EXPERIMENT -5

A python program to implement multi - layer perceptron with back propagation

AIM:

A python program to implement Simple linear regression using Least Square Method

CODE:

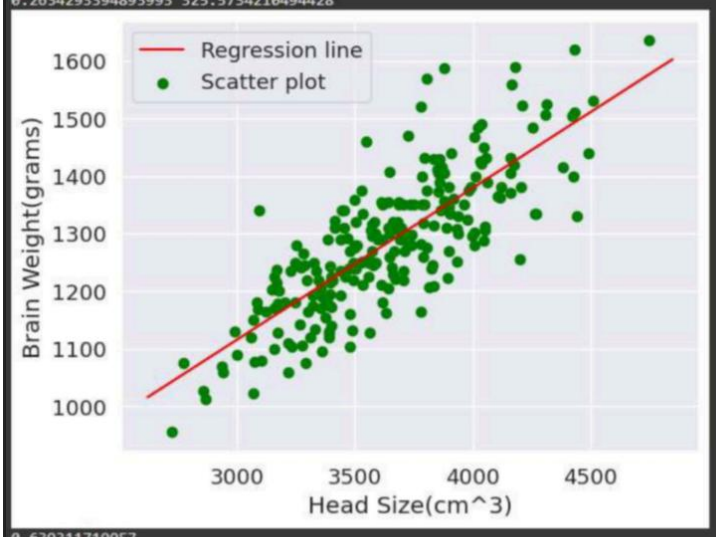
```
import pandas as pd
import numpy as np
bnotes = pd.read_csv('/content/BankNote_Authentication.csv')
bnotes.head(10)


x = bnotes.drop('class',axis=1)
y = bnotes['class']
print(x.head(2))
print(y.head(2))


from sklearn.model_selection import train_test_split
#train_test ratio = 0.2
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.2)
from sklearn.neural_network import MLPClassifier
# activation function : relu
mlp = MLPClassifier(max_iter=500,activation='relu')
mlp.fit(x_train,y_train)
MLPClassifier(max_iter=500)
pred = mlp.predict(x_test)
print(pred)


from sklearn.metrics import classification_report, confusion_matrix
```

```
confusion_matrix(y_test,pred)
print(classification_report(y_test,pred))

# activation function : logistic
mlp = MLPClassifier(max_iter=500,activation='logistic')
mlp.fit(x_train,y_train)

MLPClassifier(activation='logistic', max_iter=500)
pred = mlp.predict(x_test)
print(pred)

from sklearn.metrics import classification_report, confusion_matrix
confusion_matrix(y_test,pred)
print(classification_report(y_test,pred))
mlp = MLPClassifier(max_iter=500,activation='tanh')
mlp.fit(x_train,y_train)
pred = mlp.predict(x_test)
print(pred)

from sklearn.metrics import classification_report, confusion_matrix
confusion_matrix(y_test,pred)
print(classification_report(y_test,pred))
# activation function : identity
mlp = MLPClassifier(max_iter=500,activation='identity')
mlp.fit(x_train,y_train)
MLPClassifier(activation='identity', max_iter=500)
pred = mlp.predict(x_test)
print(pred)

from sklearn.metrics import classification_report,confusion_matrix
confusion_matrix(y_test,pred)

print(classification_report(y_test,pred))

#train_test ratio = 0.3
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.3)
from sklearn.neural_network import MLPClassifier
# activation function : relu
mlp = MLPClassifier(max_iter=500,activation='relu')
mlp.fit(x_train,y_train)
MLPClassifier(max_iter=500)
pred = mlp.predict(x_test)
print(pred)
```

```
from sklearn.metrics import classification_report,confusion_matrix
confusion_matrix(y_test,pred)
print(classification_report(y_test,pred))
# activation function : logistic
mlp = MLPClassifier(max_iter=500,activation='logistic')
mlp.fit(x_train,y_train)
MLPClassifier(max_iter=500,activation='logistic')
pred = mlp.predict(x_test)
print(pred)
MLPClassifier(max_iter=500,activation='tanh')
# activation function : tanh
mlp = MLPClassifier(max_iter=500,activation='tanh')
mlp.fit(x_train,y_train)
pred = mlp.predict(x_test)
print(pred)
from sklearn.metrics import classification_report,confusion_matrix
confusion_matrix(y_test,pred)
print(classification_report(y_test,pred))
# activation function : identity
mlp = MLPClassifier(max_iter=500,activation='identity')
mlp.fit(x_train,y_train)
MLPClassifier(max_iter=500,activation='identity')
pred = mlp.predict(x_test)
print(pred)
from sklearn.metrics import classification_report,confusion_matrix
confusion_matrix(y_test,pred)
print(classification_report(y_test,pred))
```

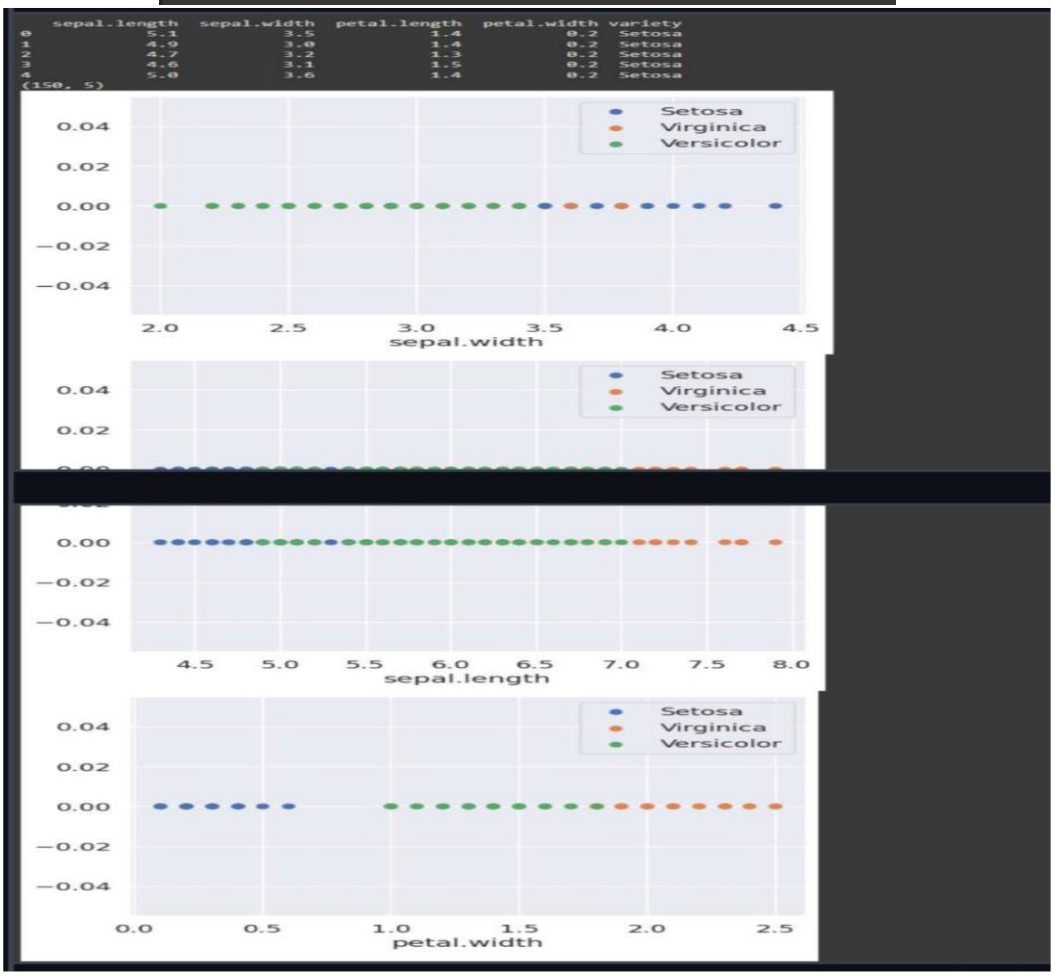
OUTPUT

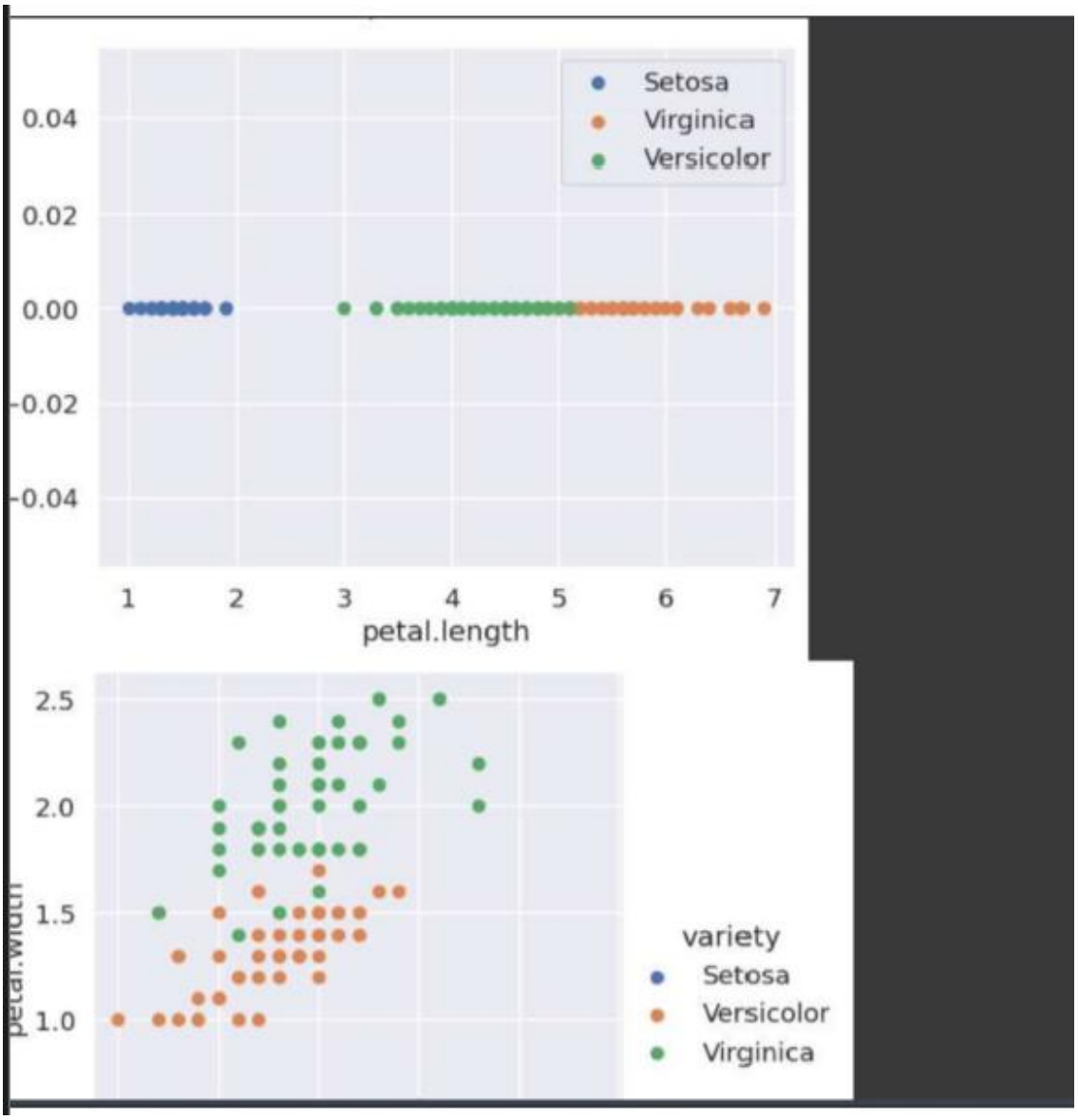
```
[4512 3738 4261 3777 4177] [1530 1297 1335 1282 1590]
3633.9915611814345 1282.873417721519
0.2634293394893993 325.5734210494428
```




```
0.639311719957
0.639311719957
```

```
   sepal.length  sepal.width  petal.length  petal.width variety
0           5.1          3.5           1.4          0.2  Setosa
1           4.9          3.0           1.4          0.2  Setosa
2           4.7          3.2           1.3          0.2  Setosa
3           4.6          3.1           1.5          0.2  Setosa
4           5.0          3.6           1.4          0.2  Setosa
(150, 5)
```




Setosa
Virginica
Versicolor
0.04
0.02
0.00
-0.02
-0.04
1
2
3
4
5
6
7
petal.length
2.5
2.0
1.5
1.0
variety
Setosa
Versicolor
Virginica

```
    User ID  Gender  Age  EstimatedSalary  Purchased
0  15624510    Male   19            19000          0
1  15810944    Male   35            20000          0
2  15668575  Female   26            43000          0
3  15603246  Female   27            57000          0
4  15804002    Male   19            76000          0
[[-1.05714987  0.53420426]
 [ 0.2798728  -0.51764734]
 [-1.05714987  0.41733186]
 [-0.29313691 -1.45262654]
 [ 0.47087604  1.23543867]
 [-1.05714987 -0.34233874]
 [-0.10213368  0.30045946]
 [ 1.33039061  0.59264046]
 [-1.15265148 -1.16044554]
 [ 1.04388575  0.47576806]]
[0 0 0 0 0 0 1 0 1 0 0 0 0 0 0 0 1 0 0 1 0 1 0 1 0 0 0 0 0 1 0 0 0 0
 0 0 1]
Confusion Matrix :
 [[31  1]
 [ 1  7]]
Accuracy :  0.95
0.7583333333333334
0.823529411764706
Accuracy is :  0.925
```

```
Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       148
           1       1.00      1.00      1.00       127

    accuracy                           1.00       275
   macro avg       1.00      1.00      1.00       275
weighted avg       1.00      1.00      1.00       275



--- Activation: identity, Test size: 0.2 ---

Confusion Matrix:
[[146   2]
 [  2 125]]

Classification Report:
              precision    recall  f1-score   support

           0       0.99      0.99      0.99       148
           1       0.98      0.98      0.98       127

    accuracy                           0.99       275
   macro avg       0.99      0.99      0.99       275
weighted avg       0.99      0.99      0.99       275



--- Activation: relu, Test size: 0.3 ---

Confusion Matrix:
[[229   0]
 [  0 183]]

Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       229
           1       1.00      1.00      1.00       183

    accuracy                           1.00       412
   macro avg       1.00      1.00      1.00       412
weighted avg       1.00      1.00      1.00       412
```

```
 [  0 183]]

Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       229
           1       1.00      1.00      1.00       183

    accuracy                           1.00       412
   macro avg       1.00      1.00      1.00       412
weighted avg       1.00      1.00      1.00       412



--- Activation: logistic, Test size: 0.3 ---

Confusion Matrix:
[[229   0]
 [  1 182]]

Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       229
           1       1.00      0.99      1.00       183

    accuracy                           1.00       412
   macro avg       1.00      1.00      1.00       412
weighted avg       1.00      1.00      1.00       412



--- Activation: tanh, Test size: 0.3 ---

Confusion Matrix:
[[229   0]
 [  0 183]]

Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       229
           1       1.00      1.00      1.00       183

    accuracy                           1.00       412
   macro avg       1.00      1.00      1.00       412
weighted avg       1.00      1.00      1.00       412
```

RESULT:

Thus a python program to implement multi- layer perceptron with back propagation is written and output is verified successfully.