PRACTICAL NUMBER:4A

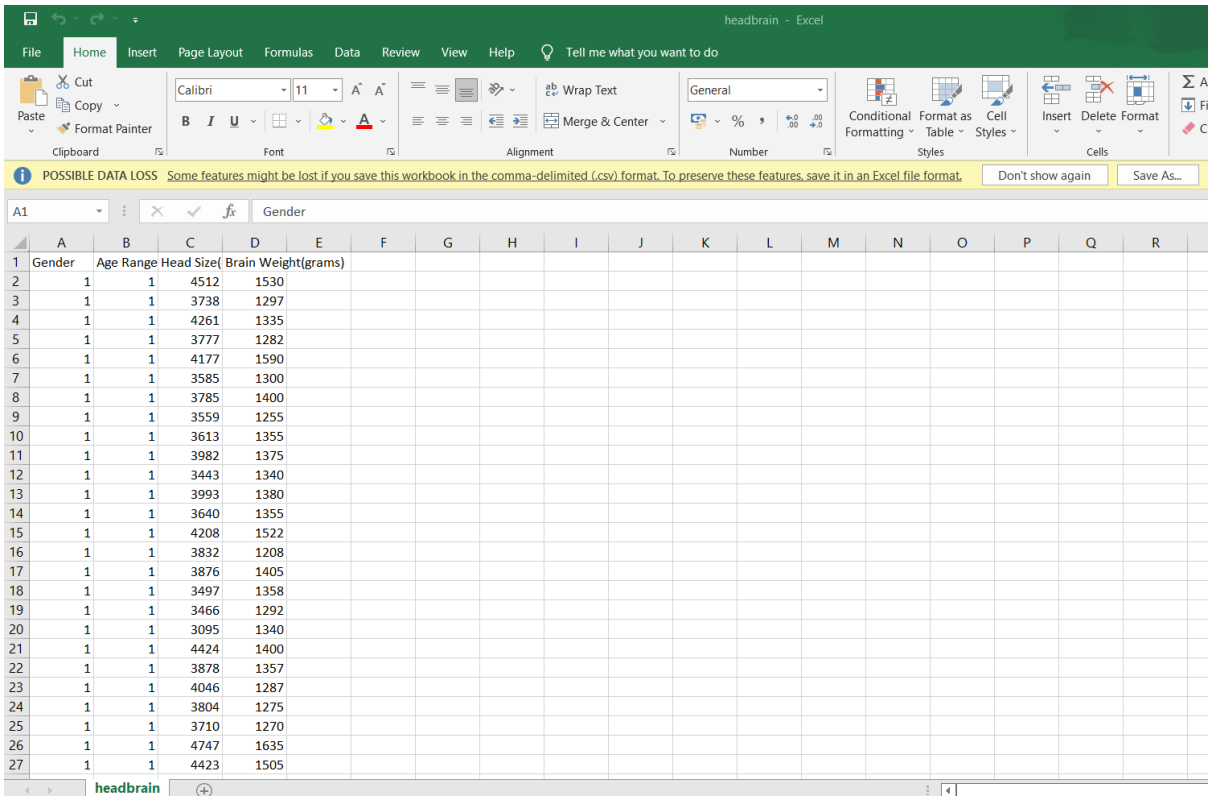
Aim: For a given set of training data examples stored in a .CSV file implement Least Square Regression algorithm.

Code:

```
#Import the required libraries

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

#Import the data set
#Reading Data
data = pd.read_csv('headbrain.csv')
#With the shape method, comes the flexibility to obtain the dimensions of any Python object.
print("Dimensions:",data.shape)
#The head function in Python displays the first five rows of the dataframe by default.
print("First 5 rows:",data.head())

#Assigning ‘X’ as independent variable and ‘Y’ as dependent variable
# Coomputing X and Y
X = data['Head Size(cm^3)'].values
Y = data['Brain Weight(grams)'].values

#Next, in order to calculate the slope and y-intercept we first need to compute the means of ‘x’ and ‘y’.
#Mean X and Y
mean_x = np.mean(X)
mean_y = np.mean(Y)
# Total number of values
n = len(X)
#Calculate the values of the slope and y-intercept
#Using the formula to calculate 'm' and 'c'
numer = 0
denom = 0
for i in range(n):
   numer += (X[i] - mean_x) * (Y[i] - mean_y)
   denom += (X[i] - mean_x) ** 2
m = numer / denom
c = mean_y - (m * mean_x)

#Printing coefficients
print("Coefficients")
print(m, c)
```

```
#Plotting the line of best fit
#Plotting Values and Regression Line

max_x = np.max(X) + 100
min_x = np.min(X) - 100

#Calculating line values x and y
x = np.linspace(min_x, max_x, 1000)
y = c + m * x

#Ploting Line
plt.plot(x, y, color='#58b970', label='Regression Line')
#Ploting Scatter Points
plt.scatter(X, Y, c='#ef5423', label='Scatter Plot')

plt.xlabel('Head Size in cm3')
plt.ylabel('Brain Weight in grams')
plt.legend()
plt.show()
```

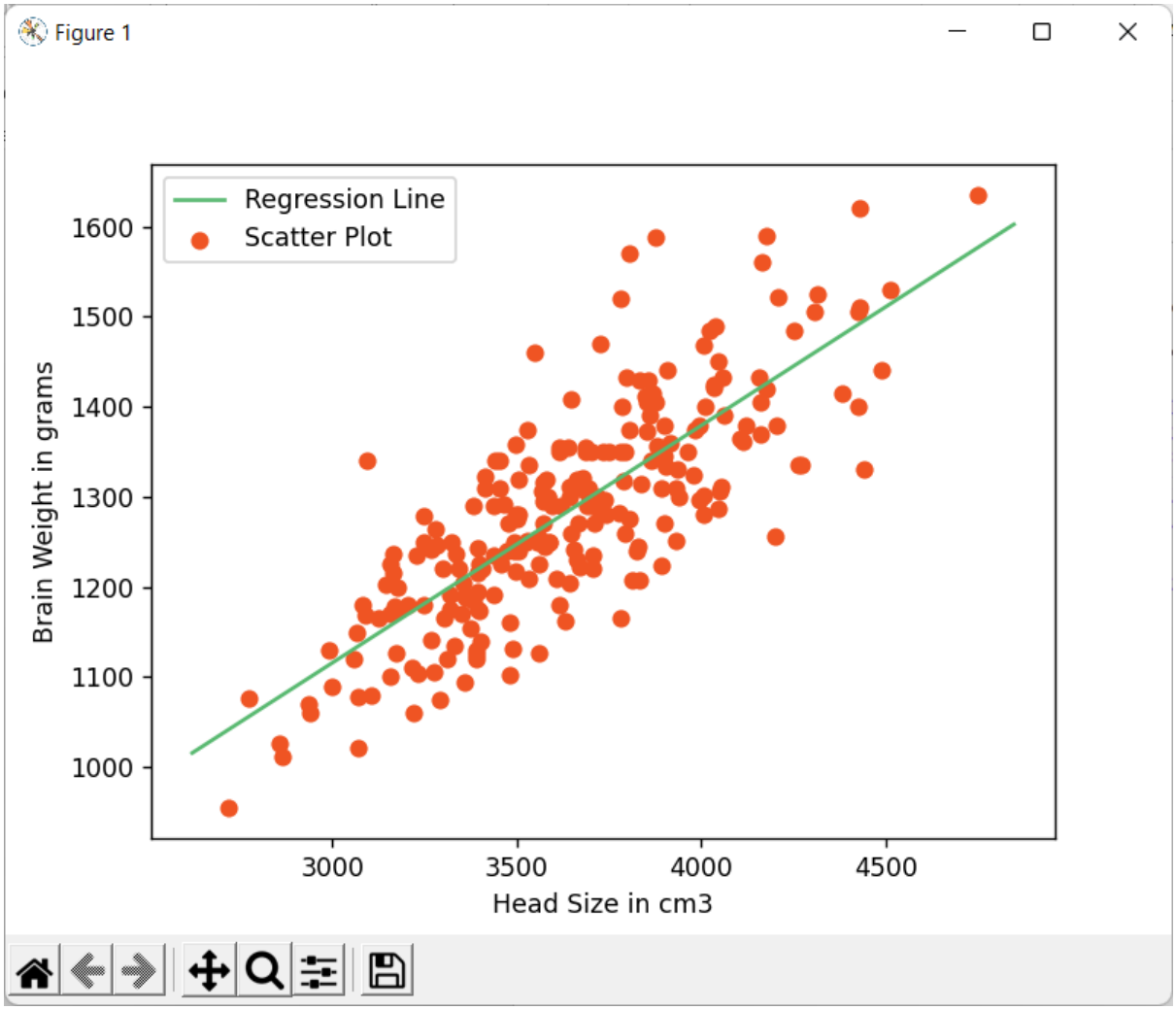
headbrain.csv



	A	B	C	D
1	Gender	Age Range	Head Size(	Brain Weight(grams)
2	1	1	4512	1530
3	1	1	3738	1297
4	1	1	4261	1335
5	1	1	3777	1282
6	1	1	4177	1590
7	1	1	3585	1300
8	1	1	3785	1400
9	1	1	3559	1255
10	1	1	3613	1355
11	1	1	3982	1375
12	1	1	3443	1340
13	1	1	3993	1380
14	1	1	3640	1355
15	1	1	4208	1522
16	1	1	3832	1208
17	1	1	3876	1405
18	1	1	3497	1358
19	1	1	3466	1292
20	1	1	3095	1340
21	1	1	4424	1400
22	1	1	3878	1357
23	1	1	4046	1287
24	1	1	3804	1275
25	1	1	3710	1270
26	1	1	4747	1635
27	1	1	4423	1505

Output:

```
*Python 3.7.9 Shell*
File  Edit  Shell  Debug  Options  Window  Help
Python 3.7.9 (tags/v3.7.9:13c94747c7, Aug 17 2020, 18:58:18) [MSC v.1900 64 bit
(AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
================== RESTART: D:/MSc IT/Sem3/ML Pracs/Prac4A.py ==================
Dimensions: (237, 4)
First 5 rows:    Gender  Age Range  Head Size(cm^3)  Brain Weight(grams)
0       1          1             4512                 1530
1       1          1             3738                 1297
2       1          1             4261                 1335
3       1          1             3777                 1282
4       1          1             4177                 1590
Coefficients
0.26342933948939945 325.57342104944223
```



PRACTICAL NUMBER:4B

Aim: For a given set of training data examples stored in a .CSV file implement Logistic Regression algorithm.

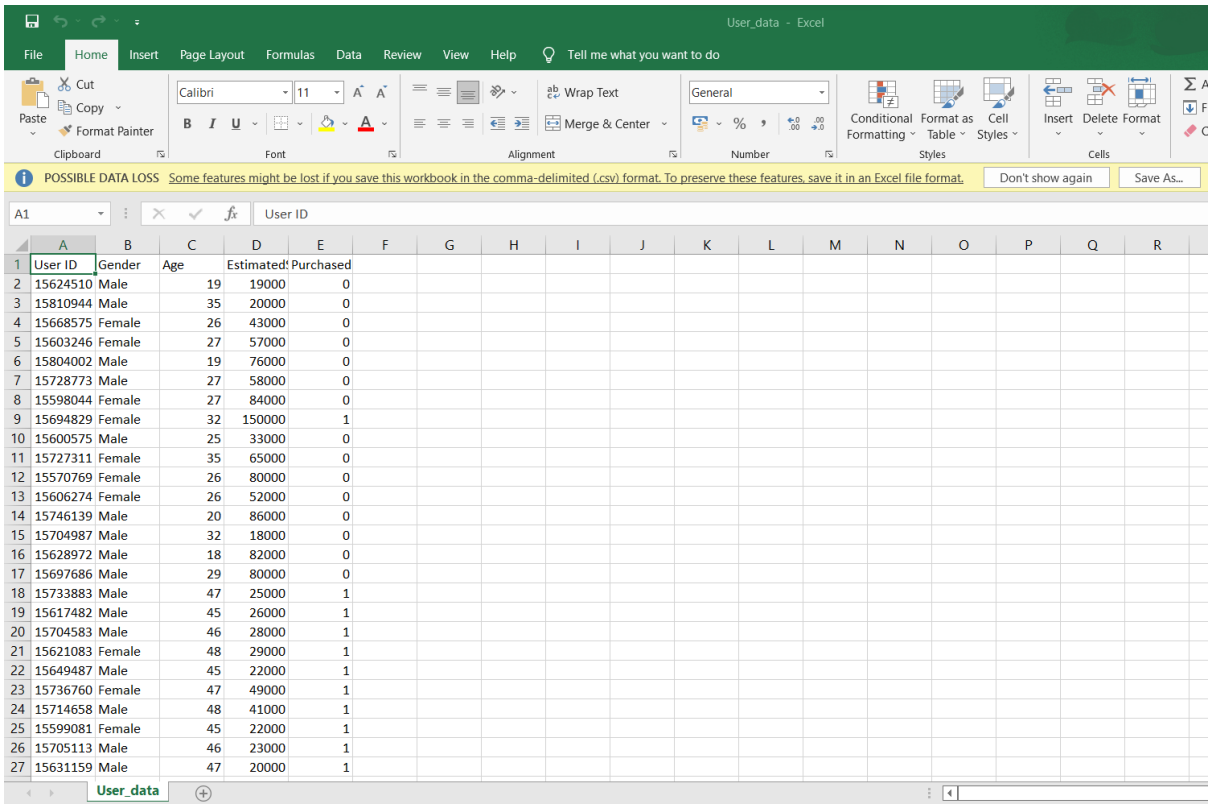
Code:

```
#Importing libraries

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split

from sklearn.preprocessing import StandardScaler

from sklearn.linear_model import LogisticRegression

from sklearn.metrics import confusion_matrix

from sklearn.metrics import accuracy_score

from matplotlib.colors import ListedColormap


#Reading the data

dataset = pd.read_csv('User_Data.csv')

#Now, to predict whether a user will purchase the product or not, one needs to find out the relationship between Age and Estimated Salary

# input

x = dataset.iloc[:, [2, 3]].values

# output

y = dataset.iloc[:, 4].values


#Splitting the dataset to train and test. 75% of data is used for training the model and 25% of it is used to test the performance of our model.

xtrain, xtest, ytrain, ytest = train_test_split(x, y, test_size = 0.25, random_state = 0)


#Performing feature scaling

sc_x = StandardScaler()
```

```
xtrain = sc_x.fit_transform(xtrain)
xtest = sc_x.transform(xtest)
print (xtrain[0:10, :])

#Training The Model
classifier = LogisticRegression(random_state = 0)
classifier.fit(xtrain, ytrain)

#Performing predictions on test data
Y_prediction = classifier.predict(xtest)

#Testing the performance of model using confusion matrix
co_mat = confusion_matrix(ytest, Y_prediction)
print ("Confusion Matrix : \n", co_mat)

#Accuracy of model
print ("Accuracy : ", accuracy_score(ytest, Y_prediction))

#Visualizing the performance of our model.
X_set, y_set = xtest, ytest
X1, X2 = np.meshgrid(np.arange(start = X_set[:, 0].min() - 1,
stop = X_set[:, 0].max() + 1, step = 0.01),
np.arange(start = X_set[:, 1].min() - 1,
stop = X_set[:, 1].max() + 1, step = 0.01))

plt.contourf(X1, X2, classifier.predict(
np.array([X1.ravel(), X2.ravel()]).T).reshape(
X1.shape), alpha = 0.75, cmap = ListedColormap(('red', 'green')))
```

```
plt.xlim(X1.min(), X1.max())

plt.ylim(X2.min(), X2.max())

for i, j in enumerate(np.unique(y_set)):

    plt.scatter(X_set[y_set == j, 0], X_set[y_set == j, 1], c = ListedColormap(('red', 'green'))(i), label = j)

plt.title('Classifier (Test set)')

plt.xlabel('Age')

plt.ylabel('Estimated Salary')

plt.legend()

plt.show()
```

User_data.csv



User_data - Excel

File Home Insert Page Layout Formulas Data Review View Help Tell me what you want to do

POSSIBLE DATA LOSS Some features might be lost if you save this workbook in the comma-delimited (.csv) format. To preserve these features, save it in an Excel file format. Don't show again Save As...

A1 User ID

	A	B	C	D	E
1	User ID	Gender	Age	Estimated	Purchased
2	15624510	Male	19	19000	0
3	15810944	Male	35	20000	0
4	15668575	Female	26	43000	0
5	15603246	Female	27	57000	0
6	15804002	Male	19	76000	0
7	15728773	Male	27	58000	0
8	15598044	Female	27	84000	0
9	15694829	Female	32	150000	1
10	15600575	Male	25	33000	0
11	15727311	Female	35	65000	0
12	15570769	Female	26	80000	0
13	15606274	Female	26	52000	0
14	15746139	Male	20	86000	0
15	15704987	Male	32	18000	0
16	15628972	Male	18	82000	0
17	15697686	Male	29	80000	0
18	15733883	Male	47	25000	1
19	15617482	Male	45	26000	1
20	15704583	Male	46	28000	1
21	15621083	Female	48	29000	1
22	15649487	Male	45	22000	1
23	15736760	Female	47	49000	1
24	15714658	Male	48	41000	1
25	15599081	Female	45	22000	1
26	15705113	Male	46	23000	1
27	15631159	Male	47	20000	1

User_data

Output:

```
*Python 3.7.9 Shell*
File  Edit  Shell  Debug  Options  Window  Help
Python 3.7.9 (tags/v3.7.9:13c94747c7, Aug 17 2020, 18:58:18) [MSC v.1900 64 bit
(AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
================== RESTART: D:/MSc IT/Sem3/ML Pracs/Prac4b1.py ==================
[[ 0.58164944 -0.88670699]
 [-0.60673761  1.46173768]
 [-0.01254409 -0.5677824 ]
 [-0.60673761  1.89663484]
 [ 1.37390747 -1.40858358]
 [ 1.47293972  0.99784738]
 [ 0.08648817 -0.79972756]
 [-0.01254409 -0.24885782]
 [-0.21060859 -0.5677824 ]
 [-0.21060859 -0.19087153]]
Confusion Matrix :
 [[65  3]
 [ 8 24]]
Accuracy :  0.89
```

1. **Explain Least Square Regression Algorithm**
2. **Explain Logistic Algorithm.**