linear programming assignment 2

March 25, 2024

[29]:
```
#import necessary libraries
import pandas as pd
import numpy as np
```

[30]:
```
data =pd.read_csv("C:\\Users\\DELL\\Desktop\\LINEAR PROGRAMMING␣
 ↪ASSIGNMENT\\students_score_dataset.csv")
data
```

[30]:
```
    Study Hours  Exam Scores
0           3.7         87.9
1           9.5        143.6
2           7.3        123.7
3           6.0         99.9
4           1.6         64.5
..          …           …
95          4.9         95.3
96          5.2        101.9
97          4.3         94.5
98          0.3         53.9
99          1.1         64.9

[100 rows x 2 columns]
```

[31]:
```
x = np.array(data["Study Hours"]).reshape(-1,1)
y = np.array(data["Exam Scores"])
```

[32]:
```
x
```

[32]:
```
array([[3.7],
       [9.5],
       [7.3],
       [6. ],
       [1.6],
       [1.6],
       [0.6],
       [8.7],
       [6. ],
       [7.1],
```

```
[0.2],
[9.7],
[8.3],
[2.1],
[1.8],
[1.8],
[3. ],
[5.2],
[4.3],
[2.9],
[6.1],
[1.4],
[2.9],
[3.7],
[4.6],
[7.9],
[2. ],
[5.1],
[5.9],
[0.5],
[6.1],
[1.7],
[0.7],
[9.5],
[9.7],
[8.1],
[3. ],
[1. ],
[6.8],
[4.4],
[1.2],
[5. ],
[0.3],
[9.1],
[2.6],
[6.6],
[3.1],
[5.2],
[5.5],
[1.8],
[9.7],
[7.8],
[9.4],
[8.9],
[6. ],
[9.2],
[0.9],
```

```
       [2. ],
       [0.5],
       [3.3],
       [3.9],
       [2.7],
       [8.3],
       [3.6],
       [2.8],
       [5.4],
       [1.4],
       [8. ],
       [0.7],
       [9.9],
       [7.7],
       [2. ],
       [0.1],
       [8.2],
       [7.1],
       [7.3],
       [7.7],
       [0.7],
       [3.6],
       [1.2],
       [8.6],
       [6.2],
       [3.3],
       [0.6],
       [3.1],
       [3.3],
       [7.3],
       [6.4],
       [8.9],
       [4.7],
       [1.2],
       [7.1],
       [7.6],
       [5.6],
       [7.7],
       [4.9],
       [5.2],
       [4.3],
       [0.3],
       [1.1]])
```

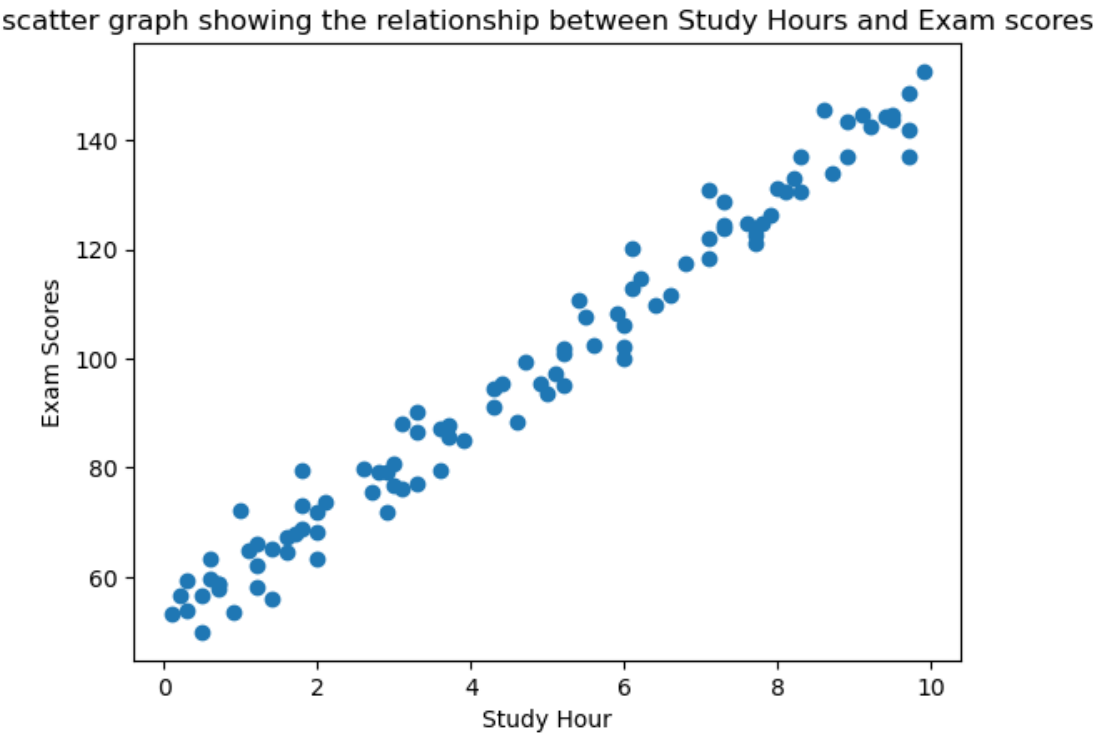
[33]:
```
y
```

[33]:
```
array([ 87.9, 143.6, 123.7,  99.9,  64.5,  67.4,  63.2, 134. , 106.1,
       118.3,  56.6, 148.6, 130.6,  73.8,  68.7,  73.2,  76.9, 100.8,
        91.2,  71.8, 112.7,  65.3,  79.2,  85.5,  88.5, 126.4,  68.3,
        97.4, 108.4,  56.7, 120.2,  67.9,  57.8, 144.5, 137. , 130.7,
        80.8,  72.1, 117.5,  95.5,  62. ,  93.7,  59.2, 144.7,  79.8,
       111.7,  88.2,  95. , 107.6,  79.4, 142. , 124.7, 144.4, 137. ,
       102. , 142.5,  53.5,  72. ,  49.9,  90.3,  85. ,  75.5, 136.9,
        79.5,  79.2, 110.8,  56.1, 131.1,  58.8, 152.6, 121. ,  63.3,
        53.2, 133. , 121.9, 124.6, 123.7,  58.6,  87.3,  58. , 145.6,
       114.7,  77.1,  59.6,  76.2,  86.5, 128.8, 109.7, 143.5,  99.3,
        66.1, 130.8, 124.9, 102.4, 122.6,  95.3, 101.9,  94.5,  53.9,
        64.9])
```

[34]:
```
#checking the missing data
data.isna().sum()
```

[34]:
```
Study Hours    0
Exam Scores    0
dtype: int64
```

[35]:
```
#visualization of the graph
import matplotlib.pyplot as plt
plt.scatter(x,y)
plt.xlabel("Study Hour")
plt.ylabel("Exam Scores")
plt.title("scatter graph showing the relationship between Study Hours and Exam␣
 ↪scores")
plt.show()
```



[36]:
```
#spliting the data
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2)

#standardizing data
from sklearn.preprocessing import StandardScaler
scaler=StandardScaler()
x_train_scaled = scaler.fit_transform(x_train)
x_test_scaled = scaler.transform(x_test)

#buliding the model
from sklearn.linear_model import LinearRegression
model = LinearRegression().fit(x_train_scaled,y_train)
model
```

[36]:
```
LinearRegression()
```

[37]:
```
#make predictions
y_pred = model.predict(x_test_scaled)
y_pred
```

[37]:
```
array([120.0580792 , 142.34837848,  68.69347653, 101.64435371,
       120.0580792 ,  67.72433308, 143.31752192,  64.81690274,
```

```
       137.50266124,  86.13805857,  86.13805857,  79.35405444,
       135.56437435,  78.38491099,  66.75518963, 117.15064886,
       147.19409571,  70.63176342, 113.27407507,  92.92206269])
```

[38]:
```
#calculate intercepts and coefficients
model.coef_
model.intercept_
print(model.coef_)
print(model.intercept_)
```

```
[28.56401317]
95.53875000000001
```

[39]:
```
#evaluating the model accuracy
model.score(x_train_scaled,y_train)
```

[39]:
```
0.9735234812035534
```

[40]:
```
model.score(x_test_scaled,y_test)
```

[40]:
```
0.9842750199525561
```

[41]:
```
from sklearn.metrics import␣
 ↪mean_absolute_error,r2_score,mean_squared_error,accuracy_score
mean=mean_absolute_error(y_test,y_pred)
r2 = r2_score(y_test,y_pred)
mse=mean_squared_error(y_test,y_pred)

print(f"mean:{mean}")
print(f"r2:{r2}")
print(f"mse:{mse}")
```

```
mean:2.523486584607764
r2:0.9842750199525561
mse:13.919350957881651
```

[42]:
```
from sklearn.preprocessing import PolynomialFeatures
poly=PolynomialFeatures(degree=2)
x_train_poly=poly.fit_transform(x_train_scaled)
x_test_poly=poly.transform(x_test_scaled)

#re_train the model
model_poly=LinearRegression().fit(x_train_poly,y_train)
pred_poly=model_poly.predict(x_test_poly)
mea =mean_absolute_error(y_test,pred_poly)
r2=r2_score(y_test,pred_poly)
mse=mean_squared_error(y_test,pred_poly)
```

```
print(f"mean:{mean}")
print(f"r2:{r2}")
print(f"mse:{mse}")
```

```
mean:2.523486584607764
r2:0.9857203553728175
mse:12.639976936052829
```

[ ]: