Regression and Iris dataset

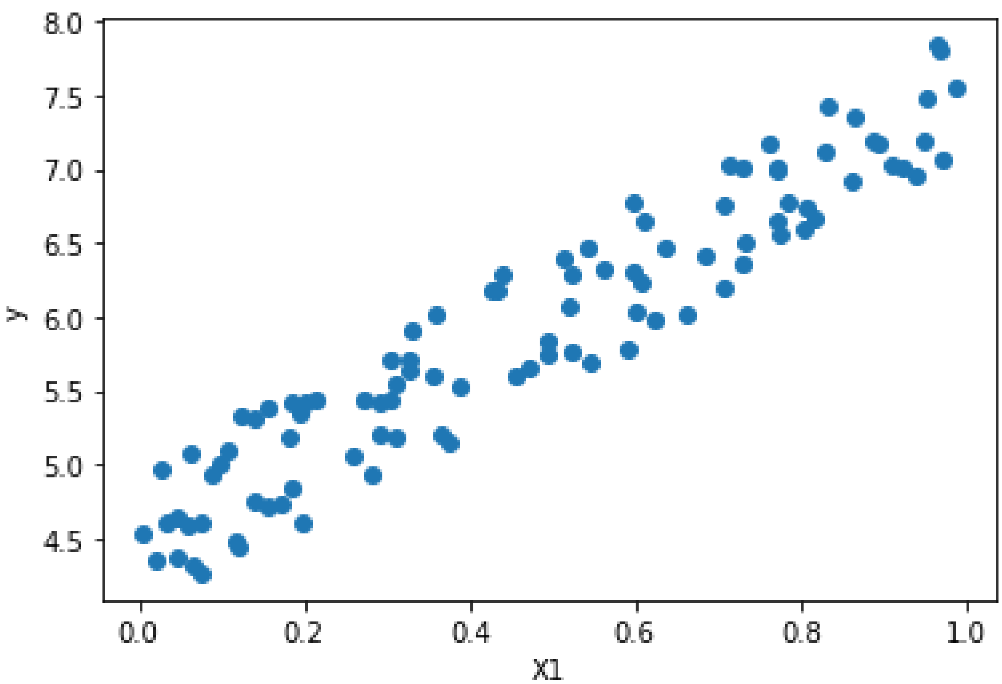
September 2, 2022

[91]:
```python
# 1.Create a dataset of 100 random values and store in X1.
import numpy as np
np.random.seed(42)
X1=np.random.rand(100,1)
X1.shape
```

[91]:
```
(100, 1)
```

[73]:
```python
# 2.Create a model for target y=4+3X1+some_random_values
y = 4 + 3 * X1 + np.random.rand(100,1)
```

[76]:
```python
# 3.Use matplotlib to plot X and y.
import matplotlib.pyplot as plt
plt.scatter(X1,y)
plt.xlabel("X1")
plt.ylabel("y")
plt.show()
```



[77]:
```
#4.Add x0=1 to each instanc of X1(use np.c_[np.ones((100,1)),X1])and store in X.
X=np.c_[np.ones((100,1)),X1]
```

[78]:
```
#5.Compute theta using the normal equation.Use the inv() function from
# np.linalg to compute the inverse of a matix and the dot() method for
# matrix multiplication.
theta = np.linalg.inv(X.T.dot(X)).dot(X.T).dot(y)
```

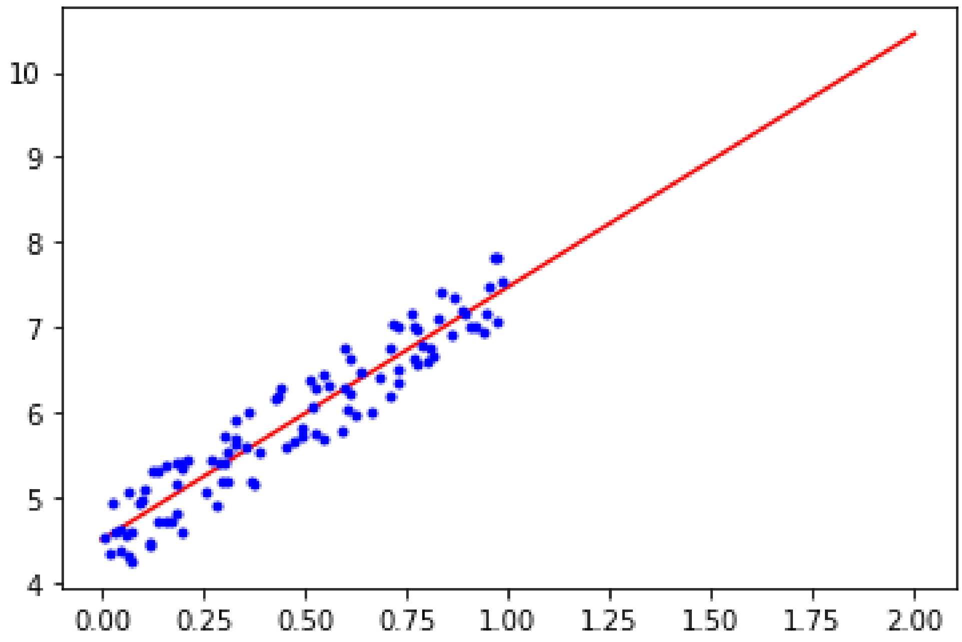
[79]:
```
# 6. Show the best value of  0 and  1 obtained in above computation.
print(theta)
```

```
[[4.51359766]
 [2.96646836]]
```

[80]:
```
#7. a= np.array( [ [0], [2] ] ), add x0 = 1 to a and store it in a1, as you did
# in 4, now make prediction for a1 and store the predicted values in pred, and
# show the values stored in pred
a = np.array([[0],[2]])
a1 = np.c_[np.ones((2,1)),a]
pred = a1.dot(theta)
pred
```

[80]:
```
array([[ 4.51359766],
       [10.44653437]])
```

[81]:
```
# 8. Plot the line using new value a (as given above) and its predicted values
# pred along with values of X1.
plt.plot(a,pred,'-r')
plt.plot(X1,y,'b.')
plt.show()
```



[82]:
```
# 9. Use LinearRegression model from linear_model of sklearn library, and train
# the model using values of X1, and target y.
from sklearn.linear_model import LinearRegression
lin_reg = LinearRegression()
lin_reg.fit(X1,y)
```

[82]: LinearRegression()

[83]:
```
#10. Show the value of intercept_ and coef_ of trained linear regression model
print("Intercept: ",lin_reg.intercept_)
print("Coefficient: ",lin_reg.coef_)
```

```
Intercept:  [4.51359766]
Coefficient:  [[2.96646836]]
```

[84]:
```
# 11. Predict the values for a (given in 7) and show the predictions.
lin_reg.predict(a)
```

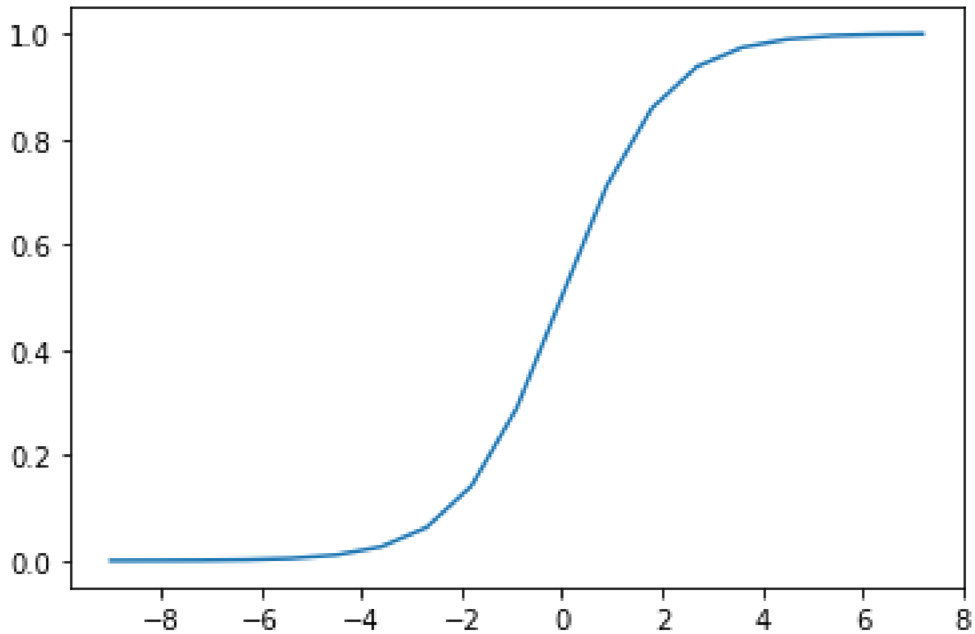
[84]:
```
array([[ 4.51359766],
       [10.44653437]])
```

[85]:
```
#12. Plot the logistic regression. The formula for logistic regression is as␣
 ↪given below:
import math

def sigmoid(t):
```

```
    m = []
    for item in t:
        m.append(1/(1+math.exp(-item)))
    return m
t = np.arange(-9., 8., 0.9) #take any random array
sig = sigmoid(t)

plt.plot(t,sig)
plt.show()
```



[86]:
```
# 13. Download iris dataset from sklearn.datasets and store the data in X1 and
# target in y1.Analyze the dataset by reading its description.
from sklearn import datasets
iris = datasets.load_iris()
X1 = iris["data"]
y1 = iris["target"]
# we have four feature sepal length , sepal width , petal length , petal width.
# And we have the target sentosa , versicolor , virginica.
```

[87]:
```
# 14. Store the data of petal length and petal width in X. Store values for
# setosa or versicolor in y
X = iris["data"][:,2:]  # petal length , petal width
#store value for sentosa or versicolor
y = ((iris["target"]==0) | (iris["target"]==1)).astype(int)
```

[88]:
```python
# 15. Update X to store data for those flowers which have target setosa or
# versicolor.(target =0 for setosa, and target = 1 for versicolor,
# X=X[(y==0)|(y==1)] and same treatmentto y= y[(y==0)|(y==1)] )
X=X[(y==0)|(y==1)]
y= y[(y==0)|(y==1)]
```

[89]:
```python
# 16. Train SVM classifier model using linear "kernel".
# (hint SVC(kernel="linear", C=float("inf")) then use fit on X and y).
from sklearn import svm
s = svm.SVC(kernel ='linear' , C=1).fit(X,y)
s
```

[89]:
```
SVC(C=1, kernel='linear')
```