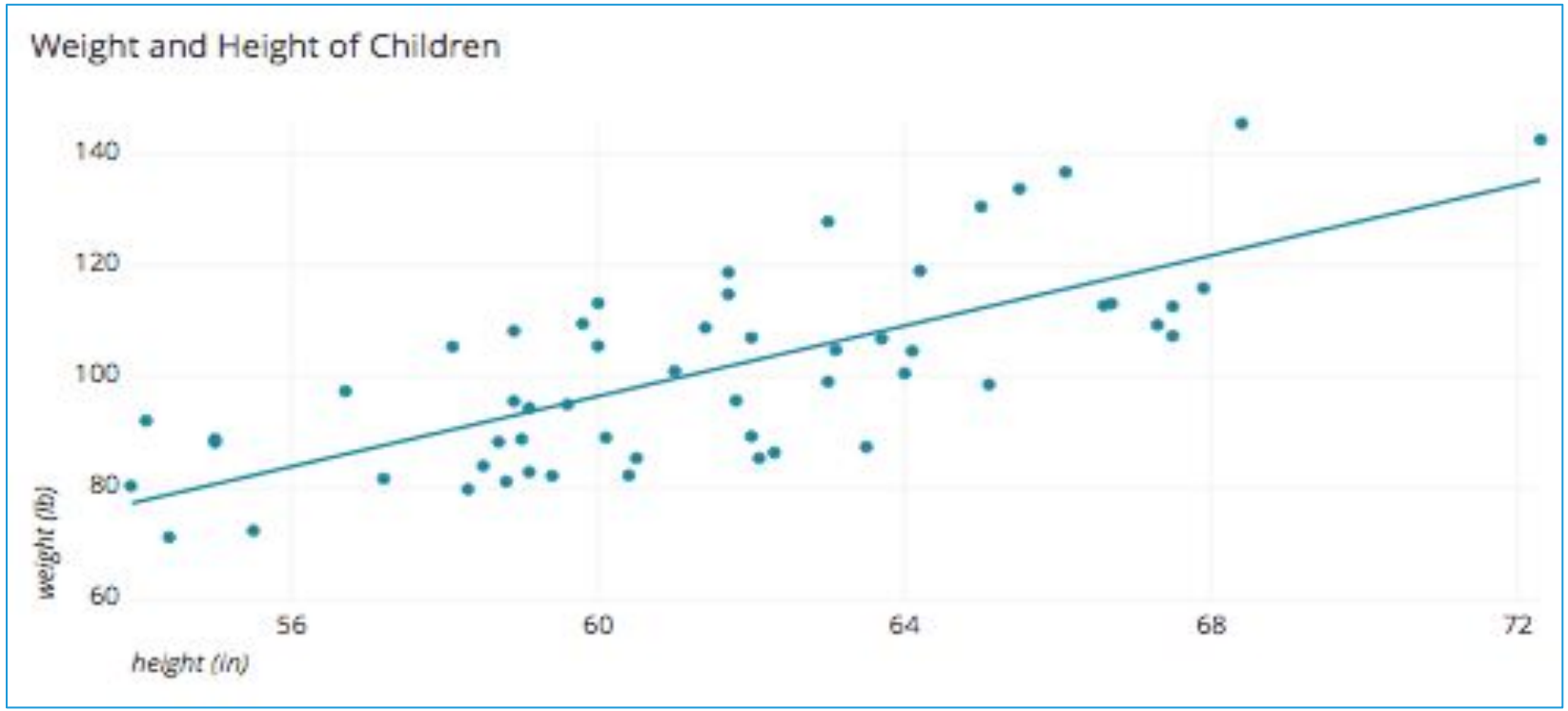
Visualizing Relationships in Python

Contents

Scatter Plot



Each dot represents one child with his or her height measured along the x-axis and weight measured along the y-axis

Case Study

Background

A company has the scores of various attribute tests of their employees

Objective

To understand the factors contributing to the Job Proficiency of an employee.
To see the relationship between these various factors

Sample Size

25

Data Snapshot

JOB PROFICIENCY DATA

Variables

Observations

empno	aptitude	testofen	tech_	g_k_	job_prof
1	86	110	100	87	88
2	62	62	99	100	80
3	110	107	103	103	96

Columns	Description	Type	Measurement	Possible values
empno	Employee No	Numeric	-	-
aptitude	Aptitude	Numeric	-	positive values
testofen	Test of English	Numeric	-	positive values
tech_	Technical Score	Numeric	-	positive values
g_k_	General Knowledge	Numeric	-	positive values
job_prof	Job Proficiency	Numeric	-	positive values

ScatterPlot with Regression Line in Python

```
#Importing Data
```

```
import pandas as pd
job=pd.read_csv('JOB PROFICIENCY DATA.csv', index_col=0)
```

index_col= 0 instead of None (take first column as index by default)

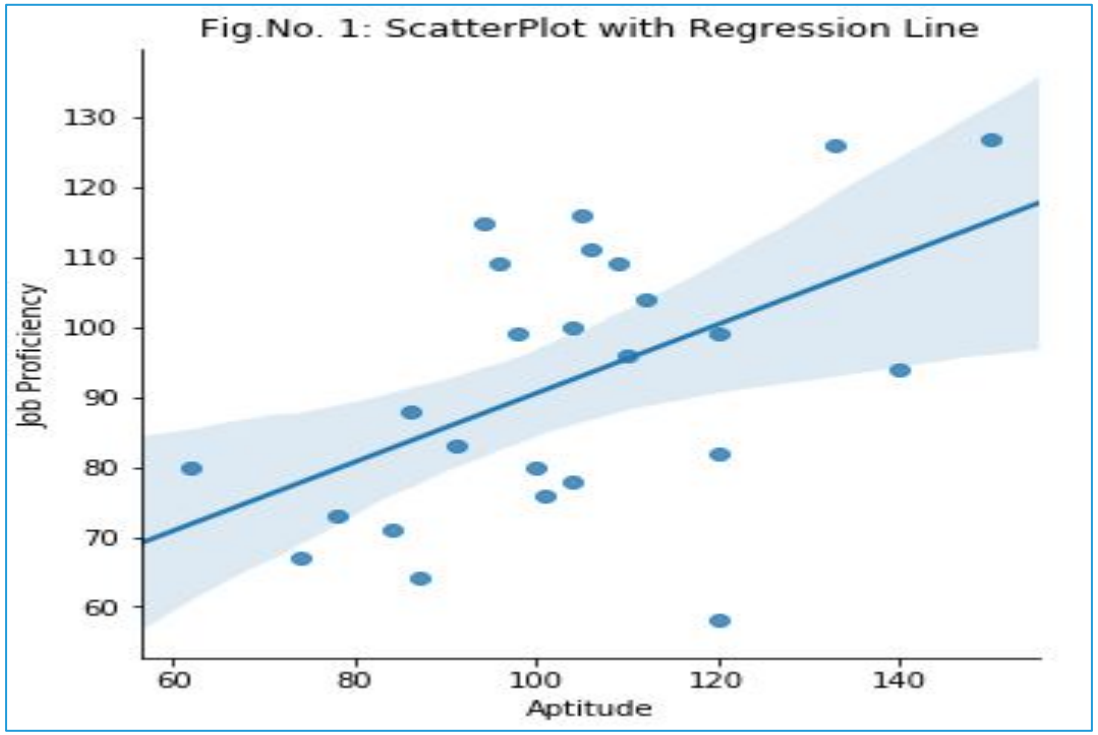
```
#Importing Library Seaborn
```

```
import seaborn as sns
import matplotlib.pyplot as plt
```

```
#Scatterplot of job proficiency against aptitude with Regression Line
sns.lmplot('aptitude','job_prof',data=job);plt.xlabel('Aptitude');plt.ylabel('Job Proficiency');plt.title('Fig.No. 1: ScatterPlot with Regression Line')
```

- **sns.lmplot()** calls a scatter plot from sns object with regression line
- **plt.xlabel** provides a user defined label for the variable on x axis
- **plt.ylabel** provides a user defined label for the variable on y axis
- **plt.title** gives title to the plot

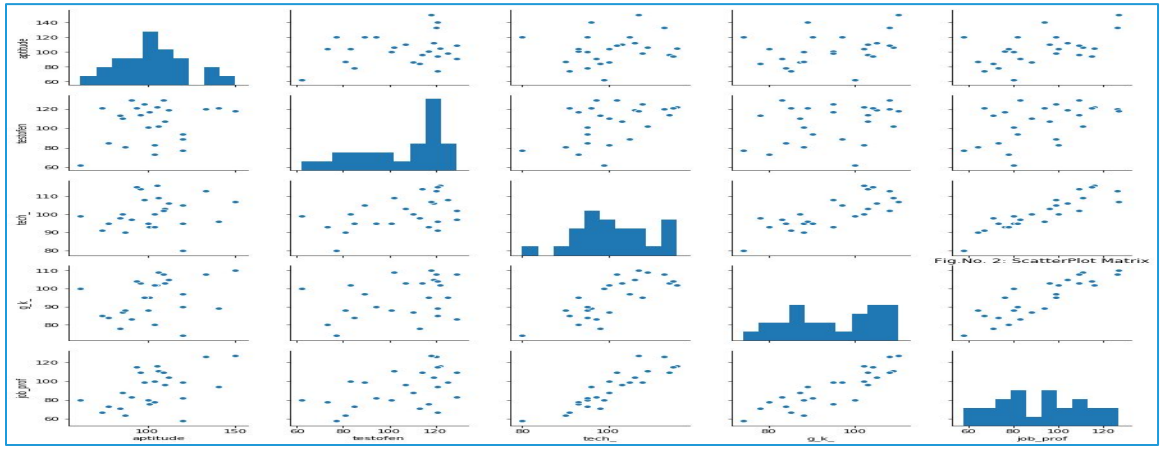
ScatterPlot with Regression Line in Python



Interpretation :

- Scatter plot above shows that, as the aptitude score increases job proficiency also increases.
- For a given aptitude score, the job proficiency can be estimated and vice-a-versa using the regression line.

Scatter Plot Matrix



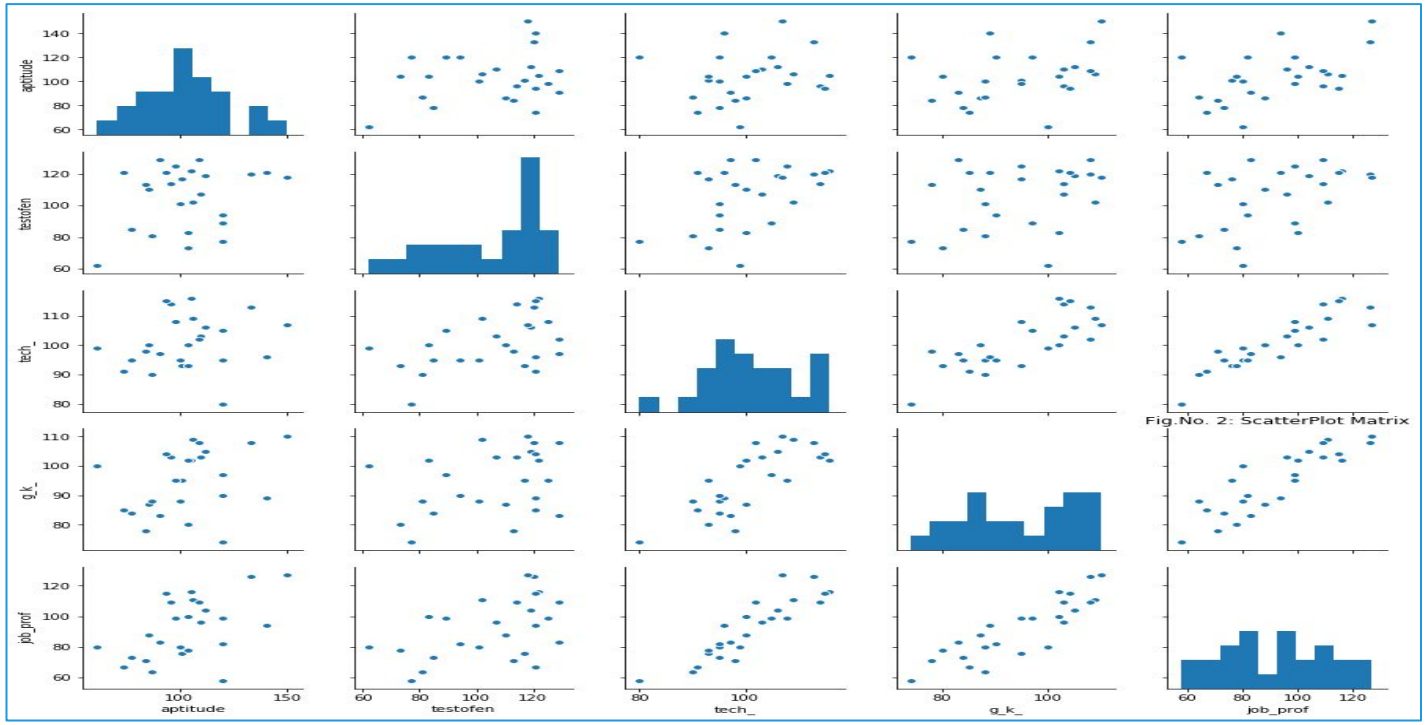

```
# ScatterPlot Matrix
sns.pairplot(job);plt.title('Fig.No. 2: ScatterPlot Matrix')
```

- **pairplot()** from sns is used to plot pairwise comparison

8

Scatter Plot Matrix in Python

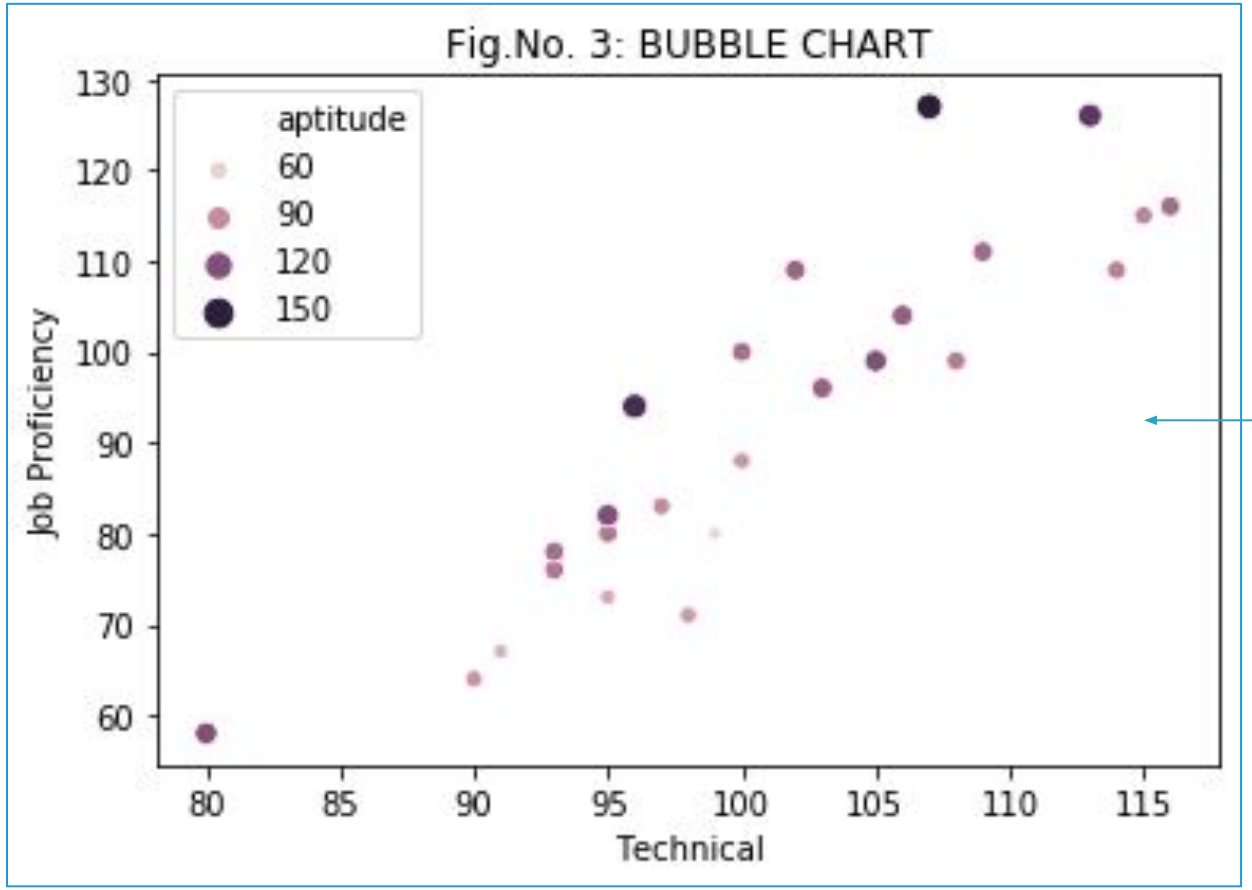
```
# Output
```



Interpretation :

- Scatter plot matrix above shows that, as the aptitude score, English language score, technical score and general knowledge score increases job proficiency also increases.
- Technical score and GK score has slight positive relation but other variables are not related to each other.

Bubble Chart



Interpretation :

- Here we observe that as Technical score increases Job Proficiency also increases, however, Aptitude score does not show any such consistent direction.

Bubble Chart in Python

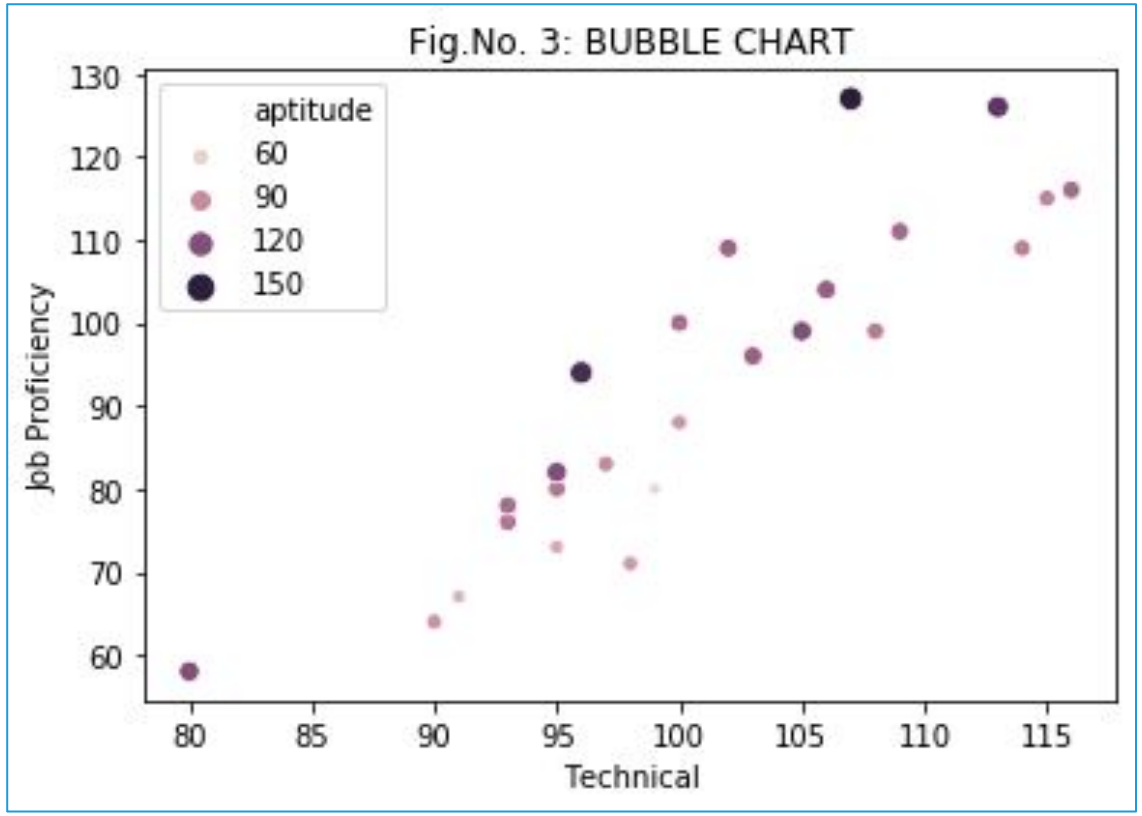
```
# Bubble Chart
```

```
sns.scatterplot('tech_', 'job_prof', data=job,
hue='aptitude',size='aptitude'); plt.title('Fig.No. 3: BUBBLE CHART');
plt.xlabel('Technical'); plt.ylabel('Job Proficiency')
```

- **sns.scatterplot()** calls a scatter plot from sns object
- **tech_, job_prof** are variables to be plotted on x and y axis
- **hue** gives colors based on aptitude score
- **size** assigns the size to the bubble based on aptitude score

Bubble Chart in Python

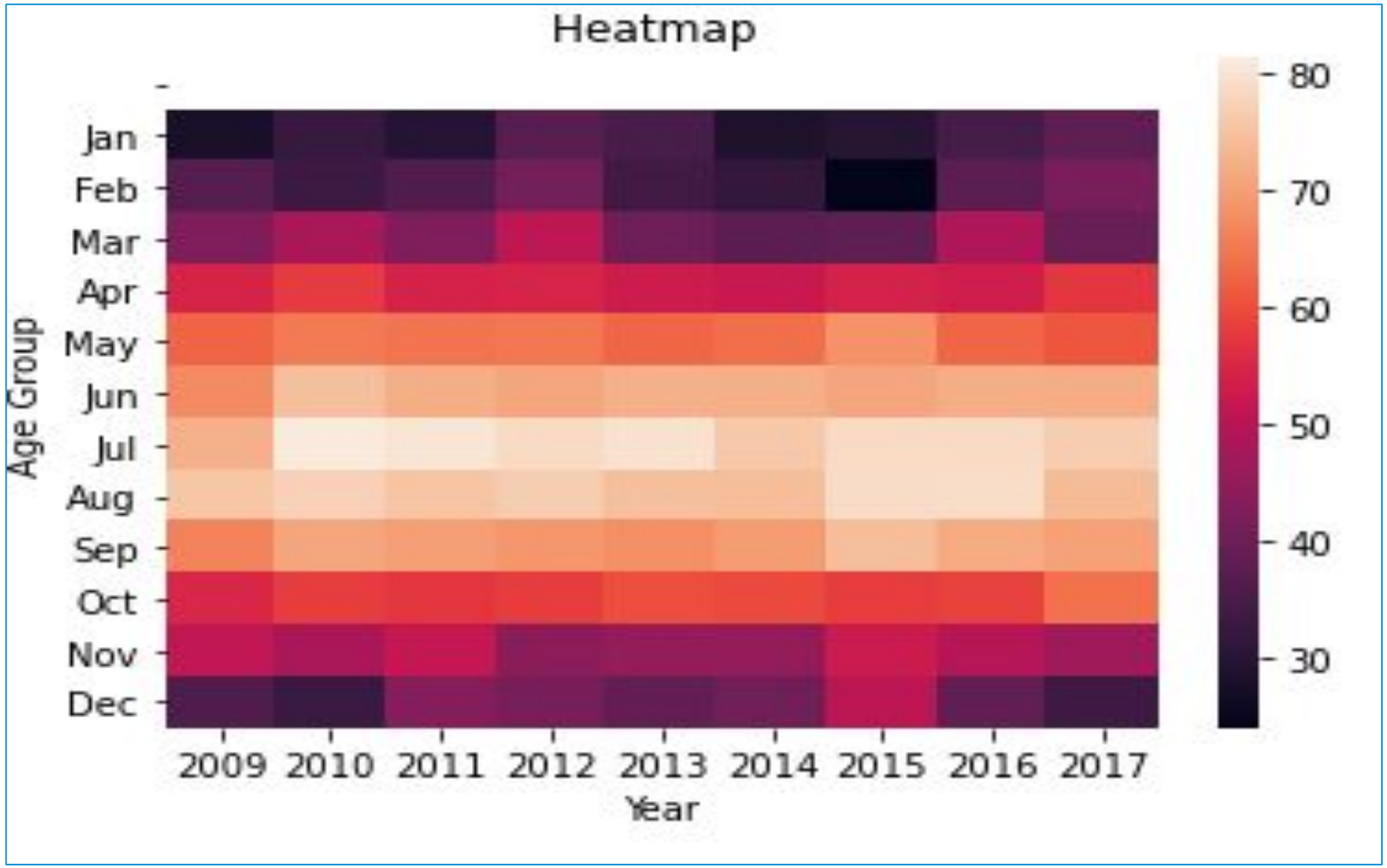
```
# Output
```



Interpretation :

- Here we observe that as Technical score increases Job Proficiency also increases however, Aptitude score does not show any such consistent direction.

Heat Map


Heatmap
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
Age Group
2009
2010
2011
2012
2013
2014
2015
2016
2017
Year
80
70
60
50
40
30

Case Study

To get a better understanding of the subject, we shall consider the below case as an example.

Background

NY Temperature varies across months over the years

Objective

To visually see the hottest months in the years
To see how temperature has fluctuated over the years

Sample Size

108

Data Snapshot

Average Temperatures in NY

Variables

Observations

Year	Month	Temperature
2009	Jan	27.9
2009	Feb	36.7
2009	Mar	42.4
2009	Apr	54.5
2009	May	62.5
2009	Jun	67.5

Columns	Description	Type	Measurement	Possible values
Year	Years listed from 2009-2017	Categorical	2009 – 2017	9
Month	Months of the year	Categorical	Jan - Dec	12
Temperature	Average Temperature in degree Fahrenheit	Numeric	-	-

Heat Map in Python

```
# Installing and calling the package
import seaborn as sns
import calendar
```

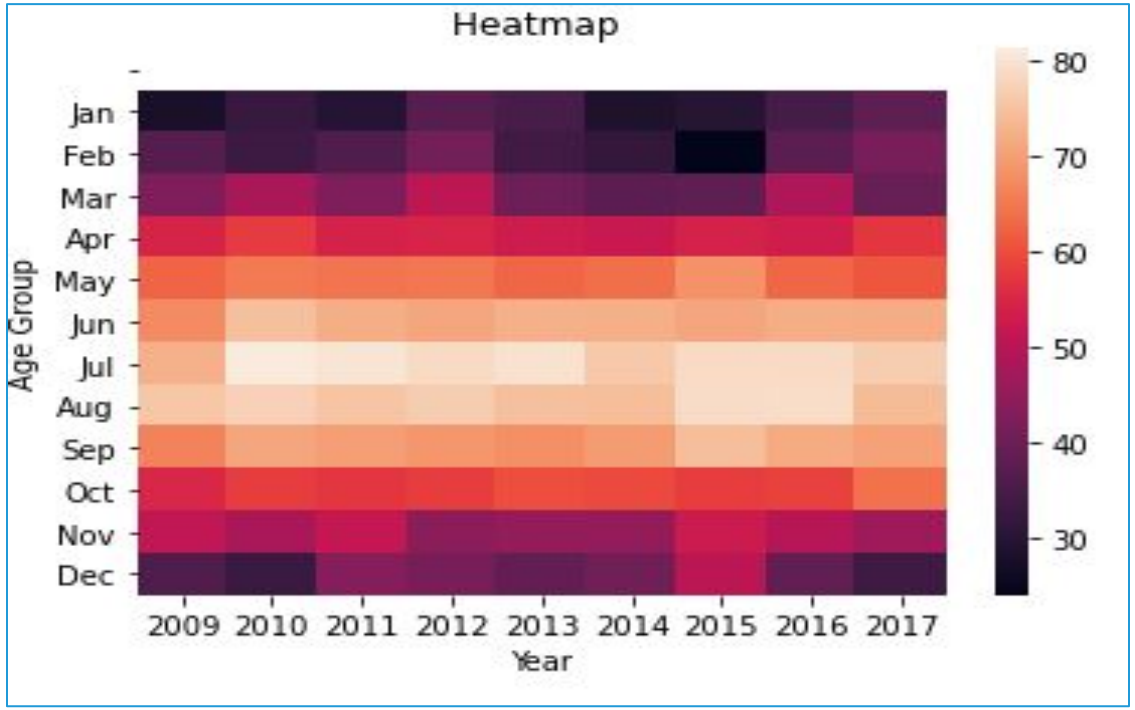
```
# Importing Data and Arranging the Months in the right order :
heatmapdata=pd.read_csv('Average Temperatures in NY.csv')
agg=pd.pivot_table(heatmapdata, index=['Month '], columns=['Year '])
agg.columns = (heatmapdata['Year ']).unique()
agg = agg.reindex(list(calendar.month_abbr))
```

- **calendar** library gives the functions related to calendar manipulations such as Year, Month.

```
# Heat Map
plt.show; ax=sns.heatmap(agg);ax.set(xlabel='Year', ylabel='Age
Group',title='Heatmap ')
```

Heat Map in Python

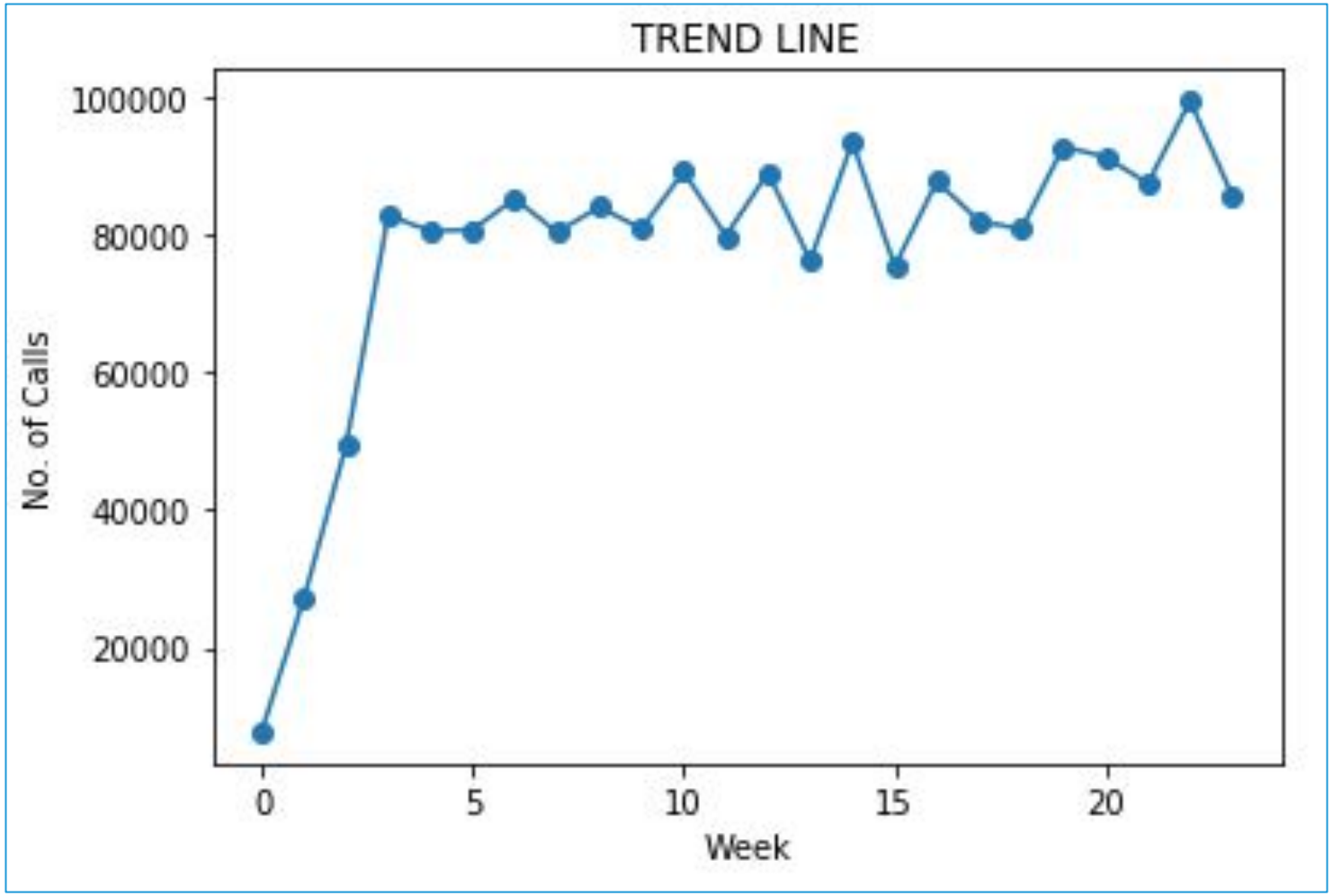
```
# Output for Heat Map :
```



Interpretation :

- Heat map above shows that July is the hottest season across the year .
- 2015 showed a longer hot period as compared to other years extending from may to September

Trend Line


TREND LINE
100000
80000
60000
40000
20000
No. of Calls
0
5
10
15
20
Week

Case Study

To get a better understanding of the subject, we shall consider the below case as an example.

Background

Telecom Weekly Data for 24 weeks

Objective

To visually observe the trend of total calls over 24 weeks

Sample Size

21902

Data Snapshot

Plotting a trendline requires time-element. Consider the following datasets. Week can be taken as the time element.

TelecomData_WeeklyData

Variables

Observations

CustID	Week	Calls	Minutes	Amt
1001	1	56	292	78.4

Columns	Description	Type	Measurement	Possible values
CustID	Customer ID	Numeric	-	-
Week	Week no.	Numeric	1-24	24
Calls	No. of Calls	Numeric	-	positive values
Minutes	Total Minutes	Numeric	Minutes	positive values
Amt	Amount Charged	Numeric	Rs.	positive values

Trend Line in Python

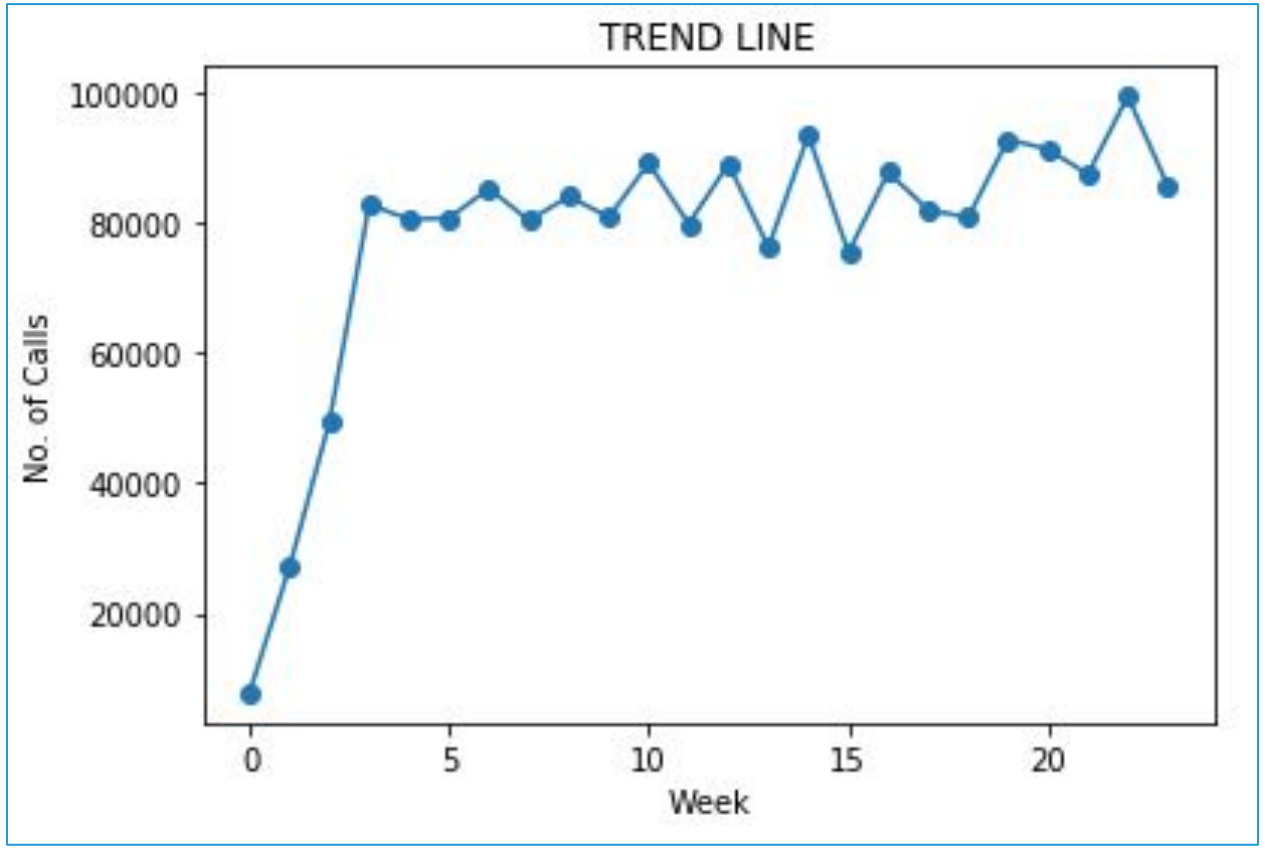
```
# Importing Data
transaction = pd.read_csv("TelecomData_WeeklyData.csv")

# Merging and Formatting Data
trend=(transaction.groupby('Week')['Calls'].sum().to_frame()).reset_index(
)
# Trend Line
plt.plot(trend['Calls'], marker='o');plt.xlabel('Week');plt.ylabel('No.
of Calls');plt.title('TREND LINE')
```

- The basic function is plot(x, data, marker, color)
- **x** is a vector containing the numeric values.
- **marker** plots simple line, "o" used to draw both points and lines.
- **plt.xlabel** is the label for x axis.
- **plt.ylabel** is the label for y axis.
- **plt.title** is the Title of the chart.
- **color** is used to give colors to both the points and lines.

Trend Line in Python

```
# Output
```



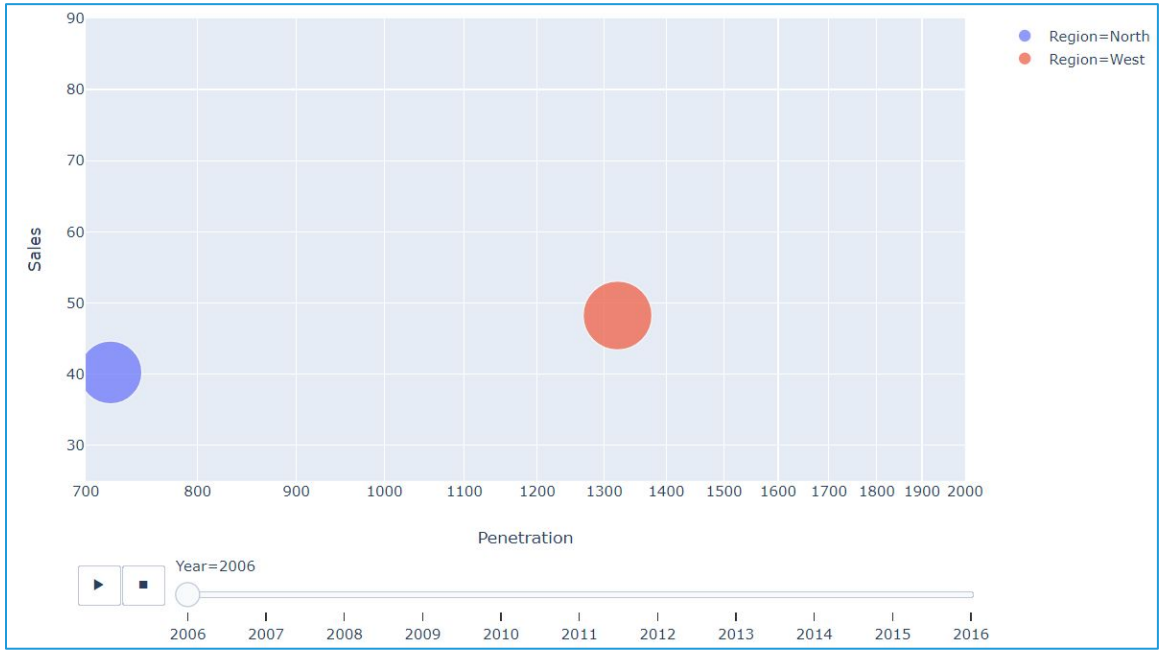
Interpretation :

- Upto first 4 weeks, number of calls increases continuously. After 5th week there are more ups and down in number of calls among customers.

Motion Chart

- A Motion Chart is a dynamic bubble chart which allows efficient and interactive exploration and visualization of longitudinal multivariate Data.
- It allows you to plot the dimension values in your report against up to four metrics across time.

Fig. No. 6 : MOTION CHART



Case Study

To get a better understanding of the subject, we shall consider the below case as an example.

Background

Sales Data & it's penetration in each Region over the years

Objective

To visually observe the sales & penetration in motion over the years

Sample Size

22

Data Snapshot

Sales Data (Motion Chart)

Variables

Year	Region	Sales	Penetrartion
2006	North	40.23	721

Observations

Columns	Description	Type	Measurement	Possible values
Year	Year	Numeric	2006-2016	11
Region	Region	Categorical	North,West	2
Sales	Sales in a particular Year	Numeric	Rs.	Positive values
Penetration	Penetration in a particular Year	Numeric		Positive values

Motion Chart in Python

To create a motion chart in python execute the following code in Jupyter Notebook.

```
#Importing Data
sales = pd.read_csv("Sales Data (Motion Chart).csv")
```

```
#Installing plotly-express
pip install plotly-express
```

Install **plotly-express** using pip installer with this command in anaconda prompt

```
#Installing
import plotly.express as px
```

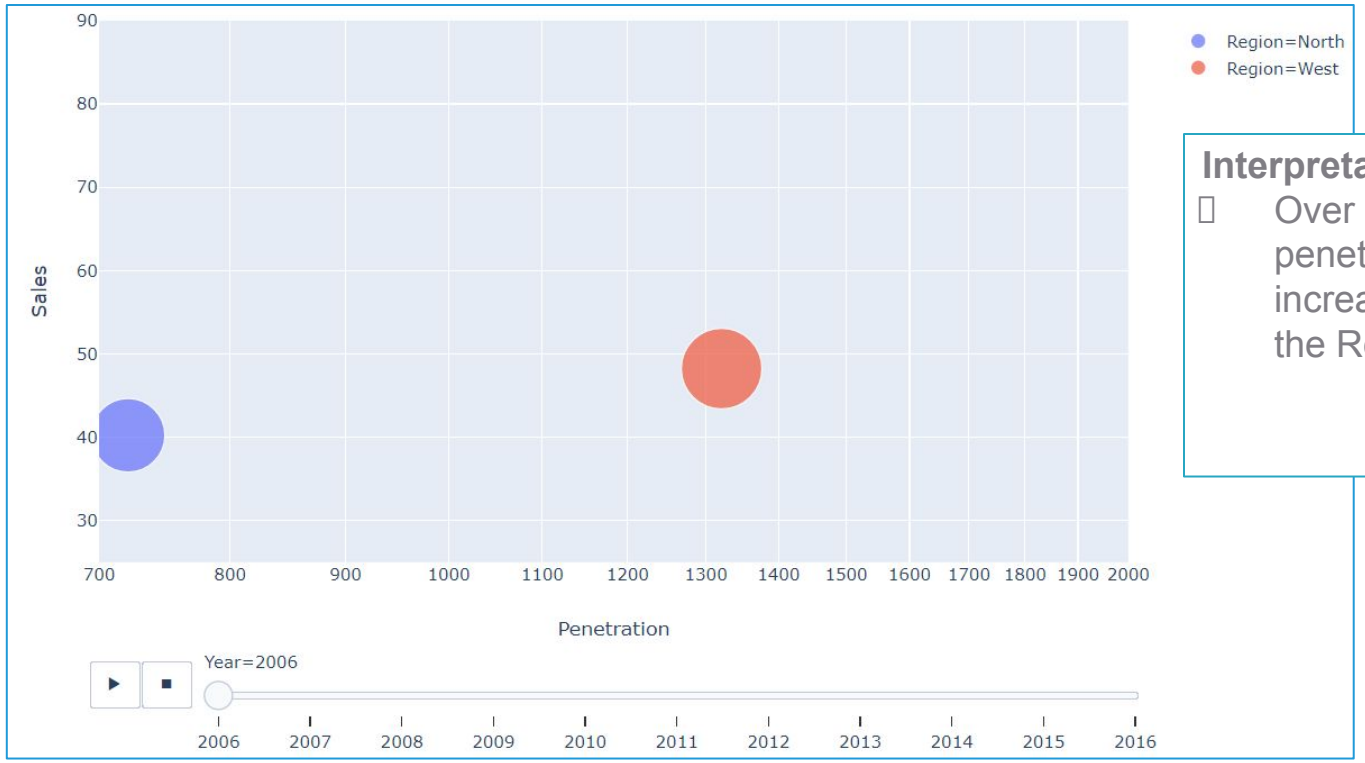
plotly-express is the best package we can use to plot an effective Motion Chart in Python

```
# Motion Chart
px.scatter(sales, x="Penetration", y="Sales", animation_frame="Year",
animation_group="Region",size="Sales", color="Region",
hover_name="Region", log_x=True, size_max=55, range_x=[700,2000],
rang
```

- **px.scatter** is the function used to create a motion chart
- **sales** is the data that is used
- **animation_frame=** inputs time variable
- **animation_group=** inputs of categorical variable
- **log_x**=(default False)If True, the x-axis is log-scaled in cartesian coordinates.

Motion Chart in Python

```
# Output
```



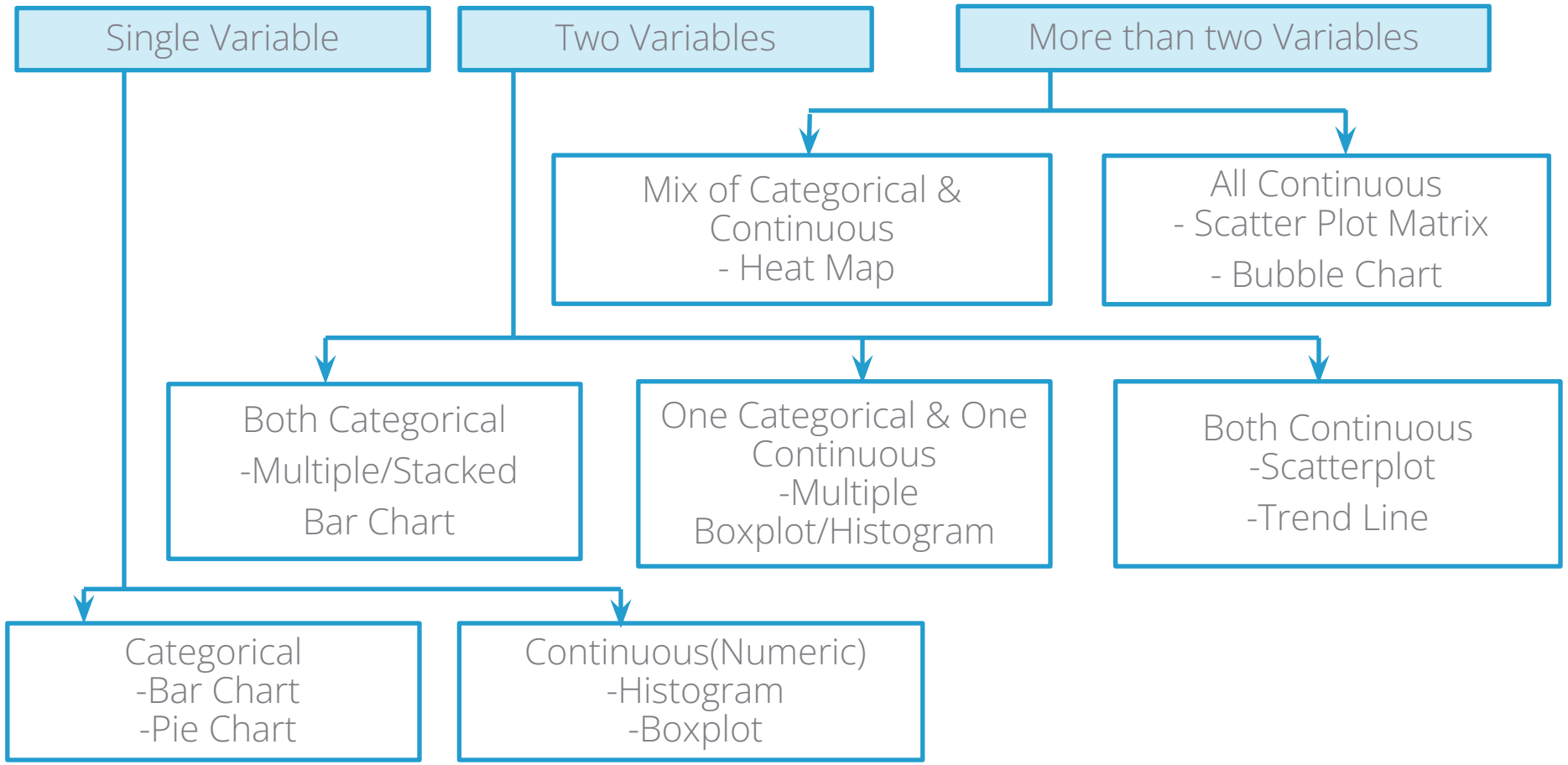
Interpretation :

- Over time, as sales increases penetration has also increased parallelly for both the Regions.

* Motion chart can be run only with Jupyter Notebook using plotly-express

Get an Edge!

Choosing the right graph



Quick Recap

In this session, we learnt data visualisation using basics graphs

Chart Types and Functions in Python	
	• Scatterplot with Regression Line `sns.lmplot()` • Scatterplot Matrix – `sns.pairplot()` • Bubble Chart – `sns.lmplot()` • Heat Map – `sns.heatmap()` • Trend Line – `plot()` • Motion Chart – `px.scatter` from package "plotly.express"