THE SPARKS FOUNDATION

DATA SCIENCE AND BUSINESS ANALYTICS INTERN ### ### GRIPDECEMBER22 ### ### DECEMBER 2022

MRIDUL KAPOOR ### ### mridul.kapoor2002@gmail.com

TASK-1 ### ### PREDICTION USING SUPERVISED MACHINE LEARNING

In [ ]:
```
# IMPORTING MODULES
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [ ]:
```
data_url = "http://bit.ly/w-data"
data_file = pd.read_csv(data_url)
print("\nData imported successfully")

data_file.head()
```

```
Data imported successfully
```

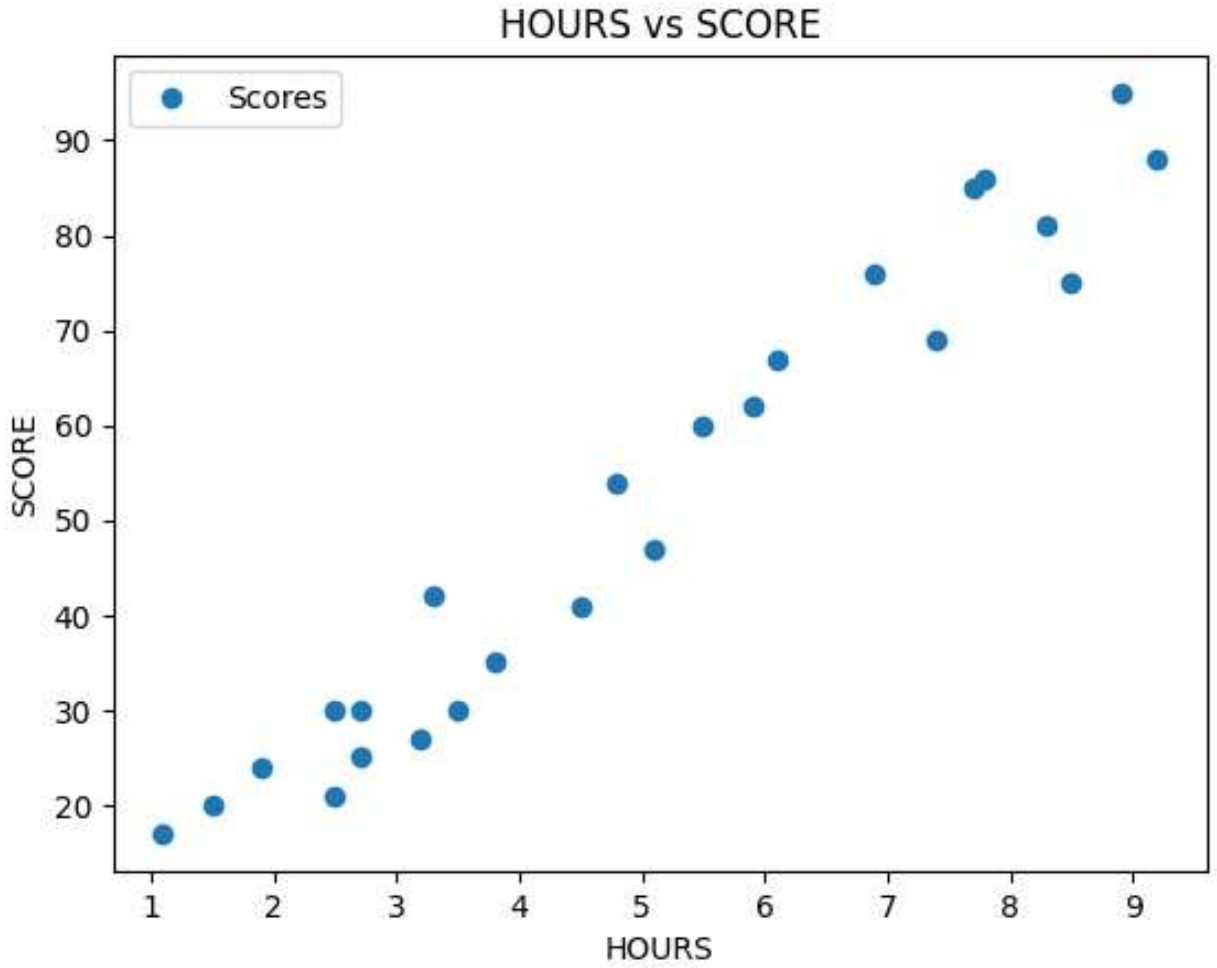
Out[ ]:

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

In [ ]:
```
data_file.isnull().sum()
```

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

In [ ]:
```
# DATA PLOTTING
data_file.plot(x = "Hours", y = "Scores", style = "o")

plt.title("HOURS vs SCORE")
plt.xlabel("HOURS")
plt.ylabel("SCORE")
plt.show()
```



In [ ]:
```
# X AND Y AXIS DATA VALUES
X = data_file.iloc[:, :-1].values
```

```
Y = data_file.iloc[:, 1].values
```

In [ ]:
```
# SPLITTING DATA INTO TEST SET AND TRAINING SET
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, random_state=0)
```

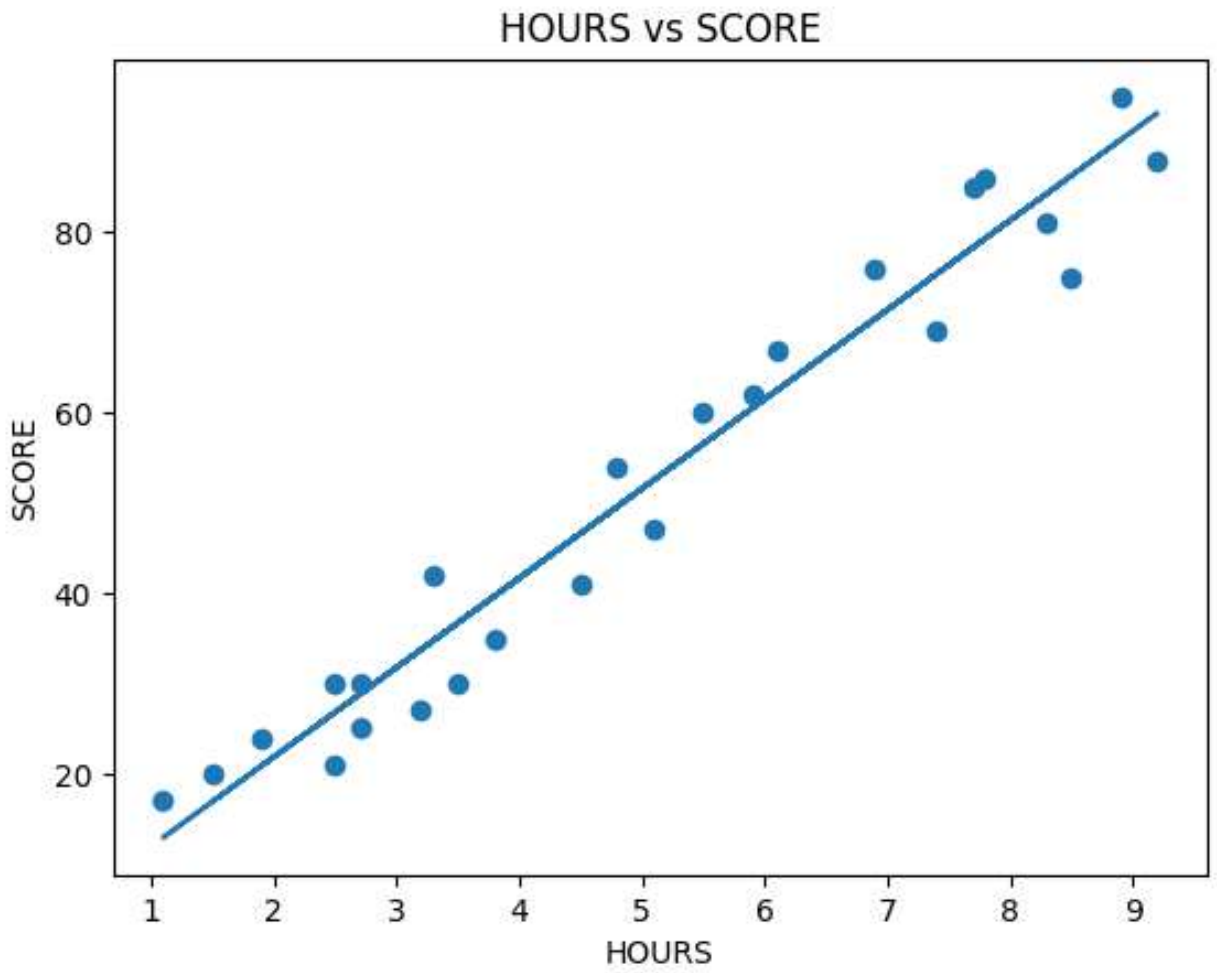
In [ ]:
```
# IMPORTING LINEAR REGRESSION MODEL
from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
regressor.fit(X_train, Y_train)

print("\nTraining complete.")
```

```
Training complete.
```

In [ ]:
```
# CREATING INSTANCE AND CREATING REGRESSION LINE
print("\nRegression Points")
line =regressor.coef_*X+regressor.intercept_
print(line)
```

```
Regression Points
[[26.79480124]
 [52.56250809]
 [33.73226078]
 [86.25874013]
 [36.70545772]
 [16.88414476]
 [93.19619966]
 [56.52677068]
 [84.27660883]
 [28.77693254]
 [78.33021494]
 [60.49103328]
 [46.6161142 ]
 [34.72332643]
 [12.91988217]
 [90.22300272]
 [26.79480124]
 [20.84840735]
 [62.47316457]
 [75.357018  ]
 [28.77693254]
 [49.58931115]
 [39.67865467]
 [70.40168976]
 [79.32128059]]
```

In [ ]:
```
plt.title("HOURS vs SCORE")
plt.xlabel("HOURS")
plt.ylabel("SCORE")
plt.scatter(X, Y)
plt.plot(X, line);
plt.show()
```



In [ ]:
```
print("\nTest Points")
print(X_test)
Y_pred = regressor.predict(X_test)
```

```
Test Points
[[1.5]
 [3.2]
 [7.4]
 [2.5]
 [5.9]]
```

In [ ]:
```
# for comparison purpose only
# COMPARING ACTUAL DATA WITH PREDICTED DATA
data_file = pd.DataFrame({'Actual': Y_test, 'Predicted': Y_pred})
data_file
```

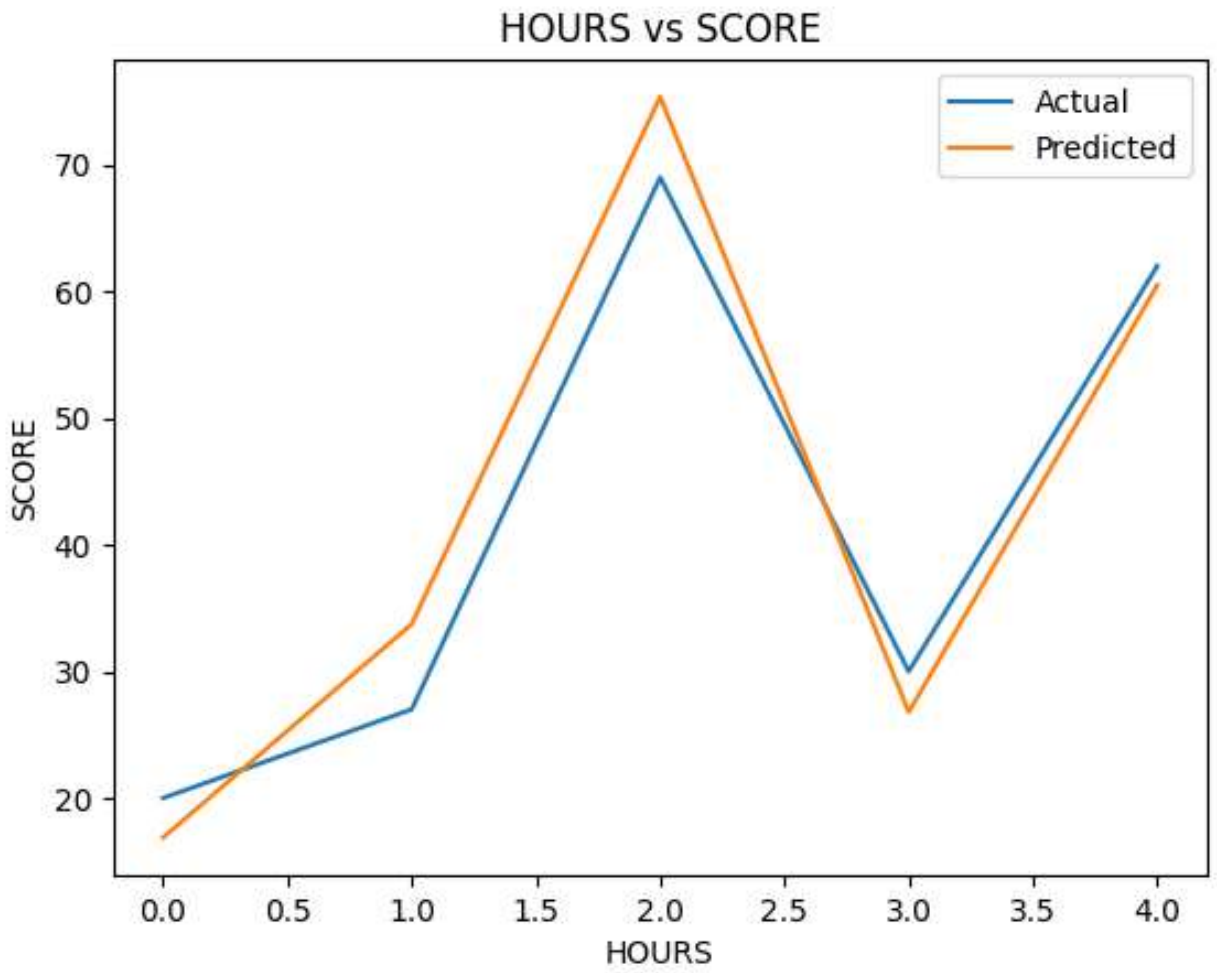
Out[ ]:

	Actual	Predicted
0	20	16.884145
1	27	33.732261
2	69	75.357018
3	30	26.794801
4	62	60.491033

In [ ]:
```
print("\nTraining score:", regressor.score(X_train, Y_train))
print("Testing score:", regressor.score(X_test, Y_test))
```

```
Training score: 0.9515510725211552
Testing score: 0.9454906892105355
```

In [ ]:
```
data_file.plot(kind='line')
plt.title("HOURS vs SCORE")
plt.xlabel("HOURS")
plt.ylabel("SCORE")
plt.show()
```



In [ ]:
```
# PREDICTING FOR USER SPECIFIED HOUR INPUT
# HERE PREDICTING FOR 9.25 HRS/DAY
hours_input = float(input("\nEnter hours student studied: "))
hours = np.array(hours_input).reshape(1, -1)
predict_score = regressor.predict(hours)
print("\nIf the student reads for %0.3f hours then he will score %0.3f"%(hours_input, predict_score[0]))
```

```
If the student reads for 9.250 hours then he will score 93.692
```

In [ ]:
```
# FOR MODEL EVALUATION
# PROVIDES ACCURACY OF MODEL
from sklearn import metrics
print("\nMean Absolute Error:", metrics.mean_absolute_error(Y_test, Y_pred))
print("Mean Squared Error:", metrics.mean_squared_error(Y_test, Y_pred))
print("Root mean squared Error:", np.sqrt(metrics.mean_squared_error(Y_test, Y_pred)),"\n")
```

```
Mean Absolute Error: 4.183859899002975
Mean Squared Error: 21.598769307217406
Root mean squared Error: 4.647447612100367
```

END