ASSIGNMENT # 2

Perform linear or polynomial Regression is upto you on the Given dataset and predict brain weight from head size

```
# Simple Linear Regression

# Importing the libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

# Importing the dataset
record = pd.read_csv('dataset.csv')
head_size = record.iloc[:,2:3].values
brain_wt = record.iloc[:, 3].values


# Splitting the dataset into the Training set and Test set
from sklearn.model_selection import train_test_split
head_size_train, head_size_test, brain_wt_train, brain_wt_test = train_test_split(head_size, brain_wt, test_size = 1/3, random_state = 0)

# Fitting Simple Linear Regression to the Training set
from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
regressor.fit(head_size_train, brain_wt_train)
```

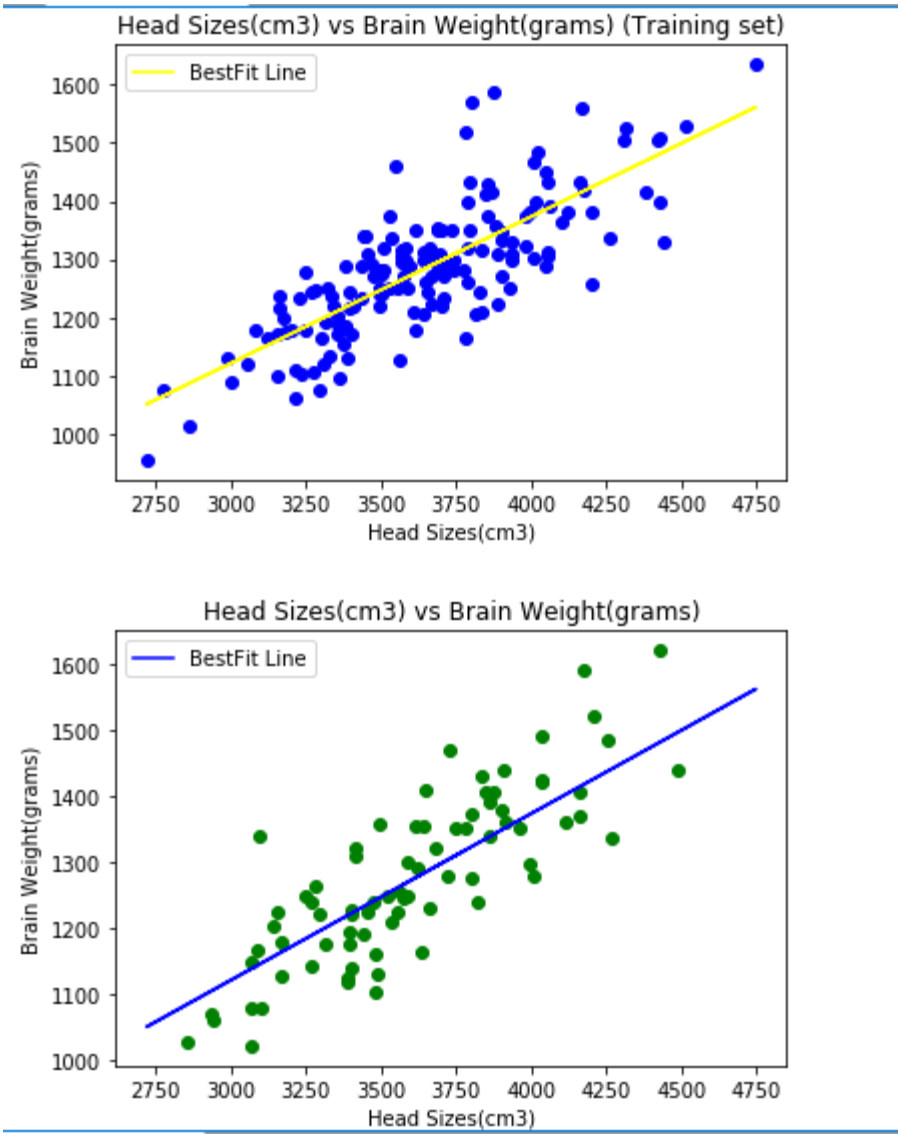
```
# Fitting Simple Linear Regression to the Training set
from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
regressor.fit(head_size_train, brain_wt_train)

# Predicting the Test set results
brain_wt_pred = regressor.predict(head_size_test)

# Visualising the Training set results
plt.scatter(head_size_train, brain_wt_train, color = 'blue')
plt.plot(head_size_train, regressor.predict(head_size_train), color = 'yellow', label= 'BestFit Line')
plt.title('Head Sizes(cm3) vs Brain Weight(grams) (Training set)')
plt.xlabel('Head Sizes(cm3)')
plt.ylabel('Brain Weight(grams)')
plt.legend()
plt.show()

# Visualising the Test set results
plt.scatter(head_size_test, brain_wt_test, color = 'green')
plt.plot(head_size_train, regressor.predict(head_size_train), color = 'blue', label= 'BestFit Line')
plt.title('Head Sizes(cm3) vs Brain Weight(grams)')
plt.xlabel('Head Sizes(cm3)')
plt.ylabel('Brain Weight(grams)')
plt.legend()
plt.show()
```

Name	Type	Size	Value
brain_wt	int64	(237,)	[1530 1297 1335 ... 1104 1170 1120]
brain_wt_pred	float64	(79,)	[1303.83322923 1292.73537163 1381.5182324 ... 1105.3329127 1363.8625 ...
brain_wt_test	int64	(79,)	[1280 1321 1425 ... 1070 1350 1522]
brain_wt_train	int64	(158,)	[1282 1165 1635 ... 1270 1215 1316]
head_size	int64	(237, 1)	[[4512] [3738]]
head_size_test	int64	(79, 1)	[[3724] [3680]]
head_size_train	int64	(158, 1)	[[3777] [3302]]
record	DataFrame	(237, 4)	Column names: Gender, Age Range, Head Size(cm^3), Brain Weight(grams)



CLASS TASK:

CREATE TWO RANDOM ARRAYS A AND B, AND MULTIPLY THEM. GET THEIR RESULT IN C AND ADD 1 TGO EVERY ELEMENT OF C.

```
import numpy as np
#Creating Random Arrays
a= np.random.randn(3,4)
b= np.random.randn(3,4)
print(" Array A")
print(a)
print(" Array B")
print(b)

#Multiplication
print ("Multiplication of two Arrays: ")
c=np.multiply(a,b)
print(c)

#Defining an Arrray
ones=np.ones((3,4))

#Adding Array 'one' to Array c
result= np.add(ones,c)
print(result)
```

```
 Array A
[[-0.56250916  0.86927162  0.69352436  0.13546132]
 [-1.64977301 -0.15369153  0.93053903  0.64207115]
 [ 0.25091951 -0.12022381  0.85429112 -1.16715981]]
 Array B
[[-0.13013601  0.64671916 -0.84040676 -1.10920176]
 [-0.1725768   0.51389745  0.92838344  1.14345089]
 [-0.45123261 -0.16945984  0.47716349 -0.5878841 ]]
Multiplication of two Arrays:
[[ 0.0732027   0.56217461 -0.58284256 -0.15025393]
 [ 0.28471255 -0.07898169  0.86389702  0.73417683]
 [-0.11322306  0.02037311  0.40763653  0.68615469]]
[[1.0732027  1.56217461 0.41715744 0.84974607]
 [1.28471255 0.92101831 1.86389702 1.73417683]
 [0.88677694 1.02037311 1.40763653 1.68615469]]
```