Outreach Program

on

Data Processing

using

Python

SL No.	Topics	SL No.	Topics
1	Descriptive Statistics	11	Advanced Imputation
2	Introduction to Python	12	Feature Selection
3	Data Aggregation	13	Linear Regression
4	Data Preprocessing	14	Dummy Variable Regression
5	Data Visualization	15	Binary Logistic Regression

DESCRIPTIVE STATISTICS

DESCRIPTIVE STATISTICS

Statistics

Science of collection, analysis, interpretation and presentation of data

Analytics

Process of extracting meaningful insights by discovering patterns and relationships in the data

DESCRIPTIVE STATISTICS

Data set

Number of tasks completed per hour

Productivity	
69	71
70	68
72	70
71	67
70	69
68	72
73	70
69	71

DESCRIPTIVE STATISTICS

Data

The values or figures assigned to a metric

Can be numeric or non numeric

Example

Metric: Productivity

Data: 70, 72, 69, etc

DESCRIPTIVE STATISTICS

Use of Statistics

To know what happened in the past

To know approximately what will happen in future if things remain more or less the same

Example

Approximately how many tasks will be completed in the next hour?

Is it same as the last value as the last value is close to the future?

The difference between the last value and previous value is only 1 (71 – 70).So adding 1 to last value will give the next future value?

Approximately how many tasks will be completed two hours down the line?

Is mean, median, etc give better estimate of future value? Why?

DESCRIPTIVE STATISTICS

Use of Statistics

Generally the values in a data set will not be same

Values will be not only changing but change without any particular trend or pattern

Example

If we collect another 16 hours productivity data, the value may not be same as 69, second value as 70, etc?

Similarly the difference between the 1st and 2nd value or 2nd and 3rd value , etc may not be same.

It is possible that none of the values will be repeated in the new dataset

How will you make projections about future?

Explore what is remaining more or less consistent even if the values are changing.

Use it for future projections, generalizations, etc

DESCRIPTIVE STATISTICS

Productivity Data

69	71
70	68
72	70
71	67
70	69
68	72
73	70
69	71

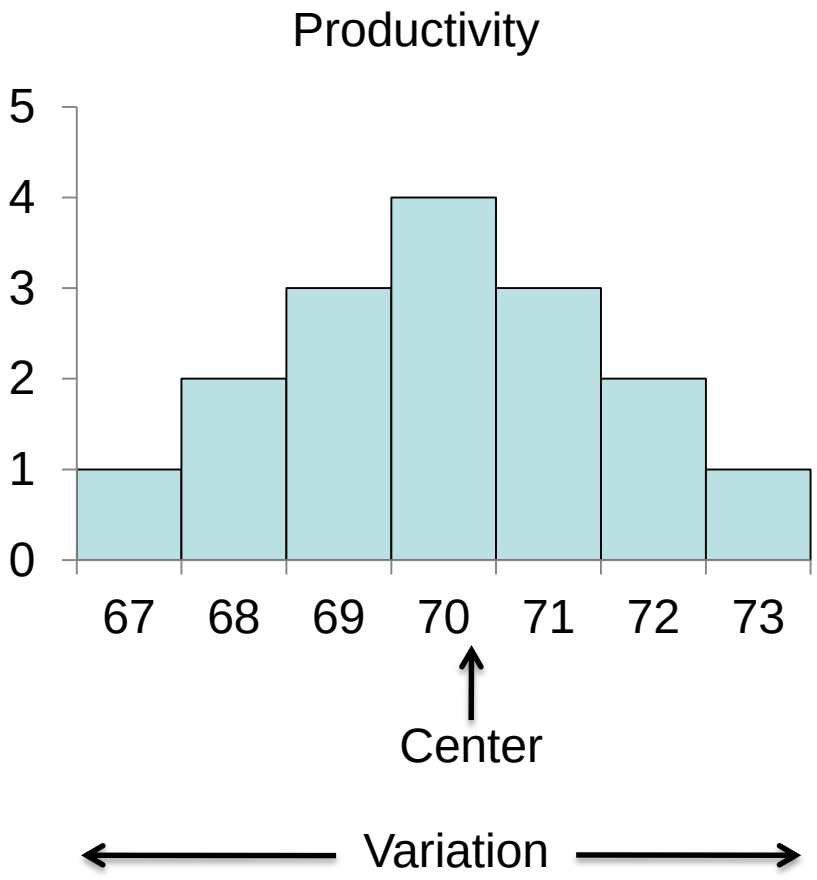
Demonstration

67 68 69 70 71 72 73

DESCRIPTIVE STATISTICS

Interpretation

- Only thing remain more or less consistent is the shape
- All projections or generalizations need to be made using shape
- Shape represents the distribution
- The properties of the shape also will remain more or less constant
- The properties can be used to make projections



DESCRIPTIVE STATISTICS

Measures of Central Tendency

Mean or Average

69	71
70	68
72	70
71	67
70	69
68	72
73	70
69	71

Computation & Interpretation

Centre of gravity of the data

Sum of all values divided by the number of values

Mean = (69 + 70 + - - - + 71) / 16
= 70

On an average 70 tasks are completed in an hour

If we collect data on productivity for another 16 or 20 hours and calculate the mean it will be equal or very close to 70.

DESCRIPTIVE STATISTICS

Measures of Central Tendency

Median

69	71
70	68
72	70
71	67
70	69
68	72
73	70
69	71

Computation & Interpretation

Value dividing the data set into two equal parts

After arranging the data in ascending or descending order, the value in the middle

Suppose there are n values:

If n is odd, median is the value in the $(n + 1)/2^{th}$ position

If n is even, median is the average of $n/2^{th}$ and $(n + 2)/2^{th}$ observation

DESCRIPTIVE STATISTICS

Measures of Central Tendency

Median

Position	Value	Position	Value
1	67	9	70
2	68	10	70
3	68	11	71
4	69	12	71
5	69	13	71
6	69	14	72
7	70	15	72
8	70	16	73

Computation & Interpretation

n = 16

n/2 = 8

(n+2)/2 = 9

Median = Average of 8^{th} and 9^{th} position values

Median = (70 + 70)/2 = 70

Half of the hours the productivity will be less than 70 tasks and remaining 50% of hours productivity will be higher than 70

DESCRIPTIVE STATISTICS

Variable Data Summarization: Measure of Variation or Spread

Range: Definition

Range: Maximum value – Minimum Value

Example:

5	4	7	3	2
15	9	8	5	2

Maximum Value = 15

Minimum Value = 2

Range = 15 – 2 = 13

DESCRIPTIVE STATISTICS

Variable Data Summarization: Measure of Variation or Spread

Range: Issues

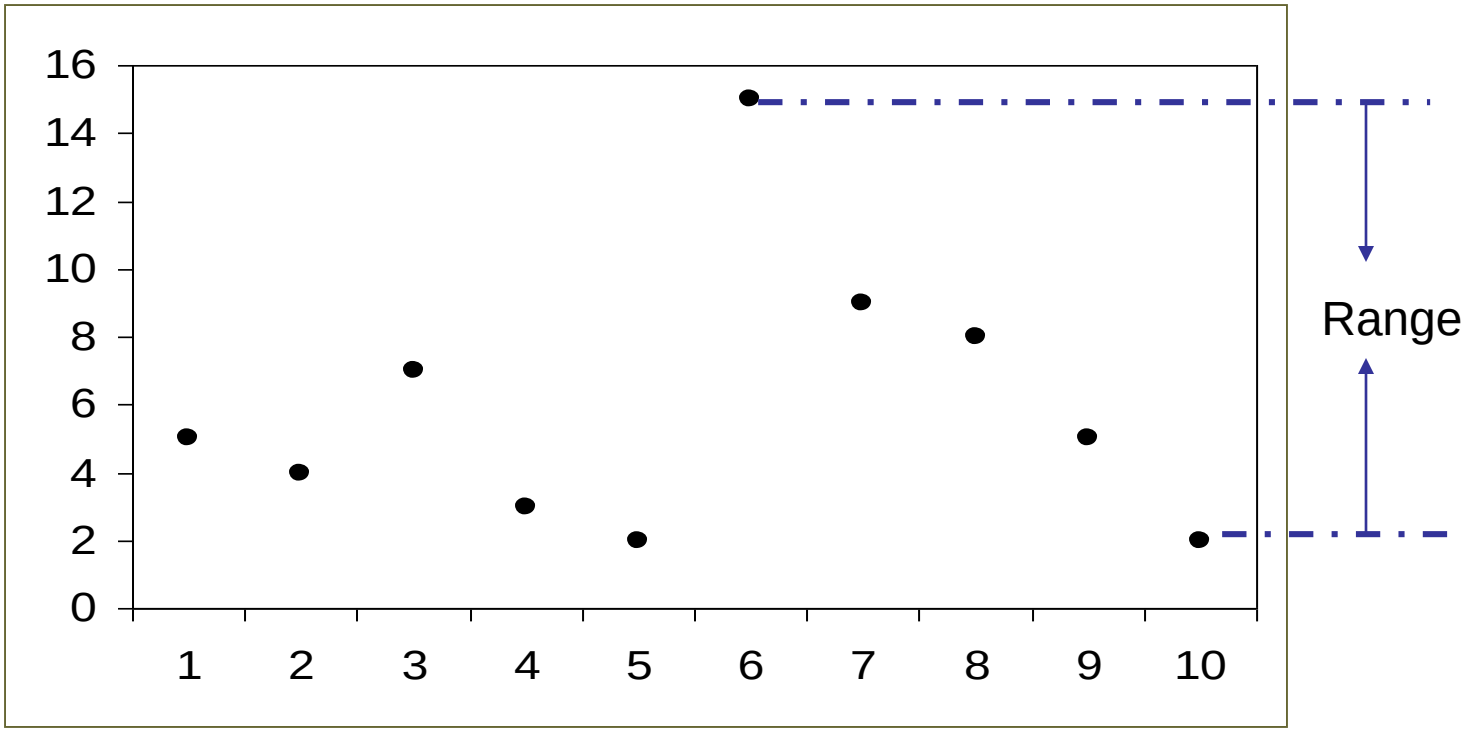
It depends only on extreme values

Hence affected by outliers

DESCRIPTIVE STATISTICS

Variable Data Summarization: Measure of Variation or Spread

Range: Issues



DESCRIPTIVE STATISTICS

Variable Data Summarization: Measure of Variation or Spread

Standard Deviation: Example:

5	4	7	3	2
15	9	8	5	2

Step 1:

Calculate Mean

Mean = 6

DESCRIPTIVE STATISTICS

Variable Data Summarization: Measure of Variation or Spread

Standard Deviation: Example:

5	4	7	3	2
15	9	8	5	2

Step 2:

Take deviations from Mean

-1	-2	1	-3	-4
9	3	2	-1	-4

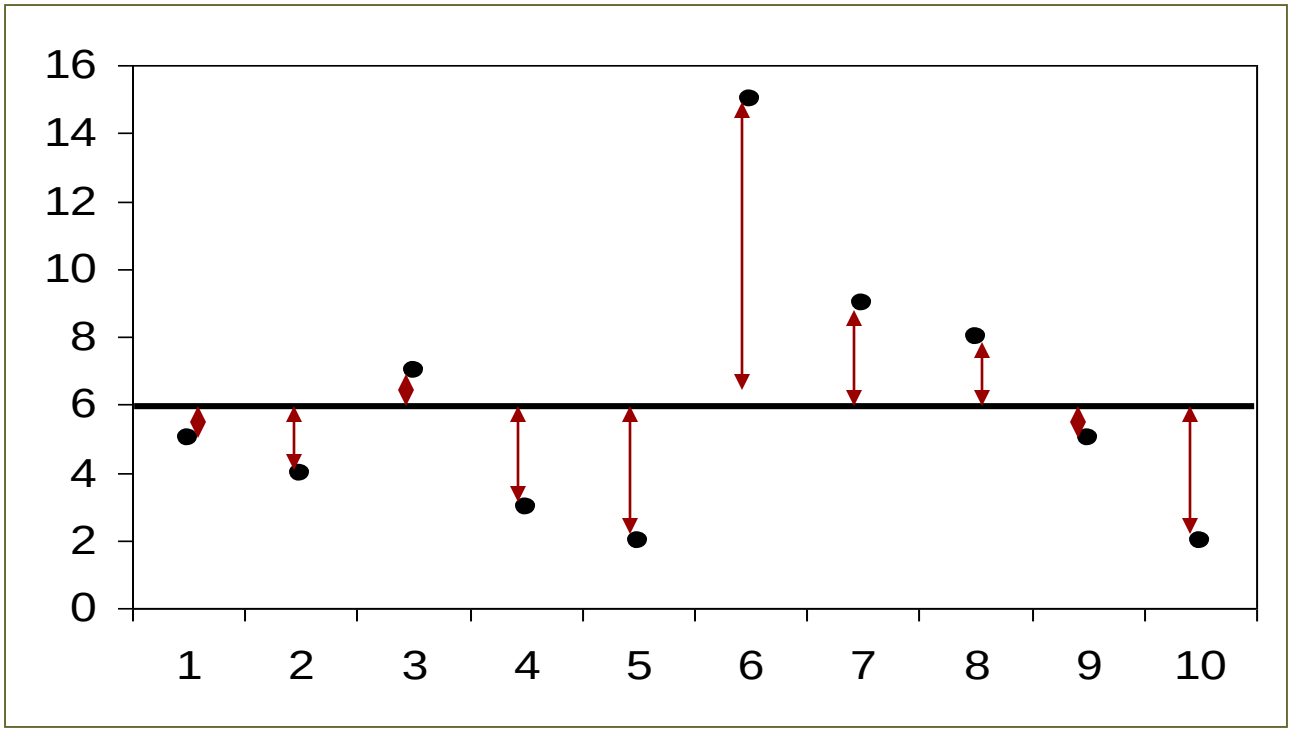
DESCRIPTIVE STATISTICS

Variable Data Summarization: Measure of Variation or Spread

Standard Deviation: Example:

Step 2:

Take deviations from Mean



DESCRIPTIVE STATISTICS

Variable Data Summarization: Measure of Variation or Spread

Standard Deviation: Example:

Step 3:

Since some values are positive & rest are negative, while taking sum they will cancel out.

So square the values & Sum

1	4	1	9	16
81	9	4	1	16

Sum = 142

DESCRIPTIVE STATISTICS

Variable Data Summarization: Measure of Variation or Spread

Standard Deviation: Example:

Step 4:

$$\text{Standard Deviation} = \sqrt{(\text{Sum of Squares} / (n - 1))}$$

$$= \sqrt{(142 / (10 - 1))}$$

$$= \sqrt{15.77} = 3.972$$

MS Excel Function

Standard Deviation = stdev(range of values)

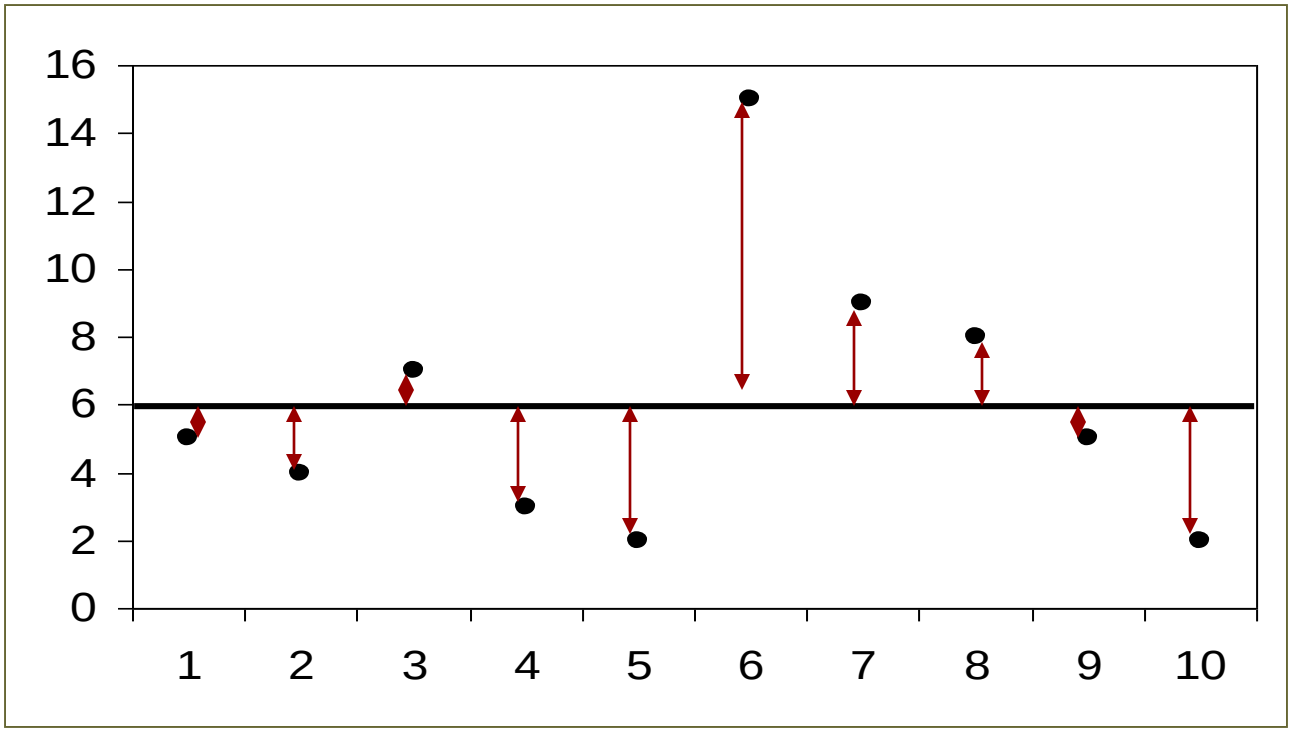
DESCRIPTIVE STATISTICS

Variable Data Summarization: Measure of Variation or Spread

Standard Deviation: Definition

Square root of the average squared deviation from mean

Indicates On an average how much each value is away from the Mean



DESCRIPTIVE STATISTICS

Measures of Variation

Range

69	71
70	68
72	70
71	67
70	69
68	72
73	70
69	71

Computation & Interpretation

Difference between the maximum and minimum value of the data

Maximum = 73

Minimum = 67

Range = 73 – 67 = 6

DESCRIPTIVE STATISTICS

Measures of Variation

Standard Deviation

69	71
70	68
72	70
71	67
70	69
68	72
73	70
69	71

Computation & Interpretation

Square root of the average of the square of the deviations from the mean

On an average how much each values are distributed around the center

Step 1

Compute mean = 70

DESCRIPTIVE STATISTICS

Measures of Variation

Standard Deviation

-1	1
0	-2
2	0
1	-3
0	-1
-2	2
3	0
-1	1

Computation & Interpretation

Step 2

Take deviations from mean

DESCRIPTIVE STATISTICS

Measures of Variation

Standard Deviation

1	1
0	4
4	0
1	9
0	1
4	4
3	0
1	1

Computation & Interpretation

Step 3

Square the deviations

DESCRIPTIVE STATISTICS

Measures of Variation

Standard Deviation

1	1
0	4
4	0
1	9
0	1
4	4
3	0
1	1

Computation & Interpretation

Step 4

Sum the square of deviation

Sum of Squares = 40

DESCRIPTIVE STATISTICS

Measures of Variation

Standard Deviation

1	1
0	4
4	0
1	9
0	1
4	4
3	0
1	1

Computation & Interpretation

Step 5

Variance = Average of the sum of square deviation

Variance = 40 / (16 – 1) = 2.667

Std Deviation = √Variance

= √2.667

= 1.633

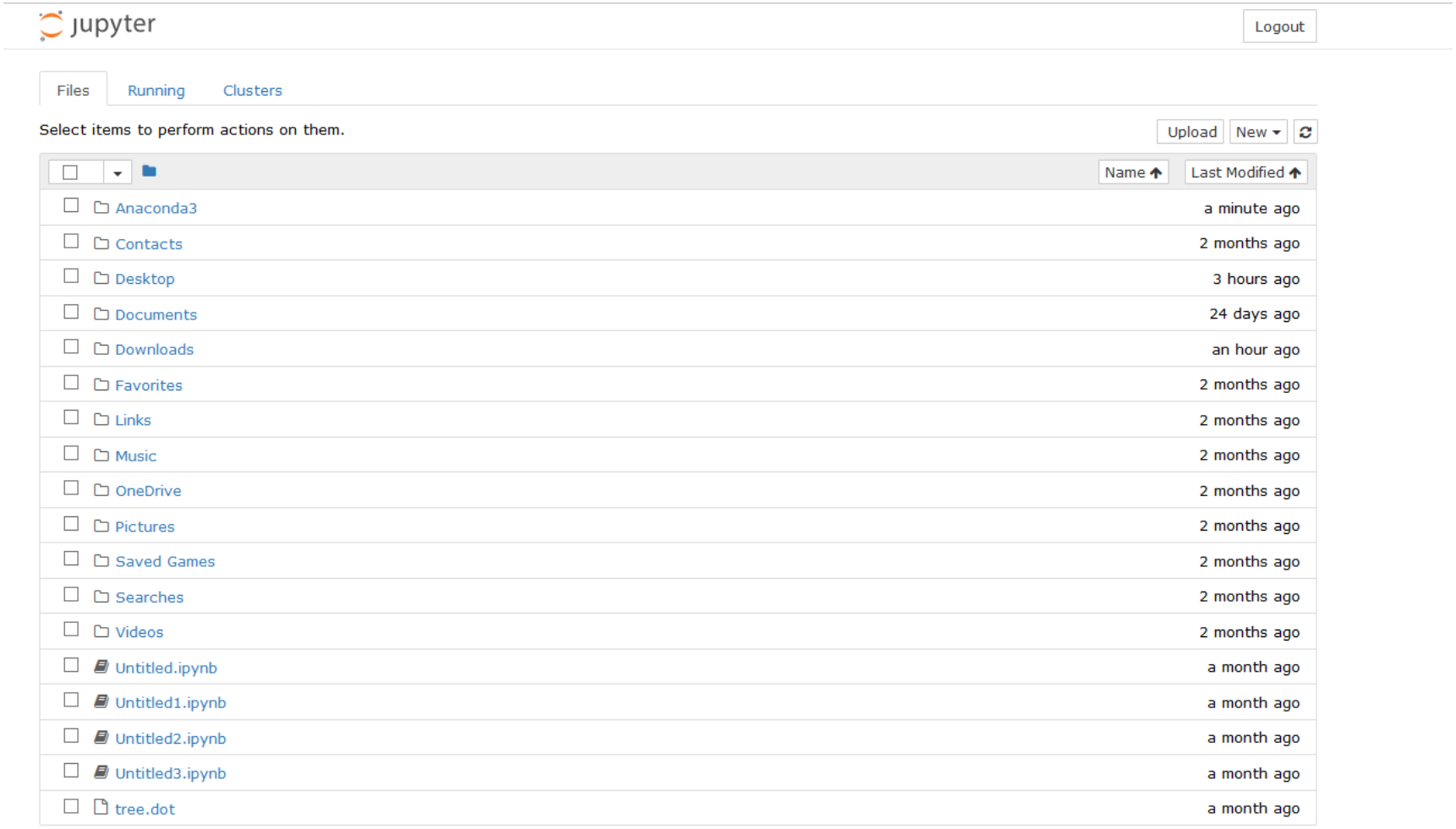
Introduction *to* Python

PYTHON INSTALLATION

1. Download Anaconda from http://jupyter.readthedocs.io/en/latest/install.html
2. Run the set up (exe) file and follow instructions
3. Check Jupyter notebook is installed

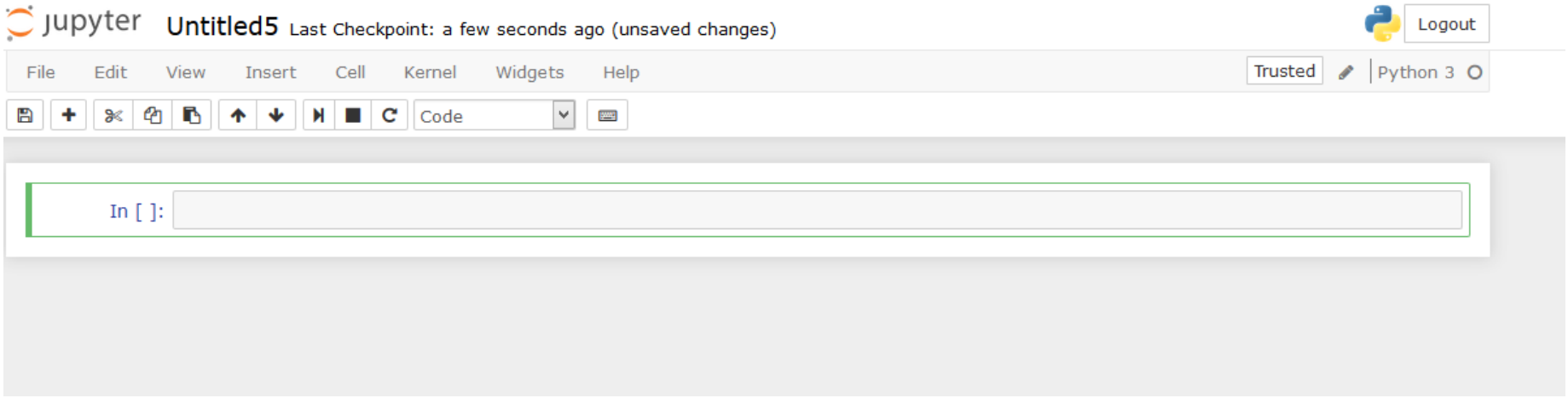
PYTHON INSTALLATION

3. Open Jupyter Notebook



PYTHON INSTALLATION

3. Open Jupyter Notebook



DESCRIPTIVE STATISTICS *using* Python

DESCRIPTIVE STATISTICS

Exercise 1: The monthly credit card expenses of an individual in 1000 rupees is given in the file Monthly_Expenses.csv.

a. Read the dataset to R studio

b. Compute mean, median minimum, maximum, range, variance, standard deviation, skewness, kurtosis and quantiles of Expenses

c. Compute default summary of Monthly Expenses

d. Draw Histogram of Monthly Expenses

DESCRIPTIVE STATISTICS

Reading a csv file : Source code

```
import pandas as mypd
mydata =
mypd.read_csv("E:/ISI/ML02/Course_Material/Dataset/Monthly_Expenses.csv")
mydata
mydata.head()
mydata.info()
```

To read a particular column or variable of data set to a ne variable

Example: Read Expenses to myexp

```
myexp = mydata.Expenses
myexp
```

DESCRIPTIVE STATISTICS

Operators - Arithmetic

Operator	Description
+	addition
-	subtraction
*	multiplication
/	division
**	exponentiation
%	modulus (x mod y) 5%2 is 1

DESCRIPTIVE STATISTICS

Operators - Logical

Operator	Description
<	less than
<=	less than or equal to
>	greater than
>=	greater than or equal to
==	exactly equal to
! =	not equal to

DESCRIPTIVE STATISTICS

Descriptive Statistics

Computation of descriptive statistics for variable myexp

Function	Code	Value
Mean	myexp.mean()	59.2
Median	myexp.median()	59
Mode	myexp.mode()	59
Standard deviation	myexp.std()	3.105
Variance	myexp.var()	9.642
Minimum	myexp.min()	53
Maximum	myexp.max()	65
Percentile	myexp.quantile(0.9)	63
Skewness	myexp.skew()	-0.09
Kurtosis	myexp.kurt()	-0.436

DESCRIPTIVE STATISTICS

Descriptive Statistics

Statistics	Code
Summary	myexp.describe()

Statistics	Value
Count	20
Mean	59.2
Standard Deviation	3.1052
Minimum	53
Q1	57
Median	59
Q3	61
Maximum	65

DESCRIPTIVE STATISTICS

Descriptive Statistics

Arithmetic functions for variable CC

Function	Code	Value
Count	myexp.count()	20
Sum	myexp.sum()	1184
Product	myexp.prod()	6.21447E+18

Function	Code	Value
Square root	Import math as mymath mymath.sqrt(49)	7
Sum of Squares	Sum(myexp**2)	70276

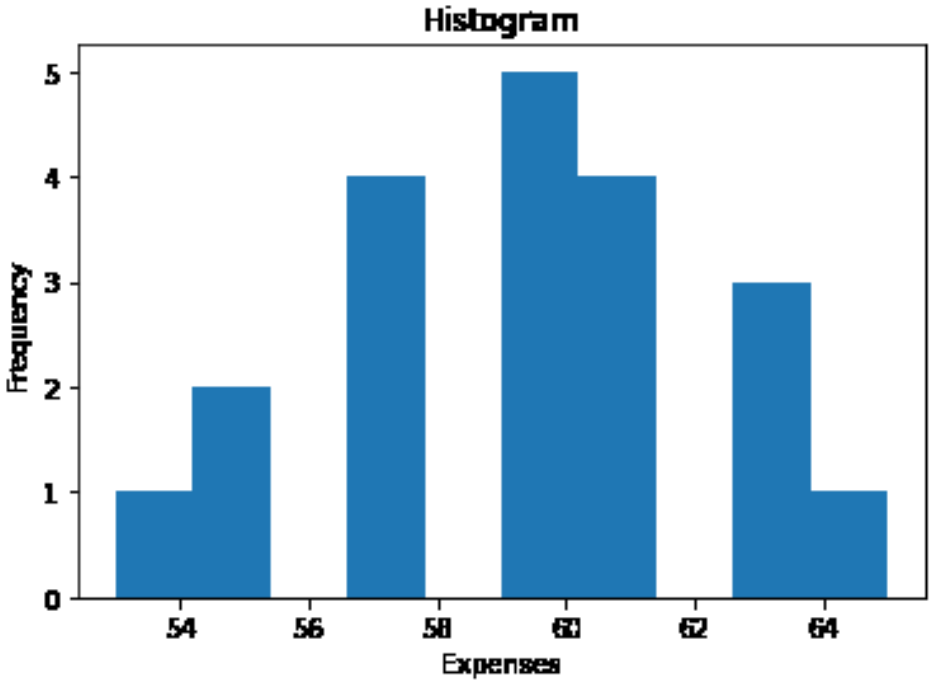
DESCRIPTIVE STATISTICS

Graphs:

Graph	Code
Histogram	import matplotlib.pyplot as myplot myplot.hist(myexp) myplot.title("Histogram") myplot.xlabel(“Expenses") myplot.ylabel("Frequency") myplot.show()
Box Plot	myplot.boxplot(myexp) myplot.title("BoxPlot") myplot.xlabel(“Expenses") myplot.show()

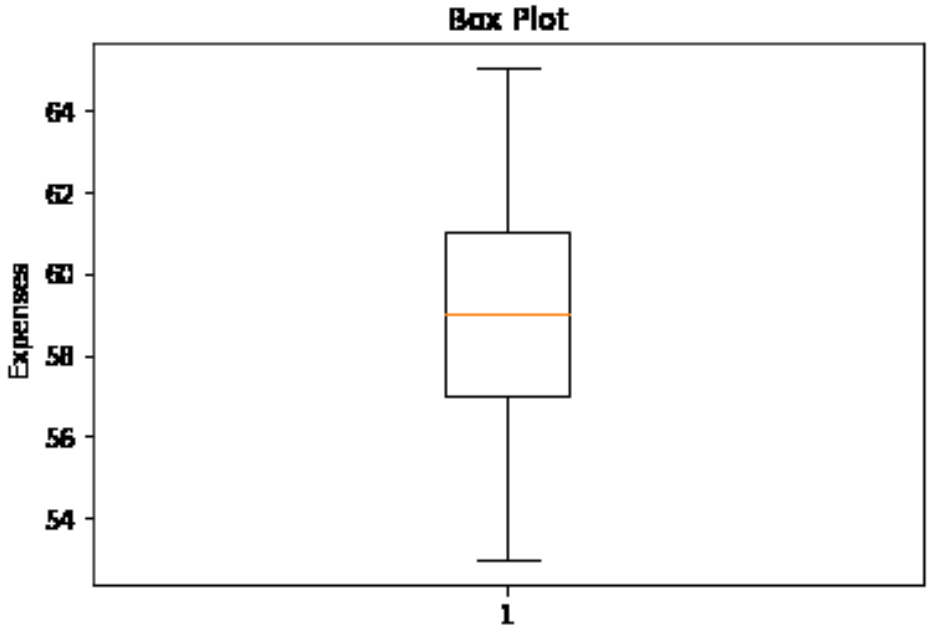
DESCRIPTIVE STATISTICS

Graphs:



DESCRIPTIVE STATISTICS

Graphs:



DESCRIPTIVE STATISTICS

Exercise 2: The data on productivity (number of tasks completed), developer experience (1: Experienced, 2: Fresher), Code reuse (1:High, 2: Low) and usage of knowledge repository usage (1: High , 2: Low) of an technical support process are given in file Productivty.csv.

a. Import the file to Python

b. Copy first 20 records from the file to another dataset and save it

c. Compute descriptive summary of variable Productivity

d. Check whether the average productivity varies with developer experience?

e. Check whether the average productivity vary with code reuse?

f. Check whether the average productivity vary with knowledge repository usage?

g. Compute the aggregate average of productivity with developer experience & code reuse?

h. Compute the aggregate average of usage with all three factors?

DESCRIPTIVE STATISTICS

Exercise 2: The data on productivity (number of tasks completed), developer experience (1: Experienced, 2: Fresher), Code reuse (1:High, 2: Low) and usage of knowledge repository usage (1: High , 2: Low) of an technical support process are given in file Productivty.xlsx.

Reading dataset to variable: mydata

```
import pandas as mypd
import matplotlib.pyplot as myplot
mydata = mypd.read_excel("E:/ISI/ML-02/Course_Material/Productivity.xlsx")
Mydata.head()
```

Reading the variable

```
dev_exp = mydata.Developer_Experience
reuse = mydata.Code_Reuse
kr_usage= mydata.Knowledge_Repository_Usage
prod = mydata.Productivity
```

DESCRIPTIVE STATISTICS

Exercise 2: The data on productivity (number of tasks completed), developer experience (1: Experienced, 2: Fresher), Code reuse (1:High, 2: Low) and usage of knowledge repository usage (1: High , 2: Low) of an technical support process are given in file Productivty.xlsx.

Copying first twenty records to a new data set

```
mynewdata = mydata.iloc[0:20, :]
mynewdata
```

Saving the new dataset

```
mynewdata.to_excel("E:/ISI/ML-02/Course_Material/Newdata.xlsx")
```

DESCRIPTIVE STATISTICS

Exercise 2: The data on productivity (number of tasks completed), developer experience (1: Experienced, 2: Fresher), Code reuse (1:High, 2: Low) and usage of knowledge repository usage (1: High , 2: Low) of an technical support process are given in file Productivty.xlsx.

Computing descriptive statistics for variable : Productivity

prod.describe()

Count	Mean	SD	Minimum	25%	50%	75%	Maximum
30	64.83	39.14	20	32.5	60	86.25	150

DESCRIPTIVE STATISTICS

Exercise 2: The data on productivity (number of tasks completed), developer experience (1: Experienced, 2: Fresher), Code reuse (1:High, 2: Low) and usage of knowledge repository usage (1: High , 2: Low) of an technical support process are given in file Productivty.xlsx.

Computing average productivity for different developer experience

```
prod.groupby(dev_exp).mean()
```

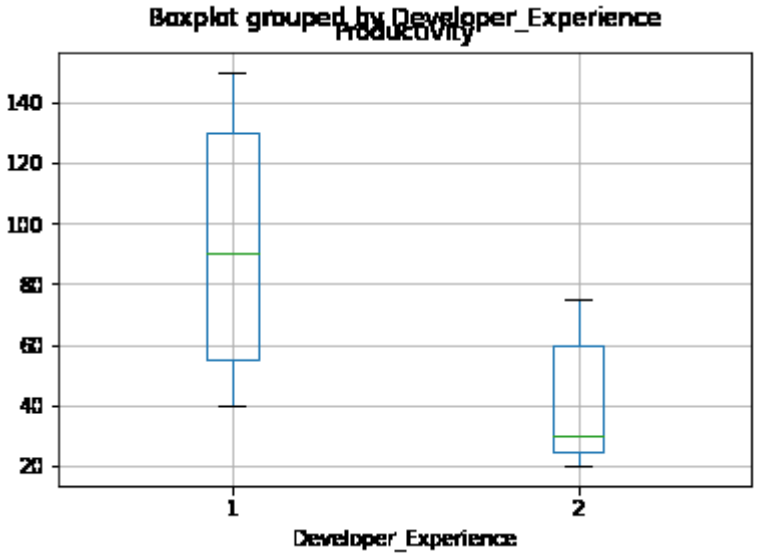
Group	Developer Experience	Average Productivity
1	Experienced	88.67
2	Fresher	41.00

DESCRIPTIVE STATISTICS

Exercise 2: The data on productivity (number of tasks completed), developer experience (1: Experienced, 2: Fresher), Code reuse (1:High, 2: Low) and usage of knowledge repository usage (1: High , 2: Low) of an technical support process are given in file Productivty.xlsx.

Box plot of Productivity by Developer Experience

```
mydata.boxplot(column= "Productivity", by = "Developer_Experience")
myplot.show()
```



DESCRIPTIVE STATISTICS

Exercise 2: The data on productivity (number of tasks completed), developer experience (1: Experienced, 2: Fresher), Code reuse (1:High, 2: Low) and usage of knowledge repository usage (1: High , 2: Low) of an technical support process are given in file Productivty.xlsx.

Computing aggregate average of productivity for different developer experience and code reuse

```
prod.groupby([dev_exp, reuse]).mean()
```

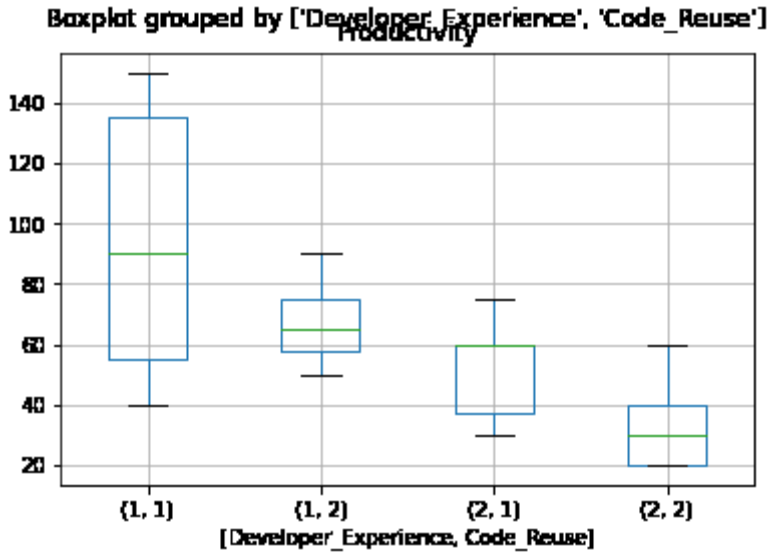
Developer Experience	Code Reuse	Average Productivigy
Experienced	High	96.36
Experienced	Low	67.50
Fresher	High	52.50
Fresher	Low	33.33

DESCRIPTIVE STATISTICS

Exercise 2: The data on productivity (number of tasks completed), developer experience (1: Experienced, 2: Fresher), Code reuse (1:High, 2: Low) and usage of knowledge repository usage (1: High , 2: Low) of an technical support process are given in file Productivty.xlsx.

Box plot of productivity bydeveloper experience and code reuse

```
mydata.boxplot(column= "Productivity", by = ['Developer_Experience',
    'Code_Reuse'])
myplot.show()
```



DESCRIPTIVE STATISTICS

Exercise 2: The data on productivity (number of tasks completed), developer experience (1: Experienced, 2: Fresher), Code reuse (1:High, 2: Low) and usage of knowledge repository usage (1: High , 2: Low) of an technical support process are given in file Productivty.xlsx.

Computing aggregate average of productivity by 3 factors

```
prod.groupby([dev_exp, reuse, kr_usage]).mean()
```

Developer Experience	Code Reuse	Knowledge Repository Usage	Average Productivity
Experienced	High	High	130
Experienced	High	Low	56
Experienced	Low	High	80
Experienced	Low	Low	55
Fresher	High	High	75
Fresher	High	Low	48
Fresher	Low	Low	33.33

DESCRIPTIVE STATISTICS

Exercise 2: The data on productivity (number of tasks completed), developer experience (1: Experienced, 2: Fresher), Code reuse (1:High, 2: Low) and usage of knowledge repository usage (1: High , 2: Low) of an technical support process are given in file Productivty.xlsx.

Computing aggregate summary of credit card usage by 3 factors

```
prod.groupby([dev_exp, reuse, kr_usage]).describe()
```

Developer Experience	Code Reuse	Knowledge Repository Usage	count	mean	std	min	25%	50%	75%	max
Experienced	High	High	6	130	20.9762	90	130	135	140	150
Experienced	High	Low	5	56	20.7364	40	40	50	60	90
Experienced	Low	High	2	80	14.1421	70	75	80	85	90
Experienced	Low	Low	2	55	7.07107	50	52.5	55	57.5	60
Fresher	High	High	1	75		75	75	75	75	75
Fresher	High	Low	5	48	16.4317	30	30	60	60	60
Fresher	Low	low	9	33.33	16.5831	20	20	30	40	60

DESCRIPTIVE STATISTICS

Exercise 3: In IT service provider has conducted a customer satisfaction survey. The four important questions asked are given below: The respondents have to answer each question in a 7 point scale with 1: least satisfied and 7: most satisfied. The data is given in Csat_Freq_table.csv

```
import pandas as mypd
import matplotlib.pyplot as myplot
```

Reading the data set to variable: mydata

```
mydata = mypd.read_csv("E:\LKQ_India\Dataset/Csat_Freq_Table.csv")
mydata.head()
```

Computing Frequency table for Q4

```
myq4 = mydata.q4
n = myq4.count()
mytable = myq4.value_counts()
mytable
```

```
mytable = mytable.sort_index()
```

Rating	Frequency
6	108
4	35
3	13
7	11
5	11
2	1

DESCRIPTIVE STATISTICS

Exercise 3: In IT service provider has conducted a customer satisfaction survey. The four important questions asked are given below: The respondents have to answer each question in a 7 point scale with 1: least satisfied and 7: most satisfied. The data is given in Csat_Freq_table.csv

round(mytable*100/n,2)

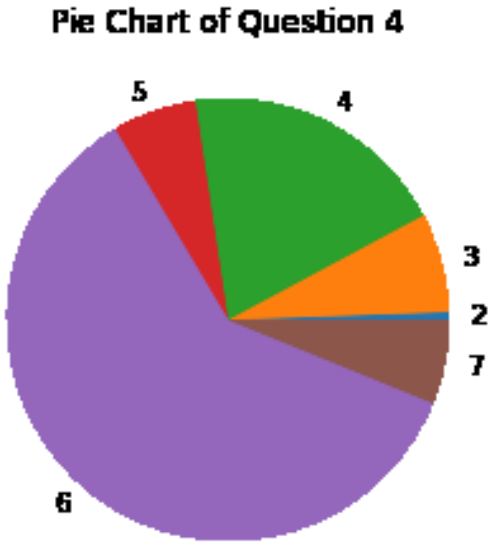
Rating	%
6	60.34
4	19.55
3	7.26
7	6.15
5	6.15
2	0.56

DESCRIPTIVE STATISTICS

Exercise 3: In IT service provider has conducted a customer satisfaction survey. The four important questions asked are given below: The respondents have to answer each question in a 7 point scale with 1: least satisfied and 7: most satisfied. The data is given in Csat_Freq_table.csv

Creating pie chart for Q4

```
myplot.pie(mytable.values, labels = mytable.index)
myplot.title("Pie Chart of Question 4")
myplot.show()
```

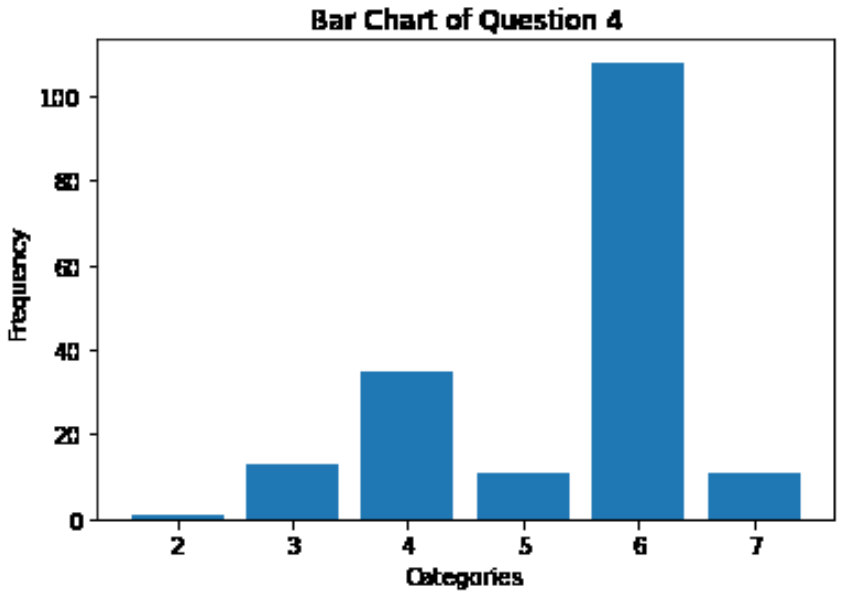


DESCRIPTIVE STATISTICS

Exercise 3: In IT service provider has conducted a customer satisfaction survey. The four important questions asked are given below: The respondents have to answer each question in a 7 point scale with 1: least satisfied and 7: most satisfied. The data is given in Csat_Freq_table.csv

Creating bar chart for Q4

```
myplot.bar(mytable.index, mytable.values)
myplot.xlabel("Categories")
myplot.ylabel("Frequency")
myplot.title("Bar Chart of Question 4")
myplot.show()
```



DATA PREPROCESSING

1. Missing value replenishment
2. Merging data files
3. Appending the data files
4. Transformation or normalization
5. Random Sampling

MISSING VALUE HANDLING

Example:The data on sprint productivity along with software development process variables are given in Preprocesing_Data1 file. Handle the missing values

Reading the data

```
import pandas as mypd
mydata = mypd.read_excel("D:/ISI/ML-02/Dataset/Preprocessing_Data_I.xlsx")
mydata.head()
mydata

mydata.count()
```

MISSING VALUE HANDLING

Option 1: Discard all records with missing values

```
mynewdata = mydata.dropna()
mynewdata
mynewdata.to_excel("D:/ISI/ML-02/Modified_Data.xlsx")
```

MISSING VALUE HANDLING

Option 2: Replace the missing values with variable mean, median, etc

Replacing the missing values with appropriate values

```
rev_time = mydata.Review_Time
rev_time_mean = rev_time.mean()
rev_time.fillna(rev_time_mean, inplace= True)

test_cov = mydata.Test_Coverage
test_cov_median = test_cov.median()
test_cov.fillna(test_cov_median, inplace= True)

rev_cov = mydata.Review_Coverage
rev_cov.fillna(95, inplace= True)
```

Saving the new file

```
mydata.to_excel("D:/ISI/ML-02/Modified_Data.xlsx")
```

DATA MERGING

Exercise: The data is collected for optimizing a mailing campaign. The features are given in Mail_Repond_Features.csv file and the response is given in Mail_Respond_Response.txt file. Can you merge the two files into a single data set?

Read the files

```
import pandas as mypd
myfeatures = mypd.read_csv("D:/ISI/ML 02/Dataset/Mail_Respond_Features.csv")
myresponse = mypd.read_csv("D:/ISI/ML-02/Mail_Respond_Response.txt",
                           delimiter= "\t")
```

Merge the files by “ID” field

```
mydata = mypd.merge(myfeatures, myresponse, on = "SL_No")
```

Exporting the merged data

```
mydata.to_excel("D:/ISI/ML-02/Merged_data.xlsx")
```

DATA APPEND

Exercise: The data is collected from a software development process to study the relationship between the sprint productivity and process features. The data collected is given in two files namely Preprocessing_Data_I and Preprocessing_Data_II. Can you append the second file with the first one?

Read the files

```
import pandas as mypd
pre_data1 = mypd.read_excel("D:/ISI/ML-02//Dataset/Preprocessing_Data_I.xlsx")
pre_data2 = mypd.read_csv("D:/ISI/ML-02/Dataset/Preprocessing_Data_II.csv")

pre_data1.head()
pre_data2.head()
```

Append class1 with class2

```
mydata = pre_data1.append(pre_data2)
mydata.to_excel("D:/ISI/ML-02/Append_Data.xlsx")
```

TRANSFORMATION / NORMALIZATION

z transform:

Transformed data = (Data – Mean) / SD

Min – Max transform:

Transformed data = (Data – Min)/ (Max – Min)

TRANSFORMATION / NORMALIZATION

Exercise : The TAT data of a tech support process is given in TAT file. Normalize the variables in the TAT data?

Read the files

```
import pandas as mypd
from sklearn.preprocessing import StandardScaler as z
from sklearn.preprocessing import MinMaxScaler as minmax

mydata = mypd.read_excel("D:/ISI/ML-02/Course_Material/Dataset/TAT.xlsx")
Mydata.head()
```

TRANSFORMATION / NORMALIZATION

Exercise : The TAT data of a tech support process is given in TAT file. Normalize the variables in the TAT data?

z transform

```
std_data = z().fit_transform(mydata)
std_data = mypd.DataFrame(std_data, columns= mydata.columns.values)
std_data.to_excel("D:/ISI/ML-02/Standardized_Data.xlsx")
```

Min Max transform

```
tr_data = minmax().fit_transform(mydata)
tr_data = mypd.DataFrame(tr_data, columns= mydata.columns.values)
tr_data.to_excel("D:/ISI/ML-02/Tr_data.xlsx")
```

DATA VISUALIZATION

Methodology for exploring the relationship between fields

Generally used to explore relationship between response and explanatory variables in supervised learning

Explanatory variable	Response	Plot
Continuous	Continuous	Scatter plot
Continuous	Categorical	Boxplot, Density plot

Example:

The data on temperature, time, viscosity and yield are given in Chemical_Yield file.

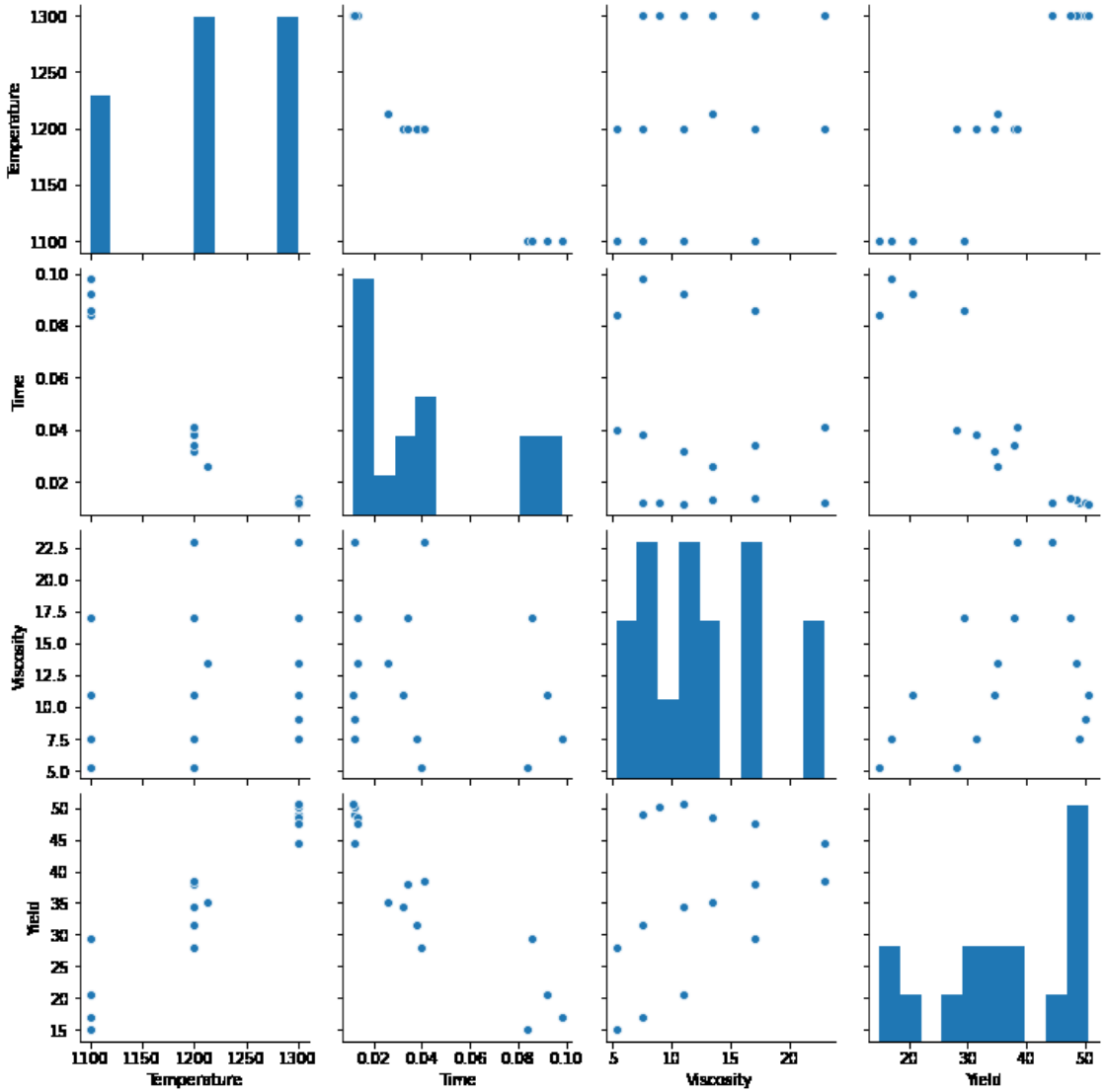
1. Replace the missing values using imputation?
2. Explore the relationship of temperature, time and viscosity to yield graphically

Example:

```
# Importing packages
import pandas as mypd
import matplotlib.pyplot as myplot
import seaborn as mysb

# Importing the dataset
mydata = mypd.read_excel("E:/Caterpillar/Data/Chemical_Yield.xlsx")
mydata.head()

# Checking for missing values
mydata.describe()
mydata.info()
```

Example:

```
# Replacing missing value in temperature with mean
temp = mydata.Temperature
temp_mean = temp.mean()
temp.fillna(temp_mean, inplace= True)
mydata.head(12)

# Removing SL No field
mydata = mydata.iloc[:, 1:5]

# Visualizing the relationship
mysb.pairplot(mydata)
myplot.show()
```


Temperature
Time
Viscosity
Yield
1100
1150
1200
1250
1300
0.02
0.04
0.06
0.08
0.10
5.0
7.5
10.0
12.5
15.0
17.5
20.0
22.5
15
20
25
30
35
40
45
50
5
10
15
20

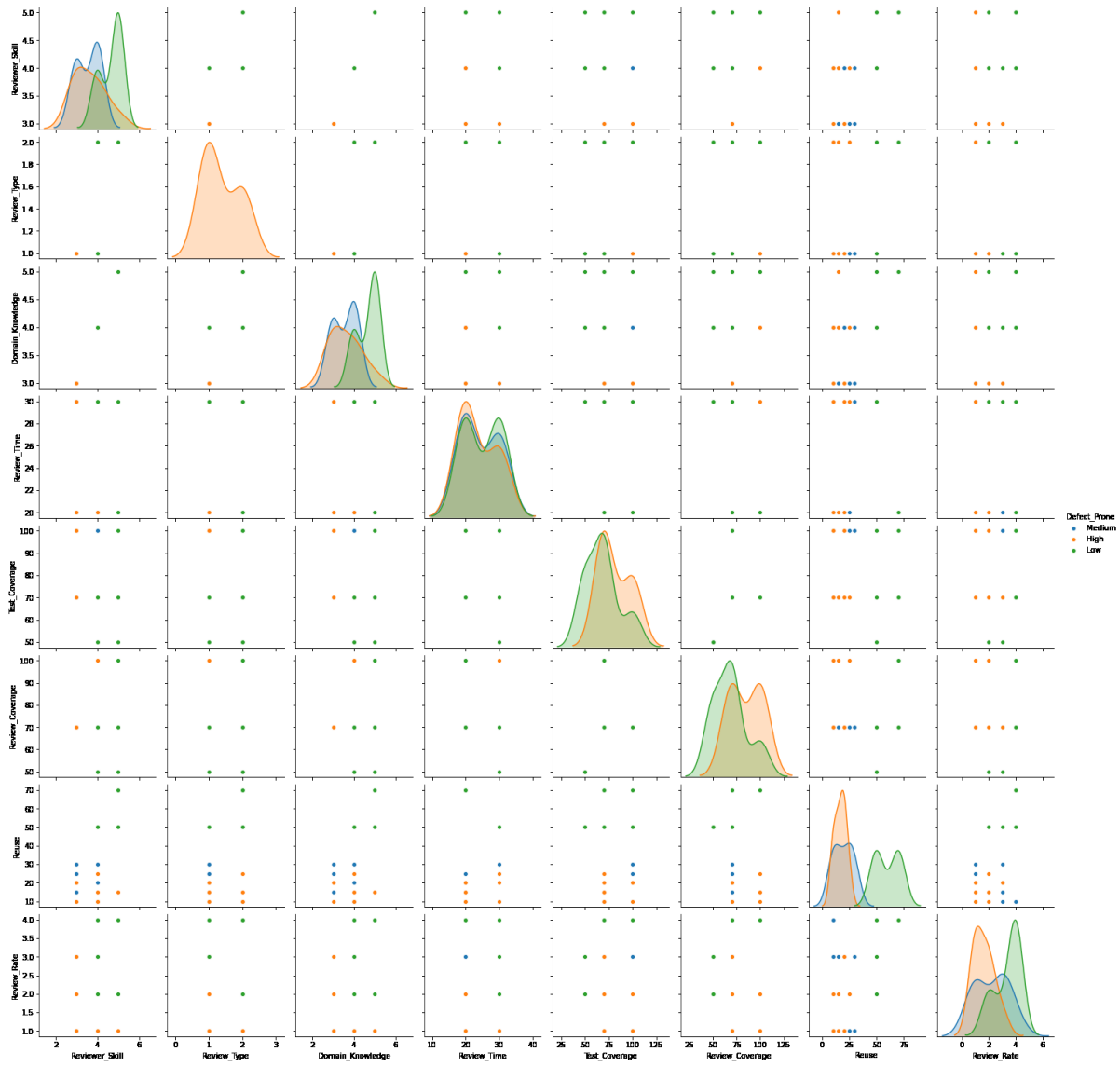
Example:

The data on defect proneness of a tech support process is given in Defect_Proneness file. Explore the relationship between defect proneness with the process variables graphically

Example:

```
# Importing packages
import pandas as mypd
import matplotlib.pyplot as myplot
import seaborn as mysb

# Visualizing the relationship
mysb.pairplot(mydata, hue= "Defect_Prone")
myplot.show()
```

Pair Plot:


Reviewer_Skill
Review_Type
Domain_Knowledge
Review_Time
Test_Coverage
Review_Coverage
Reuse
Review_Rate
Defect_Prone
Medium
High
Low

ADVANCED IMPUTATION METHODS

ADVANCED IMPUTATION METHODS

Advanced techniques for replacing missing values:

k nearest neighbor method

Bagging method

ADVANCED IMPUTATION METHODS

k nearest neighbor method

Example: The data on temperature, time, viscosity and the yield of a chemical process is given below. Can you replace the missing value in using k-nn method with k = 3?

SL No.	Temperature	Time	Viscosity	Yield
1	1300	0.012	7.5	49
2	1300	0.012	9	50.2
3	1300	0.0115	11	50.5
4	1300	0.013	13.5	48.5
5	1300	0.0135	17	47.5
6	1300	0.012	23	44.5
7	1200	0.04	5.3	28
8	1200	0.038	7.5	31.5
9	1200	0.032	11	34.5
10		0.026	13.5	35
11	1200	0.034	17	38
12	1200	0.041	23	38.5
13	1100	0.084	5.3	15
14	1100	0.098	7.5	17
15	1100	0.092	11	20.5
16	1100	0.086	17	29.5

ADVANCED IMPUTATION METHODS

k nearest neighbor method

Replacing missing temperature value (10th record)

Step 1: Identify k = 3 neighbors of 10 the record time, viscosity & yield variables

SL No.	Temperature	Time	Viscosity	Yield
10		0.026	13.5	35

ADVANCED IMPUTATION METHODS

k nearest neighbor method

Replacing missing temperature value (10th record)

Step 2: Compute Euclidean distance of each record to 10th record using time, viscosity & yield fields

SL No.	Temperature	Time	Viscosity	Yield	ED^2	ED
1	1300	0.012	7.5	49	232.00	15.23
2	1300	0.012	9	50.2	251.29	15.85
3	1300	0.0115	11	50.5	246.50	15.70
4	1300	0.013	13.5	48.5	182.25	13.50
5	1300	0.0135	17	47.5	168.50	12.98
6	1300	0.012	23	44.5	180.50	13.44
7	1200	0.04	5.3	28	116.24	10.78
8	1200	0.038	7.5	31.5	48.25	6.95
9	1200	0.032	11	34.5	6.50	2.55
10		0.026	13.5	35		
11	1200	0.034	17	38	21.25	4.61
12	1200	0.041	23	38.5	102.50	10.12
13	1100	0.084	5.3	15	467.24	21.62
14	1100	0.098	7.5	17	360.01	18.97
15	1100	0.092	11	20.5	216.50	14.71
16	1100	0.086	17	29.5	42.50	6.52

ADVANCED IMPUTATION METHODS

k nearest neighbor method

Replacing missing temperature value (10th record)

Step 3: Identify 3 neighbors as records having lowest Euclidean distance

SL No.	Temperature	Time	Viscosity	Yield	ED^2	ED
9	1200	0.032	11	34.5	6.50004	2.55
10		0.026	13.5	35		
11	1200	0.034	17	38	21.2501	4.61
16	1100	0.086	17	29.5	42.5036	6.52

Step 4: Replace the temperature missing value with the average temperature of the nearest neighbors

10th record temperature = (1200 + 1200 + 1100)/3 = 1166.7

ADVANCED IMPUTATION METHODS

KNN Imputation using Python - Example

The sprint productivity data is given in Preprocessing_Data_I and Preprocessing_Data_II files. Kindly append the files and replace the missing values using k-nn (k = 4)

```
# Import packages
import pandas as mypd
from sklearn.impute import KNNImputer

# Import Preprocessing_Data_I data set
SP_I = mypd.read_excel("E:/hp/hp_2020/Module3/Data/Preprocessing_Data_I.xlsx")
SP_I.head()

# Import Preprocessing_Data_II data set
SP_II = mypd.read_csv("E:/hp/hp_2020/Module3/Data/Preprocessing_Data_II.csv")
SP_II.head()
```

ADVANCED IMPUTATION METHODS

KNN Imputation using Python - Example

The sprint productivity data is given in Preprocessing_Data_I and Preprocessing_Data_II files. Kindly append the files and replace the missing values using k-nn (k = 4)

```
# Append the datasets
mydata = SP_I.append(SP_II)
mydata

# Descriptive summary of dataset
mydata.describe()
mydata.info()
```

ADVANCED IMPUTATION METHODS

KNN Imputation using Python - Example

The sprint productivity data is given in Preprocessing_Data_I and Preprocessing_Data_II files. Kindly append the files and replace the missing values using k-nn (k = 4)

```
# knn imputation
knn = KNNImputer(n_neighbors=4)
cleaned_data = knn.fit_transform(mydata)

# Making the cleaned as a dataframe object
cleaned_data = mypd.DataFrame(cleaned_data, columns= mydata.columns)
cleaned_data
cleaned_data.describe()
cleaned_data.info()
```

FEATURE SELECTION

FEATURE SELECTION

Recursive Feature Elimination Method

A methodology to select most relevant features in a dataset

A way to select the important factors or fields in a dataset based on its usefulness in predicting a target or response variable

An effective way to eliminate the features not useful for predicting the response variable

FEATURE SELECTION

Recursive Feature Elimination Method

1. Fit a model with all features
2. Validate the model
3. Calculate accuracy or score
4. Remove the least contributing feature
5. Fit the model with remaining feature and go back to step 2

The procedure stops when all the remaining features are important.

Generally accuracy is estimated using cross validation

FEATURE SELECTION

Recursive Feature Elimination Method - R

Response	Commonly used models
Continuous	Linear regression General additive models Random Forest Bagging, etc
Categorical	Logistic regression Linear discriminant analysis General additive models Random Forest Bagging, etc

FEATURE SELECTION

Recursive Feature Elimination Method - Python

Example: Select the 5 important features in the Boston housing data. The response variable is MEDV.

```
# Import packages
import pandas as mypd
from sklearn.feature_selection import RFE
from sklearn import tree

# Import dataset
mydata = mypd.read_csv("E:/hp/hp_2020/Module3/Data/Boston_Housing_Data.csv")
mydata.head()

# Separate x's & y
x = mydata.iloc[:, 0:13]
y = mydata.MEDV

# Set up the model
mymodel = tree.DecisionTreeRegressor()
```

FEATURE SELECTION

Recursive Feature Elimination Method - Python

Example: Select the 5 important features in the Boston housing data. The response variable is MEDV.

```
# Set up Recursive feature Elimination
mysearch = RFE(mymodel, n_features_to_select=5, step=1)

# Fit RFE to the data
mysearch = mysearch.fit(x,y)

# identify important features
decision = mysearch.support_
decision = mypd.DataFrame(decision, columns=["Decision"])

# featue ranking
rank = mysearch.ranking_
rank = mypd.DataFrame(rank, columns=["Rank"])
```

FEATURE SELECTION

Recursive Feature Elimination Method - Python

Example: Select the 5 important features in the Boston housing data. The response variable is MEDV.

```
# getting the feature names
features = x.columns
features = mypd.DataFrame(features, columns= ["Features"])

# Preparing the result
myresult = features.join(decision)
myresult = myresult.join(rank)
myresult

# Exporting the result
myresult.to_excel("E:/hp/hp_2020/Module3/rfe_result.xlsx")
```

FEATURE SELECTION

Recursive Feature Elimination Method - Python

Example: Select the 5 important features in the Boston housing data. The response variable is MEDV.

Features	Decision	Rank
CRIM	TRUE	1
ZN	FALSE	8
INDUS	FALSE	6
CHAS	FALSE	7
NOX	TRUE	1
RM	TRUE	1
AGE	FALSE	3
DIS	TRUE	1
RAD	FALSE	9
TAX	FALSE	2
PTRATIO	FALSE	5
B	FALSE	4
LSTAT	TRUE	1

FEATURE SELECTION

Recursive Feature Elimination Method - Python

Note: To get importance ranking of features select n_features_to_select = 1

```
mysearch = RFE(mymodel, n_features_to_select=1, step=1)
```

Features	Rank
CRIM	3
ZN	13
INDUS	11
CHAS	12
NOX	5
RM	1
AGE	7
DIS	4
RAD	10
TAX	6
PTRATIO	8
B	9
LSTAT	2

FEATURE SELECTION

Recursive Feature Elimination Method - Python

Example: Identify 5 most important features deciding whether a customer will take pep or not in bank-data?

```
# Import packages
import pandas as mypd
from sklearn.feature_selection import RFE
from sklearn import tree

# Import dataset
mydata = mypd.read_csv("E:/hp/hp_2020/Module3/Data/bank-data.csv")
mydata.head()

# Separate x's & y
x = mydata.iloc[:, 0:10]
y = mydata.pep

# Set up the model
mymodel = tree.DecisionTreeClassifier()
```

FEATURE SELECTION

Recursive Feature Elimination Method - Python

Example: Identify 5 most important features deciding whether a customer will take pep or not in bank-data?

```
# Set up Recursive feature Elimination
mysearch = RFE(mymodel, n_features_to_select=5, step=1)

# Fit RFE to the data
mysearch = mysearch.fit(x,y)

# identify important features
decision = mysearch.support_
decision = mypd.DataFrame(decision, columns=["Decision"])

# featue ranking
rank = mysearch.ranking_
rank = mypd.DataFrame(rank, columns=["Rank"])
```

FEATURE SELECTION

Recursive Feature Elimination Method - Python

Example: Identify 5 most important features deciding whether a customer will take pep or not in bank-data?

```
# getting the feature names
features = x.columns
features = mypd.DataFrame(features, columns= ["Features"])

# Preparing the result
myresult = features.join(decision)
myresult = myresult.join(rank)
myresult

# Exporting the result
myresult.to_excel("E:/hp/hp_2020/Module3/rfe_result.xlsx")
```

FEATURE SELECTION

Recursive Feature Elimination Method - Python

Example: Identify 5 most important features deciding whether a customer will take pep or not in bank-data?

Features	Decision	Rank
age	TRUE	1
sex	FALSE	6
region	FALSE	3
Income	TRUE	1
married	FALSE	2
children	TRUE	1
car	FALSE	4
save_act	TRUE	1
current_act	FALSE	5
mortgage	TRUE	1

CORRELATION
&
REGRESSION

CORRELATION & REGRESSION

Correlation:

Correlation analysis is a technique to identify the relationship between two variables.

Type and degree of relationship between two variables.

CORRELATION & REGRESSION

Correlation: Usage

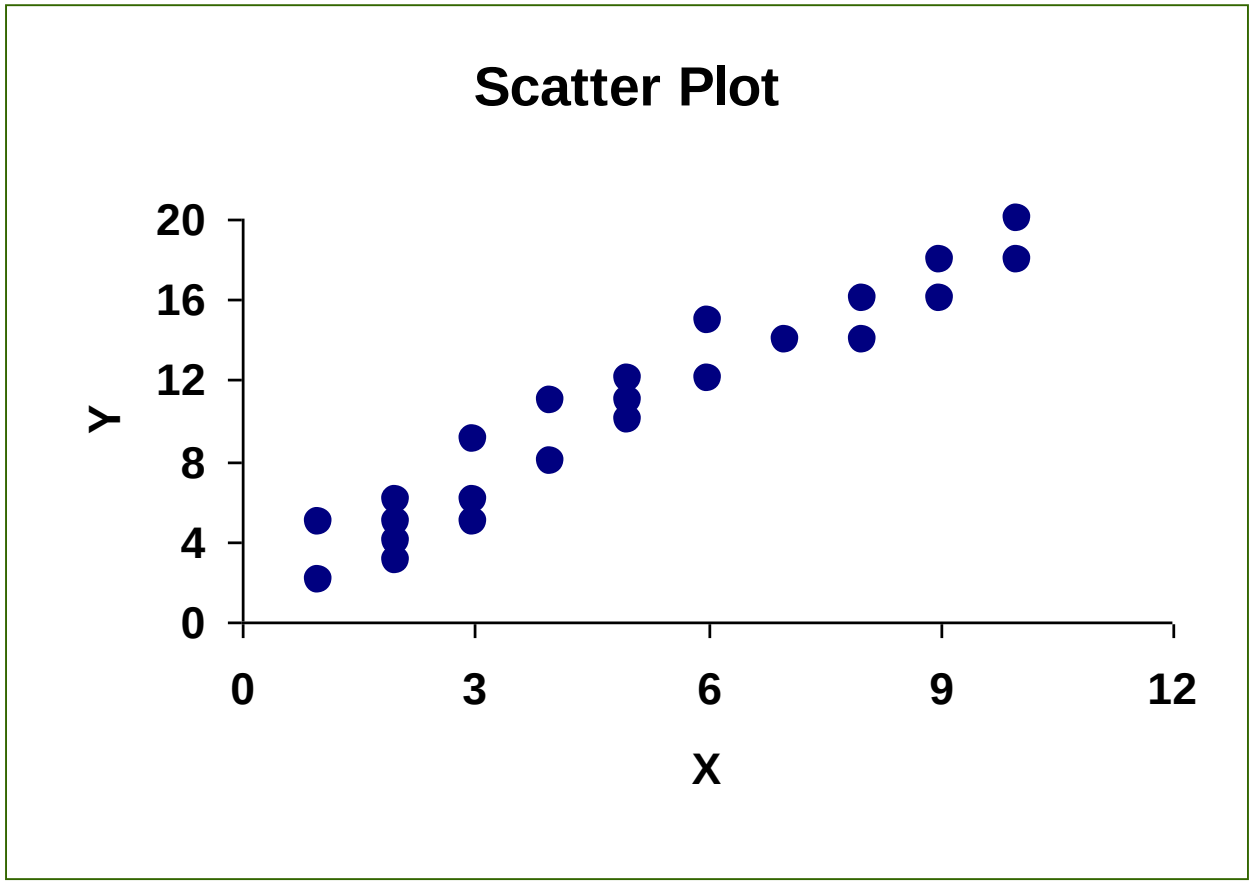
Explore the relationship between the output characteristic and input or process variable.

Output variable : y : Dependent variable

Input / Process variable : x : Independent variable

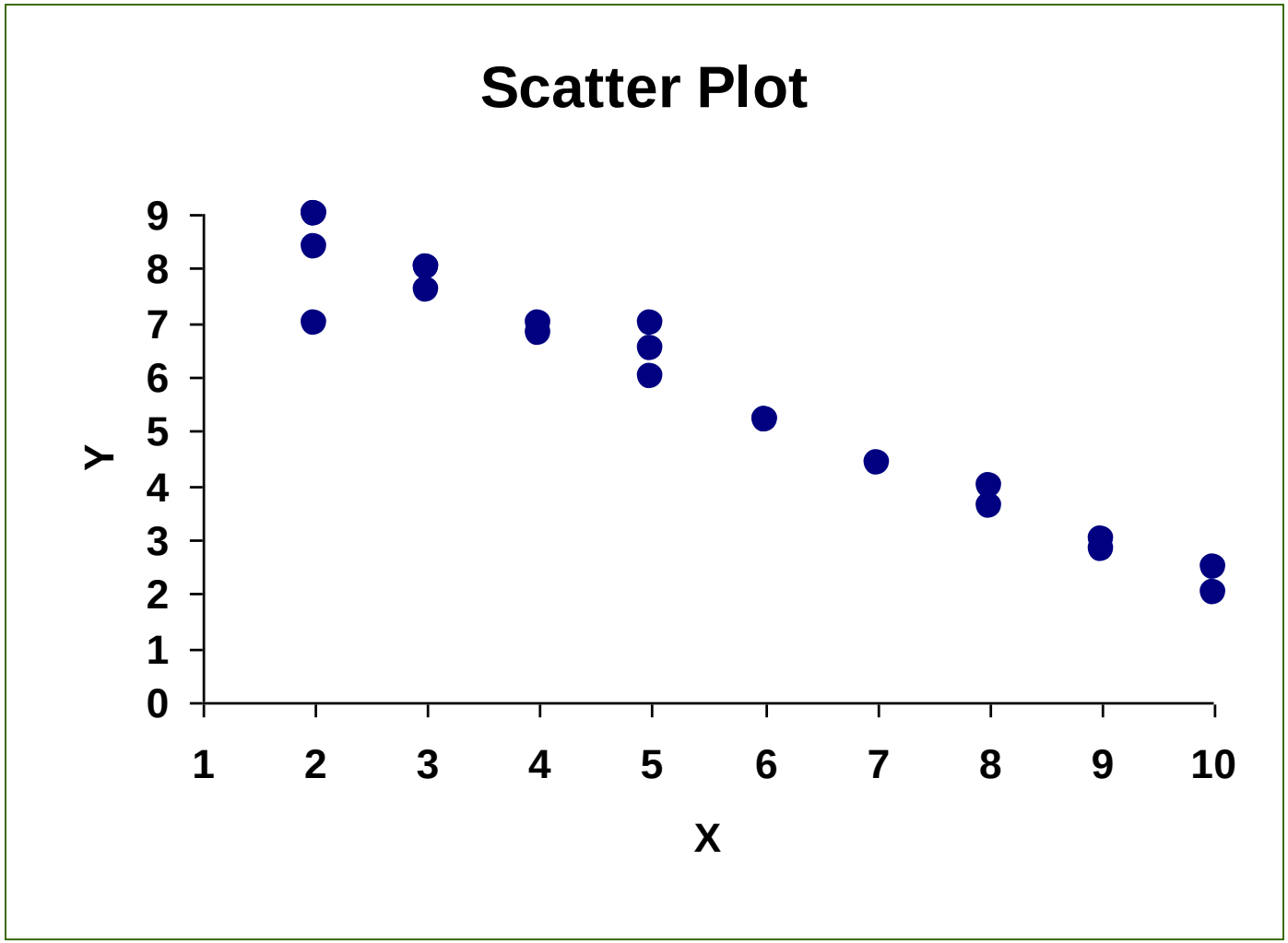
CORRELATION & REGRESSION

Positive Correlation: y increases as x increases & vice versa



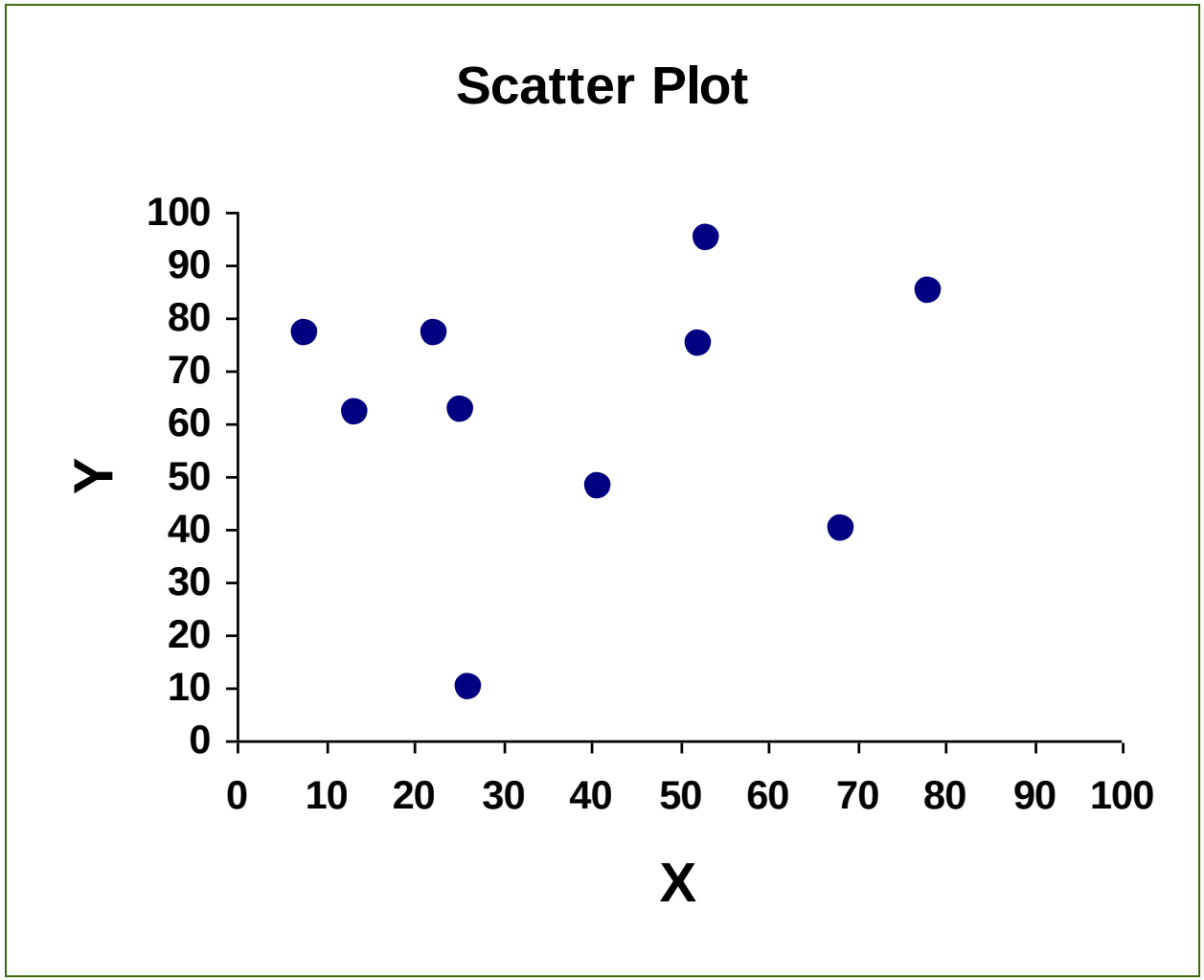
CORRELATION & REGRESSION

Negative Correlation: y decreases as x increases & vice versa



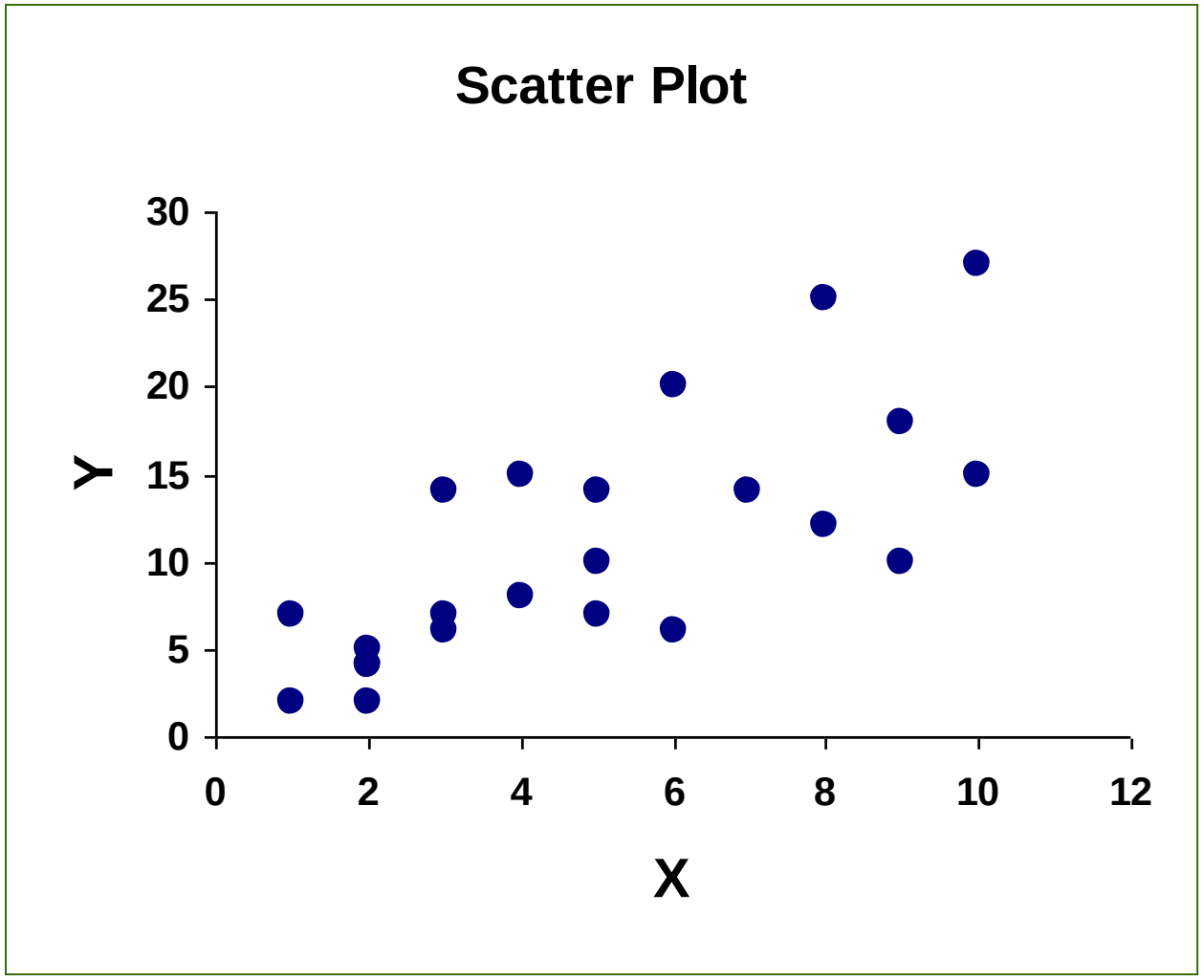
CORRELATION & REGRESSION

No Correlation: Random Distribution of points



CORRELATION & REGRESSION

Is there any correlation ?



CORRELATION & REGRESSION

Measure of Correlation: Coefficient of Correlation

Symbol : r

Range : -1 to 1

Sign : Type of correlation

Value : Degree of correlation

Examples:

r = 0.6 , 60 % positive correlation

r = -0.82, 82% negative correlation

r = 0, No correlation

CORRELATION & REGRESSION

Coefficient of Correlation Computation: Positive Correlation

Collect data on x and y: When x is low, y is also low & vice versa

x	y
2	5
3	7
1	3
5	11
6	12
7	15

CORRELATION & REGRESSION

Coefficient of Correlation Computation : Positive Correlation

Calculate Mean of x & y values

SL No.	x	y
1	2	5
2	3	7
3	1	3
4	5	11
5	6	12
6	7	15
Mean	4	8.83

CORRELATION & REGRESSION

Coefficient of Correlation Computation : Positive Correlation

Take x – Mean x and y – Mean y

SL No.	x – Mean x	y – Mean y
1	-2	-3.83
2	-1	-1.83
3	-3	-5.83
4	1	2.17
5	2	3.17
6	3	6.17

Conclusion:

Low values will become negative & high values will become positive

CORRELATION & REGRESSION

Coefficient of Correlation Computation : Positive Correlation

Generally when x values are negative, y values are also negative & vice versa

SL No.	x – Mean x	y – Mean y
1	-2	-3.83
2	-1	-1.83
3	-3	-5.83
4	1	2.17
5	2	3.17
6	3	6.17

CORRELATION & REGRESSION

Coefficient of Correlation Computation : Positive Correlation

Then

Product of x & y values will be generally positive

SL No.	x – Mean x	y – Mean y	Product
1	-2	-3.83	7.66
2	-1	-1.83	1.83
3	-3	-5.83	17.49
4	1	2.17	2.17
5	2	3.17	6.34
6	3	6.17	18.51
		Sum = Sxy	54

CORRELATION & REGRESSION

Coefficient of Correlation Computation : Positive Correlation

Sum of Product of x & y values (Sxy) will be positive

SL No.	x – Mean x	y – Mean y	Product
1	-2	-3.83	7.66
2	-1	-1.83	1.83
3	-3	-5.83	17.49
4	1	2.17	2.17
5	2	3.17	6.34
6	3	6.17	18.51
		Sum = Sxy	54

CORRELATION & REGRESSION

Coefficient of Correlation Computation : Negative Correlation

Collect data on x and y: When x is low then y will be high & vice versa

x	y
2	12
3	11
1	15
5	7
6	5
7	3

CORRELATION & REGRESSION

Coefficient of Correlation Computation : Negative Correlation

Calculate Mean of x & y values

SL No.	x	y
1	2	12
2	3	11
3	1	15
4	5	7
5	6	5
6	7	3
Mean	4	8.83

CORRELATION & REGRESSION

Coefficient of Correlation Computation : Negative Correlation

Take x – Mean x and y – Mean y

SL No.	x – Mean x	y – Mean y
1	-2	3.67
2	-1	2.67
3	-3	6.67
4	1	-1.33
5	2	-3.33
6	3	-5.33

Conclusion:

Low values will become negative & high values will become positive

CORRELATION & REGRESSION

Coefficient of Correlation Computation : Negative Correlation

Generally when x values are negative, y values are positive & vice versa

SL No.	x – Mean x	y – Mean y
1	-2	3.67
2	-1	2.67
3	-3	6.67
4	1	-1.33
5	2	-3.33
6	3	-5.33

CORRELATION & REGRESSION

Coefficient of Correlation Computation : Negative Correlation

Then

Product of x & y values will be generally negative

SL No.	x – Mean x	y – Mean y	Product
1	-2	3.67	-7.34
2	-1	2.67	-2.67
3	-3	6.67	-20.01
4	1	-1.33	-1.33
5	2	-3.33	-6.66
6	3	-5.33	-15.99
		Sum = Sxy	- 54

CORRELATION & REGRESSION

Coefficient of Correlation Computation : Negative Correlation

Sum of Product of x & y values Sxy will be negative

SL No.	x – Mean x	y – Mean y	Product
1	-2	3.67	-7.34
2	-1	2.67	-2.67
3	-3	6.67	-20.01
4	1	-1.33	-1.33
5	2	-3.33	-6.66
6	3	-5.33	-15.99
		Sum = Sxy	- 54

CORRELATION & REGRESSION

Coefficient of Correlation Computation :

In Short

If correlation is positive

Sxy will be positive

If correlation is negative

Sxy will be negative

CORRELATION & REGRESSION

Coefficient of Correlation Computation :

Sxy is divided by $\sqrt{}$ (Sxx.Syy)

Sxy = Σ(x-Mean x)(y-Mean y)

Sxx = Σ(x-Mean x)^2

Syy = Σ(y-Mean y)^2

Correlation Coefficient r = Sxy / $\sqrt{}$ (Sxx.Syy)

CORRELATION & REGRESSION

Coefficient of Correlation Computation :

SL No.	x – Mean x	y – Mean y	Product	$(x – Mean\ x)^2$	$(y – Mean\ y)^2$
1	-2	3.67	-7.34	4	14.6689
2	-1	2.67	-2.67	1	3.3489
3	-3	6.67	-20.01	9	33.9889
4	1	-1.33	-1.33	1	4.7089
5	2	-3.33	-6.66	4	10.0489
6	3	-5.33	-15.99	9	38.0689
Sum			Sxy: -54	Sxx: 28	Syy:104.83

$r = Sxy / \sqrt{Sxx.Syy} = -54 / \sqrt{(28 \times 104.83)} = -0.9967$

CORRELATION & REGRESSION

Exercise**:** The data on vapor pressure of water at various temperatures are given in Correlation.csv file.

1. Construct the scatter plot and interpret?
2. Compute the correlation coefficient?

CORRELATION & REGRESSION

Exercise**:** The data on vapor pressure of water at various temperatures are given in Correlation.csv file.

1. Reading the data and variables

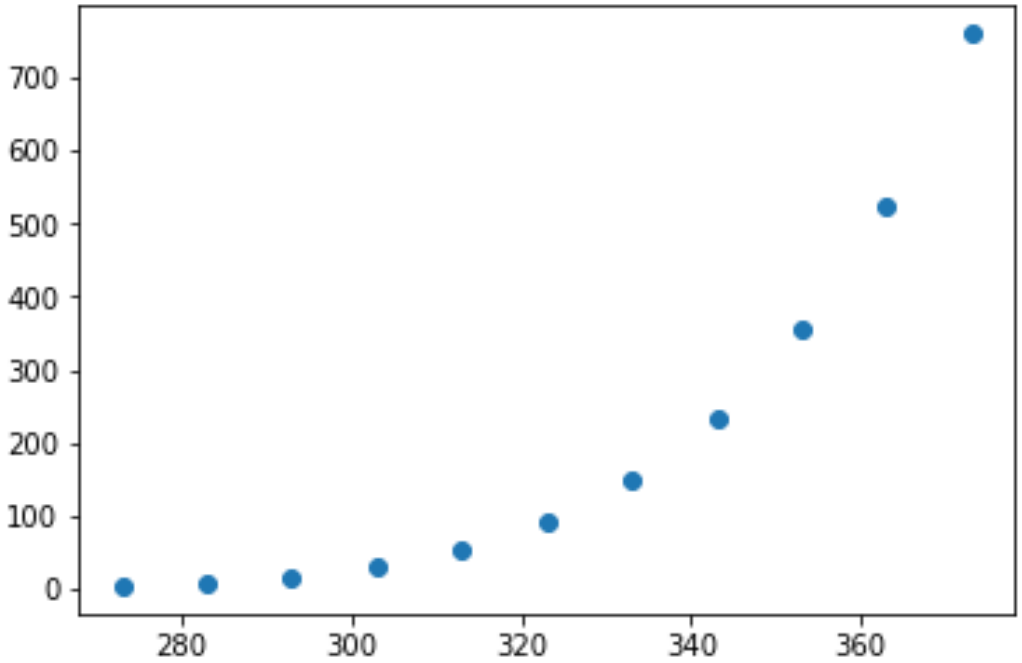
```
import pandas as mypd
import numpy as mynp
import matplotlib.pyplot as myplot
mydata = mypd.read_csv("E:/ISI/PM-01/Data/Correlation.csv")
temp = mydata.Temperature
pressure = mydata.Vapor_Pressure
```

CORRELATION & REGRESSION

Exercise**:** The data on vapor pressure of water at various temperatures are given in Correlation.csv file.

2. Constructing Scatter plot

```
myplot.scatter(temp, pressure)
myplot.show()
```



CORRELATION & REGRESSION

Exercise**:** The data on vapor pressure of water at various temperatures are given in Correlation.csv file.

Computing correlation coefficient

mynp.corrcoef(temp, pressure)

Statistics	Value
r	0.893

MULTIPLE REGRESSION ANALYSIS

CORRELATION & REGRESSION

Regression

Correlation helps

To check whether two variables are related

If related

Identify the type & degree of relationship

CORRELATION & REGRESSION

Regression

Regression helps

- To identify the exact form of the relationship
- To model output in terms of input or process variables

Examples:

Expected (Yield) = 5 + 3 x Time - 2 x Temperature

CORRELATION & REGRESSION

Simple Linear Regression Illustration

Output variable is modeled in terms of only one variable

x	y
2	7
1	4
5	16
4	13
3	10
6	19

Regression Model

$y = 1 + 3x$

Simple Linear Regression

General Form:

$$y = a + bx + \varepsilon$$

where

a: intercept (the value of y when x is equal to 0)

b: slope (indicates the amount of change in y with every unit change in x)

CORRELATION & REGRESSION

Simple Linear Regression: Parameter Estimation

Model: $y = a + bx + \varepsilon$

$$\hat{a} = \bar{y} - \hat{b}\bar{x}$$

$$\hat{b} = S_{xy} / S_{xx}$$

Test for Significance (Testing b = 0 or not) of relation between x & y

H0: $b = 0$

H1: $b \neq 0$

Test Statistic $t_0 = (\hat{b} - 0)/se(\hat{b})$

If p value < 0.05, then H0 is rejected & y can be modeled with x

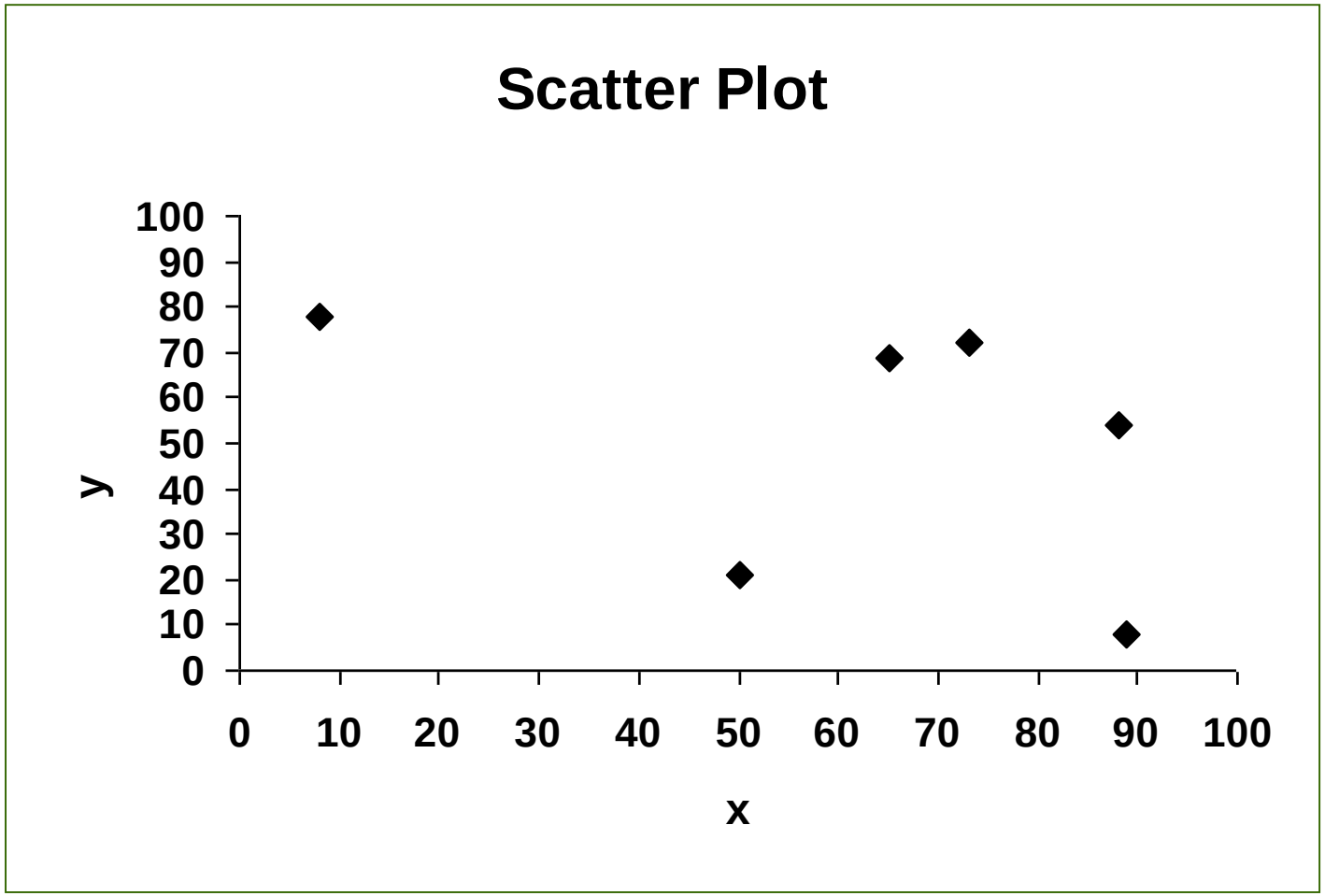
CORRELATION & REGRESSION

Regression illustration: Issues

x	y
65	69
8	78
89	8
88	21
50	24
73	72

CORRELATION & REGRESSION

Regression Model $y = 76.32 - 0.42x + \varepsilon$



CORRELATION & REGRESSION

Regression: Issues

For any set of data,

a & b can be calculated

Regression model $y = a + bx + \varepsilon$ can be build

But all the models may not be useful

CORRELATION & REGRESSION

Coefficient of Regression: Measure of degree of Relationship

Symbol : R^2

$R^2 = SS_R / Syy = b.Sxy / Syy$

$SS_R = \Sigma(y_{predicted} - \text{Mean } y)^2$

$Syy = \Sigma(y_{actual} - \text{Mean } y)^2$

R^2 : amount variation in y explained by x

Range of R^2 : 0 to 1

If $R^2 \geq 0.6$, the model is reasonably good

Coefficient of Regression: Testing the significance of Regression

Regression ANOVA

Model	SS	df	MS	F	p value
Regression	SS_R				
Residual	$Syy - SS_R$				
Total	Syy				

If p value < 0.05, then the regression model is significant

REGRESSION ANALYSIS

Multiple Linear Regression

To model output variable y in terms of two or more variables.

General Form:

$$y = a + b_1x_1 + b_2x_2 + - - - + b_kx_k + \varepsilon$$

Two variable case:

$$y = a + b_1x_1 + b_2x_2 + \varepsilon$$

Where

a: intercept (the predicted value of y when all x's are zero)

b_j: slope (the amount change in y for unit change in x_j keeping all other x's constant, j = 1,2,---,k)

REGRESSION ANALYSIS

Exercise : The effect of temperature and reaction time affects the % yield. The data collected in given in the Mult-Reg_Yield file. Develop a model for % yield in terms of temperature and time?

Step 1: Read data

```
import pandas as mypd
from scipy import stats
import matplotlib.pyplot as myplot
from pandas.tools.plotting import scatter_matrix
from statsmodels.formula.api import ols
from statsmodels.stats.anova import anova_lm

mydata = mypd.read_csv("E:/ISI/PM-01/Data/Mult_reg_Yield.csv")
time = mydata.Time
temp = mydata.Temperature
output = mydata.Yield
```

REGRESSION ANALYSIS

Exercise : The effect of temperature and reaction time affects the % yield. The data collected in given in the Mult-Reg_Yield file. Develop a model for % yield in terms of temperature and time?

Step 1: Correlation Analysis

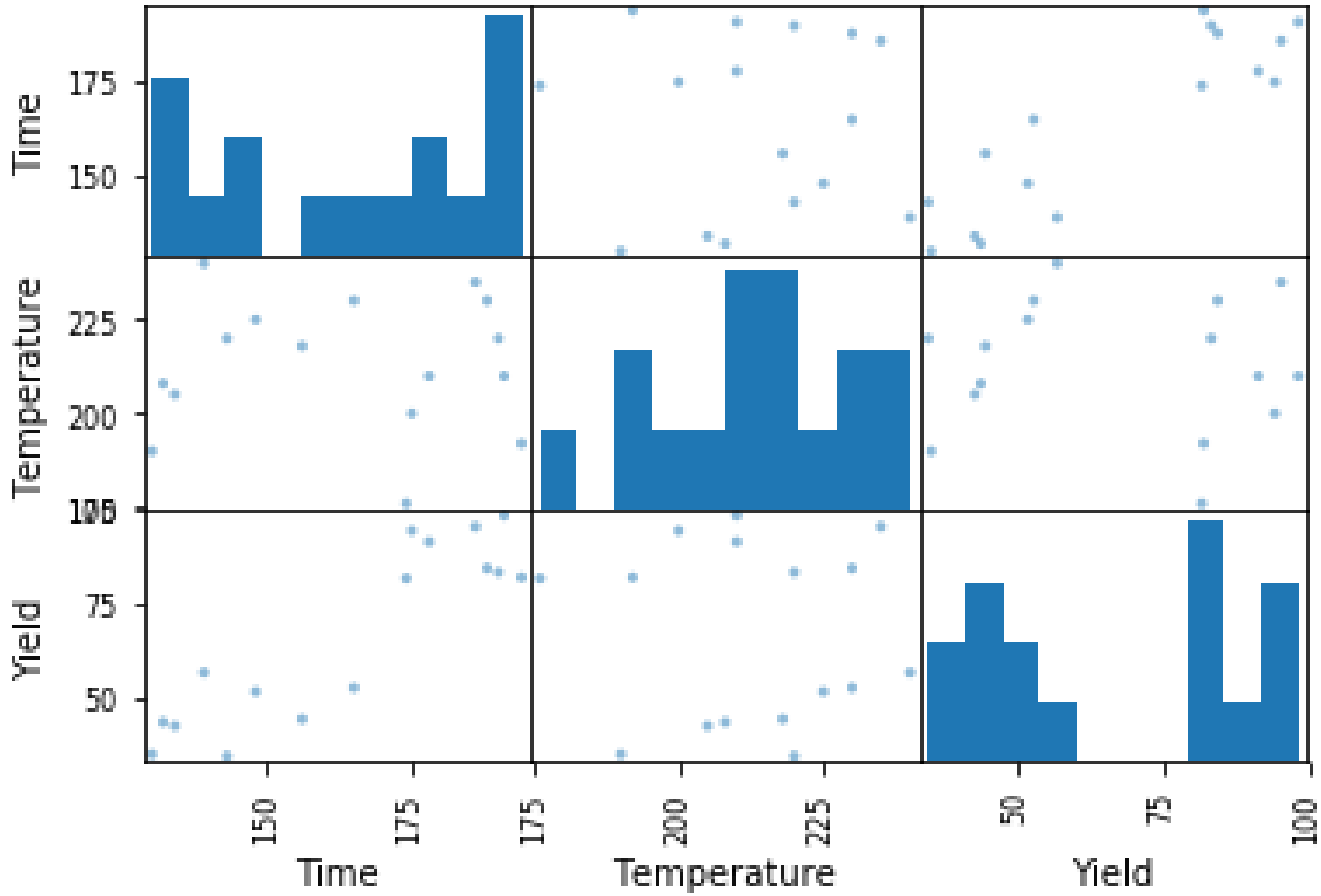
```
scatter_matrix(mydata)
myplot.show()
```

Correlation between xs & y should be high

Correlation between xs should be low

REGRESSION ANALYSIS

Exercise : The effect of temperature and reaction time affects the % yield. The data collected in given in the Mult-Reg_Yield file. Develop a model for % yield in terms of temperature and time?



REGRESSION ANALYSIS

Step 2: Regression Output

```
mymodel = ols("output ~ time + temp", mydata).fit()
mymodel.summary()
```

Statistics	Value	Criteria
R-squared:	0.806	≥ 0.6
Adj. R-squared:	0.777	≥ 0.6
F-statistic:	27.07	
Prob (F-statistic):	2.32e-05	< 0.05
Log-Likelihood:	-59.703	
AIC:	125.4	
BIC:	127.7	

REGRESSION ANALYSIS

Step 2: Regression Output

```
anova_table = anova_lm(mymodel)
anova_table
```

	df	SS	MS	F	p-value
Time	1	6777.81	6777.81	53.98722	0.000006
Temp	1	19.25253	19.25253	0.153352	0.701696
Residual	13	1632.081	125.5447		

Criteria: p value < 0.05

REGRESSION ANALYSIS

Step 2: Regression Output

Regression ANOVA

Model	SS	df	MS	F	p value
Regression	6797.063	2	3398.531	27.07	0.0000
Residual	1632.08138	13	125.5447		
Total	8429.14438	15			

Criteria: P value < 0.05

REGRESSION ANALYSIS

Step 2: Regression Output – Identify the model

	Coefficients	Std error	t	p-value	[0.025	0.975]
Intercept	-67.8844	40.587	-1.673	0.118	-155.57	19.797
Time	0.9061	0.123	7.344	0.000	0.64	1.173
Temp	-0.0642	0.164	-0.392	0.702	-0.418	0.29

Interpretation: Only time is related to yield or output as p value < 0.05

REGRESSION ANALYSIS

Step 2: Regression Output – Identify the model

	Coefficients	Std error	t	p-value	[0.025	0.975]
Intercept	- 81.6205	19.791	-4.124	0.001	-124.067	-39.174
Time	0.9065	0.120	7.580	0.000	0.650	1.163

Model Yield= 0.9065 x Time - 81.621

Statistics	Value	Criteria
R-squared:	0.804	≥ 0.6
Adj. R-squared:	0.79	≥ 0.6
F-statistic:	57.46	
Prob (F-statistic):	2.55e-06	< 0.05

REGRESSION ANALYSIS

Step 3: Residual Analysis

```
pred = mymodel.predict()
pred = mypd.DataFrame(pred, columns=["Predicted"])

res = mymodel.resid
res = mypd.DataFrame(res, columns= ['Residuals'])

myresult = mydata.join(pred)
myresult = myresult.join(res)
myresult
```

REGRESSION ANALYSIS

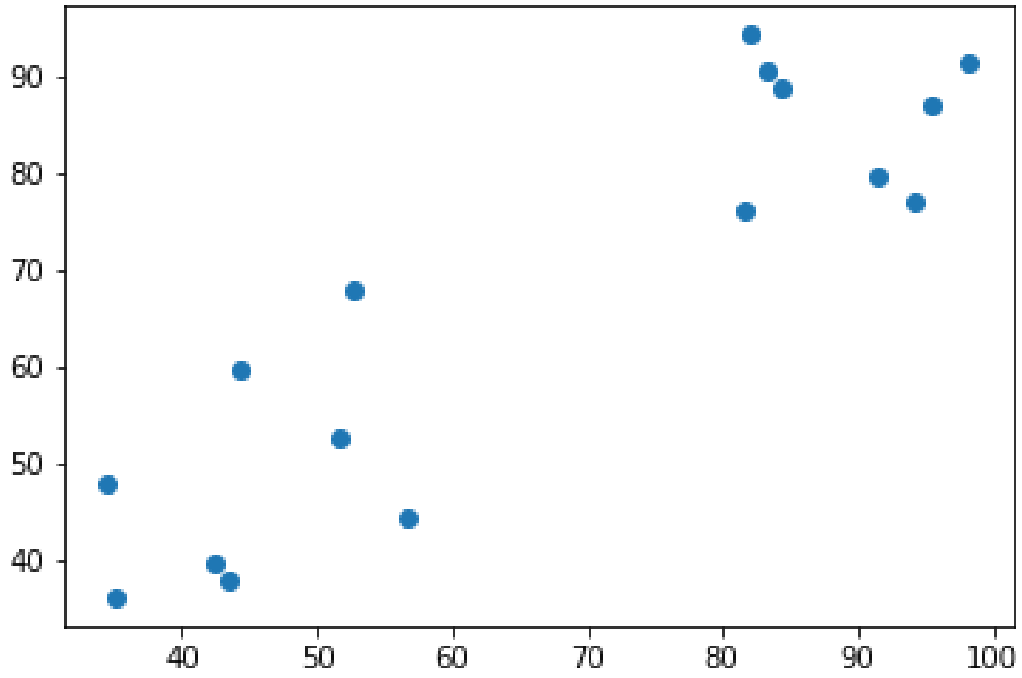
Step 3: Residual Analysis

SL No	Time	Temperature	Yield	Predicted	Residuals
0	130	190	35	36.2185	-1.2185
1	174	176	81.7	76.1024	5.5976
2	134	205	42.5	39.8443	2.6557
3	191	210	98.3	91.5122	6.7878
4	165	230	52.7	67.9444	-15.2444
5	194	192	82	94.2315	-12.2315
6	143	220	34.5	48.0024	-13.5024
7	186	235	95.4	86.9799	8.4201
8	139	240	56.7	44.3766	12.3234
9	188	230	84.4	88.7928	-4.3928
10	175	200	94.3	77.0089	17.2911
11	156	218	44.3	59.7863	-15.4863
12	190	220	83.3	90.6057	-7.3057
13	178	210	91.4	79.7283	11.6717
14	132	208	43.5	38.0314	5.4686
15	148	225	51.7	52.5346	-0.8346

REGRESSION ANALYSIS

Step 3: Residual Analysis – Actual Vs Fitted

```
myplot.scatter(output, pred)
myplot.show()
```



Note: There need to be strong positive correlation between actual and fitted response

REGRESSION ANALYSIS

Step 3: Residual Analysis: Normality test

```
stats.mstats.normaltest(res)
```

Normality Test: Yield data	
W	p value
1.8945	0.3878

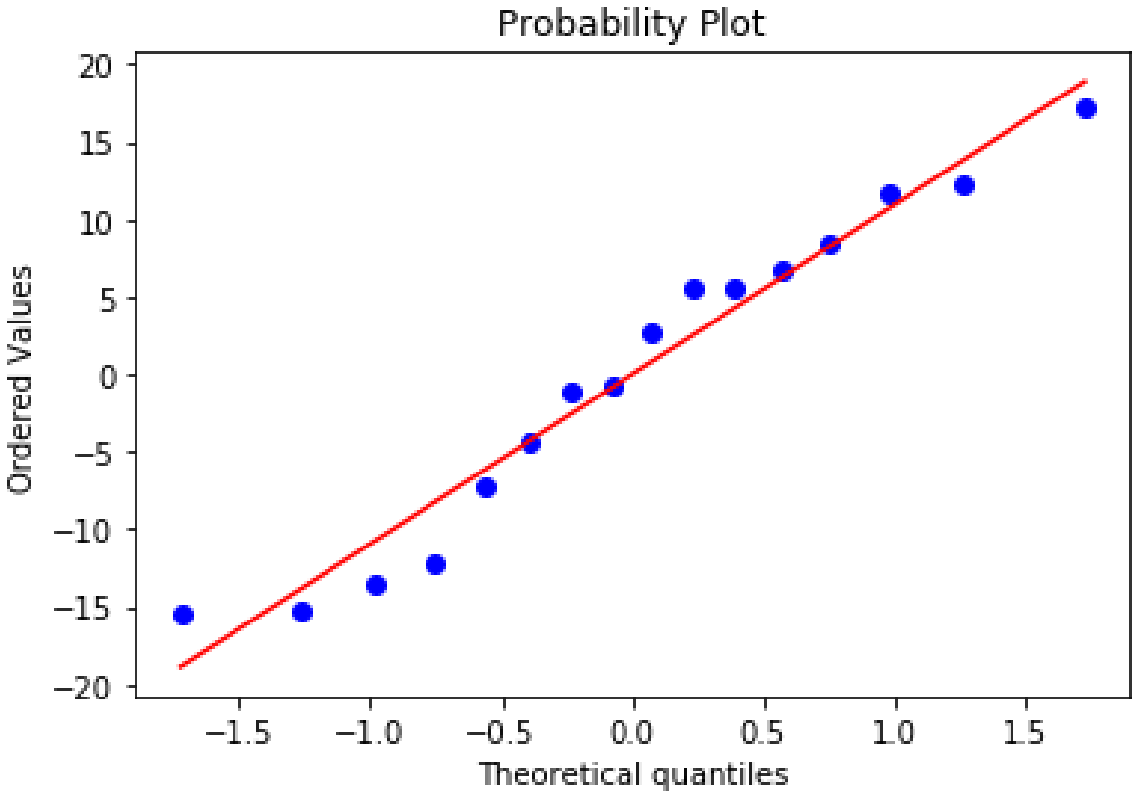
```
res_sq = res**2
mse = res_sq.mean()

import math as mymath
rmse = mymath.sqrt(mse)
rmse
```

Statistic	Value
MSE	103.21
RMSE	10.159

REGRESSION ANALYSIS

7: Residual Analysis: Normality test

```
stats.probplot(res, plot = myplot)
myplot.show
```

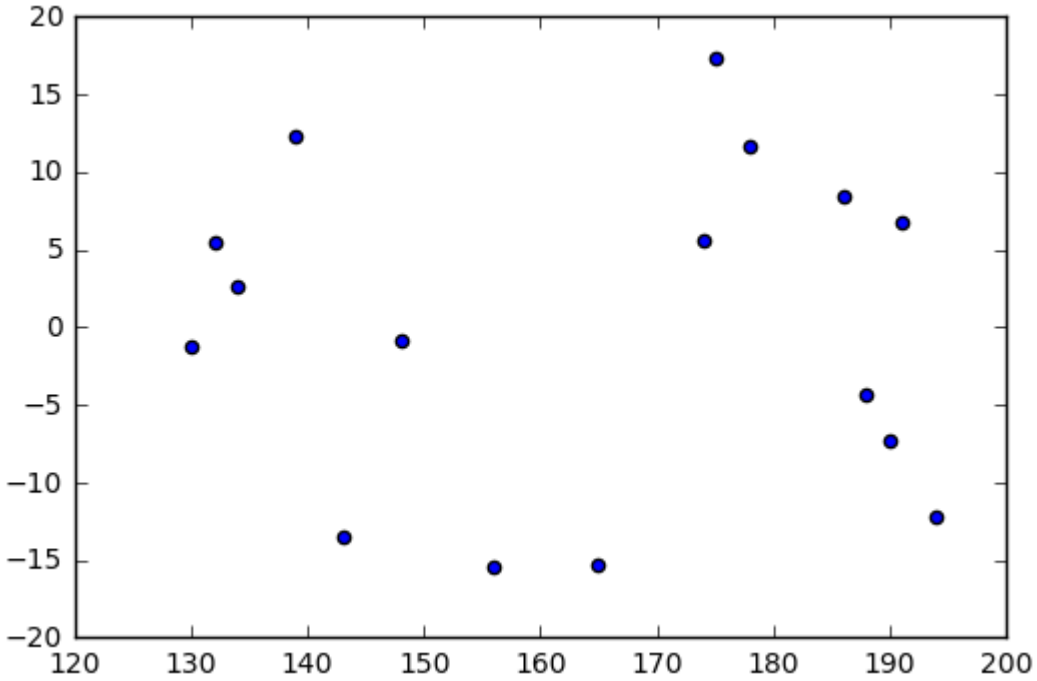


REGRESSION ANALYSIS

7: Model adequacy check

Residuals Vs Independent variables

```
myplot.scatter(time, res)
myplot.show()
```



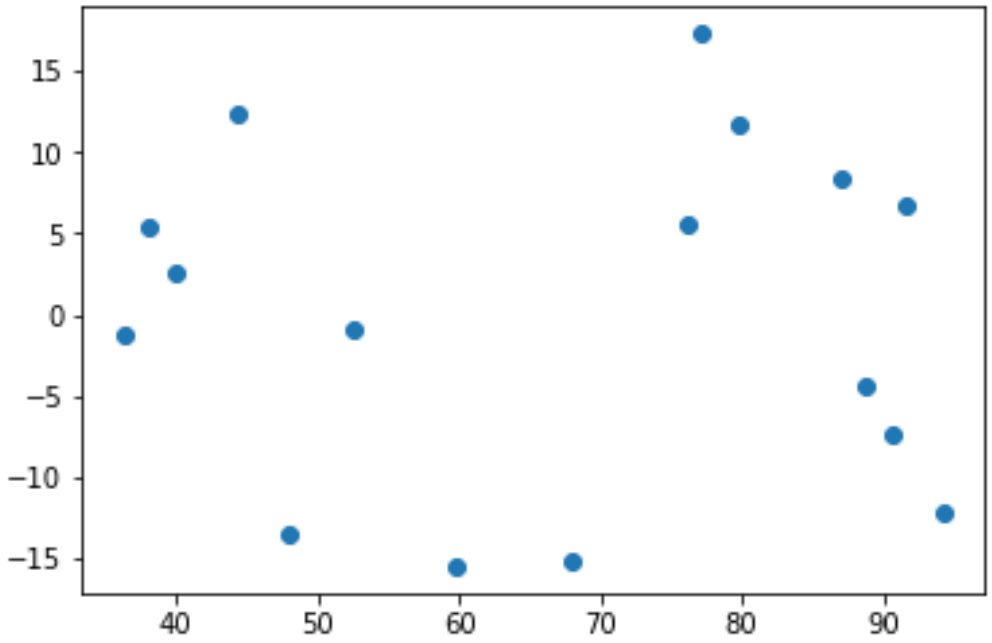
Note: There should not be any pattern or trend, the points should be distributed randomly

REGRESSION ANALYSIS

7: Model adequacy check

Residuals Vs Fitted

```
myplot.scatter(pred, res)
myplot.show()
```



Note: There should not be any pattern or trend, the points should be distributed randomly

REGRESSION ANALYSIS

Exercise : The effect of temperature and reaction time affects the % yield. The data collected in given in the Mult-Reg_Yield file. Develop a model for % yield in terms of temperature and time?

Step 1: Read packages

```
# importing the packages
import pandas as mypd
import matplotlib.pyplot as myplot
from scipy import stats
import math as mymath
from sklearn.metrics import mean_squared_error
from sklearn.model_selection import cross_val_score
from sklearn.linear_model import LinearRegression
import seaborn as mysb
```

REGRESSION ANALYSIS

Exercise : The effect of temperature and reaction time affects the % yield. The data collected in given in the Mult-Reg_Yield file. Develop a model for % yield in terms of temperature and time?

Step 2: Read data

```
#importing the dataset
mydata = mypd.read_csv("E:/hp/hp_2020/Module1/Dataset/Mult_Reg_Yield.csv")
mydata.head()


# Seperating x and y
x = mydata.iloc[:, 0:2]
y = mydata.Yield
```

REGRESSION ANALYSIS

Exercise : The effect of temperature and reaction time affects the % yield. The data collected in given in the Mult-Reg_Yield file. Develop a model for % yield in terms of temperature and time?

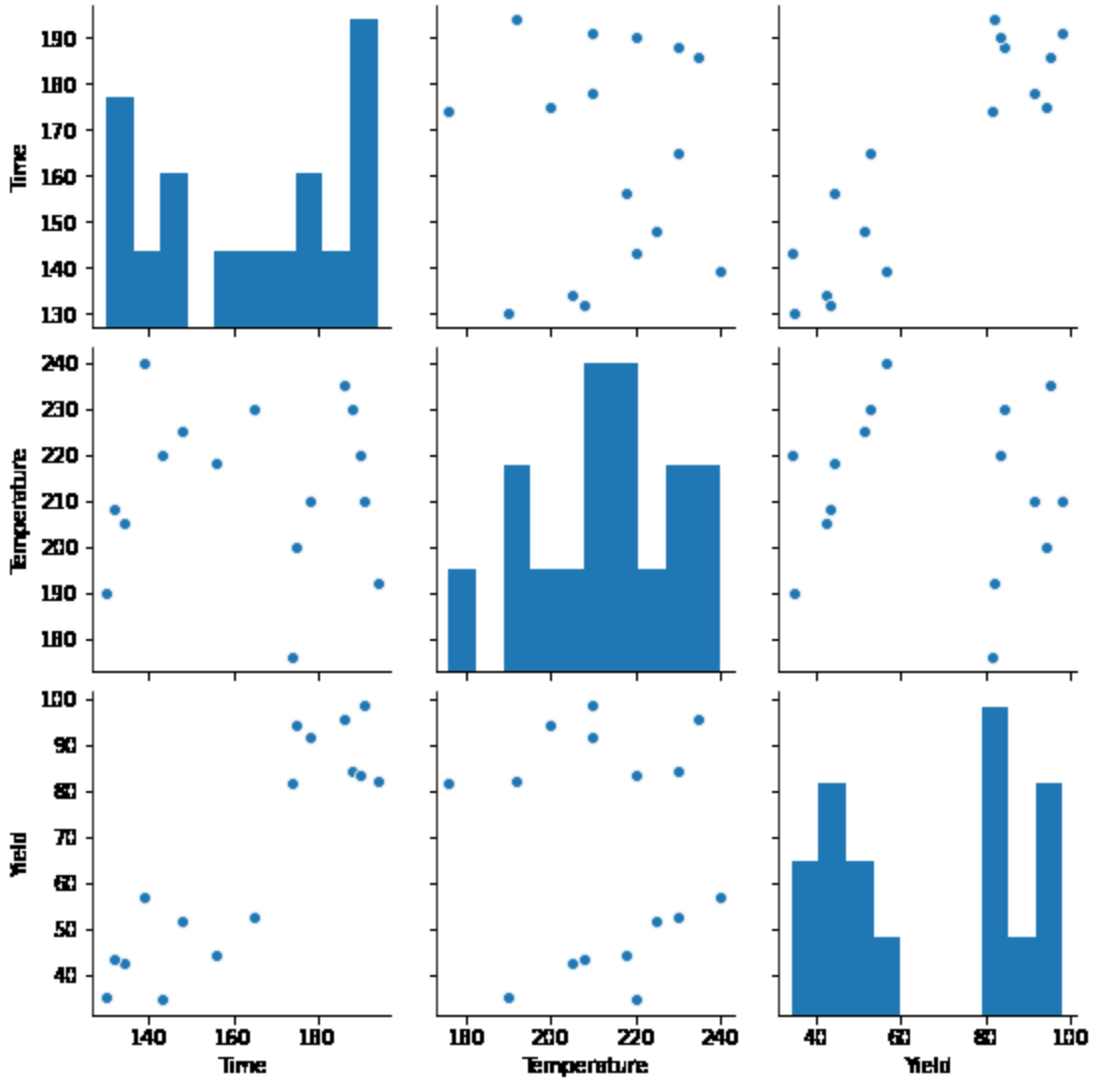
Step 3: Correlation Analysis

```
# Scatter plot
mysb.pairplot(mydata)
myplot.show()
```

Correlation between xs & y should be high

Correlation between xs should be low

REGRESSION ANALYSIS



REGRESSION ANALYSIS

Step 4: Regression Modeling

```
# fitting the model
mymodel = LinearRegression()
mymodel = mymodel.fit(x,y)
mymodel.intercept_
mymodel.coef_
```

	Coefficient
Intercept	-67.8845
Time	0.9061
Temperature	-0.0642

Yield = -67.8845 + 0.9061 x Time – 0.0642 x Temperature

REGRESSION ANALYSIS

Step 4: Regression Modeling

```
# Model accuracy
rsq = mymodel.score(x,y)
pred = mymodel.predict(x)

# Model Adequacy
mse = mean_squared_error(y, pred)
rmse = mymath.sqrt(mse)
```

Statistic	Coefficient
R^2	0.8064
MSE	102.0051
RMSE	10.0998

REGRESSION ANALYSIS

Step 4: Residual Analysis

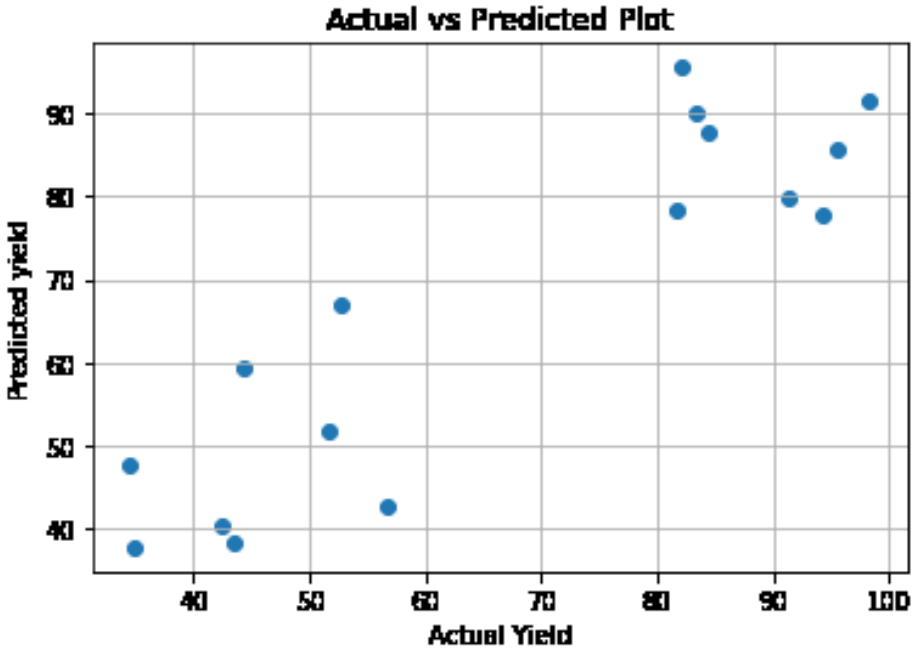
```
# Residual Analysis
res = y – pred
myresult = [y, pred, res]
myresult = mypd.DataFrame(myresult)
myresult =myresult.transpose()
```

SL No	Yield	Predicted	Residuals
0	35	37.71	-2.71
1	81.7	78.48	3.22
2	42.5	40.37	2.13
3	98.3	91.70	6.60
4	52.7	66.86	-14.16
5	82	95.57	-13.57
6	34.5	47.56	-13.06
7	95.4	85.56	9.84
8	56.7	42.66	14.04
9	84.4	87.70	-3.30
10	94.3	77.84	16.46
11	44.3	59.47	-15.17
12	83.3	90.15	-6.85
13	91.4	79.92	11.48
14	43.5	38.37	5.13
15	51.7	51.77	-0.07

REGRESSION ANALYSIS

Step 4: Residual Analysis – Actual Vs Predicted Plot

```
myplot.scatter(y, pred)
myplot.title("Actual vs Predicted Plot")
myplot.xlabel("Actual Yield")
myplot.ylabel("Predicted yield"
myplot.grid()
myplot.show()
```

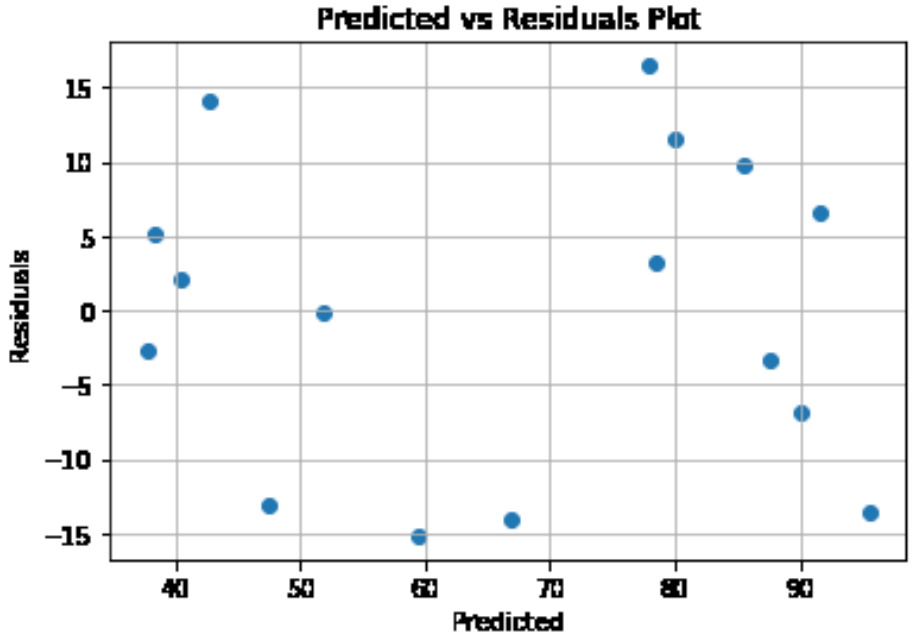


Note: There need to be strong positive correlation between actual and fitted response

REGRESSION ANALYSIS

Step 4: Residual Analysis – Predicted Vs Residuals Plot

```
myplot.scatter(pred, res)
myplot.title("Predicted vs Residuals Plot")
myplot.xlabel("Predicted")
myplot.ylabel("Residuals")
myplot.grid()
myplot.show()
```



Note: There need to be strong positive correlation between actual and fitted response

REGRESSION ANALYSIS

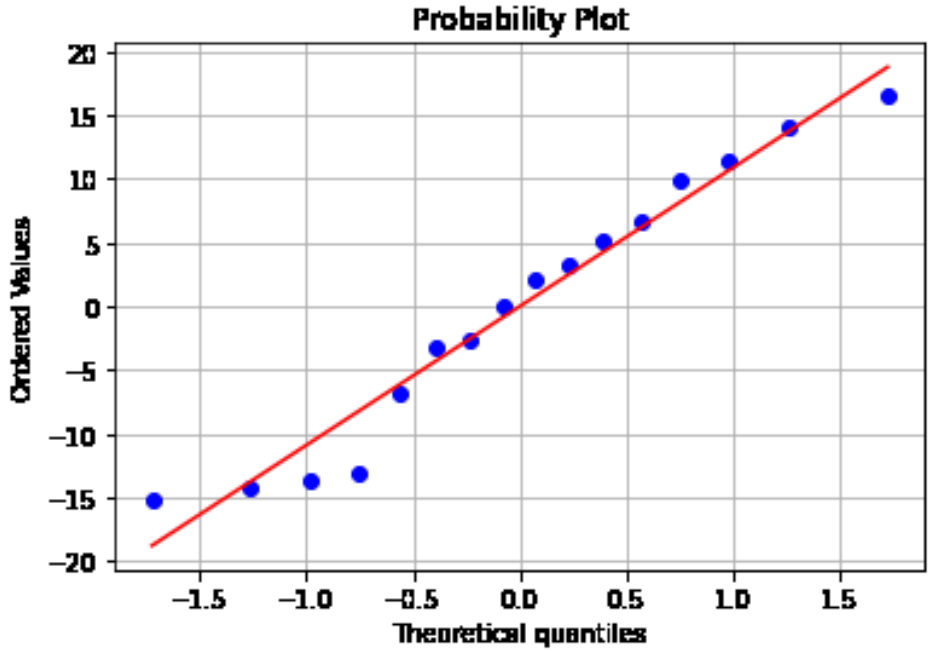
Step 4: Residual Analysis: Normality test

```
norm_test = stats.normaltest(res)
w = norm_test[0]
p_value = norm_test[1]

stats.probplot(res, plot= myplot)
myplot.grid()
myplot.show()
```

Normality Test: Yield data	
w	p value
1.9835	0.3709

REGRESSION ANALYSIS

Step 4: Residual Analysis: Normality test



REGRESSION ANALYSIS

Step 5: Cross Validation

```
# Cross Validation
myscore = cross_val_score(mymodel, x, y, scoring='neg_mean_squared_error', cv = 4)
cv_mse = -1*myscore.mean()
rmse = mymath.sqrt(cv_mse)
```

Statistic	Training	Test
MSE	102.0051	122.5726
RMSE	10.0998	11.0713

REGRESSION ANALYSIS

Regression with dummy variables

When x's are not numeric but nominal

Each nominal or categorical variable is converted into dummy variables

Dummy variables takes values 0 or 1

Number of dummy variable for one x variable is equal to number of distinct values of that variable - 1

Example: A study was conducted by an IT company to develop a model to estimate sprint productivity of agile projects in telecom vertical. The explonatory variables chosen are developer skill, review type and code reuse. Data was collected from 34 projects and is given in Agile_Productivity file.?

REGRESSION ANALYSIS

Regression with dummy variables

Variable		Dummy
Review Type	Code	$Review_{Peer}$
Fagan	1	0
Peer	2	1

Variable		Dummy	
Skill	Code	$Skill_{Experienced}$	$Skill_{Master}$
Fresher	1	0	0
Experienced	2	1	0
Master	3	0	1

REGRESSION ANALYSIS

Regression with dummy variables

Read the fie and variables

```
import pandas as mypd
from scipy import stats
import matplotlib.pyplot as myplot
from statsmodels.formula.api import ols
Import seaborn as mysb

mydata = mypd.read_excel("E:/ISI/ML-02/Agile_Sprint_Productivity.xlsx")
dev_skill = mydata.Developer_Skill
rev_skill = mydata.Review_Type
reuse = mydata.iloc[:,2]
sp = mydata.Sprint_Productivty
```

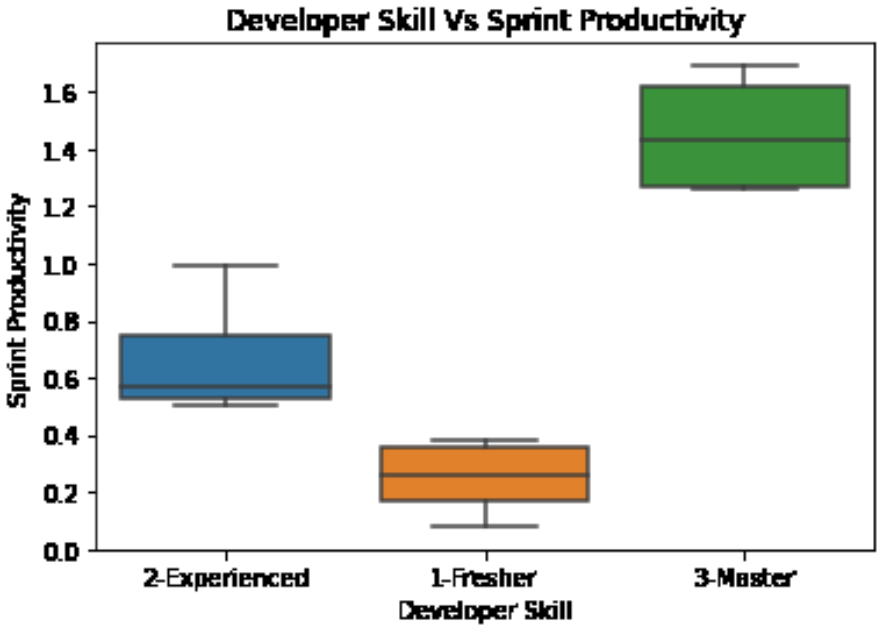
REGRESSION ANALYSIS

Regression with dummy variables

Checking relation between x and y

Developer Skill Vs Sprint Productivity

```
mysb.boxplot(x = dev_skill, y = sp)
myplot.title("Developer Skill Vs Sprint Productivity")
myplot.xlabel("Developer Skill")
myplot.ylabel("Sprint Productivity")
myplot.show()
```



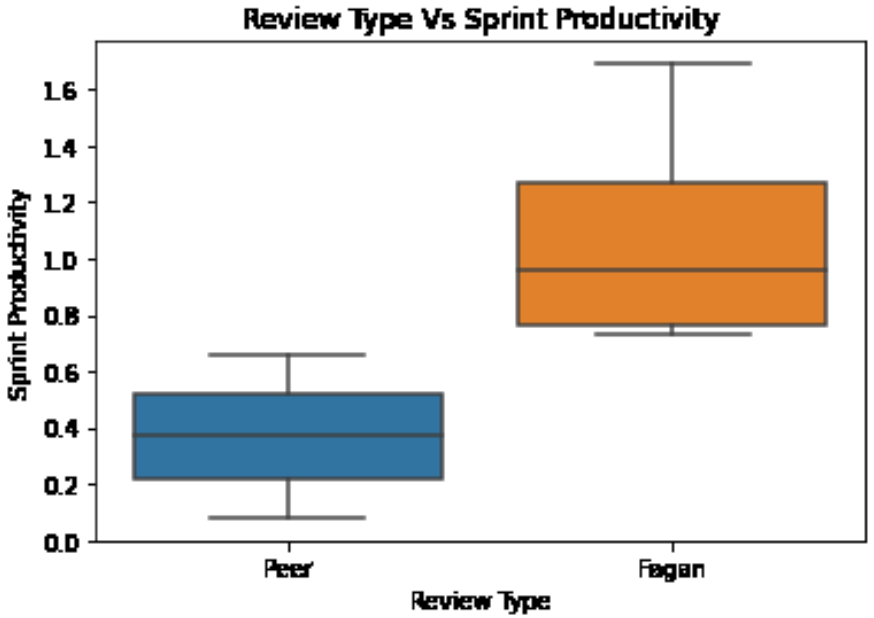
REGRESSION ANALYSIS

Regression with dummy variables

Checking relation between x and y

Review Type Vs Sprint Productivity

```
mysb.boxplot(x = rev_type, y = sp)
myplot.title("Review Type Vs Sprint Productivity")
myplot.xlabel("Review Type")
myplot.ylabel("Sprint Productivity")
myplot.show()
```



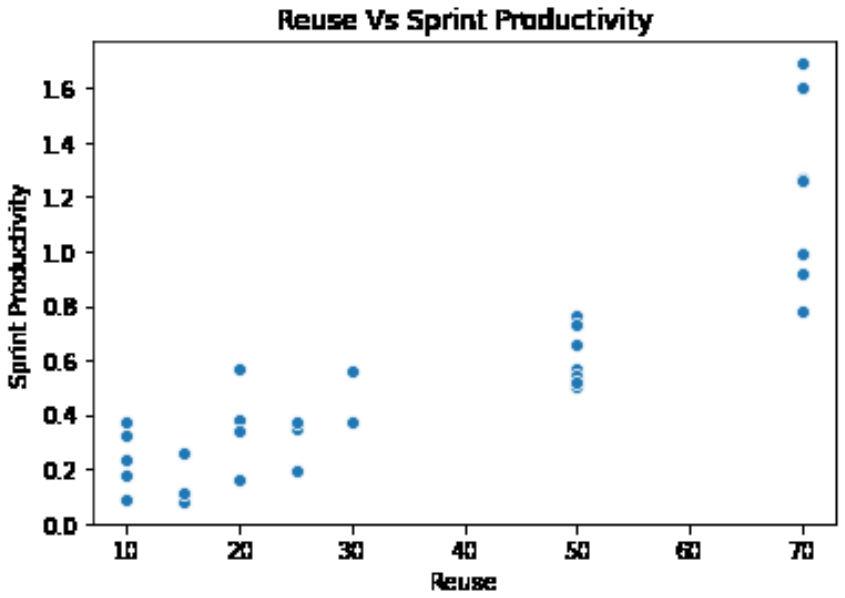
REGRESSION ANALYSIS

Regression with dummy variables

Checking relation between x and y

Reuse Vs Sprint Productivity

```
mysb.scatterplot(x = reuse, y = sp)
myplot.title("Reuse Vs Sprint Productivity")
myplot.xlabel("Reuse")
myplot.ylabel("Sprint Productivity")
myplot.show()
```



REGRESSION ANALYSIS

Regression with dummy variables – Output

```
mymodel  = ols('sp ~ C(dev_skill) + C(rev_type) + reuse', mydata).fit()
mymodel.summary()
```

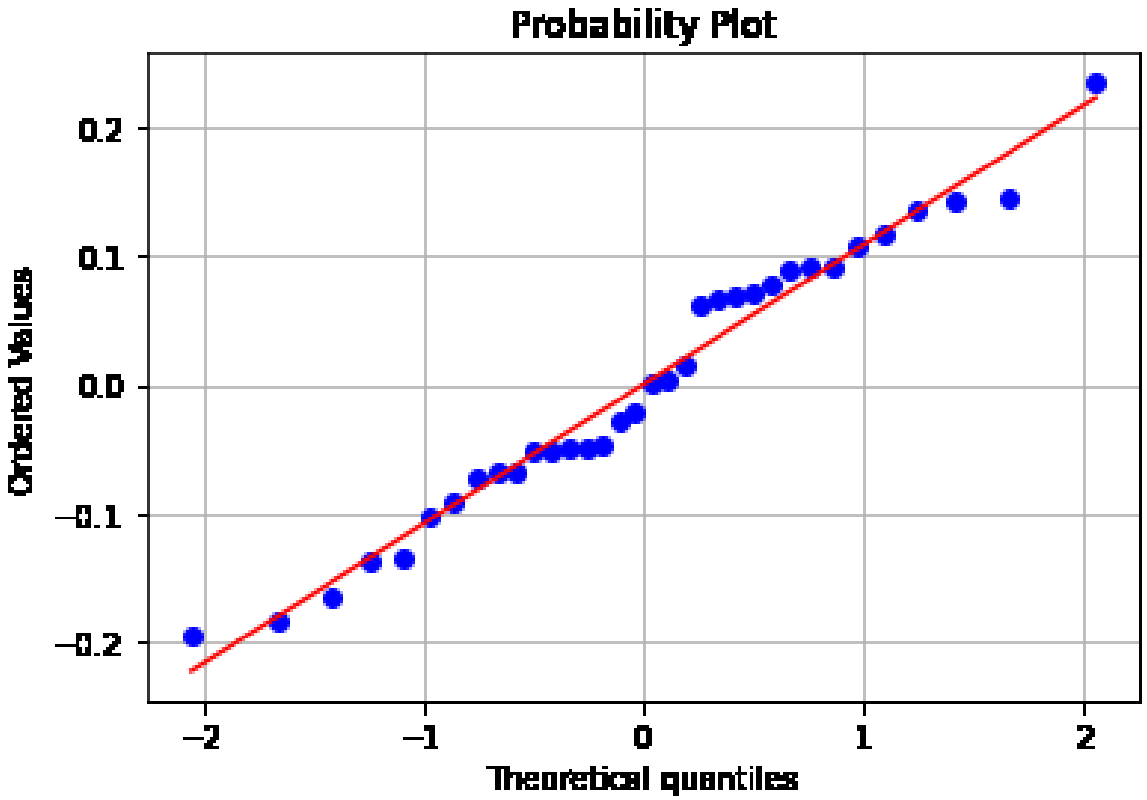
R^2	0.931
Adjusted R^2	0.921
F Statistics	97.69
p value	0.0000

	Coefficient	Std Error	t	p-value	[0.025	0.975]
Intercept	0.4055	0.103	3.939	0.000	0.195	0.616
C(dev_skill)[T.2-Experienced]	0.1989	0.080	2.471	0.020	0.034	0.363
C(dev_skill)[T.3-Master]	0.7999	0.125	6.424	0.000	0.545	1.055
C(rev_type)[T.Peer]	-0.2140	0.070	-3.038	0.005	-0.358	-0.070
reuse	0.0036	0.002	1.494	0.146	-0.001	0.008

REGRESSION ANALYSIS

Regression with dummy variables – Normality test of Residuals

```
predy = mymodel.predict()
res = sp - predy
stats.probplot(res, plot= myplot)
myplot.grid()
myplot.show()
```



REGRESSION ANALYSIS

Regression with dummy variables – Normality test of Residuals

```
mytest = stats.normaltest(res)
test_stat = round(mytest[0],4)
test_stat
p_value = round(mytest[1],4)
p_value
```

Statistics	Value
w	0.9569
p-value	0.6197

BINARY LOGISTIC REGRESSION

BINARY LOGISTIC REGRESSION

Used to develop models when the output or response variable y is binary

The output variable will be binary, coded as either success or failure

Models probability of success p which lies between 0 and 1

Linear model is not appropriate

$$p = \frac{e^{a+b_1x_1+b_2x_2+---+b_kx_k}}{1+e^{a+b_1x_1+b_2x_2+---+b_kx_k}}$$

p: probability of success
x_i's : independent variables
a, b_1, b_2, ---: coefficients to be estimated

If estimate of $p \geq 0.5$, then classified as success, otherwise as failure

BINARY LOGISTIC REGRESSION

Example: Develop a model to predict the non payment of overdrafts by customers of a multinational banking institution. The data collected is given in Logistic_Reg.csv file. The factors and response considered are given below.

SL No	Factor
1	Individual expected level of activity score
2	Transaction speed score
3	Peer comparison score in terms of transaction volume

Response	Values
Outcome	0: Not Paid and 1: Paid

BINARY LOGISTIC REGRESSION

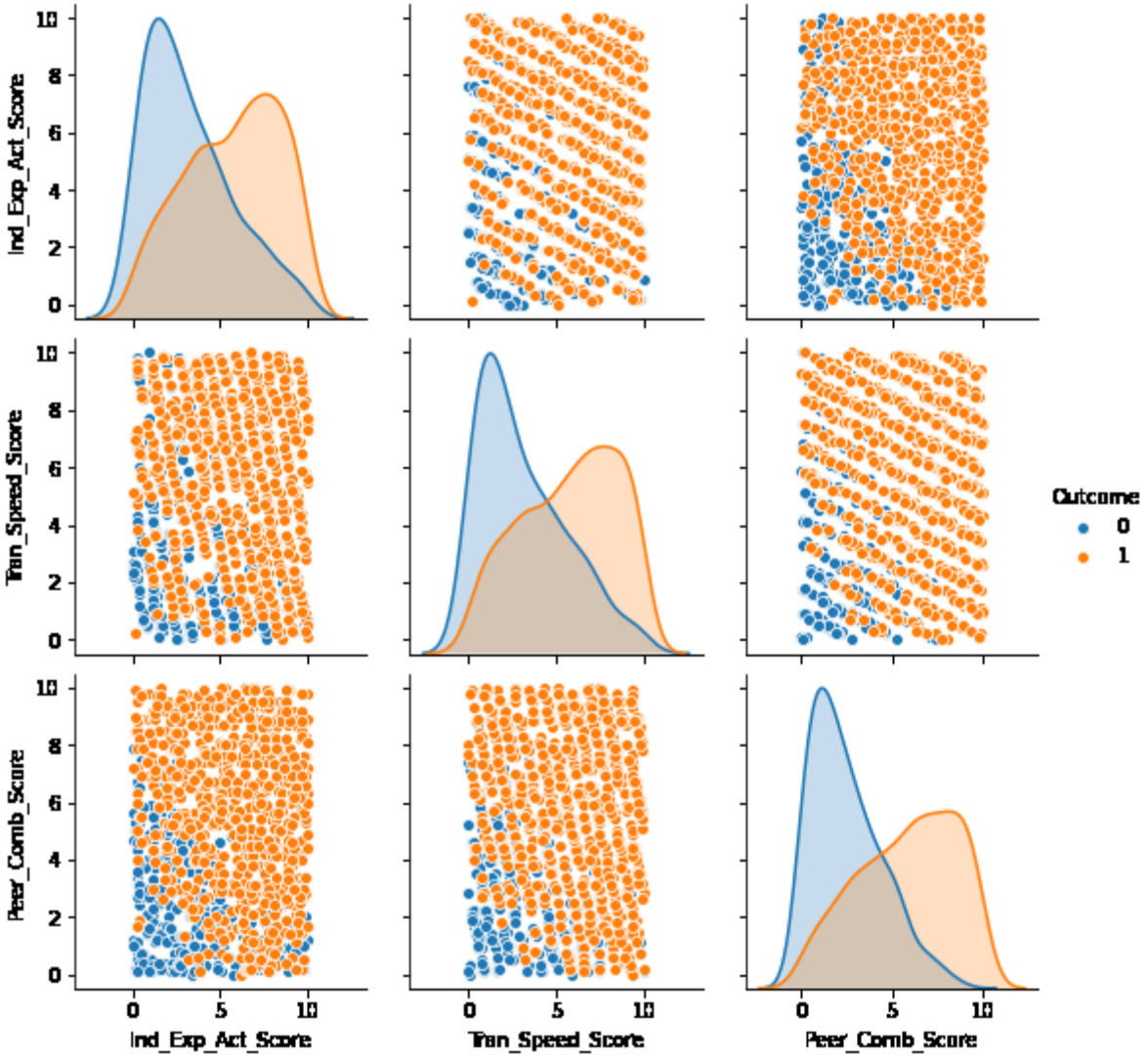
Example: Develop a model to predict the non payment of overdrafts by customers of a multinational banking institution. The data collected is given in Logistic_Reg.csv file.

Reading the file and variables

```
import pandas as mypd
from sklearn.linear_model import LogisticRegression
import statsmodels.api as mysm
Import seabron as mysm

mydata = mypd.read_csv("E:/ISI/PM-01/Data/Logistic_Reg.csv")

mysb.pairplot(mydata, hue= "Outcome")
myplot.show()
```

BINARY LOGISTIC REGRESSION

Example



BINARY LOGISTIC REGRESSION

Example: Develop a model to predict the non payment of overdrafts by customers of a multinational banking institution. The data collected is given in Logistic_Reg.csv file.

Reading the file and variables

```
x  = mydata.iloc[:, 0:3]
y = mydata.Outcome
x["Intercept"]=1
```

BINARY LOGISTIC REGRESSION

Example: Develop a model to predict the non payment of overdrafts by customers of a multinational banking institution. The data collected is given in Logistic_Reg.csv file.

Developing the model

```
mymodel = mysm.Logit(y,x)
myresult = mymodel.fit()
myresult.summary()
```

Logistic Regression Results

Statistic	Value	Statistic	Value
Response	Outcome	No. of values	980
Model	Logit	Df Residuals	976
Method	MLE	Df Model	3

BINARY LOGISTIC REGRESSION

Example: Develop a model to predict the non payment of overdrafts by customers of a multinational banking institution. The data collected is given in Logistic_Reg.csv file.

Logistic Regression Results

Statistic	Value	Criteria
Pseudo R^2	0.893	≥ 0.6
Log-Likelihood:	-63.416	
LL-Null:	-577.85	
LLR p-value:	0.00	< 0.05

BINARY LOGISTIC REGRESSION

Example: Develop a model to predict the non payment of overdrafts by customers of a multinational banking institution. The data collected is given in Logistic_Reg.csv file.

Logistic Regression Results

	Code	Coef	Std err	z	p-value	95 % CI
Ind_Exp_Act_Score	x_1	2.7957	0.355	7.867	0.00	2.099 3.492
Tran_Speed_Score	x_2	2.7532	0.343	8.032	0.00	2.081 3.425
Peer_Comb_Score	x_3	3.5153	0.434	8.095	0.00	2.664 4.366
Intercept		-35.5062	4.406	-8.058	0.00	-71.012

Criteria: p-value < 0.05

BINARY LOGISTIC REGRESSION

Example: Develop a model to predict the non payment of overdrafts by customers of a multinational banking institution. The data collected is given in Logistic_Reg.csv file.

Logistic Regression Results

	Code	Coef	Std err	z	p-value	95 % CI
Ind_Exp_Act_Score	x_1	2.7957	0.355	7.867	0.00	2.099 3.492
Tran_Speed_Score	x_2	2.7532	0.343	8.032	0.00	2.081 3.425
Peer_Comb_Score	x_3	3.5153	0.434	8.095	0.00	2.664 4.366
Intercept		-35.5062	4.406	-8.058	0.00	-71.012

The Model

$$y = \frac{e^{-35.5062+2.7957\,x_1+2.7532\,x_2+3.5153\,x_3}}{1+e^{-35.5062+2.7957\,x_1+2.7532\,x_2+3.5153\,x_3}}$$

BINARY LOGISTIC REGRESSION

Example: Develop a model to predict the non payment of overdrafts by customers of a multinational banking institution. The data collected is given in Logistic_Reg.csv file.

Exporting the Predicted values

```
pred = myresult.predict(x)
predclass = [1 if i >= 0.5 else 0 for i in pred]
myoutput = mypd.DataFrame(predclass)
myoutput.to_csv("E:\ISI\PM-01/output.csv")
```

Actual	Predicted	
	0	1
0	257	14
1	14	695

Statistics	Computation	Value
Accuracy %	(257 + 695) / (257 + 695 + 14 + 14)	97.14
Misclassification Error %	100 – Accuracy %	2.85

Accuracy of $\geq$ 80 % is good

BINARY LOGISTIC REGRESSION

Example: Develop a model to predict the non payment of overdrafts by customers of a multinational banking institution. The data collected is given in Logistic_Reg.csv file.

```
import pandas as mypd
from matplotlib import pyplot as myplot
import seaborn as mysb
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
from sklearn.model_selection import cross_val_score

mydata = mypd.read_csv("E:/Caterpillar/Data/Logistic_Reg.csv")

mysb.pairplot(mydata, hue= "Outcome")
myplot.show()
```

BINARY LOGISTIC REGRESSION

Example: Develop a model to predict the non payment of overdrafts by customers of a multinational banking institution. The data collected is given in Logistic_Reg.csv file.

```
x = mydata.iloc[:, 0:3]
y = mydata.Outcome

mymodel = LogisticRegression(C=1e8)
mymodel = mymodel.fit(x,y)
mymodel.intercept_
mymodel.coef_
```

Variable	Coefficient
Intercept	-35.4617
Ind_Exp_Act_Score	2.7922
Tran_Speed_Score	2.7498
Peer_Comb_Score	3.511

BINARY LOGISTIC REGRESSION

Example: Develop a model to predict the non payment of overdrafts by customers of a multinational banking institution. The data collected is given in Logistic_Reg.csv file.

```
accuracy = mymodel.score(x,y)
ypred = mymodel.predict(x)
ypredprob = mymodel.predict_proba(x)
myresult = [y, ypred]
myresult =mypd.DataFrame(myresult)
ypredprob = mypd.DataFrame(ypredprob)

accuracy = accuracy_score(y, ypred)
```

Statistic	Value
Accuracy	97.14%

BINARY LOGISTIC REGRESSION

Example: Develop a model to predict the non payment of overdrafts by customers of a multinational banking institution. The data collected is given in Logistic_Reg.csv file.

```
mytable  = mypd.crosstab(y, ypred)
mytable
```

Actual	Predicted	
	0	1
0	257	14
1	14	695

BINARY LOGISTIC REGRESSION

Example: Develop a model to predict the non payment of overdrafts by customers of a multinational banking institution. The data collected is given in Logistic_Reg.csv file.

```
myscore = cross_val_score(mymodel, x, y, scoring="accuracy", cv = 5)
cv_accuracy = myscore.mean()
round(cv_accuracy*100,2)
```

Statistic	Accuracy %
Training	97.10
Cross Validation	96.64

BINARY LOGISTIC REGRESSION

Addition Performance Measures

	Predicted Count	
Actual Count	Negative (0)	Positive (1)
Negative (0)	True Negative (TN)	False Positive (FP)
Positive (1)	False Negative (FN)	True Positive (TP)

$$Sensitivity\ or\ Recall = \frac{TP}{TP + FN}$$

$$Specificity = \frac{TN}{TN + FP}$$

$$Precision = \frac{TP}{TP + FP}$$

$$F - Measure = \frac{2\ x\ Precision\ x\ Recall}{Precision + Recall}$$

BINARY LOGISTIC REGRESSION

Addition Performance Measures

	Predicted Count	
Actual Count	0	1
0	257	14
1	14	695

Statistic	Value
Sensitivity	0.9803
Specificity	0.9483
Precision	0.9803
F-Measure	0.9803

Outreach Program

on

Data Processing

using

Python

Thank You