Slip 1

Q1.Write a R program to add, multiply and divide two vectors of integertype. (Vector length should be minimum 4)

```
a<-c(1,3,5,7)
b<-c(2,4,6,8)
print(a+b)
print(a-b)
print(a/b)
print(a%%b)
```

Q2.Consider the student data set. It can be downloaded from: https://drive.google.com/open?id=1oakZCv7g3mlmCSdv9J8kdSaqO 5_6dIOw . Write a programme in python to apply simple linear regression and find out mean absolute error, mean squared error and root mean squared error. [20 Marks]

```
import numpy as nm
import pandas as pd
data_set= pd.read_csv('student_scores.csv')
print(data_set)
y = data_set['Scores'].values.reshape(-1, 1)
X = data_set['Hours'].values.reshape(-1, 1)
print(X)
print(y)
print(X.shape)
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)
print(X_train)
print(X_test)
from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
regressor.fit(X_train, y_train)
print(regressor.intercept_)
print(regressor.coef_)
score = regressor.predict([[9.5]])
print(score)
y_pred = regressor.predict(X_test)
print(y_pred)
from sklearn.metrics import mean_absolute_error, mean_squared_error
mae = mean_absolute_error(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)
rmse = nm.sqrt(mse)
print(mae)
print(mse)
print(rmse)
print('Actual',y_test)
print('Predicted',y_pred)
```

slip 2

Q1. Write an R program to calculate the multiplication table using a function. [10 Marks]

```
num = as.integer(readline(prompt = "Enter a number: "))
for(i in 1:10)
 {
 print(paste(num,'x', i, '=', num*i))
}
```

Q2. Write a python program to implement k-means algorithms on asynthetic dataset. [20 Marks]

```
import seaborn as sns

import matplotlib.pyplot as plt

from sklearn.datasets import make_blobs

data = make_blobs(n_samples=300, n_features=2, centers=5, cluster_std=1.8,random_state=101)

data[0].shape

data[1]

plt.scatter(data[0][:,0],data[0][:,1],c=data[1],cmap='brg')

from sklearn.cluster import KMeans

kmeans = KMeans(n_clusters=5)

kmeans.fit(data[0])

kmeans.cluster_centers_

kmeans.labels_f, (ax1, ax2) = plt.subplots(1, 2, sharey=True,figsize=(10,6))

ax1.set_title('K Means')

ax1.scatter(data[0][:,0],data[0][:,1],c=kmeans.labels_,cmap='brg')

ax2.set_title("Original")

ax2.scatter(data[0][:,0],data[0][:,1],c=data[1],cmap='brg')
```

Slip 3

Q1. Write a R program to reverse a number and also calculate the sum of digits of that number. [10 Marks]

```
n = as.integer(readline(prompt = "Enter a number :"))
sum = 0
while (n > 0) {
  r = n %% 10
  sum = sum + r
  n = n %/% 10
}
print(paste("Sum of digit is :", sum))
```

Q2. Consider the following observations/data. And apply simple linear regression and find out estimated coefficients b0 and b1.(use numpypackage) x=[0,1,2,3,4,5,6,7,8,9,11,13]
y = ([1, 3, 2, 5, 7, 8, 8, 9, 10, 12,16, 18] [20 Marks]

```
import numpy as np
from sklearn.linear_model import LinearRegression
x= np.array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9,11,13]).reshape((-1, 1))
y = np.array([1, 3, 2, 5, 7, 8, 8, 9, 10, 12,16, 18])
print(x)
print(y)
model = LinearRegression()
model.fit(x, y)
print('intercept:-',model.intercept_)
print('Slope:- ', model.coef_)
```

slip 4

Q1. Write a R program to calculate the sum of two matrices of given size. [10 Marks]

```
m1 = matrix(c(1, 2, 3, 4, 5, 6), nrow = 2)
print("Matrix-1:")
print(m1)
m2 = matrix(c(0, 1, 2, 3, 0, 2), nrow = 2)
print("Matrix-2:")
print(m2)
result = m1 + m2
print("Result of addition")
print(result)
```

Q2. Consider following dataset

weather=['Sunny','Sunny','Overcast','Rainy','Rainy','Rainy','Overcast','Sunny','Sunny','Rainy','Sunny','Overcast','Overcast','Rainy']

temp=['Hot','Hot','Hot','Mild','Cool','Cool','Cool','Mild','Cool','Mild','Mild','Mild','Hot','Mild']

play=['No','No','Yes','Yes','Yes','No','Yes','No','Yes','Yes','Yes','Yes','Yes','No'].

Use Naïve Bayes algorithm to predict [0: Overcast, 2: Mild]tuple belongs to which class whether to play the sports or not.[20 Marks]

```
weather=['sunny','sunny','overcast','rainy','rainy','rainy','overcast','sunny','sunny','rainy','
        sunny','overcast','overcast','rainy']
temp=['hot','hot','hot','mild','cool','cool','cool','mild','cool','mild','mild','mild','hot','mild']
play=['No','No','Yes','Yes','Yes','No','Yes','No','Yes','Yes','Yes','Yes','Yes','No']
from sklearn import preprocessing
le = preprocessing.LabelEncoder()
wheather_encoded = le.fit_transform(weather)
print(wheather_encoded)
temp_encoded = le.fit_transform(temp)
label = le.fit_transform(play)
print("Temp:",temp_encoded)
print("Play:",label)
features = list(zip(wheather_encoded,temp_encoded))
print(features)
from sklearn.naive_bayes import GaussianNB
model = GaussianNB()
model.fit(features,label)
```

```
predicted = model.predict([[0,2]])
print("Predicted Value:",predicted)
```

slip 5

Q1. Write a R program to concatenate two given factors. [10 Marks]

```
f1 <- factor(sample(LETTERS, size=6, replace=TRUE))
f2 <- factor(sample(LETTERS, size=6, replace=TRUE))
print("Original factors:")
print(f1)
print(f2)
f = factor(c(levels(f1)[f1], levels(f2)[f2]))
print("After concatenate factor becomes:")
print(f)
```

Q2. Write a Python program build Decision Tree Classifier using Scikit- learn package for diabetes data set (download database from https://www.kaggle.com/uciml/pima-indians-diabetes-database)

[20 Marks]

Slip 6

Q1. Write a R program to create a data frame using two given vectors and display the duplicate elements. [10 Marks]

```
a = c(10,20,10,10,40,50,20,30)
b = c(10,30,10,20,0,50,30,30)
print("Original data frame:")
ab = data.frame(a,b)
print(ab)
print("Duplicate elements of the said data frame:")
print(duplicated(ab))
print("Unique rows of the said data frame:")
print(unique(ab))
```

Q2. Write a python program to implement hierarchical Agglomerative clustering algorithm. (Download Customer.csv dataset from github.com).[20 Marks]

```
dataset = pd.read_csv('Mall_Customers.csv')

x = dataset.iloc[:, [3, 4]].values

import scipy.cluster.hierarchy as shc

dendro = shc.dendrogram(shc.linkage(x, method="ward"))

mtp.title("Dendrogrma Plot")

mtp.ylabel("Euclidean Distances")

mtp.xlabel("Customers")
mtp.show()

from sklearn.cluster import AgglomerativeClustering

hc= AgglomerativeClustering(n_clusters=5, affinity='euclidean', linkage='ward')

y_pred= hc.fit_predict(x)

mtp.scatter(x[y_pred == 0, 0], x[y_pred == 0, 1], s = 100, c = 'blue', label = 'Cluster 1')

mtp.scatter(x[y_pred == 1, 0], x[y_pred == 1, 1], s = 100, c = 'green', label = 'Cluster 2')

mtp.scatter(x[y_pred== 2, 0], x[y_pred == 2, 1], s = 100, c = 'red', label = 'Cluster 3')

mtp.scatter(x[y_pred == 3, 0], x[y_pred == 3, 1], s = 100, c = 'cyan', label = 'Cluster 4')

mtp.scatter(x[y_pred == 4, 0], x[y_pred == 4, 1], s = 100, c = 'magenta', label = 'Cluster 5')

mtp.title('Clusters of customers')

mtp.xlabel('Annual Income (k$)')

mtp.ylabel('Spending Score (1-100)')

mtp.legend()

mtp.show()
```

slip 7

Q1. Write a R program to create a sequence of numbers from 20 to 50 and find the mean ofnumbers from 20 to 60 and sum of numbers from 51 to 91.[10 Marks]

```
print("Sequence of numbers from 20 to 50:")

print(seq(20,50))

print("Mean of numbers from 20 to 60:")

print(mean(20:60))

print("Sum of numbers from 51 to 91:")

print(sum(51:91))
```

Q2. Consider the following observations/data. And apply simple linear regression and find out estimated coefficients b1 and b1 Also analyse theperformance of the model

(Use sklearn package)

```
x = np.array([1,2,3,4,5,6,7,8])
```

```
y =  np.array([7,14,15,18,19,21,26,23])                                  [20 Marks]
 import numpy as np
 from sklearn.linear_model import LinearRegression
 x= np.array([1,2,3,4,5,6,7,8]).reshape((-1, 1))
 print(x)
y = np.array([7,14,15,18,19,21,26,23])
 print(y)
model = LinearRegression()
 model.fit(x, y)
 x_new = np.array(9).reshape((-1, 1))
 y_new_pred = model.predict(x_new)
 print(y_new_pred)
print('Slope:- ', model.coef_)
```

slip 8

Q1. Write a R program to get the first 10 Fibonacci numbers. [10 Marks]

```
Fibonacci <- numeric(10)
Fibonacci[1] <- Fibonacci[2] <- 1
for (i in 3:10) Fibonacci[i] <- Fibonacci[i - 2] + Fibonacci[i - 1]
print("First 10 Fibonacci numbers:")
print(Fibonacci)
```

Q2. Write a python program to implement k-means algorithm to build prediction model (Use Credit Card Dataset CC GENERAL.csv Download from kaggle.com) [20 Marks]

```
import numpy as nm
import matplotlib.pyplot as mtp
import pandas as pd
dataset = pd.read_csv('creditcard.csv')
dataset
x = dataset.iloc[:, [3, 4]].values
print(x)
from sklearn.cluster import KMeans
wcss_list= []
for i in range(1, 11):
         kmeans = KMeans(n_clusters=i, init='k-means++', random_state= 42)
        kmeans.fit(x)
wcss_list.append(kmeans.inertia_)
mtp.plot(range(1, 11), wcss_list)
mtp.title('The Elobw Method Graph')
mtp.xlabel('Number of clusters(k)')
mtp.ylabel('wcss_list')
mtp.show()
kmeans = KMeans(n_clusters=3, init='k-means++', random_state= 42)
y_predict= kmeans.fit_predict(x)
mtp.scatter(x[y_predict == 0, 0], x[y_predict == 0, 1], s = 100, c = 'blue', label = 'Cluster 1') #for first cluster
mtp.scatter(x[y_predict == 1, 0], x[y_predict == 1, 1], s = 100, c = 'green', label = 'Cluster 2') #for second cluster
mtp.scatter(x[y_predict== 2, 0], x[y_predict == 2, 1], s = 100, c = 'red', label = 'Cluster 3') #for third cluster
mtp.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:, 1], s = 300, c = 'yellow', label = 'Centroid')
mtp.title('Clusters of Credit Card')
mtp.xlabel('V3')
mtp.ylabel('V4')
mtp.legend()
mtp.show()
```

slip 9

Q1. Write an R program to create a Data frames which contain details of 5 employees and display summary of the data. [10 Marks]

```
Employees = data.frame(Name=c("Ram","sham","swati", "pooja","Arun"),
                      Gender=c("M","M","F","F","M"),
                      Age=c(23,22,25,26,32),

Designation=c("Clerk","Manager","Exective","CEO","ASSISTANT"),
                      SSN=c("123-34-2346","123-44-779","556-24-433","123-
98-987","679-77-576")
                      )
print("Details of the employees:")
print(Employees)
```

Q2. Write a Python program to build an SVM model to Cancer dataset. The dataset is available in the scikit-learn library. Check the accuracy of model with precision and recall.

```
import numpy as nm
import pandas as pd
data_set= pd.read_csv('cancer.csv')
data_set
x= data_set.iloc[:, [2,3]].values
y= data_set.iloc[:, 4].values
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=0)
from sklearn.preprocessing import StandardScaler
st_x= StandardScaler()
x_train= st_x.fit_transform(x_train)
x_test= st_x.transform(x_test)
print(x_train)
print(x_test)
from sklearn.svm import SVC
classifier = SVC(kernel='linear', random_state=0)
classifier.fit(x_train, y_train)
y_pred= classifier.predict(x_test)
from sklearn.metrics import confusion_matrix
cm= confusion_matrix(y_test, y_pred)
print(cm)
from sklearn.metrics import accuracy_score
result2 = accuracy_score(y_test,y_pred)
print("Accuracy:",result2)
```

Slip 10 [20 Marks]

Q1. Write a R program to find the maximum and the minimum value of a given vector [10 Marks]

```
x = c(10, 20, 30, 25, 9, 26)
print("Original Vectors:")
print(x)
print("Maximum value of the above Vector:")
```

```
   print(max(x))
   print("Minimum value of the above Vector:")
   print(min(x))
```

Q2. Write a Python Programme to read the dataset (“Iris.csv”). dataset download from (https://archive.ics.uci.edu/ml/datasets/iris) and apply Apriori algorithm. [20 Marks]

```
import numpy as nm
import matplotlib.pyplot as mtp
import pandas as pd
dataset = pd.read_csv('Iris.csv')
dataset
transactions=[]
for i in range(0, 150):
    transactions.append([str(dataset.values[i,j])  for j in range(0,6)])
from apyori import apriori
rules= apriori(transactions= transactions, min_support=0.003, min_confidence = 0.2, min_lift=3, 
min_length=2, max_length=2)
results= list(rules)
results
for item in results:
    pair = item[0]
    items = [x for x in pair]
    print("Rule: " + items[0] + " -> " + items[1])

    print("Support: " + str(item[1]))
    print("Confidence: " + str(item[2][0][2]))
    print("Lift: " + str(item[2][0][3]))
    print("=====================================")
```

slip 11

Q1. Write a R program to find all elements of a given list that are not in another given list.

```
                A =  st("x", "y", ' z")
                B =  st("X", "Y", "Z", "x", "y", "z")
```

[10 Marks]

```
 l1 = list("x", "y", "z")
 l2 = list("X", "Y", "Z", "x", "y", "z")
 print("Original lists:")
 print(l1)
 print(l2)
 print("All elements of l2 that are not in l1:")
 setdiff(l2, l1)
```

Q2. Write a python program to implement hierarchical clustering algorithm.(Download Wholesale customers data dataset from github.com).[20 Marks]

```
import numpy as nm

import matplotlib.pyplot as mtp

import pandas as pd

dataset = pd.read_csv('Wholesale customers data.csv')

dataset
```

```
x = dataset.iloc[:, [3, 4]].values

print(x)

import scipy.cluster.hierarchy as shc

dendro = shc.dendrogram(shc.linkage(x, method="ward"))

mtp.title("Dendrogrma Plot")

mtp.ylabel("Euclidean Distances")

mtp.xlabel("Customers")

mtp.show()

from sklearn.cluster import AgglomerativeClustering

hc= AgglomerativeClustering(n_clusters=5, affinity='euclidean', linkage='ward')

y_pred= hc.fit_predict(x)

mtp.scatter(x[y_pred == 0, 0], x[y_pred == 0, 1], s = 100, c = 'blue', label = 'Cluster 1')

mtp.scatter(x[y_pred == 1, 0], x[y_pred == 1, 1], s = 100, c = 'green', label = 'Cluster 2')

mtp.scatter(x[y_pred== 2, 0], x[y_pred == 2, 1], s = 100, c = 'red', label = 'Cluster 3')

mtp.scatter(x[y_pred == 3, 0], x[y_pred == 3, 1], s = 100, c = 'cyan', label = 'Cluster 4')

mtp.scatter(x[y_pred == 4, 0], x[y_pred == 4, 1], s = 100, c = 'magenta', label = 'Cluster 5')

mtp.title('Clusters of customers')

mtp.xlabel('Milk')

mtp.ylabel('Grocery')

mtp.legend()

mtp.show()
```

slip 12

Q1. Write a R program to create a Dataframes which contain details of 5employees anddisplay the details.

Employee contain (empno,empname,gender,age,designation)[10 Marks]

```
Employees = data.frame(Name=c("Anastasia S","Dima R","Katherine S", "JAMES
A","LAURA MARTIN"),
                    Gender=c("M","M","F","F","M"),
                    Age=c(23,22,25,26,32),
                    Designation=c("Clerk","Manager","Exective","CEO","ASSISTANT"),
                    SSN=c("123-34-2346","123-44-779","556-24-433","123-98-987","679-77-576")
                    )
           print("Details of the employees:")
           print(Employees)
```

Q2. Write a python program to implement multiple Linear Regression model for a car dataset. Dataset can be downloaded from: https://www.w3schools.com/python/python_ml_multiple_regression.asp[20 Marks]

```
import pandas
from sklearn import linear_model
df = pandas.read_csv("data.csv")
print(df)
X = df[['Weight', 'Volume']]
print(X)
y = df['CO2']
print(y)
regr = linear_model.LinearRegression()
regr.fit(X, y)
predictedCO2 = regr.predict([[2300, 1300]])
print(predictedCO2)
```

Slip 13

Q1. Draw a pie chart using R programming for the following data distribution:

Digits on Dice	1	2	3	4	5	6
Frequency of getting each number	7	2	6	3	4	8

[10 Marks]

```
Labels<- c(1,2,3,4,5,6)
Frequency<- c(7,2,6,3,4,8)
Pie(Frequency,Labels)
```

Q2. Write a Python program to read "StudentsPerformance.csv" file. Solve following:

- To display the shape of dataset.
- To display the top rows of the dataset with their columns.Note: Download dataset from following link :

(https://www.kaggle.com/spscientist/students-performance-inexams?select=StudentsPerformance.csv) [20 Marks]

Slip 14

Q1. Write a script in R to create a list of employees (name) and perform thefollowing:

a. Display names of employees in the list.

b. Add an employee at the end of the list

c. Remove the third element of the list. [10 Marks]

```
emp<-list("Ram","sham","swati", "pooja","Arun"),
print(emp)
emp<-append(emp,”Rohan”)
print(emp)
newlist<-emp[-3]
print(newlist)
```

Q2. Write a Python Programme to apply Apriori algorithm on Groceries dataset. Dataset can be downloaded from (https://github.com/amankharwal/Websitedata/blob/master/Groceries_dataset.csv).

Also display support and confidence for each rule. [20 Marks]

```
import numpy as nm
import matplotlib.pyplot as mtp
import pandas as pd
dataset = pd.read_csv('store_data.csv')
dataset
transactions=[]
for i in range(0, 7500):
   transactions.append([str(dataset.values[i,j])  for j in range(0,20)])
from apyori import apriori
```

slip 15

Q1.Write a R program to add, multiply and divide two vectors of integer type.(vector length should be minimum 4) [10 Marks]

```
a<-c(1,3,5,7)

b<-c(2,4,6,8)

print(a+b)

print(a-b)

print(a/b)

print(a%%b)
```

Q2. Write a Python program build Decision Tree Classifier for shows.csvfrom pandas and predict class label for show starring a 40 years old American comedian, with 10 years of experience, and a comedy ranking of 7? Create a csv file as shown in https://www.w3schools.com/python/python_ml_decision_tree.asp[20 Marks]

```
import pandas
from sklearn import tree
from sklearn.tree import DecisionTreeClassifier

df = pandas.read_csv("data.csv")
```

```
d = {'UK': 0, 'USA': 1, 'N': 2}
df['Nationality'] = df['Nationality'].map(d)
d = {'YES': 1, 'NO': 0}
df['Go'] = df['Go'].map(d)

features = ['Age', 'Experience', 'Rank', 'Nationality']
X = df[features]
y = df['Go']

dtree = DecisionTreeClassifier()
dtree = dtree.fit(X, y)

print(dtree.predict([[40, 10, 7, 1]]))

print("[1] means 'GO'")
print("[0] means 'NO'")
```

Slip 16

Q1. Write a R program to create a simple bar plot of given data

Year	Export	Import
2001	26	35
2002	32	40
2003	35	50

[10 Marks]

```
year=c(2001,2002,2003);
export=c(26,32,35);
import=c(35,40,50);

par(mfrow=c(1,2));
barplot(export,legend.text=year,col=rainbow(3),main="Export");

barplot(import,legend.text=year,col=rainbow(3),main="Import");
```

Q2. Write a Python program build Decision Tree Classifier using Scikit-learn package for diabetes data set (download database from https://www.kaggle.com/uciml/pima-indians-diabetes-database)[20 Marks]

Slip 17

Q1. Write a R program to get the first 20 Fibonacci numbers.[10 Marks]

```
length_fib = 20
fibonacci = numeric(length_fib)
fibonacci[1] = 1
fibonacci[2] = 1
for (i in 3:length_fib) {
  fibonacci[i] = fibonacci[i-1] + fibonacci[i-2]
}
```

Q2. Write a python programme to implement multiple linear regression model for stock market data frame as follows:
Stock_Market = {'Year':
[2017,2017,2017,2017,2017,2017,2017,2017,2017,2017,2017,2017,2016,2
016,20,16,2016,2016,2016,2016,2016,2016,2016,2016,2016],
'Month': [12, 11,10,9,8,7,6,5,4,3,2,1,12,11,10,9,8,7,6,5,4,3,2,1],
'Interest_Rate': [2.75,2.5,2.5,2.5,2.5,2.5,2.5,2.25,2.25,2.25,2,2,2,1.75,1.75,1.75,1.75,1.75,1
.75,1.75,1.75,1.75,1.75,1.75],
'Unemployment_Rate':
[5.3,5.3,5.3,5.3,5.4,5.6,5.5,5.5,5.5,5.6,5.7,5.9,6,5.9,5.8,6.1,6.2,6.1,6.1,6.1,5
.9,6.2,6.2,6.1],
'Stock_Index_Price': [1464,1394,1357,1293,1256,1254,1234,1195,1159,1167,1130,1075,1047,
965,943,958,971,949,884,866,876,822,704,719] }
And draw a graph of stock market price verses interest rate.[20 Marks]

```
import pandas as pd
from sklearn import linear_model
data = {'year':
            [2017,2017,2017,2017,2017,2017,2017,2017,2017,2017,2017,2017,2016,2016,
            2016,2016,2016,2016,2016,2016,2016,2016,2016,2016],
        'month': [12,11,10,9,8,7,6,5,4,3,2,1,12,11,10,9,8,7,6,5,4,3,2,1],
        'interest_rate':
            [2.75,2.5,2.5,2.5,2.5,2.5,2.5,2.25,2.25,2.25,2,2,2,1.75,1.75,1.75,1.75,1.75,1.75,1
            .75,1.75,1.75,1.75,1.75],
        'unemployment_rate':
            [5.3,5.3,5.3,5.3,5.4,5.6,5.5,5.5,5.5,5.6,5.7,5.9,6,5.9,5.8,6.1,6.2,6.1,6.1,6.1,5.9,6.
            2,6.2,6.1],
        'index_price':
            [1464,1394,1357,1293,1256,1254,1234,1195,1159,1167,1130,1075,1047,965,9
            43,958,971,949,884,866,876,822,704,719]
        }
 df = pd.DataFrame(data)
print(df)
x = df[['interest_rate','unemployment_rate']]
print(x)
y = df['index_price']
print(y)
 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)
 print(X_train)
 print(X_test)
 regr = linear_model.LinearRegression()
regr.fit(X_train, y_train)
print('Intercept: \n', regr.intercept_)
 print('Coefficients: \n', regr.coef_)
 y_pred=regr.predict(X_test)
print(y_pred)
from sklearn.metrics import r2_score
Accuracy=r2_score(y_test,y_pred)*100
print(Accuracy)
```

```
import matplotlib.pyplot as plt
 plt.scatter(y_test,y_pred);
 plt.xlabel('Actual');
 plt.ylabel('Predicted');
import seaborn as sns
sns.regplot(x=y_test,y=y_pred,ci=None,color ='red');
```

slip 18

Q1. Write a R program to find the maximum and the minimum value of a given vector [10 Marks]

```
x = c(10, 20, 30, 25, 9, 26)

print("Original Vectors:")

print(x)

print("Maximum value of the above Vector:")

print(max(x))

print("Minimum value of the above Vector:")

print(min(x))
```

Q2. Consider the following observations/data. And apply simple linear regression and find out estimated coefficients b1 and b1 Also analyse the performance of the model

(Use sklearn package)

x = np.array([1,2,3,4,5,6,7,8])

y = np.array([7,14,15,18,19,21,26,23])

[20 Marks]

```
 import numpy as np
 from sklearn.linear_model import LinearRegression
 x= np.array([1,2,3,4,5,6,7,8]).reshape((-1, 1))
 print(x)
y = np.array([7,14,15,18,19,21,26,23])
 print(y)
model = LinearRegression()
 model.fit(x, y)
 x_new = np.array(9).reshape((-1, 1))
 y_new_pred = model.predict(x_new)
 print(y_new_pred)
print('Slope:- ', model.coef_)
```

slip 19

Q1. Write a R program to create a Data frames which contain details of 5 Students and display the details.

Students contain (Rollno,Studname,Address,Marks) [10 Marks]

```
Students = data.frame(Rollno = c(11,13,14,15,16),
                      Studname=c("Ram","sham","swati", "pooja","Arun"),
                      Address=c("pune","Mumbai","delhi","jaipur","Bhopal"),
                      Marks=c(23,22,25,26,32))
print("Details of the Students :")
print(Students)
```

Q2. Write a python program to implement multiple Linear Regression model for a car dataset.

Dataset can be downloaded from: https://www.w3schools.com/python/python_ml_multiple_regression.asp[20 Marks]

Slip 20

Q1. Write a R program to create a data frame from four given vectors.[10 Marks]

```
name = c('Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura',
'Kevin', 'Jonas')
score = c(12.5, 9, 16.5, 12, 9, 20, 14.5, 13.5, 8, 19)
attempts = c(1, 3, 2, 3, 2, 3, 1, 1, 2, 1)
qualify = c('yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes')
print("Original data frame:")
print(name)
print(score)
print(attempts)
print(qualify)
df = data.frame(name, score, attempts, qualify)
print(df)
```

Q2. Write a python program to implement hierarchical Agglomerative clustering algorithm. (Download Customer.csv dataset from github.com).[20 Marks]

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
# Importing the dataset
dataset = pd.read_csv('Mall_Customers.csv')
X = dataset.iloc[:, [3, 4]].values
# y = dataset.iloc[:, 3].values
# Splitting the dataset into the Training set and Test set
"""from sklearn.cross_validation import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state =
0)"""
# Feature Scaling
"""from sklearn.preprocessing import StandardScaler
sc_X = StandardScaler()
X_train = sc_X.fit_transform(X_train)
X_test = sc_X.transform(X_test)
sc_y = StandardScaler()
y_train = sc_y.fit_transform(y_train)"""
# Using the dendrogram to find the optimal number of clusters
import scipy.cluster.hierarchy as sch
dendrogram = sch.dendrogram(sch.linkage(X, method = 'ward'))
plt.title('Dendrogram')
plt.xlabel('Customers')
plt.ylabel('Euclidean distances')
plt.show()
# Fitting Hierarchical Clustering to the dataset
from sklearn.cluster import AgglomerativeClustering
hc = AgglomerativeClustering(n_clusters = 5, affinity = 'euclidean', linkage = 'ward')
y_hc = hc.fit_predict(X)
```

```
# Visualising the clusters
plt.scatter(X[y_hc == 0, 0], X[y_hc == 0, 1], s = 100, c = 'red', label = 'Cluster 1')
plt.scatter(X[y_hc == 1, 0], X[y_hc == 1, 1], s = 100, c = 'blue', label = 'Cluster 2')
plt.scatter(X[y_hc == 2, 0], X[y_hc == 2, 1], s = 100, c = 'green', label = 'Cluster 3')
plt.scatter(X[y_hc == 3, 0], X[y_hc == 3, 1], s = 100, c = 'cyan', label = 'Cluster 4')
plt.scatter(X[y_hc == 4, 0], X[y_hc == 4, 1], s = 100, c = 'magenta', label = 'Cluster
5')
plt.title('Clusters of customers')
plt.xlabel('Annual Income (k$)')
plt.ylabel('Spending Score (1-100)')
plt.legend()
plt.show()
```