THE SPARKS FOUNDATION

TASK 1-Prediction using supervised ML

Author-Krishna Bansal

Importing the required libraries

```
import pandas as pd
import numpy as np
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
```

Reading the data

```
data = pd.read_csv('http://bit.ly/w-data')
data.head(5)
```

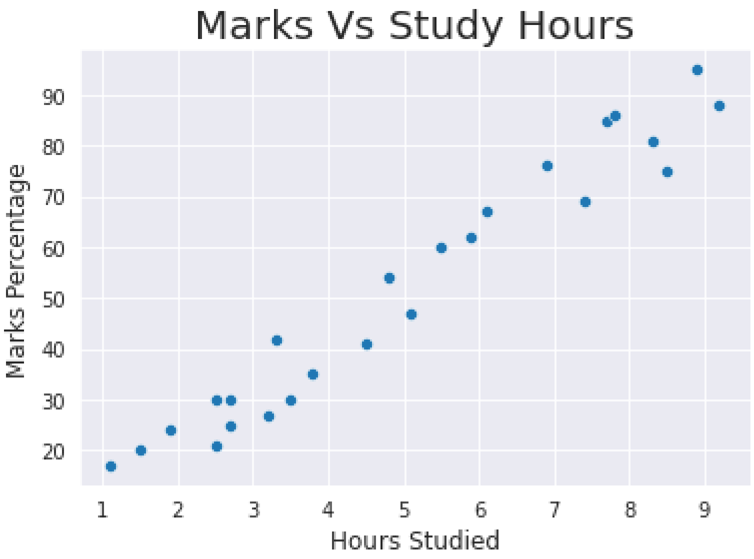
```
     Hours  Scores
  0    2.5      21
  1    5.1      47
  2    3.2      27
  3    8.5      75
  4    3.5      30
```

Check if there is null value in the dataset

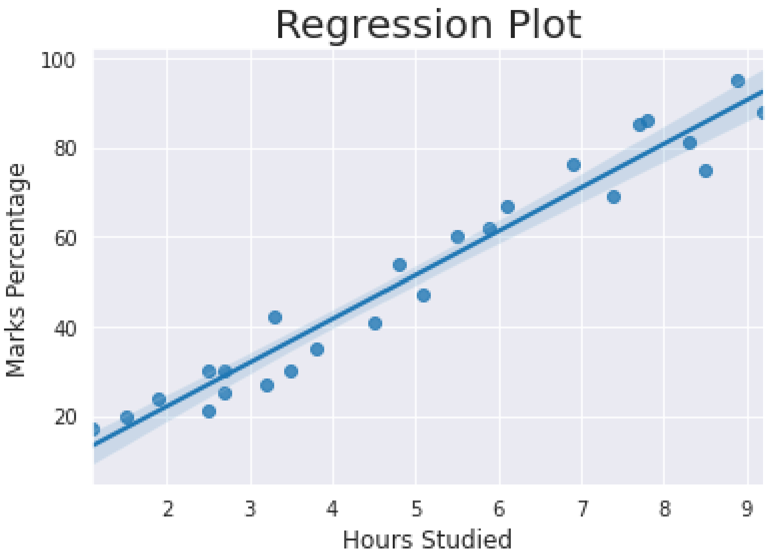
```
data.isnull == True
```

```
     False
```

```
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()
```



```
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
```

Training the Model

1.SPLITTING THE DATA

```
X = data.iloc[:, :-1].values
y = data.iloc[:, 1].values

# Spliting the Data in two
train_X, val_X, train_y, val_y = train_test_split(X, y, random_state = 0)
```

2.FITTING THE DATA INTO THE MODEL

```
regression = LinearRegression()
regression.fit(train_X, train_y)
print("---------Model Trained---------")
```

```
---------Model Trained---------
```

3. 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 pre
prediction
```

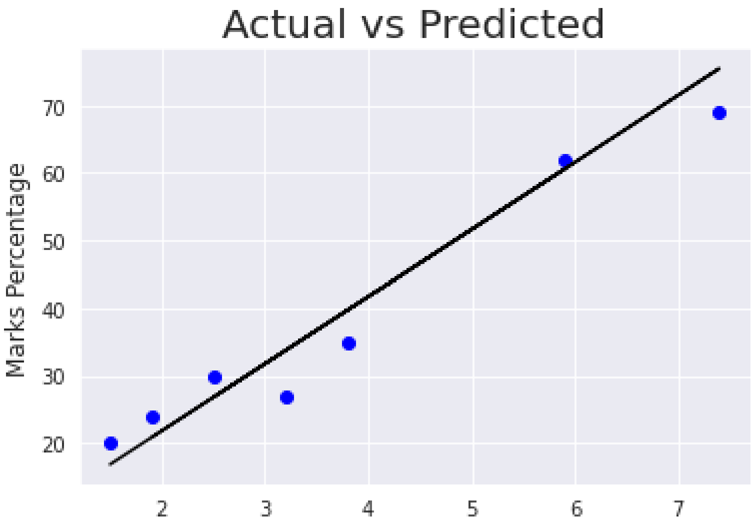
	Hours	Predicted Marks
0	1.5	16.844722
1	3.2	33.745575
2	7.4	75.500624
3	2.5	26.786400
4	5.9	60.588106
5	3.8	39.710582
6	1.9	20.821393

4. COMPARING THE PREDICTED MARKS WITH THE ACTUAL MARKS

```
compare_scores = pd.DataFrame({'Actual Marks': val_y, 'Predicted Marks': pred_y})
compare_scores
```

	Actual Marks	Predicted Marks
0	20	16.844722
1	27	33.745575
2	69	75.500624
3	30	26.786400

```
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()
```



5. EVALUATING THE MODEL

```
print('Mean absolute error: ',mean_absolute_error(val_y,pred_y))
```

```
Mean absolute error:  4.130879918502482
```

6. PREDICTED SCORE OF A STUDENT IF HE STUDIES FOR 9.25 HOURS PER DAY

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

```
Score = 93.893
```