THE Spark Foundation - Data Science and Business Analytics GRIP August 2021

Task 1 : Prediction Using Supervised ML

Author : Yukti Kapoor

To Predict the percentage of marks of the students based on the number of hours they studied.

In [1]:
```
#importing required libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error
```

In [4]:
```
#import dataset
data = pd.read_csv("https://raw.githubusercontent.com/AdiPersonalWorks/Random/master/student_scores%20-%20stude
```

In [3]:
```
data.head()
```

Out[3]:

	Hours	Scores
0	2.5	21
1	5.1	47
2	3.2	27
3	8.5	75
4	3.5	30

In [5]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 25 entries, 0 to 24
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   Hours   25 non-null     float64
 1   Scores  25 non-null     int64  
dtypes: float64(1), int64(1)
memory usage: 528.0 bytes
```

In [6]:
```
data.describe()
```

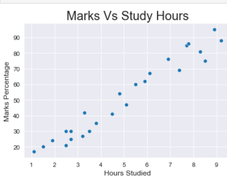
Out[6]:

	Hours	Scores
count	25.000000	25.000000
mean	5.012000	51.480000
std	2.525094	25.286887
min	1.100000	17.000000
25%	2.700000	30.000000
50%	4.800000	47.000000
75%	7.400000	75.000000
max	9.200000	95.000000

In [7]:
```
#checking null values
data.isnull().sum()
```

Out[7]:
```
Hours     0
Scores    0
dtype: int64
```

In [8]:
```
#plotting the distribution
sns.set_style('darkgrid')
sns.scatterplot(y= data['Scores'], x= data['Hours'])
plt.title('Marks Vs Study Hours',size=20)
plt.ylabel('Marks Percentage', size=12)
plt.xlabel('Hours Studied', size=12)
plt.show()
```

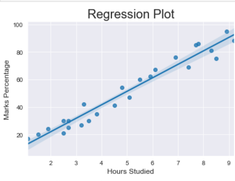


In [10]:
```
#correlation
data.corr()
```

Out[10]:

	Hours	Scores
Hours	1.000000	0.976191
Scores	0.976191	1.000000

In [12]:
```
#regression plot
sns.regplot(x= data['Hours'], y= data['Scores'])
plt.title('Regression Plot',size=20)
plt.ylabel('Marks Percentage', size=12)
plt.xlabel('Hours Studied', size=12)
plt.show()
print(data.corr())
```




```
            Hours    Scores
Hours    1.000000  0.976191
Scores   0.976191  1.000000
```

In [38]:
```
# splitting data
X = data.iloc[:, :-1].values
y = data.iloc[:, 1].values
```

In [39]:
```
#splitting into train and test set
train_X, val_X, train_y, val_y = train_test_split(X, y, random_state = 0)
```

In [40]:
```
#Fitting the Data
regression = LinearRegression()
regression.fit(train_X, train_y)
```

Out[40]:
```
LinearRegression()
```

In [41]:
```
#Predicting the Percentage of Marks

pred_y = regression.predict(val_X)
prediction = pd.DataFrame({'Hours': [i[0] for i in val_X], 'Predicted Marks': [k for k in pred_y]})
prediction
```

Out[41]:

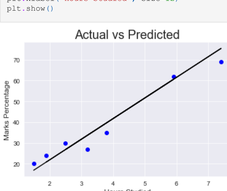
	Hours	Predicted Marks
0	1.5	16.844722
1	3.2	33.745575
2	7.4	75.500624
3	2.5	26.786400
4	5.9	60.588106
5	3.8	39.710582
6	1.9	20.821393

In [42]:
```
#comparing
compare_scores = pd.DataFrame({'Actual Marks': val_y, 'Predicted Marks': pred_y})
compare_scores
```

Out[42]:

	Actual Marks	Predicted Marks
0	20	16.844722
1	27	33.745575
2	69	75.500624
3	30	26.786400
4	62	60.588106
5	35	39.710582
6	24	20.821393

In [43]:
```
plt.scatter(x=val_X, y=val_y, color='blue')
plt.plot(val_X, pred_y, color='Black')
plt.title('Actual vs Predicted', size=20)
plt.ylabel('Marks Percentage', size=12)
plt.xlabel('Hours Studied', size=12)
plt.show()
```



In [44]:
```
# Calculating the mean absolute error
print('Mean absolute error: ',mean_absolute_error(val_y,pred_y))
```

```
Mean absolute error:  4.130879918502482
```

What will be the predicted score of a student if he/she studies for 9.25 hrs/ day?

In [46]:
```
hours = [9.25]
answer = regression.predict([hours])
print("Score = {}".format(round(answer[0],3)))
```

```
Score = 93.893
```

In [ ]:

In [ ]:

In [ ]:

In [ ]: