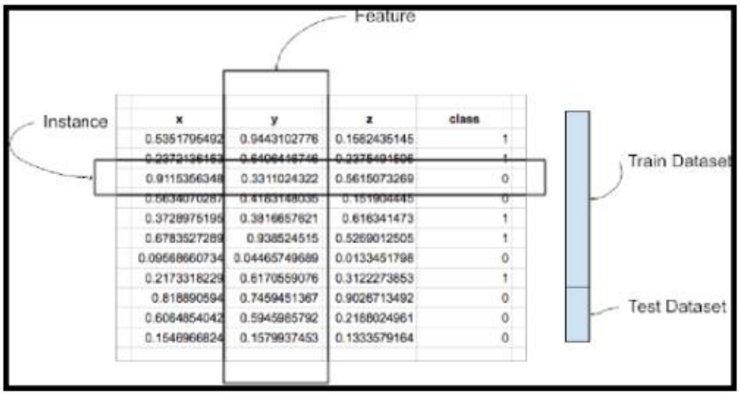
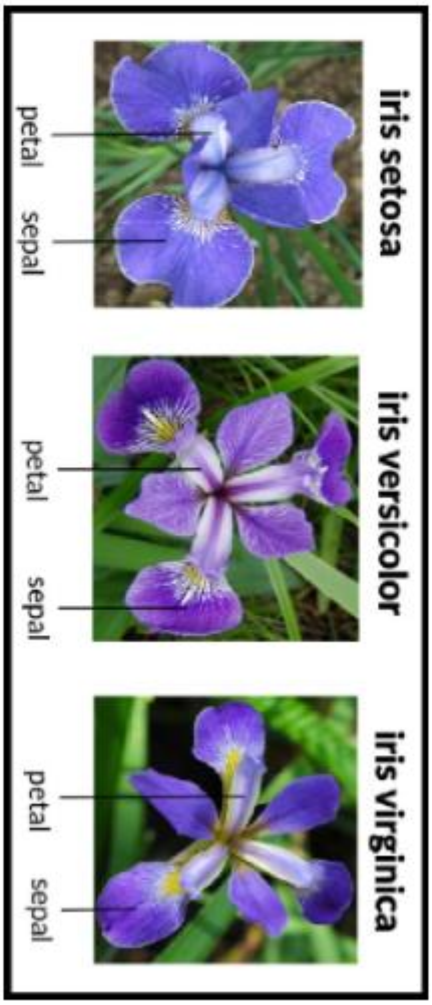
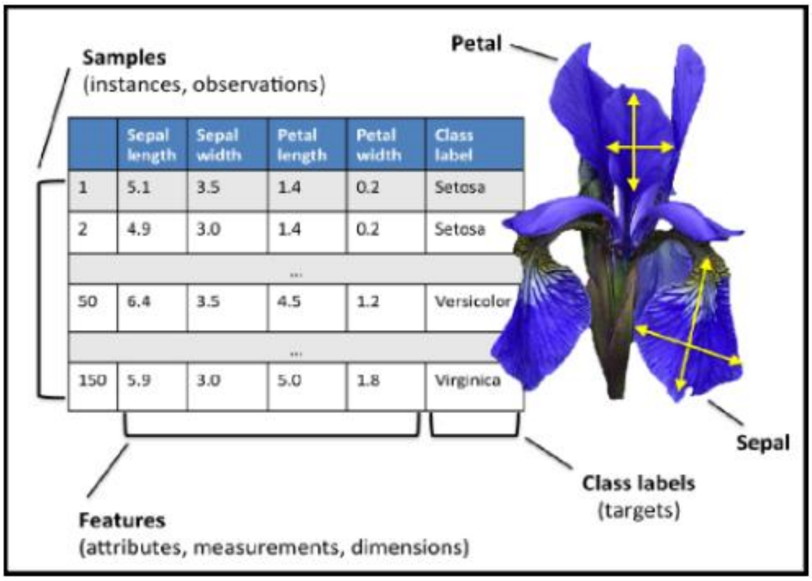




3 Different Types of Species each contain 50 Sample-

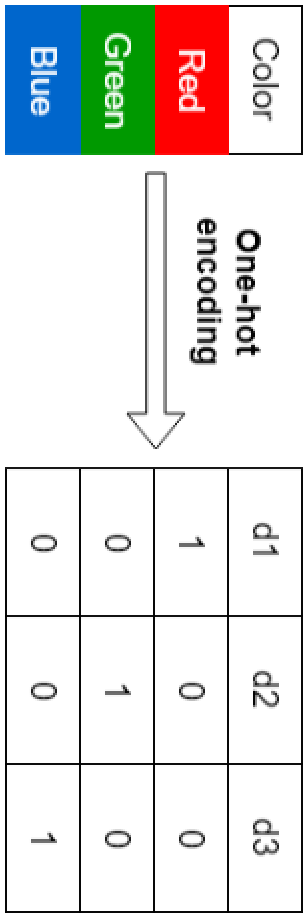
Pandas dtype	Python type	NumPy type	Usage
object	str or mixed	string_, unicode_, mixed types	Text or mixed numeric and non-numeric values
int64	int	int_, int8, int16, int32, int64, uint8, uint16, uint32, uint64	Integer numbers
float64	float	float_, float16, float32, float64	Floating point numbers
bool	bool	bool_	True/False values
datetime64	NA	datetime64[ns]	Date and time values
timedelta[ns]	NA	NA	Differences between two datetimes
category	NA	NA	Finite list of text values



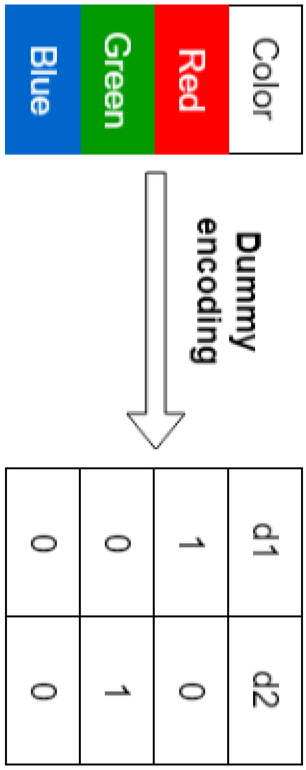
Practical no - 1

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa

Sr. No	Data Frame Function	Description
1	dataset.**head**(n=5)	**Return the first n rows.**
2	dataset.**tail**(n=5)	**Return the last n rows.**
3	**dataset.index**	The index (row labels) of the Dataset.
4	**dataset.columns**	The column labels of the Dataset.
5	**dataset.shape**	Return a tuple representing the dimensionality of the Dataset.
6	**dataset.dtypes**	Return the dtypes in the Dataset. This returns a Series with the data type of each column. The result's index is the original Dataset's columns. Columns with mixed types are stored with the object dtype.
7	**dataset.columns.values**	Return the columns values in the Dataset in array format
8	**dataset.describe(include='all')**	Generate descriptive statistics. to view some basic statistical details like percentile, mean, std etc. of a data frame or a series of numeric values. Analyzes both numeric and object series, as well as Dataset column sets of mixed data types.
9	**dataset['Column name]**	Read the Data Column wise.
10	**dataset.sort_index(axis=1, ascending=False)**	Sort object by labels (along an axis).



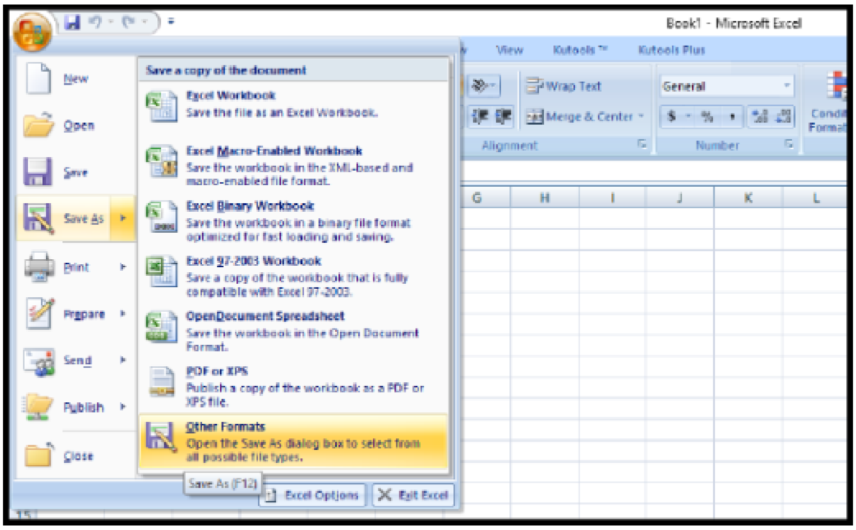
Sr. No	Data Frame Function	Description	Output
1	**dataset.iloc[3:5, 0:2]**	Slice the data	Id / SepalLengthCm 3 4 4.6 4 5 5.0
2	**dataset.iloc[[1, 2, 4], [0, 2]]**	By lists of integer position locations, similar to the NumPy/Python style:	Id / SepalWidthCm 1 2 3.0 2 3 3.2 4 5 3.6

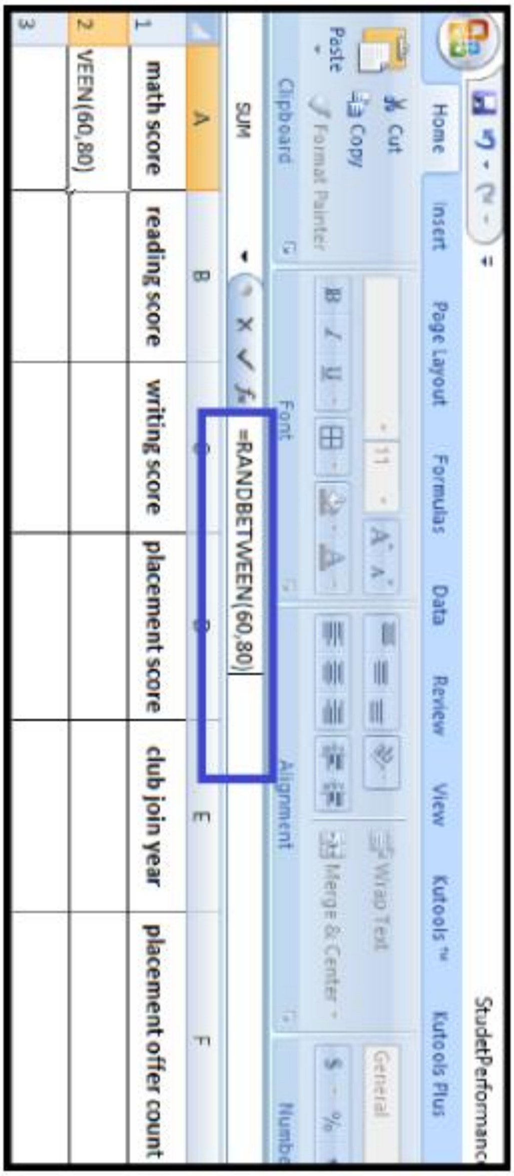


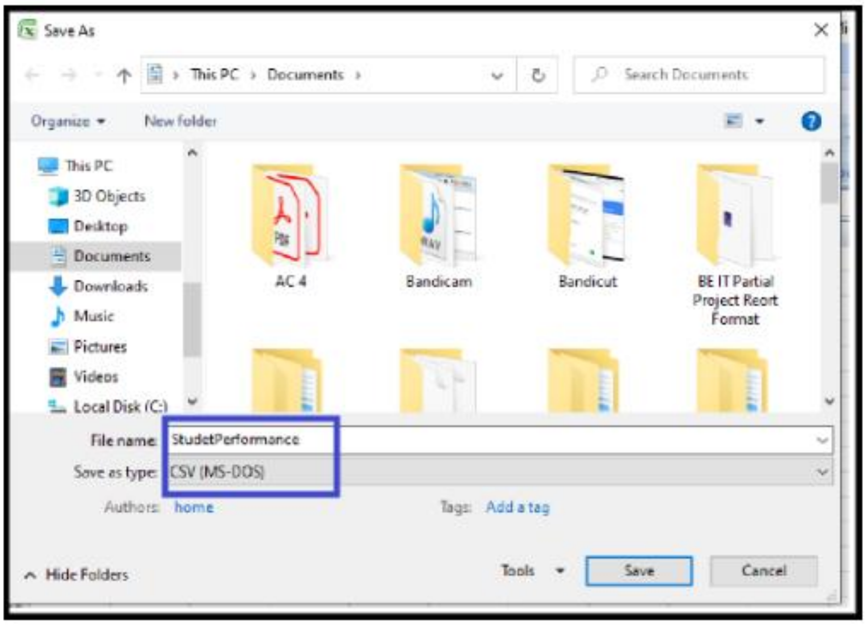
	Sepal_Length	Sepal_Width	Petal_Length	Petal_Width	Species_1	Species_2
0	5.1	3.5	1.4	0.2	0	0
1	4.9	3.0	1.4	0.2	0	0
2	4.7	3.2	1.3	0.2	0	0
3	4.6	3.1	1.5	0.2	0	0
4	5.0	3.6	1.4	0.2	0	0
...	...	...	...	...	...	...

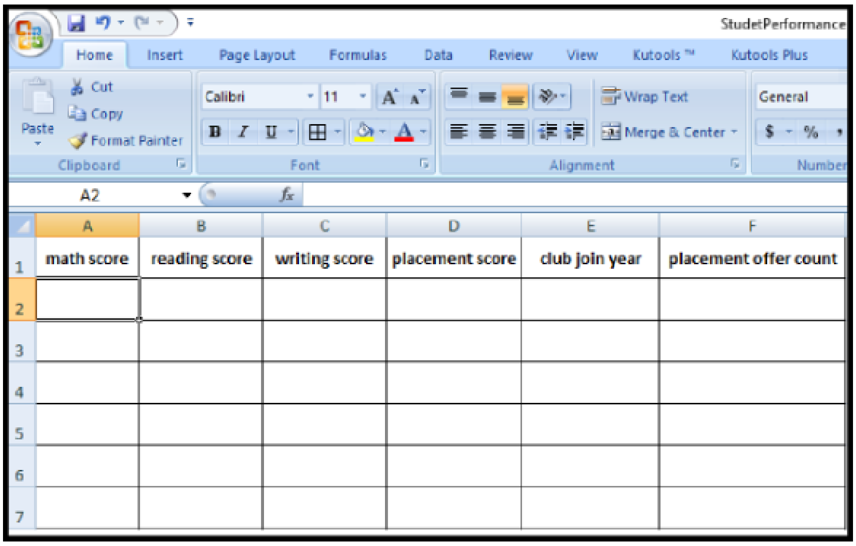
3	**dataset.iloc[1:3, :]**	For slicing rows explicitly:	Id SepalLengthCm SepalWidthCm PetalLengthCm PetalWidthCm Species 1 2 4.9 3.0 1.4 0.2 Iris-setosa 2 3 4.7 3.2 1.3 0.2 Iris-setosa
4	**dataset.iloc[:, 1:3]**	For slicing Column explicitly:	SepalLengthCm SepalWidthCm 0 5.1 3.5 1 4.9 3.0 2 4.7 3.2 3 4.6 3.1
5	**dataset.iloc[1, 1]**	For getting a value explicitly:	4.9
6	**dataset['SepalLengthCm'].iloc[5]**	Accessing Column and Rows by position	5.4
7	**cols_2_4=dataset.columns[2:4]** **dataset[cols_2_4]**	Get Column Name then get data from column	SepalWidthCm PetalLengthCm 0 3.5 1.4 1 3.0 1.4 2 3.2 1.3 3 3.1 1.5
8	**dataset[dataset.columns[2:4]].iloc[5:10]**	in one Expression answer for the above two commands	SepalWidthCm PetalLengthCm 5 3.9 1.7 6 3.4 1.4 7 3.4 1.5 8 2.9 1.4 9 3.1 1.5

Sr. No	Data Frame Function	Description	Output
1.	**df.dtypes**	To check the data type	df.dtypes sepal length (cm) float64 sepal width (cm) float64 petal length (cm) float64 petal width (cm) float64 dtype: object
2.	**df['petal length (cm)']= df['petal length (cm)'].astype("int")**	To change the data type (data type of ‘petal length (cm)'changed to int)	df.dtypes sepal length (cm) float64 sepal width (cm) float64 petal length (cm) int64 petal width (cm) float64 dtype: object



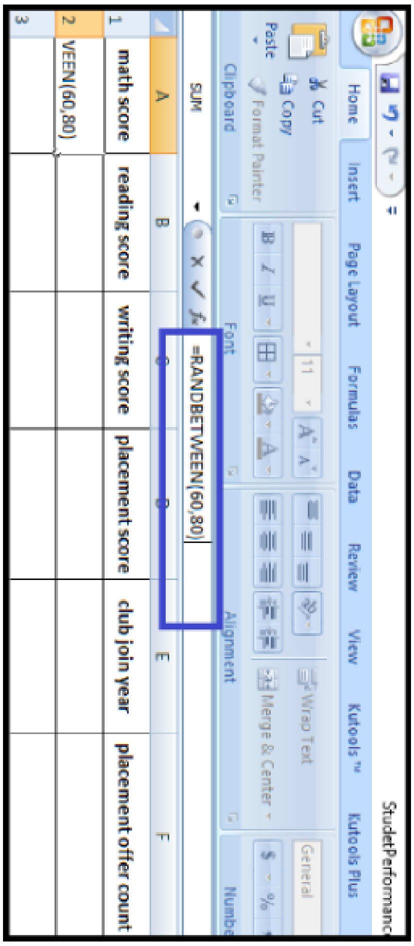




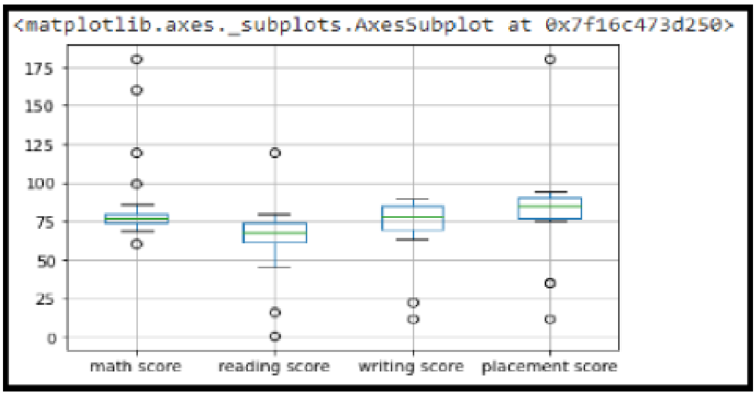


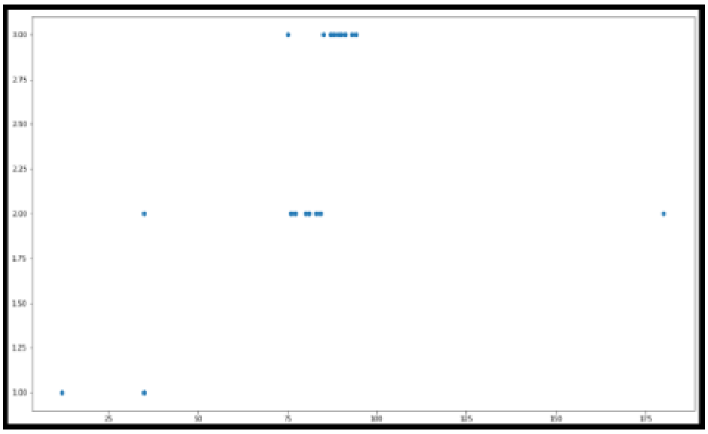

```
+------------------------+------------------------+
| with outlier           | without outlier        |
+------------------------+------------------------+
| Mean: 20.08            | Mean: 12.72            |
| Median: 14.0           | Median: 13.0           |
| Mode: 15               | Mode: 15               |
| Variance: 614.74       | Variance: 21.28        |
| Std dev: 24.79         | Std dev: 4.61          |
+------------------------+------------------------+
```

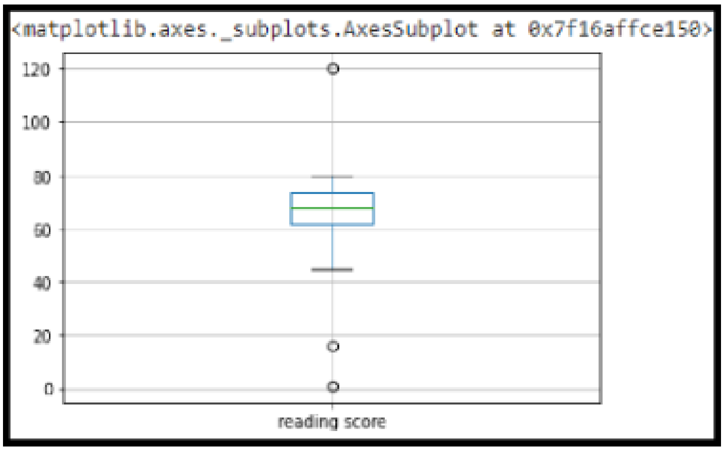
Practical No - 2



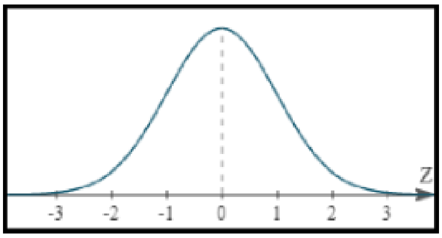
	math score	reading score	writing score	placement score	placement offer count
0	80	68	70	89	3
1	71	61	85	91	3
2	79	16	87	77	2
3	61	77	74	76	2
4	78	71	67	90	3
5	73	68	90	80	2
6	77	62	70	35	2
7	74	45	80	12	1
8	76	60	79	77	2
9	75	65	85	87	3
10	160	67	12	83	2
11	79	72	88	180	2
12	80	80	78	94	3

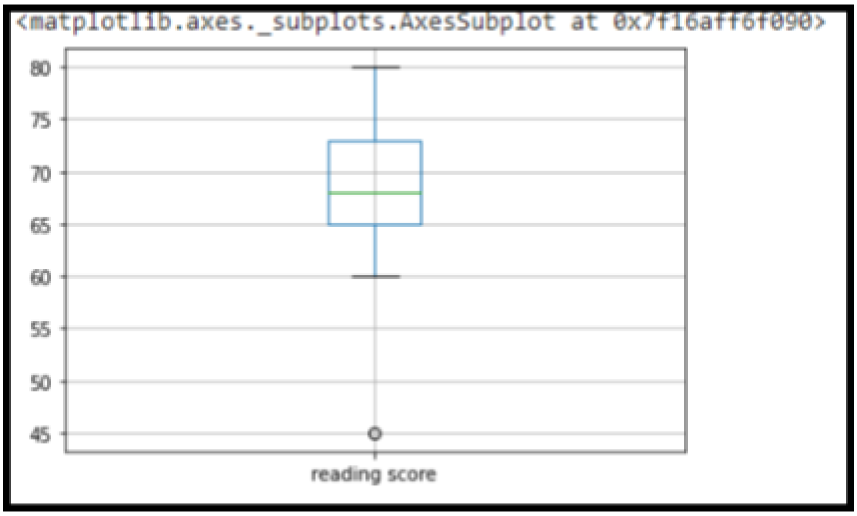


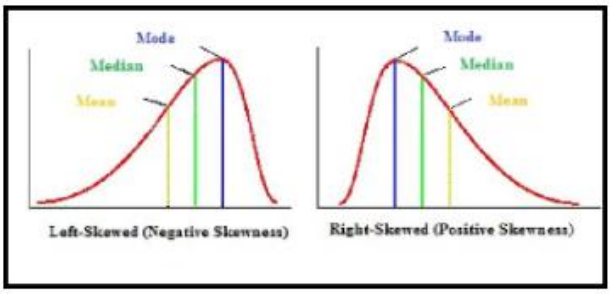


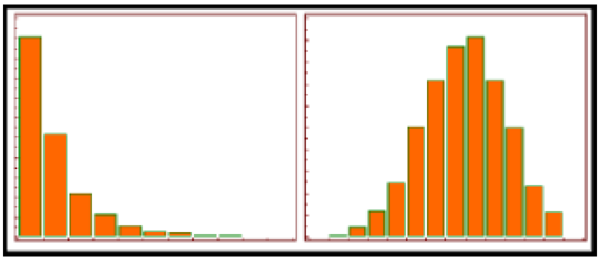


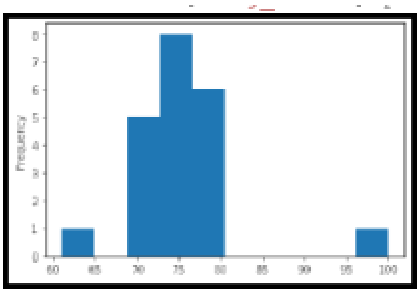

```
[0.17564553 0.5282877  0.21482799 0.92011234 0.25401045 0.44992277
 0.29319292 0.41074031 0.33237538 0.37155785 2.95895157 0.21482799
 0.17564553 0.25401045 0.37155785 0.25401045 0.05944926 0.17564553
 0.37155785 0.0972806  0.60665263 0.60800375 0.48910524 0.41074031
 0.37155785 3.74260085 0.48910524 0.5282877  1.39165302]
```

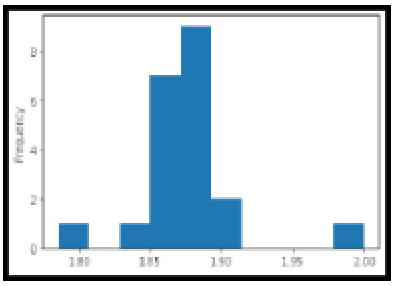












```
(600, 10)

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 600 entries, 0 to 599
Data columns (total 10 columns):
Marital_status     600 non-null object
Dependents         600 non-null int64
Is_graduate        600 non-null object
Income             600 non-null int64
Loan_amount        600 non-null int64
Term_months        600 non-null int64
Credit_score       600 non-null object
approval_status    600 non-null object
Age                600 non-null int64
Sex                600 non-null object
dtypes: int64(5), object(5)
memory usage: 47.0+ KB
None
```

```
51.0
508350.0

0    102.0
1    192.0
2    192.0
3    192.0
4    192.0
dtype: float64
```

Practical No - 3

```
0    70096.0
1    161274.0
2    125113.4
3    119853.8
4    120653.8
dtype: float64
```

```
Dependents         0.748333
Income        705541.333333
Loan_amount   323793.666667
Term_months      183.350000
Age               49.450000
dtype: float64
```

```
|        | Marital_status | Dependents | Is_graduate | Income | Loan_amount
|--------|----------------|------------|-------------|--------------|--------------
| count  | 600            | 600.000000 | 600         | 6.000000e+02 | 6.000000e+02
| unique | 2 | NaN | 2 | NaN | NaN
| top    | Yes            | NaN        | Yes         | NaN          | NaN
| freq   | 391            | NaN        | 470         | NaN          | NaN
| mean   | NaN | 0.748333 | NaN | 7.055413e+05 |
3.237937e+05 | 183.350000 | NaN | NaN | 49.450000 | NaN
| std    | NaN            | 1.026362   | NaN         | 7.114218e+05 | 7.242935e+05
| min    | NaN            | 0.000000   | NaN         | 3.000000e+04 | 1.090000e+04
| 25%    | NaN | 0.000000 | NaN | 3.849750e+05 |
6.100000e+04 | 192.000000 | NaN | NaN | 36.000000 | NaN
| 50%    | NaN | 0.000000 | NaN | 5.083500e+05 |
7.600000e+04 | 192.000000 | NaN | NaN | 51.000000 | NaN
| 75%    | NaN | 1.000000 | NaN | 7.661000e+05 |
1.302500e+05 | 192.000000 | NaN | NaN | 61.000000 | NaN
| max    | NaN | 6.000000 | NaN | 8.444900e+06 |
7.780000e+06 | 252.000000 | NaN | NaN | 76.000000 | NaN
```

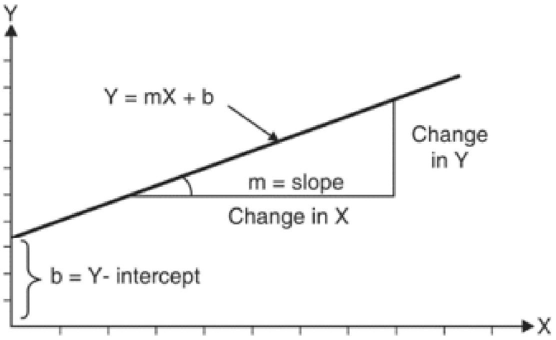
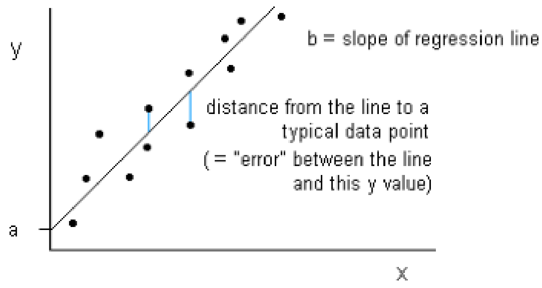
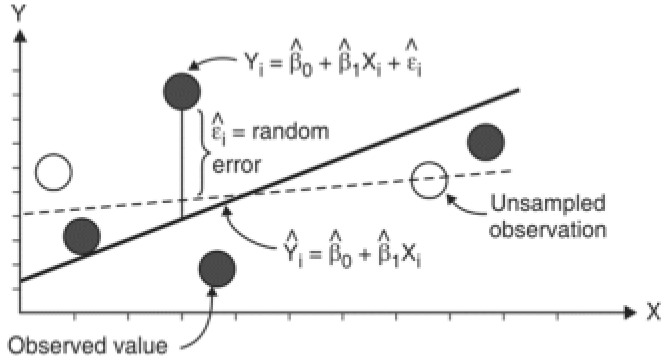
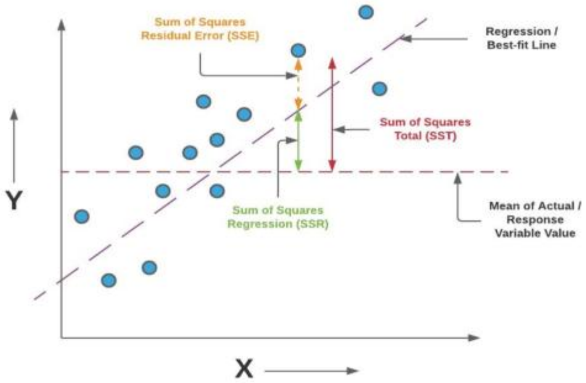


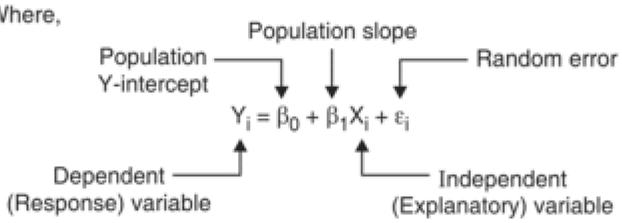
Fig. 1: geometry of linear regression

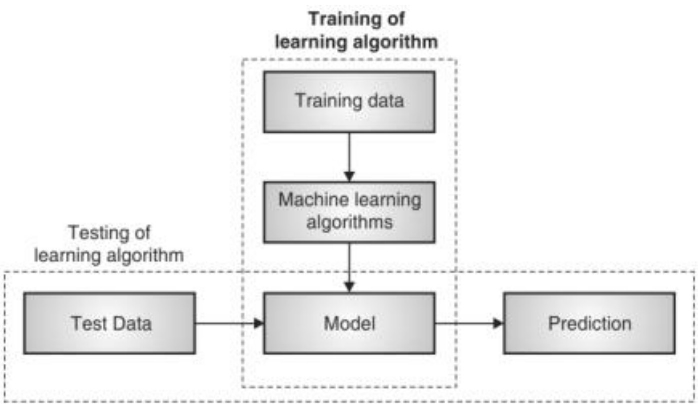


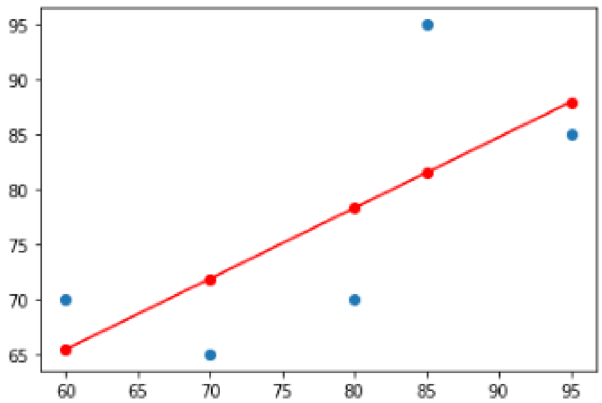




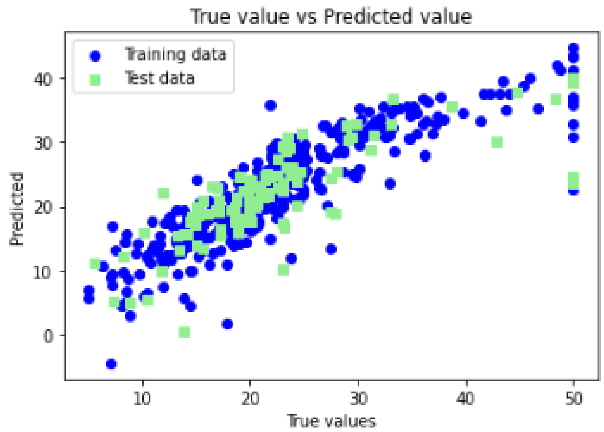
Where,



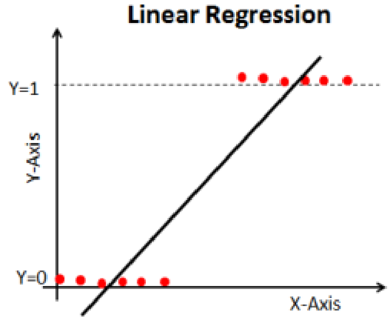


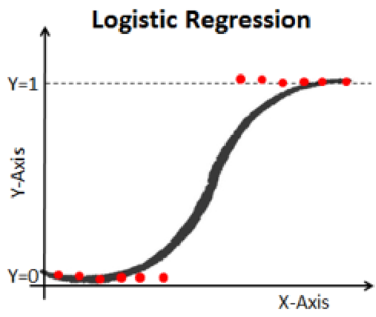


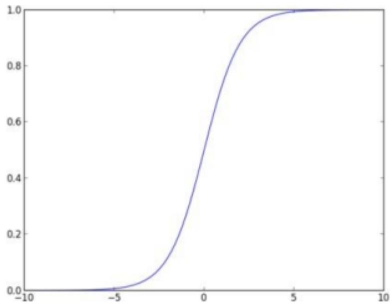
Practical No. - 4



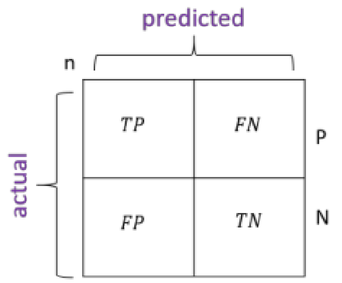
x	y	$x - x$	$y - \bar{y}$	$(x - \bar{x})^2$	$(x - \bar{x})(y - \bar{y})$
95	85	17	8	289	136
85	95	7	18	49	126
80	70	2	-7	4	-14
70	65	-8	-12	64	96
60	70	-18	-7	324	126
$\bar{x} = 78$	$\bar{y} = 77$			$\Sigma\ (x - \bar{x})^2 = 730$	$\Sigma\ (x - \bar{x})(y - \bar{y}) = 470$







Practical No . – 5



Confusion matrix

Practical No - 6

Outlook	Temp	Humidity	Windy	Play
sunny	hot	high	FALSE	no
sunny	hot	high	TRUE	no
overcast	hot	high	FALSE	yes
rainy	mild	high	FALSE	yes
rainy	cool	normal	FALSE	yes
rainy	cool	normal	TRUE	no
overcast	cool	normal	TRUE	yes
sunny	mild	high	FALSE	no
sunny	cool	normal	FALSE	yes
rainy	mild	normal	FALSE	yes
sunny	mild	normal	TRUE	yes
overcast	mild	high	TRUE	yes
overcast	hot	normal	FALSE	yes
rainy	mild	high	TRUE	no

$$X = [\underbrace{Outlook}_{x_1}, \underbrace{Temp}_{x_2}, \underbrace{Humidity}_{x_3}, \underbrace{Windy}_{x_4}$$

$$C_k = [\underbrace{Yes}_{C_1}, \underbrace{No}_{C_2}]$$

Example No.	Color	Type	Origin	Stolen?
1	Red	Sports	Domestic	Yes
2	Red	Sports	Domestic	No
3	Red	Sports	Domestic	Yes
4	Yellow	Sports	Domestic	No
5	Yellow	Sports	Imported	Yes
6	Yellow	SUV	Imported	No
7	Yellow	SUV	Imported	Yes
8	Yellow	SUV	Domestic	No
9	Red	SUV	Imported	No
10	Red	Sports	Imported	Yes

Outlook	Temp	Humidity	Windy	Play
Rainy	Cool	High	True	?

$$P(Yes \mid X) = P(Rainy \mid Yes) \times P(Cool \mid Yes) \times P(High \mid Yes) \times P(True \mid Yes) \times P(Yes)$$

$$P(Yes \mid X) = 2/9 \times 3/9 \times 3/9 \times 3/9 \times 9/14 = 0.00529 \quad 0.2 = \frac{0.00529}{0.02057 + 0.00529}$$

$$P(No \mid X) = P(Rainy \mid No) \times P(Cool \mid No) \times P(High \mid No) \times P(True \mid No) \times P(No)$$

$$P(No \mid X) = 3/5 \times 1/5 \times 4/5 \times 3/5 \times 5/14 = 0.02057 \quad 0.8 = \frac{0.02057}{0.02057 + 0.00529}$$

$$P(C_1 \mid x_1 \cap x_2 \cap x_3 \cap x_4) = \frac{P(x_1 \mid C_1) * P(x_2 \mid C_1) * P(x_3 \mid C_1) * P(x_4 \mid C_1) * P(C_1)}{P(x_1) * P(x_2) * P(x_3) * P(x_4)}$$

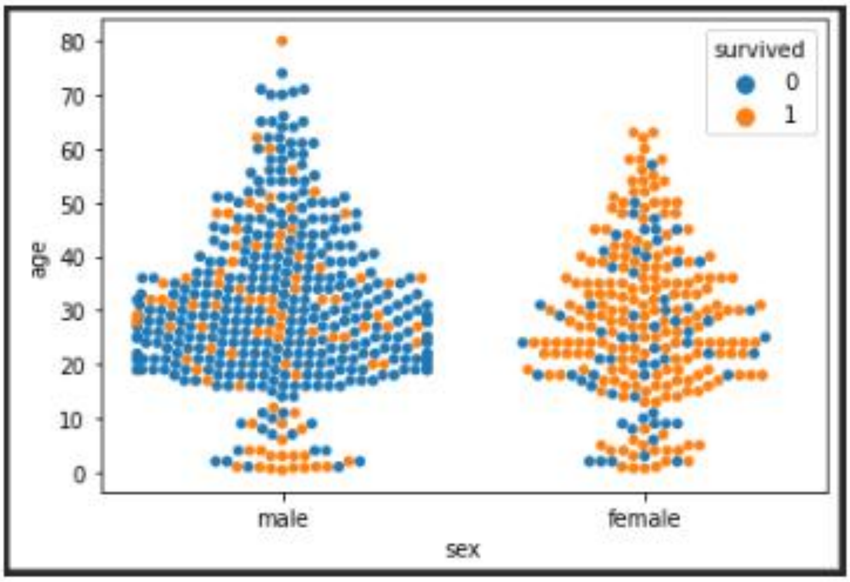
practical No - 7

Documents	Text	Total number of words in a document
A	Jupiter is the largest planet	5
B	Mars is the fourth planet from the sun	8

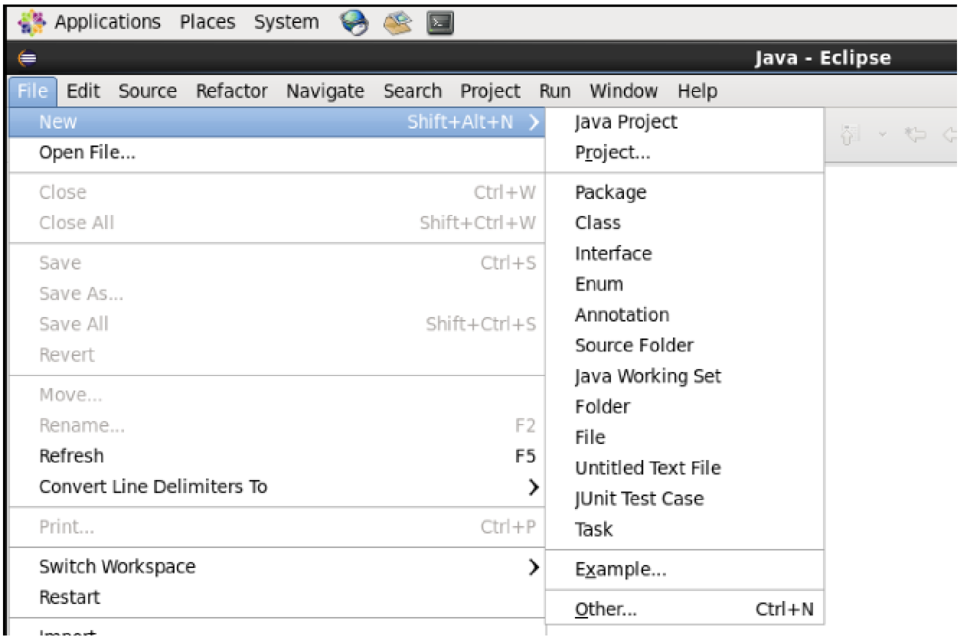
Words	TF (for A)	TF (for B)	IDF
Jupiter	1/5	0	ln(2/1) = 0.69
Is	1/5	1/8	ln(2/2) = 0
The	1/5	2/8	ln(2/2) = 0
largest	1/5	0	ln(2/1) = 0.69
Planet	1/5	1/8	ln(2/2) = 0
Mars	0	1/8	ln(2/1) = 0.69
Fourth	0	1/8	ln(2/1) = 0.69
From	0	1/8	ln(2/1) = 0.69
Sun	0	1/8	ln(2/1) = 0.69

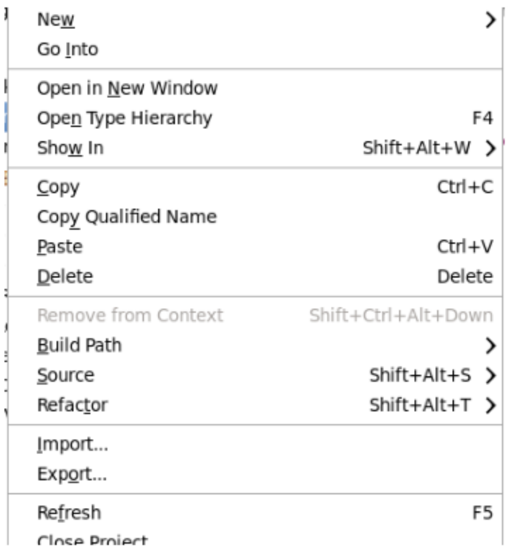
Words	TF (for A)	TF (for B)	IDF	TFIDF (A)	TFIDF (B)
Jupiter	1/5	0	ln(2/1) = 0.69	0.138	0
Is	1/5	1/8	ln(2/2) = 0	0	0
The	1/5	2/8	ln(2/2) = 0	0	0
largest	1/5	0	ln(2/1) = 0.69	0.138	0
Planet	1/5	1/8	ln(2/2) = 0	0.138	0
Mars	0	1/8	ln(2/1) = 0.69	0	0.086
Fourth	0	1/8	ln(2/1) = 0.69	0	0.086
From	0	1/8	ln(2/1) = 0.69	0	0.086
Sun	0	1/8	ln(2/1) = 0.69	0	0.086

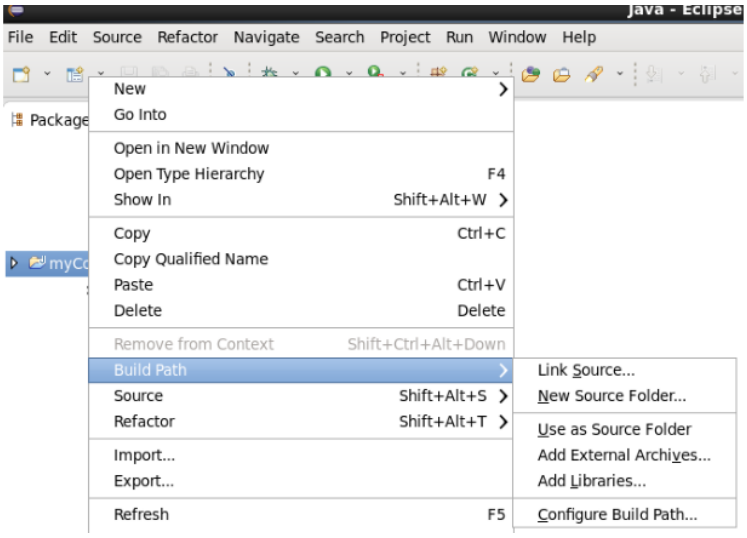
Practical No . – 8

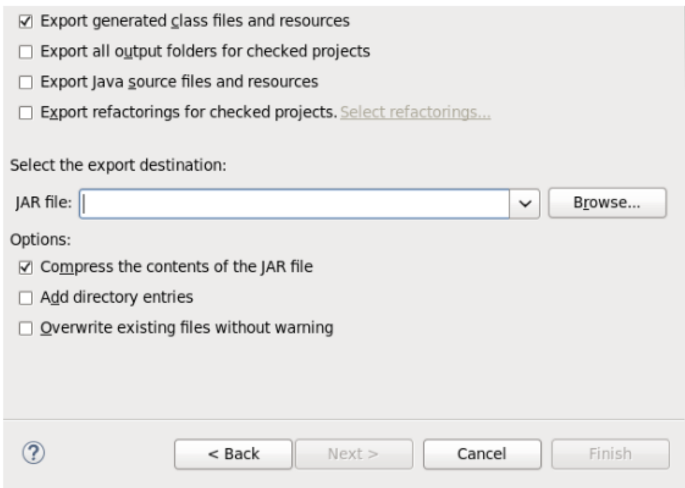


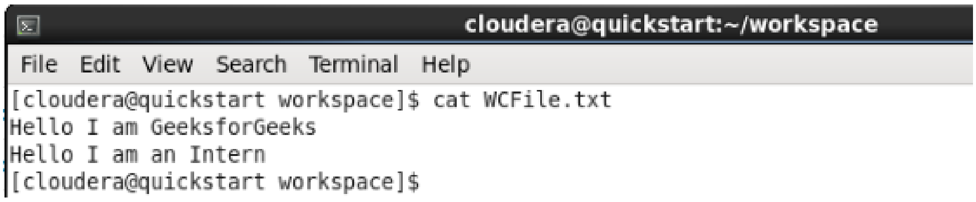
No Practical 9 & 10;

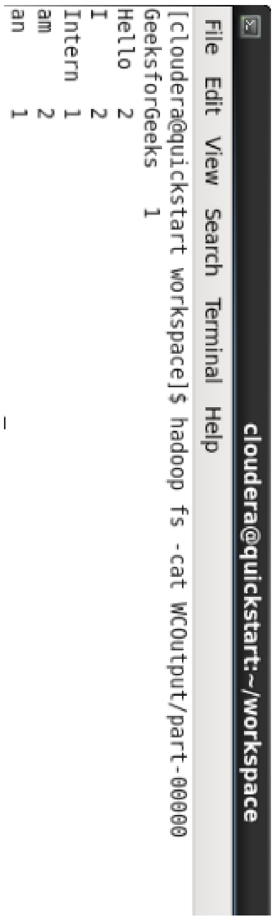












Practical No . - 11