kyomugisha Mariam .

March 24, 2024

[1]:
```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

[5]:
```python
data=pd.read_csv("C:\\Users\\SAMUEL␣
↪K\\Desktop\\Assignment\\student_scores_dataset.csv")
data.head()
```

[5]:
```
   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
```

[9]:
```python
x=np.array(data['Study Hours']).reshape(-1,1)
x
```

[9]:
```
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]])
```

[10]:

```
y=np.array(data['Exam Scores'])
y
```

[10]:

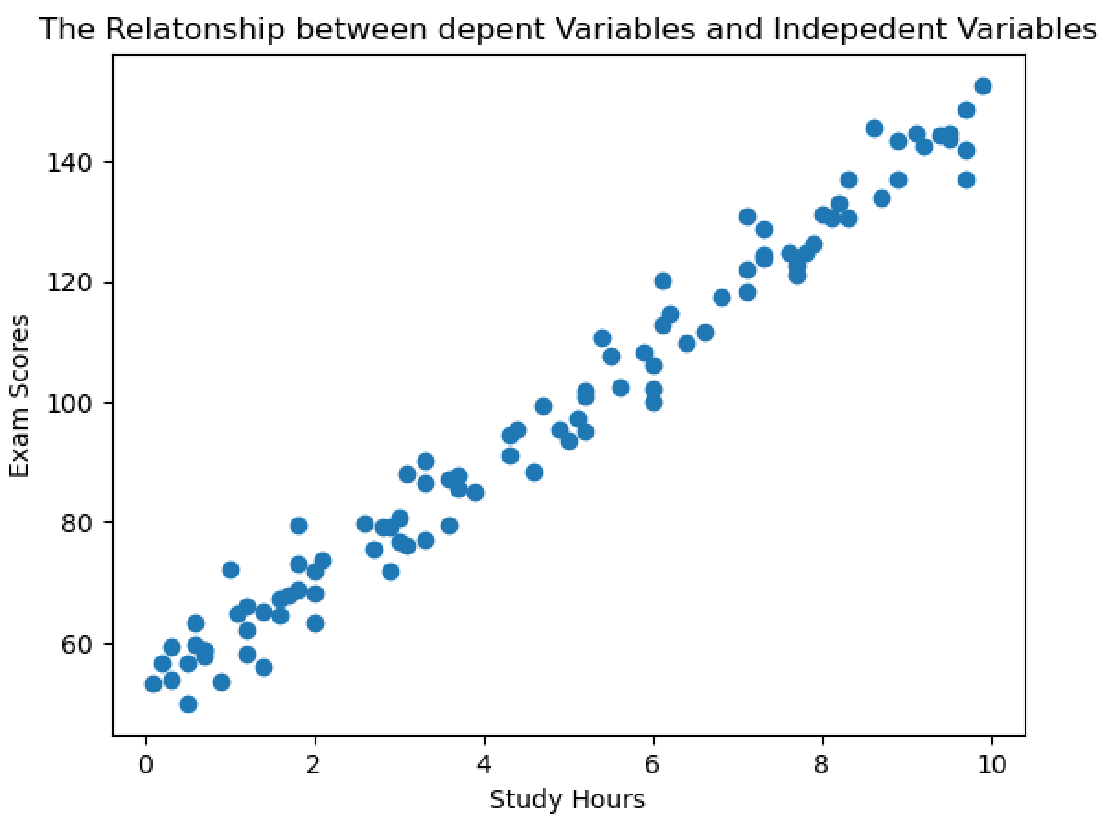
```
array([ 87.9, 143.6, 123.7,  99.9,  64.5,  67.4,  63.2, 134. , 106.1,
       118.3,  56.6, 148.6, 130.6,  73.8,  68.7,  73.2,  76.9, 100.8,
        91.2,  71.8, 112.7,  65.3,  79.2,  85.5,  88.5, 126.4,  68.3,
        97.4, 108.4,  56.7, 120.2,  67.9,  57.8, 144.5, 137. , 130.7,
        80.8,  72.1, 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])
```

[12]:
```
#checking for missing data
data.isna().sum()
```

[12]:
```
Study Hours    0
Exam Scores    0
dtype: int64
```

[13]:
```
#ploting the graphs
plt.scatter(x,y)
plt.xlabel("Study Hours")
plt.ylabel("Exam Scores")
plt.title("The Relatonship between depent Variables and Indepedent Variables")
plt.show()
```



[26]:
```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import r2_score,mean_absolute_error,mean_squared_error
from sklearn.linear_model import LinearRegression
```

[33]:
```
#splitting the data
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size =0.2,␣
↪random_state=42)
```

[34]:
```
#standardize the indepedent variabbles using scaling
scaler=StandardScaler()
x_train_scaled = scaler.fit_transform(x_train)
x_test_scaled = scaler.transform(x_test)
```

[35]:
```
#train the model using the variables of scaling
model=LinearRegression()
model.fit(x_train_scaled,y_train)
```

[35]:
```
LinearRegression()
```

[43]:
```
#making a prediction for y using scaled data
y_pred=model.predict(x_test_scaled)
print("y_pred", y_pred,("."))
```

```
y_pred [ 56.56910189 137.87943342 126.12372284 115.34765481  76.16195286
  93.79551874  79.10088051 134.94050578  52.6505317   86.9380209
  92.81587619 110.44944206 131.02193558 143.75728872  62.44695718
  66.36552738 126.12372284  57.54874444 132.00157813  67.34516992] .
```

[45]:
```
#evaluating the performance of the model
#finding the mean absolote error
MAE=mean_absolute_error(y_test,y_pred)
MAE
```

[45]:
```
2.9365732667749755
```

[46]:
```
# finding mean squared error
MSE=mean_squared_error(y_test,y_pred)
MSE
```

[46]:
```
16.202109700645348
```

[47]:
```
#finding r2 score
r2=r2_score(y_test,y_pred)
r2
```

[47]:
```
0.9826924926918468
```

[49]:
```
# finding the coefficients and intercept
model.intercept_
```

[49]:
```
96.5875
```

[52]:
```
# getting the cefficient
model.coef_
```

[52]:
```
array([28.52556103])
```

[56]:
```
#model optimization
#implementing the necessary technniques to improve the model  performance
from sklearn.preprocessing import PolynomialFeatures
```

[62]:
```
# adding a polynomial feature to improve the model
poly_features = PolynomialFeatures(degree=2)
x_train_pol = poly_features.fit_transform(x_train_scaled)# testing for a pol␣
↪featutre using the scaled x train
x_test_pol = poly_features.transform(x_test_scaled) #transforming our x test␣
↪scaled data
#print("x_train_pol",x_train_pol)
```

[66]:
```
# retrain the regression model on the updated dataset with polynomial features
model_poly= LinearRegression()
model_poly.fit(x_train_pol, y_train)
```

[66]:
```
LinearRegression()
```

[69]:
```
# making a prediction
y_pred_poly = model_poly.predict(x_test_pol)
```

[70]:
```
# evaluating the model using poly features
MAE_poly = mean_absolute_error(y_test, y_pred_poly)
MSE_poly = mean_squared_error(y_test, y_pred_poly)
r2 = r2_score(y_test, y_pred_poly)
print("mean_absolute_error",MAE)
print("mean_squared_error",MSE)
print("r2_score",r2)
```

```
mean_absolute_error 2.9365732667749755
mean_squared_error 16.202109700645348
r2_score 0.9832890659571593
```

[ ]:
```
According to the model the polynomial features optimized the model and␣
↪MAE,MSE,and r2_score had the same values
```