Linear and Logistic Regression

Set A

Q1.

Data : Sales.csv

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score,mean_squared_error
%matplotlib inline
```

In [3]:
```
data=pd.read_csv('C:\\sales_A1.csv')
data.head()
data.shape
data.info()
data.describe()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 4 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   TV         200 non-null    float64
 1   Radio      200 non-null    float64
 2   Newspaper  200 non-null    float64
 3   Sales      200 non-null    float64
dtypes: float64(4)
memory usage: 6.4 KB
```

Out[3]:

	TV	Radio	Newspaper	Sales
count	200.000000	200.000000	200.000000	200.000000
mean	147.042500	23.264000	30.554000	15.130500
std	85.854236	14.846809	21.778621	5.283892
min	0.700000	0.000000	0.300000	1.600000
25%	74.375000	9.975000	12.750000	11.000000
50%	149.750000	22.900000	25.750000	16.000000
75%	218.825000	36.525000	45.100000	19.050000
max	296.400000	49.600000	114.000000	27.000000

In [4]:
```
x= np.array(data[['TV']])
y= np.array(data[['Sales']])
print(x.shape) # Vewing the shape of x
print(y.shape) # Vewing the shape of y
```

```
(200, 1)
(200, 1)
```

In [5]:
```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, train_size = 0.7, test_size = 0.3, random_state = 10(
```

In [6]:
```
model = LinearRegression()
model.fit(x_train, y_train)
```

Out[6]:
```
▾ LinearRegression
LinearRegression()
```

In [7]:
```
model.coef_
```

Out[7]:
```
array([[0.05454575]])
```

In [8]:
```
model.intercept_
```

Out[8]:
```
array([6.9486832])
```

Simple Linear Regression Model:

y(Sales) = 0.055 TV + 6.95

In [11]:
```
y_pred=model.predict(x_test)
#Finding the value of coefficient of Determination
r2_score(y_test,y_pred)
```

Out[11]:
```
0.7921031601245662
```

Interpretation :

R2 = 79.21%

79.21 % of the variations in the dependent variable (Y) is explained by the independent variables in the model (x).

Set B

Q1.

In [13]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score,mean_squared_error
```

In [14]:
```
data=pd.read_csv('C:\\Fish.csv')
data.head()
data.shape
data.info()
data.describe()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 159 entries, 0 to 158
Data columns (total 7 columns):
 #   Column   Non-Null Count  Dtype
---  ------   --------------  -----
 0   Species  159 non-null    object
 1   Weight   159 non-null    float64
 2   Length1  159 non-null    float64
 3   Length2  159 non-null    float64
 4   Length3  159 non-null    float64
 5   Height   159 non-null    float64
 6   Width    159 non-null    float64
dtypes: float64(6), object(1)
memory usage: 8.8+ KB
```

Out[14]:

	Weight	Length1	Length2	Length3	Height	Width
count	159.000000	159.000000	159.000000	159.000000	159.000000	159.000000
mean	398.326415	26.247170	28.415723	31.227044	8.970994	4.417486
std	357.978317	9.996441	10.716328	11.610246	4.286208	1.685804
min	0.000000	7.500000	8.400000	8.800000	1.728400	1.047600
25%	120.000000	19.050000	21.000000	23.150000	5.944800	3.385650
50%	273.000000	25.200000	27.300000	29.400000	7.786000	4.248500
75%	650.000000	32.700000	35.500000	39.650000	12.365900	5.584500
max	1650.000000	59.000000	63.400000	68.000000	18.957000	8.142000

In [15]:
```
#Dependent/Target Variable
y=np.array(data[['Weight']])
x = data.iloc[:,2:7]
x.shape
```

Out[15]:
```
(159, 5)
```

In [16]:
```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=51)
```

In [17]:
```
model=LinearRegression()
model.fit(x_train, y_train)
```

Out[17]:
```
▾ LinearRegression
LinearRegression()
```

In [18]:
```
model.coef_
```

Out[18]:
```
array([[ 65.59373225,  -9.18870094, -30.48074208,  28.87858453,
         29.42203026]])
```

In [19]:
```
model.intercept_
```

Out[19]:
```
array([-500.38802944])
```

Multiple Linear Regression Equation :

Y(Weight) = 65.5937(Length 1) - 9.1887(Length 2) - 30.4807 (Length 3)

. . + 28.8486 (Height) + 29.4220 (Width) - 500.388

In [20]:
```
y_pred=model.predict(x_test)
#Finding the value of coefficient of Determination
r2_score(y_test,y_pred)
```

Out[20]:
```
0.8785422051292161
```

Interpretation :

R2 = 87.85 %

87.85 % of the variations in the dependent variable (Y) is explained by the independent variables in the model (x).

LOGISTIC REGRESSION

Assignment 1

SET A

Q3.

In [35]:
```
data=pd.read_csv('C:\\User_Dataset.csv')
data.describe()
```

Out[35]:

	User ID	Age	EstimatedSalary	Purchased
count	4.000000e+02	400.000000	400.000000	400.000000
mean	1.569154e+07	37.655000	69742.500000	0.357500
std	7.165832e+04	10.482877	34096.960282	0.479864
min	1.556669e+07	18.000000	15000.000000	0.000000
25%	1.562676e+07	29.750000	43000.000000	0.000000
50%	1.569434e+07	37.000000	70000.000000	0.000000
75%	1.575036e+07	46.000000	88000.000000	1.000000
max	1.581524e+07	60.000000	150000.000000	1.000000

In [36]:
```
data.sample(3)
```

Out[36]:

	User ID	Gender	Age	EstimatedSalary	Purchased
326	15713463	Male	41	72000	0
122	15724423	Female	40	75000	0
6	15598044	Female	27	84000	0

In [37]:
```
data.isnull().sum()
```

Out[37]:
```
User ID            0
Gender             0
Age                0
EstimatedSalary    0
Purchased          0
dtype: int64
```

In [38]:
```
x= data.iloc[:, [2, 3]].values
y= data.iloc[:, 4].values
```

In [39]:
```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.25, random_state = 25)
```

In [40]:
```
from sklearn.linear_model import LogisticRegression
model = LogisticRegression()
model.fit(x_train, y_train)
y_pred = model.predict(x_test)
```

In [41]:
```
#Confusion Matrix
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
print(cm)
```

```
[[66  0]
 [34  0]]
```

Accuracy = (TP + TN)/TOTAL

In [43]:
```
66/(66+0+0+34)
```

Out[43]:
```
0.66
```

In [44]:
```
#Accuracy
from sklearn.metrics import accuracy_score
accuracy_score(y_test,y_pred)
```

Out[44]:
```
0.66
```

Accuracy for testing dataset=66%

Assignment 1

SET B

Q2.

In [45]:
```
import pandas as np
data=pd.read_csv('C:\\iris.csv')
```

In [46]:
```
data.describe()
```

Out[46]:

	sepal.length	sepal.width	petal.length	petal.width
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.057333	3.758000	1.199333
std	0.828066	0.435866	1.765298	0.762238
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.350000	1.300000
75%	6.400000	3.300000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

In [47]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   sepal.length  150 non-null    float64
 1   sepal.width   150 non-null    float64
 2   petal.length  150 non-null    float64
 3   petal.width   150 non-null    float64
 4   variety       150 non-null    object
dtypes: float64(4), object(1)
memory usage: 6.0+ KB
```

In [49]:
```
data.shape
```

Out[49]:
```
(150, 5)
```

In [50]:
```
data.duplicated().sum()
```

Out[50]:
```
1
```

In [51]:
```
#Removing the duplicated observation
data.drop_duplicates(inplace=True)
print("Shape of the data frame: ",data.shape)
print("\n")
print("Variety categories with its count \n",data["variety"].value_counts())
```

```
Shape of the data frame:  (149, 5)


Variety categories with its count
 Setosa        50
Versicolor    50
Virginica     49
Name: variety, dtype: int64
```

In [52]:
```
#define x and y
x = data.drop(columns="variety")
y = data["variety"]
```

In [53]:
```
#Converting y into numeric form by label encoding technique
from sklearn.preprocessing import LabelEncoder
y = LabelEncoder().fit_transform(y)
```

In [54]:
```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.25, random_state = 21)
```

In [55]:
```
from sklearn.linear_model import LogisticRegression
model = LogisticRegression()
model.fit(x_train, y_train)
y_pred= model.predict(x_test)
```

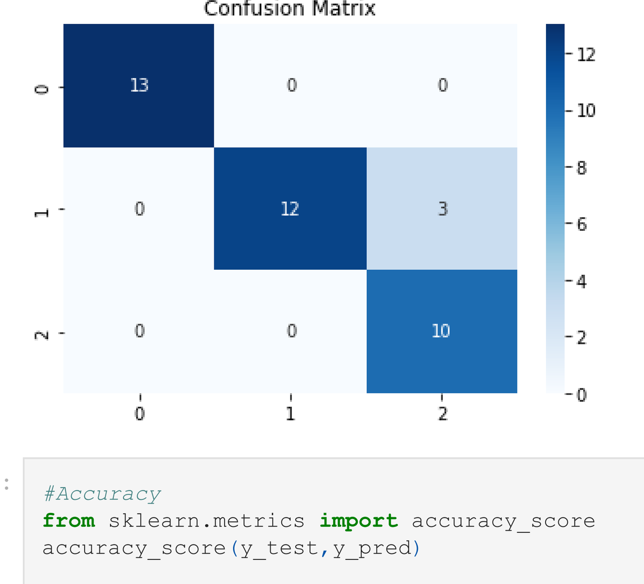
In [56]:
```
#Confusion Matrix
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
print(cm)
```

```
[[13  0  0]
 [ 0 12  3]
 [ 0  0 10]]
```

In [57]:
```
(13+12+10)/(13+12+10+3)             #Accuracy
```

Out[57]:
```
0.9210526315789473
```

In [58]:
```
#Plot cm
import seaborn as sns
import matplotlib.pyplot as plt
ax = plt.axes()
sns.heatmap(cm,annot=True,cmap="Blues")
ax.set_title('Confusion Matrix')
plt.show()
```



In [59]:
```
#Accuracy
from sklearn.metrics import accuracy_score
accuracy_score(y_test,y_pred)
```

Out[59]:
```
0.9210526315789473
```

Accuracy for testing dataset = 92.1%