

Slip no: 1

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

Answer:

```
vector1 = seq(10,40 , length.out=4)
vector2 = c(20, 10, 40, 40)
print("Original Vectors:")
print(vector1)
print(vector2)
add= vector1+vector2
cat("Sum of vector is ",add, "\n")
sub_vector= vector1-vector2
cat("Substraction of vector is ",sub_vector, "\n")
mul_vector= vector1 * vector2
cat("Multiplication of vector is ",mul_vector, "\n")
print("Division of two Vectors:")
"Division of two Vectors:"
div_vector = vector1 / vector2
print(div_vector)
```

Q2. Consider the student data set. It can be downloaded from:

https://drive.google.com/open?id=1oakZCv7g3mImCSdv9J8kdSaqO 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.

Answer: import pandas as pd

import numpy as np

from sklearn.model_selection import train_test_split

from sklearn.linear_model import LinearRegression

from sklearn.metrics import mean_absolute_error, mean_squared_error

Load the dataset

data = pd.read_csv('student_data.csv')

Assuming you have two columns: 'independent_variable' and 'dependent_variable'

```

X = data['independent_variable'].values.reshape(-1, 1)
y = data['dependent_variable']

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Create a linear regression model
model = LinearRegression()

# Fit the model on the training data
model.fit(X_train, y_train)

# Predict the values on the test data
y_pred = model.predict(X_test)

# Calculate Mean Absolute Error (MAE)
mae = mean_absolute_error(y_test, y_pred)

# Calculate Mean Squared Error (MSE)
mse = mean_squared_error(y_test, y_pred)

# Calculate Root Mean Squared Error (RMSE)
rmse = np.sqrt(mse)

print("Mean Absolute Error:", mae)
print("Mean Squared Error:", mse)
print("Root Mean Squared Error:", rmse)

```

Slip no:2

Q1. Write an R program to calculate the multiplication table using a function.

Answer:

```

table<-function(number)
{
  for(t in 1:10)
  {
    print(paste(number,'*',t,'=',number*t))
  }
}

table(2)

```

Q2. Write a python program to implement k-means algorithms on a synthetic dataset.

Answer:

```
# importing necessary libraries
```

```
import matplotlib.pyplot as plt
```

```
import numpy as np
```

```
from sklearn.datasets import make_blobs
```

```
# generating random data points
```

```
x, y = make_blobs(n_samples=100, centers=3, n_features=2)
```

```
# plotting the generated data points
```

```
plt.scatter(x[:, 0], x[:, 1], c=y, cmap='gist_rainbow')
```

```
plt.show()
```

```
# implementing K-Means
```

```
k = 3
```

```
# assigning random centers
```

```
center = 10*np.random.rand(k, 2)
```

```
# computing the distance matrix
```

```
dist_matrix = np.zeros((100, 3))
```

```
for i in range(k):
```

```
    dist_matrix[:, i] = np.sum((x-center[i, :])**2, axis=1)
```

```
# assigning labels to each data point
```

```
cluster_labels = np.argmin(dist_matrix, axis=1)
```

```
# plotting the labeled data points
```

```
plt.scatter(x[:, 0], x[:, 1], c=cluster_labels, cmap='gist_rainbow')
```

```
plt.show()
```

```
# updating the cluster centers
```

```
for i in range(k):
```

```
    center[i, :] = np.mean(x[cluster_labels == i, :], axis=0)
```

```
# recomputing the distance matrix
```

```
dist_matrix = np.zeros((100, 3))
```

```
for i in range(k):
```

```
    dist_matrix[:, i] = np.sum((x-center[i, :])**2, axis=1)
```

```
# reassigning labels to each data point
```

```

cluster_labels = np.argmin(dist_matrix, axis=1)

# plotting the labeled data points

plt.scatter(x[:, 0], x[:, 1], c=cluster_labels, cmap='gist_rainbow')

plt.show()

```

Slip no:3

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

Answer:

n=567

Reverse=function(n)

```

{
sum=0
rev=0
while(n>0)
{
r=n%%10
sum=sum+r
rev=rev*10+r
n=n%%10
}
print(rev)
print(sum)
}

```

Reverse(n)

Q2. Consider the following observations/data. And apply simple linear regression and find out estimated coefficients b0 and b1.(use numpy package)

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])

Answer:

```

import numpy as np

import matplotlib.pyplot as plt

def estimate_coef(x, y):

```

```

# number of observations/points
n = np.size(x)

# mean of x and y vector
m_x = np.mean(x)
m_y = np.mean(y)

# calculating cross-deviation and deviation about x
SS_xy = np.sum(y*x) - n*m_y*m_x
SS_xx = np.sum(x*x) - n*m_x*m_x

# calculating regression coefficients
b_1 = SS_xy / SS_xx
b_0 = m_y - b_1*m_x

return (b_0, b_1)

def plot_regression_line(x, y, b):
    # plotting the actual points as scatter plot
    plt.scatter(x, y, color = "m",
               marker = "o", s = 30)

    # predicted response vector
    y_pred = b[0] + b[1]*x

    # plotting the regression line
    plt.plot(x, y_pred, color = "g")

    # putting labels
    plt.xlabel('x')
    plt.ylabel('y')

    # function to show plot
    plt.show()

def main():
    # observations / data
    x = np.array([0,1,2,3,4,5,6,7,8,9,11,13])
    y = np.array([[1, 3, 2, 5, 7, 8, 8, 9, 10, 12,16, 18]])

    # estimating coefficients
    b = estimate_coef(x, y)

```

```

print("Estimated coefficients:\nb_0 = {} \
\nb_1 = {}".format(b[0], b[1]))
# plotting regression line
plot_regression_line(x, y, b)
if __name__ == "__main__":
    main()

```

Slip no:4

Q1. Write a R program to calculate the sum of two matrices of given size.

Answer:

```

matrix1<-matrix(c(1,2,3,4,5,6),nrow=2)
print(matrix1)
matrix2<-matrix(c(7,8,9,10,11,12),nrow=2)
print(matrix2)
result<-matrix1+matrix2
cat("Addition : ","\n")
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.

Answer:

```

# Assigning features and label variables
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']

# Import LabelEncoder
from sklearn import preprocessing

```

```

#creating labelEncoder
le = preprocessing.LabelEncoder()
# Converting string labels into numbers.
weather_encoded=le.fit_transform(weather)
print (weather_encoded)
# Converting string labels into numbers
temp_encoded=le.fit_transform(temp)
label=le.fit_transform(play)
print ("Temp:",temp_encoded)
print ("Play:",label)
#Combinig weather and temp into single listof tuples
features=zip(weather_encoded,temp_encoded)
print (features)
#Import Gaussian Naive Bayes model
from sklearn.naive_bayes import GaussianNB
#Create a Gaussian Classifier
model = GaussianNB()
# Train the model using the training sets
model.fit(features,label)
#Predict Output
predicted= model.predict([[0,2]]) # 0:Overcast, 2:Mild
print("Predicted Value:", predicted)

```

Slip no:5

Q1. Write a R program to concatenate two given factors.

Answer:

```

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-diabetesdatabase>)

Answer: import pandas as pd

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

```
from sklearn.model_selection import train_test_split
```

```
from sklearn import metrics
```

```
pima = pd.read_csv("diabetes.csv")
```

```
pima.head()
```

```
import seaborn as sns
```

```
corr = pima.corr()
```

```
ax = sns.heatmap(
```

```
corr,
```

```
vmin=-1, vmax=1, center=0,
```

```
cmap=sns.diverging_palette(20, 220, n=200),
```

```
square=True
```

```
)
```

```
ax.set_xticklabels(
```

```
ax.get_xticklabels(),
```

```
rotation=45,
```

```
horizontalalignment='right'
```

```
);
```

```
# feature selection
```

```
feature_cols = ['Pregnancies', 'Insulin', 'BMI', 'Age', 'Glucose',
```

```
'BloodPressure',
```

```
'DiabetesPedigreeFunction']
```

```
x = pima[feature_cols]
```

```
y = pima.Outcome
```

```
# split data
```

```
X_train, X_test, Y_train, Y_test = train_test_split(x, y, test_size = 0.3,
```



```

random_state=1)

# build model

classifier = DecisionTreeClassifier()

classifier = classifier.fit(X_train, Y_train)

#predict

y_pred = classifier.predict(X_test)

print(y_pred)

from sklearn.metrics import confusion_matrix

confusion_matrix(Y_test, y_pred)

print(confusion_matrix(Y_test, y_pred))

# accuracy

print("Accuracy:", metrics.accuracy_score(Y_test,y_pred))

from six import StringIO

from IPython.display import Image

from sklearn.tree import export_graphviz

import pydotplus

dot_data = StringIO()

export_graphviz(classifier, out_file=dot_data,

filled=True, rounded=True,

special_characters=True, feature_names =

feature_cols,class_names=['0','1'])

graph = pydotplus.graph_from_dot_data(dot_data.getvalue())

graph.write_png('diabetes.png')

Image(graph.create_png())

```

Slip no: 6

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

Answer:

```

companies <- data.frame(Shares = c("TCS", "Reliance", "HDFC Bank", "Infosys",
"Reliance"),
Price = c(3200, 1900, 1500, 2200, 1900))

```

companies

cat("After removing Duplicates ", "\n")

companies[duplicated(companies),]

Q2. Write a python program to implement hierarchical Agglomerative clustering algorithm.

(Download Customer.csv dataset from github.com).

Answer:

```
import numpy as np
```

```
import matplotlib.pyplot as plt
```

```
import pandas as pd
```

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

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

```
"""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)"""
```

```
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()
```

```
from sklearn.cluster import AgglomerativeClustering
```

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

```
y_hc = hc.fit_predict(X)
```

```
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()

```

Slip no:7

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

Answer:

```

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 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])

```

Answer:

```

import matplotlib.pyplot as plt
import numpy as np
from scipy import stats
x = np.array([1,2,3,4,5,6,7,8])
y = np.array([7,14,15,18,19,21,26,23])
slope, intercept, r, p, std_err = stats.linregress(x, y)
def myfunc(x):

```

```

return slope * x + intercept
mymodel = list(map(myfunc, x))
plt.scatter(x, y)
plt.plot(x, mymodel)
plt.show()

```

Slip no: 8

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

Answer:

```

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)

Answer:

```

import numpy as np
import matplotlib.pyplot as mtp
import pandas as pd
dataset = pd.read_csv('CC GENERAL.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 Elbow 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, 0], kmeans.cluster_centers_[0, 1], s = 300, c =
'yellow', label = 'Centroid')

mtp.title('Clusters of Credit Card')
mtp.xlabel('V3')
mtp.ylabel('V4')
mtp.legend()
mtp.show()

```

Slip no:9

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

Answer:

```

Employees = data.frame(Name=c("Amit S","Dikisha R","Shweta J", "Jikita A","Riya M"),
Gender=c("M","M","F","F","F"),
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.

Answer:

```
#Import scikit-learn dataset library

from sklearn import datasets

#Load dataset

cancer = datasets.load_breast_cancer()

# print the names of the 13 features

print("Features: ", cancer.feature_names)

# print the label type of cancer('malignant' 'benign')

print("Labels: ", cancer.target_names)

# print data(feature)shape

cancer.data.shape

# print the cancer data features (top 5 records)

print(cancer.data[0:5])

# print the cancer labels (0:malignant, 1:benign)

print(cancer.target)

# Import train_test_split function

from sklearn.model_selection import train_test_split

# Split dataset into training set and test set

X_train, X_test, y_train, y_test = train_test_split(cancer.data, cancer.target,

test_size=0.3,random_state=109) # 70% training and 30% test

#Import svm model

from sklearn import svm

#Create a svm Classifier

clf = svm.SVC(kernel='linear') # Linear Kernel

#Train the model using the training sets

clf.fit(X_train, y_train)

#Predict the response for test dataset

y_pred = clf.predict(X_test)

#Import scikit-learn metrics module for accuracy calculation

from sklearn import metrics

# Model Accuracy: how often is the classifier correct?
```

```
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

Slip no:10

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

Answer:

```
nums = c(10, 20, 30, 40, 50, 60)
```

```
print('Original vector:')
```

```
print(nums)
```

```
print(paste("Maximum value of the said vector:",max(nums)))
```

```
print(paste("Minimum value of the said vector:",min(nums)))
```

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

Answer:

```
"cells": [
```

```
{
```

```
"cell_type": "markdown",
```

```
"id": "b58228cb",
```

```
"metadata": {}, "source": [
```

```
"# SET - A\n",
```

```
"\n",
```

```
"### 1) Write a code to read the dataset ("Iris.csv"). dataset download from  
(https://archive.ics.uci.edu/ml/datasets/iris) and apply Apriori algorithm."
```

```
]
```

```
},
```

```
{
```

```
"cell_type": "code",
```

```
"execution_count": 1,
```

```
"id": "31f28134",
```

```
"metadata": {},
```

```
"outputs": [],
```

```
"source": [
```

```
"import numpy as np\n",
```

```
"import matplotlib.pyplot as plt\n",  
"import pandas as pd\n",  
"from apyori import apriori"  
]  
},  
{  
"cell_type": "code",  
"execution_count": null,  
"id": "91ef7af6",  
"metadata": {},  
"outputs": [],  
"source": [  
"store_data=pd.read_csv('iris.csv',header=None)"  
]  
},  
{  
"cell_type": "code",  
"execution_count": null,  
"id": "cd4c9ed9",  
"metadata": {},  
"outputs": [],  
"source": [  
"store_data.head()\n"  
]  
},  
{  
"cell_type": "code",  
"execution_count": null,  
"id": "88d01808",  
"metadata": {},  
"outputs": [],
```



```

"source": [
"records = []\n",
"for i in range(0,300):\n",
" records.append([str(store_data.values[i,j]) for j in range(0,20)])\n"
]
},
{
"cell_type": "code",
"execution_count": null,
"id": "ba30cca3",
"metadata": {},
"outputs": [],
"source": [
"association_rules=apriori(records,min_support=0.0045,min_confidence=0.2,min_lift=3,min\n\n_length=2)\n",
"association_results=list(association_rules)\n"
]
},
{
"cell_type": "code",
"execution_count": null,
"id": "8ab0102a",
"metadata": {},
"outputs": [],
"source": [
"print(len(association_results))\n"
]
},
{
"cell_type": "code",
"execution_count": null,

```

```

"id": "daa923d5",
"metadata": {},
"outputs": [],
"source": [
    "print(association_results[0])\n"
]
},
{
    "cell_type": "code",
    "execution_count": null,
    "id": "4f9ceaad",
    "metadata": {},
    "outputs": [],
    "source": [
        "for item in association_results:\n",
        " pair = item[0]\n",
        " items = [x for x in pair]\n",
        " print(\"Rule:\"+items[0]+\"->\"+items[1])\n",
        " \n",
        " print(\"Support:\"+str(item[1]))\n",
        " \n",
        " print(\"Confidence:\"+str(item[2][0][2]))\n",
        " print(\"Lift:\"+str(item[2][0][3]))\n",
        " print(\"=====\\")"
    ]
},
{
    "metadata": {
        "kernel_spec": {
            "display_name": "Python 3 (ipykernel)",
            "language": "python",

```

```

"name": "python3"
},
"language_info": {
"codemirror_mode": {
"name": "ipython",
"version": 3
},
"file_extension": ".py",
"mimetype": "text/x-python",
"name": "python",
"nbconvert_exporter": "python",
"pygments_lexer": "ipython3",
"version": "3.7.9"
}
},
"nbformat": 4,
"nbformat_minor": 5
}

```

Slip no:11

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

```

AB= list("x", "y", "z")
= list("X", "Y", "Z", "x", "y", "z")

```

Answer:

```

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).

Answer:

[https://github.com/TrainingByPackt/Data-](https://github.com/TrainingByPackt/Data-Science-with-Python/blob/master/Chapter01/Data/Wholesale%20customers%20data.csv)

[Science-with-Python/blob/master/Chapter01/Data/Wholesale%20customers%20data.csv](https://github.com/TrainingByPackt/Data-Science-with-Python/blob/master/Chapter01/Data/Wholesale%20customers%20data.csv)

```
import matplotlib.pyplot as mtp
```

```
import pandas as pd
```

```
dataset = pd.read_csv('Wholesalecustomer.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 no: 12

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

details.

Employee contain (empno,empname,gender,age,designation)

Answer:

```
Employees = data.frame(empno=c(1,2,3,4,5),
empname=c("Amit S","Dikish R","Shweta J", "Jikita A","Riya M"),
Gender=c("M","M","F","F","F"),
Age=c(23,22,25,26,32),
Designation=c("Clerk","Manager","Exective","CEO","ASSISTANT"))
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.as

Answer:

```
import pandas
from sklearn import linear_model
df = pandas.read_csv("car.csv")
X = df[['Weight', 'Volume']]
y = df['CO2']
regr = linear_model.LinearRegression()
regr.fit(X, y)
#predict the CO2 emission of a car where the weight is 2300kg, and the volume is 1300cm3:
predictedCO2 = regr.predict([[2300, 1300]])
print(predictedCO2)
```

Slip no: 13

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

Answer:

Create data for the graph.

```
digits <- c(7,2,6,3,4,8)
```

```
Frequency <- c(1,2,3,4,5,6)
```

Plot the chart.

```
pie(digits, Frequency)
```

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

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

Answer:

```
{
  "nbformat": 4,
  "nbformat_minor": 0,
  "metadata": {
    "colab": {
      "name": "Data Mining Assignment-3 SET-B-1.ipynb",
      "provenance": []
    },
    "kernelspec": {
      "name": "python3",
      "display_name": "Python 3"
    },
    "language_info": {
      "name": "python"
    }
  },
  "cells": [
    {
      "cell_type": "markdown",
      "source": [
        "### SET-B\n",
        "\n",

```

"1) Write a Python program to read \"StudentsPerformance.csv\" file. solve the

```

following :\n",

"- To display the shape of dataset.\n",

"- To display the top rows of the dataset with their columns.\n",

"- To display the number of rows randomly.\n",

"- To display the number of columns and names of the columns.\n",

"- Note : Download dataset from following link:\n",

"(https://www.kaggle.com/spscientist/students-
performanceinexams?select=StudentsPerformance.csv)"

],

"metadata": {

"id": "0hhW5uEs_wK2"

}

},

{

"cell_type": "code",

"source": [

"# Import required libraries\n",

"import numpy as np\n",

"import matplotlib.pyplot as plt\n",

"import pandas as pd\n",

],

},

{

"cell_type": "code",

"source": [

"# Read the downloaded dataset\n",

"store_data=pd.read_csv('StudentsPerformance.csv',header=None)"

],

"metadata": {

"id": "uC2jGgIFFVa3"

},

```

```
"execution_count": null,
"outputs": []
},
{
"cell_type": "code",
"source": [
"# To display the shape of dataset. (By Using shape method)\n",
"store_data.shape"
],
"metadata": {
"id": "wU6-JdtCF3ar"
},
"execution_count": null,
"outputs": []
},
{
"cell_type": "code",
"source": [
"# To display the top rows of the dataset with their columns.(By using head method\n",
"store_data.head()"
],
"metadata": {
"id": "xHtDSrSsGT2v"
},
"execution_count": null,
"outputs": []
},
{
"cell_type": "code",
"source": [
"# To display the number of rows randomly.(By using sample method)\n",
```



```

"store_data.sample(10)"
],
"metadata": {
  "id": "2Gwsi4oTG9QN"
},
"execution_count": null,
"outputs": []
},
{
  "cell_type": "code",
  "source": [
    "# To display the number of columns and names of the columns. (By using columns
    method)\n",
    "store_data.columns()"
  ],
  "metadata": {
    "id": "ZdXc3aoUHO80"
  },
  "execution_count": null,
  "outputs": []
}
]
}

```

Slip no: 14

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

- Display names of employees in the list.
- Add an employee at the end of the list
- Remove the third element of the list

Answer:

```

list_data <- list("Ram Sharma", "Sham Varma", "Raj Jadhav", "Ved Sharma")
print(list_data)

```

```

new_Emp <- "Kavya Anjali"
list_data <- append(list_data, new_Emp)
print(list_data)
list_data[3] <- NULL
print(list_data)

```

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.

Answer

```

import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from apyori import apriori

store_data = pd.read_csv('Market_Basket_Optimisation.csv', header=None)
store_data.head()

records = []

for i in range(0, 7501):
    records.append([str(store_data.values[i, j]) for j in range(0, 20)])

association_rules = apriori(records, min_support=0.0045, min_confidence=0.2, min_lift=3,
max_length=None)

association_results = list(association_rules)

print(len(association_results))

48

for item in association_results:

    # first index of the inner list
    # Contains base item and add item
    pair = item[0]

    items = [x for x in pair]

    print("Le:", items)

    print("Rule: " + items[0] + " -> " + items[1])

```

```

#second index of the inner list
print("Support: " + str(item[1]))

#third index of the list located at 0th
#of the third index of the inner list
print("Confidence: " + str(item[2][0][2]))
print("Lift: " + str(item[2][0][3]))
print("=====")

```

Slip no:15

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

Answer:

```

vector1 = seq(10,40 , length.out=4)
vector2 = c(20, 10, 40, 40)
print("Original Vectors:")
print(vector1)
print(vector2)
add= vector1+vector2
cat("Sum of vector is ",add, "\n")
sub_vector= vector1-vector2
cat("Substraction of vector is ",sub_vector, "\n")
mul_vector= vector1 * vector2
cat("Multiplication of vector is ",mul_vector, "\n")
print("Division of two Vectors:")
div_vector = vector1 / vector2
print(div_vector)

```

Q2. Write a Python program build Decision Tree Classifier for shows.csv from 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

Answer:

```
import pandas
```

```

from sklearn import tree
from sklearn.tree import DecisionTreeClassifier
import matplotlib.pyplot as plt
df = pandas.read_csv("Age.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']
print(X)
print(y)

```

Slip no:16

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

Answer:

```

# Import lattice
library(lattice)

# Create data
gfg <- data.frame(x = c(26,35,32,40,35,50),
  grp = rep(c("group 1", "group 2",
    "group 3"),
    each = 2),
  subgroup = LETTERS[1:2])

# Create grouped barplot using lattice
barchart(x ~ grp, data = gfg, groups = subgroup)

```

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-diabetesdatabase>)

Answer:

```

import pandas as pd
from sklearn.tree import DecisionTreeClassifier

```

```
from sklearn.model_selection import train_test_split

from sklearn import metrics

pima = pd.read_csv("diabetes.csv")

pima.head()

import seaborn as sns

corr = pima.corr()

ax = sns.heatmap(

    corr,

    vmin=-1, vmax=1, center=0,

    cmap=sns.diverging_palette(20, 220, n=200),

    square=True

)

ax.set_xticklabels(

    ax.get_xticklabels(),

    rotation=45,

    horizontalalignment='right'

);

# feature selection

feature_cols = ['Pregnancies', 'Insulin', 'BMI', 'Age', 'Glucose', 'BloodPressure',

'DiabetesPedigreeFunction']

x = pima[feature_cols]

y = pima.Outcome

# split data

X_train, X_test, Y_train, Y_test = train_test_split(x, y, test_size = 0.3,

random_state=1)

# build model

classifier = DecisionTreeClassifier()

classifier = classifier.fit(X_train, Y_train)

#predict

y_pred = classifier.predict(X_test)

print(y_pred)
```

```

from sklearn.metrics import confusion_matrix
confusion_matrix(Y_test, y_pred)
print(confusion_matrix(Y_test, y_pred))

# accuracy
print("Accuracy:", metrics.accuracy_score(Y_test,y_pred))

from six import StringIO
from IPython.display import Image
from sklearn.tree import export_graphviz
import pydotplus

dot_data = StringIO()
export_graphviz(classifier, out_file=dot_data,
filled=True, rounded=True,
special_characters=True, feature_names =
feature_cols,class_names=['0','1'])
graph = pydotplus.graph_from_dot_data(dot_data.getvalue())
graph.write_png('diabetes.png')
Image(graph.create_png())

```

Slip no:17

Q1. Write a R program to get the first 20 Fibonacci numbers.

Answer:

```

Fibonacci <- numeric(20)
Fibonacci[1] <- Fibonacci[2] <- 1
for (i in 3:20) Fibonacci[i] <- Fibonacci[i - 2] + Fibonacci[i - 1]
print("First 20 Fibonacci numbers:")
print(Fibonacci)

```

Q2. Write a python programme to implement multiple linear regression model for stock market data frame as follows:

```

Stock_Market = {'Year': [2017,2017,2017,2017,2017,2017,2017,2017,2017,2017,2017,2017,2016,2016,2016,2016,2016,2016,2016,2016,2016],
'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

```

```
'Unemployment_Rate': [5.3,5.3,5.3,5.3,5.4,5.6,5.5,5.5,5.5,5.6,5.7,5.9,6.5,5.9,5.8,6.1,6.2,6.1,6.1,6.1,5.9,6.2,6.2,6.1],
```

And draw a graph of stock market price verses interest rate.

```
plt.show()
```

Slip no: 18

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

Answer:

```
nums = c(10, 20, 30, 40, 50, 60)
print('Original vector:')
print(nums)
print(paste("Maximum value of the said vector:",max(nums)))
print(paste("Minimum value of the said vector:",min(nums)))
```

Q2. Consider the following observations/data. And apply simple linear regression and find out estimated coefficients b1 and b0. 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])
```

Answer:

```
import matplotlib.pyplot as plt
import numpy as np
from scipy import stats
x = np.array([1,2,3,4,5,6,7,8])
y = np.array([7,14,15,18,19,21,26,23])
slope, intercept, r, p, std_err = stats.linregress(x, y)
def myfunc(x):
    return slope * x + intercept
mymodel = list(map(myfunc, x))
plt.scatter(x, y)
plt.plot(x, mymodel)
plt.show()
```

Slip no: 19

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

Students contain (Rollno,Studname,Address,Marks)

Answer:


```

Students = data.frame(Rollno=c(21,22,23,24,25),
Name=c("Riya M","Shweta J","Aarya D", "JAMES A","LAURA M"),
Address=c("Bhekrai nagar","Hadapsar","Uruli kanchan","Hadapsar","Bhekrai nagar"),
Marks=c(80,67,90,92,70))
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

Answer:

```

import pandas
from sklearn import linear_model
from sklearn.linear_model import LinearRegression
df = pandas.read_csv("car.csv")
X = df[['Weight', 'Volume']]
y = df['CO2']
regr = linear_model.LinearRegression()
regr.fit(X, y)
test_y = regr.predict(X)
#predict the CO2 emission of a car where the weight is 2300kg, and the volume is 1300cm3:
predictedCO2 = regr.predict([[2300, 1300]])
print(predictedCO2)

```

Slip no: 20

Q1. Write a R program to create a data frame from four given vectors.

Answer:

```

name = c('Aarya', 'Riya', 'Shweta', 'Anjali', 'Geeta', 'Mayuri', 'Kirti', 'Akansha', 'Kavita', 'Jagruti')
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).

Answer:

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

# Importing the dataset
dataset = pd.read_csv('Customer.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()
```