Name: Gore Aniket Machhindra

Email: aniket.m.gore.1901@gmail.com

Role: Data Science and Business Analytics Intern

Task 1: Prediction using Supervised ML

```
# Importing all libraries required in this notebook
import pandas as pd
import numpy as np
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
```

```
url =  "http://bit.ly/w-data"
data = pd.read_csv(url)
print("Data imported successfully")
data.head()
```

```
Data imported successfully
```

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

```
data.tail()
```

	Hours	Scores

```
1 data.shape
```

```
(25, 2)
```

22 3.8 35

```
1 data.describe()
```

	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

```
1 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
```

```
1 data.corr()
```

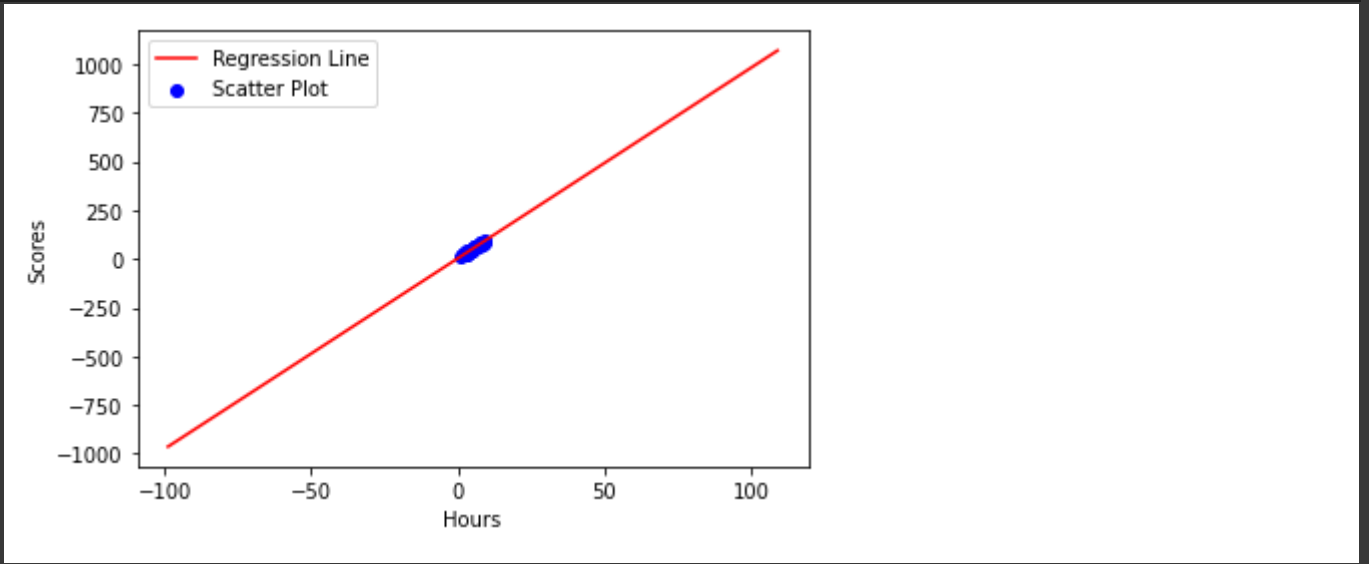
	Hours	Scores
Hours	1.000000	0.976191
Scores	0.976191	1.000000

```
1 #Collecting X and Y
2 X=data['Hours'].values
3 Y=data['Scores'].values
```

```
#Mean X and Y
mean_x=np.mean(X)
mean_y=np.mean(Y)
#Total number of values
n=len(X)
#Using the formula to calculate b1 and b2
numer=0
denom=0
for i in range(n):
    numer+=(X[i]-mean_x)*(Y[i]-mean_y)
    denom+=(X[i]-mean_x)**2
b1=numer/denom
b0=mean_y-(b1*mean_x)
#Print coefficients
print(b1,b0)
```

```
9.775803390787475 2.4836734053731746
```

```
#Plotting Values and Regression Line
max_x=np.max(X)+100
min_x=np.min(X)-100
#Calculating line values x and y
x=np.linspace(min_x,max_x,1000)
y=b0+b1*x
#Plotting Line
plt.plot(x,y,color='red',label='Regression Line')
#Plotting Scatter Points
plt.scatter(X,Y,c='blue',label='Scatter Plot')
plt.xlabel('Hours')
plt.ylabel('Scores')
plt.legend()
plt.show()
```




```
ss_t=0
ss_r=0
for i in range(n):
```

```
    y_pred=b0+b1*X[i]
    ss_t+=(Y[i]-mean_y)**2
    ss_r+=(Y[i]-y_pred)**2
r2=1-(ss_r/ss_t)
print(r2)
```

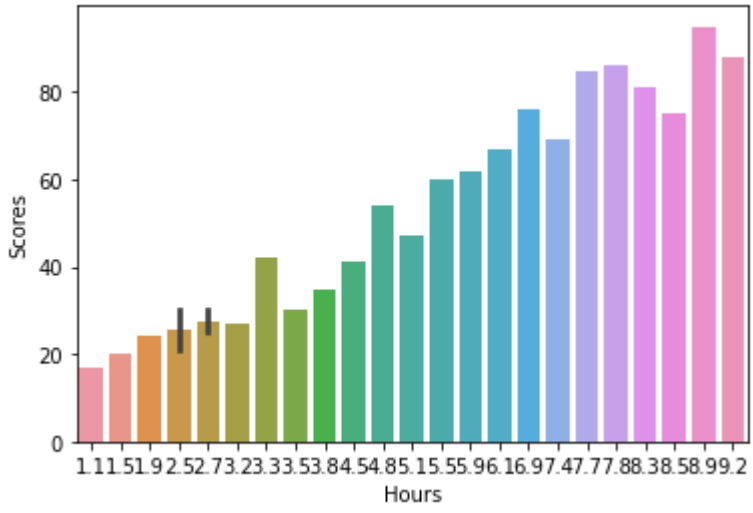
```
0.9529481969048356
```

```
#Creating Model using Scikit Learn Library
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
#Cannot use Rank 1 matrix in scikit Learn
X=X.reshape((n,1))
#Creating Model
reg=LinearRegression()
#Fitting training data
reg=reg.fit(X,Y)
#Y Prediction
Y_pred=reg.predict(X)
#Calculating R2 Score
r2_score=reg.score(X,Y)
print(r2_score)
```

```
0.9529481969048356
```

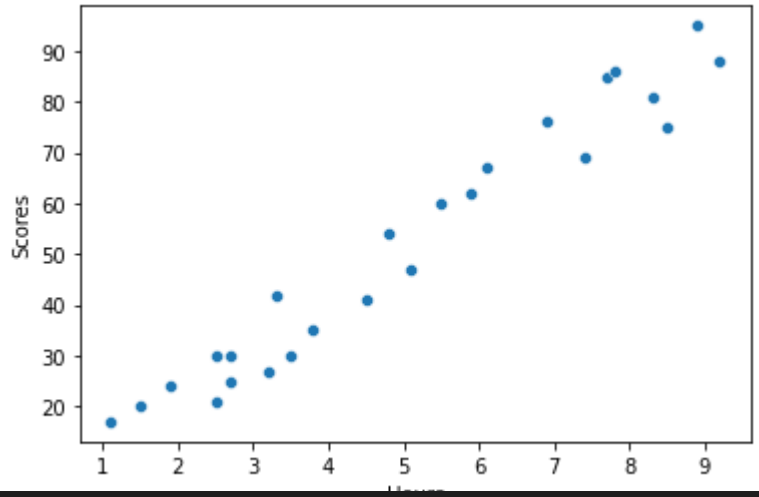
```
sns.barplot(data['Hours'], data['Scores'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass th
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7ffa51262a50>
```

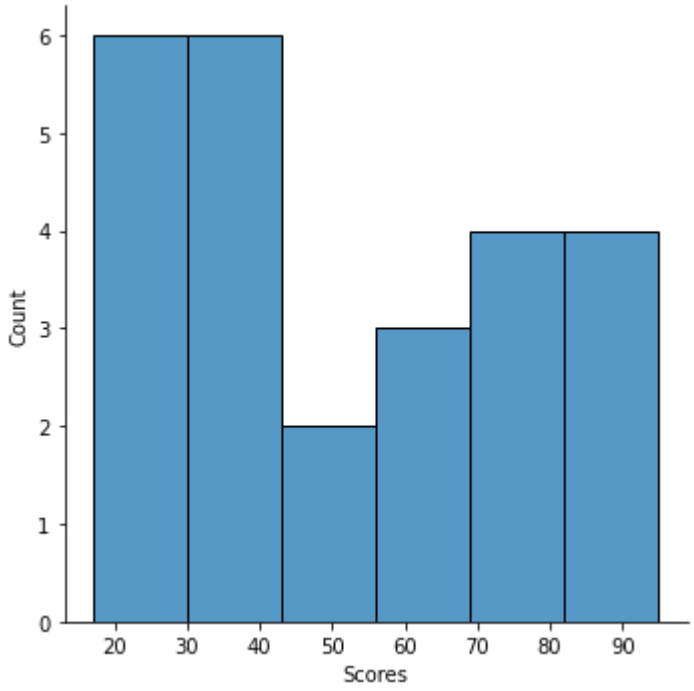



```
sns.scatterplot(data['Hours'], data['Scores'])
```

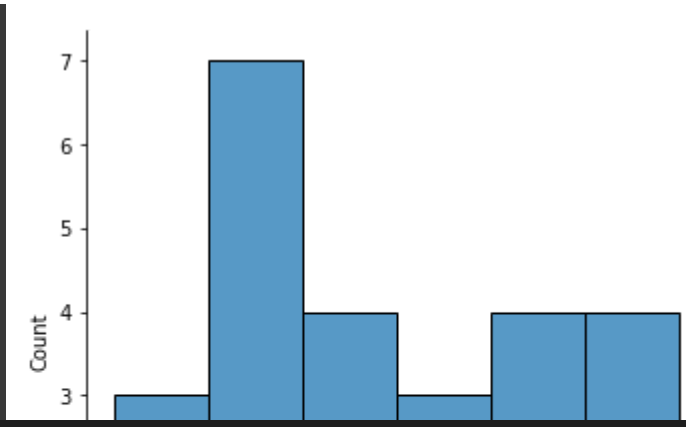
```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass th
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7ffa516c2990>
```




```
1 sns.displot(data['Scores'])
2 plt.show()
```

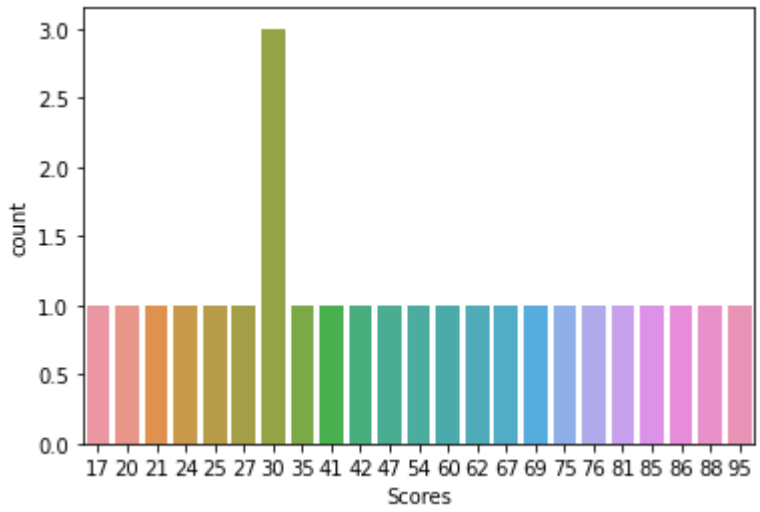



```
1 sns.displot(data['Hours'])
2 plt.show()
```




```
1 sns.countplot(data['Scores'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass th
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7ffa4f35ca90>
```

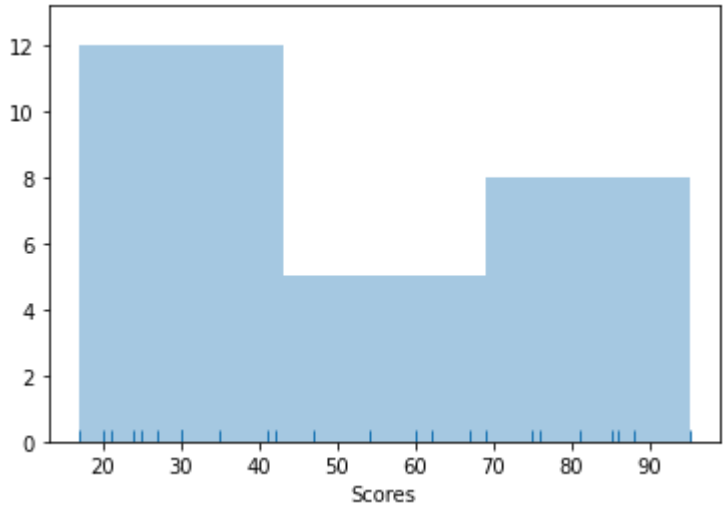



```
1 sns.countplot(data['Hours'])
2
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass th
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7ffa469f7250>
```

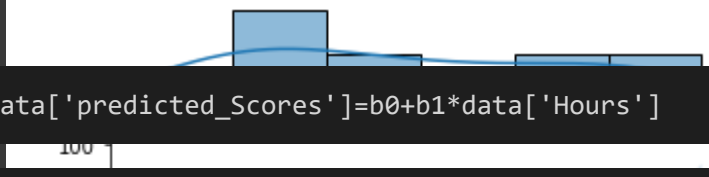
```
1 sns.distplot(data['Scores'],kde=False,rug=True)
2 plt.show()
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619: FutureWarning: `di
  warnings.warn(msg, FutureWarning)
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2103: FutureWarning: The
  warnings.warn(msg, FutureWarning)
```




```
1 sns.jointplot(data['Hours'],data['Scores'],kind="reg")
2 plt.show()
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass th
  FutureWarning
```



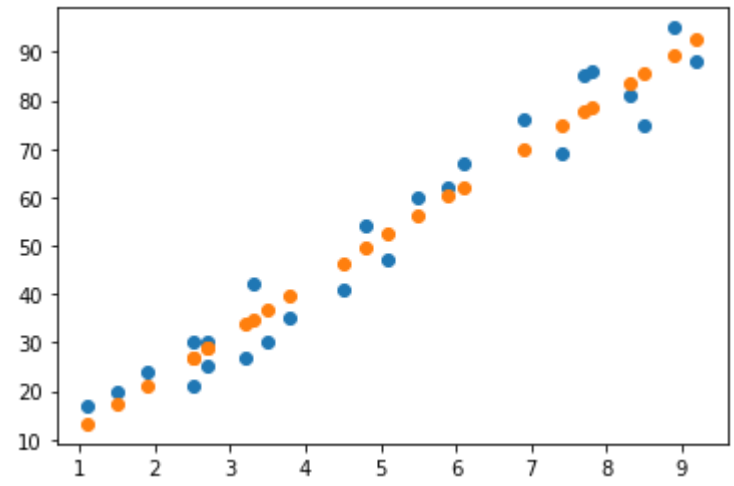
```
1 data['predicted_Scores']=b0+b1*data['Hours']
```

```
1 data.head()
```

	Hours	Scores	predicted_Scores
0	2.5	21	26.923182
1	5.1	47	52.340271
2	3.2	27	33.766244
3	8.5	75	85.578002
4	3.5	30	36.698985

```
1 plt.scatter(data['Hours'],data['Scores'])
2 plt.scatter(data['Hours'],data['predicted_Scores'])
3 plt.plot()
```

```
[]
```




```
1 b0+b1*9.25
```

```
92.90985477015732
```

```
1 y=list(data['Scores'].values)
2 y_pred=list(data['predicted_Scores'].values)
```

```
1 s=sum([(y_pred[i]-y[i])**2 for i in range (len(data))])
2 rmse=(np.sqrt(s/len(data)))/mean_y
```

```
3 rmse
```

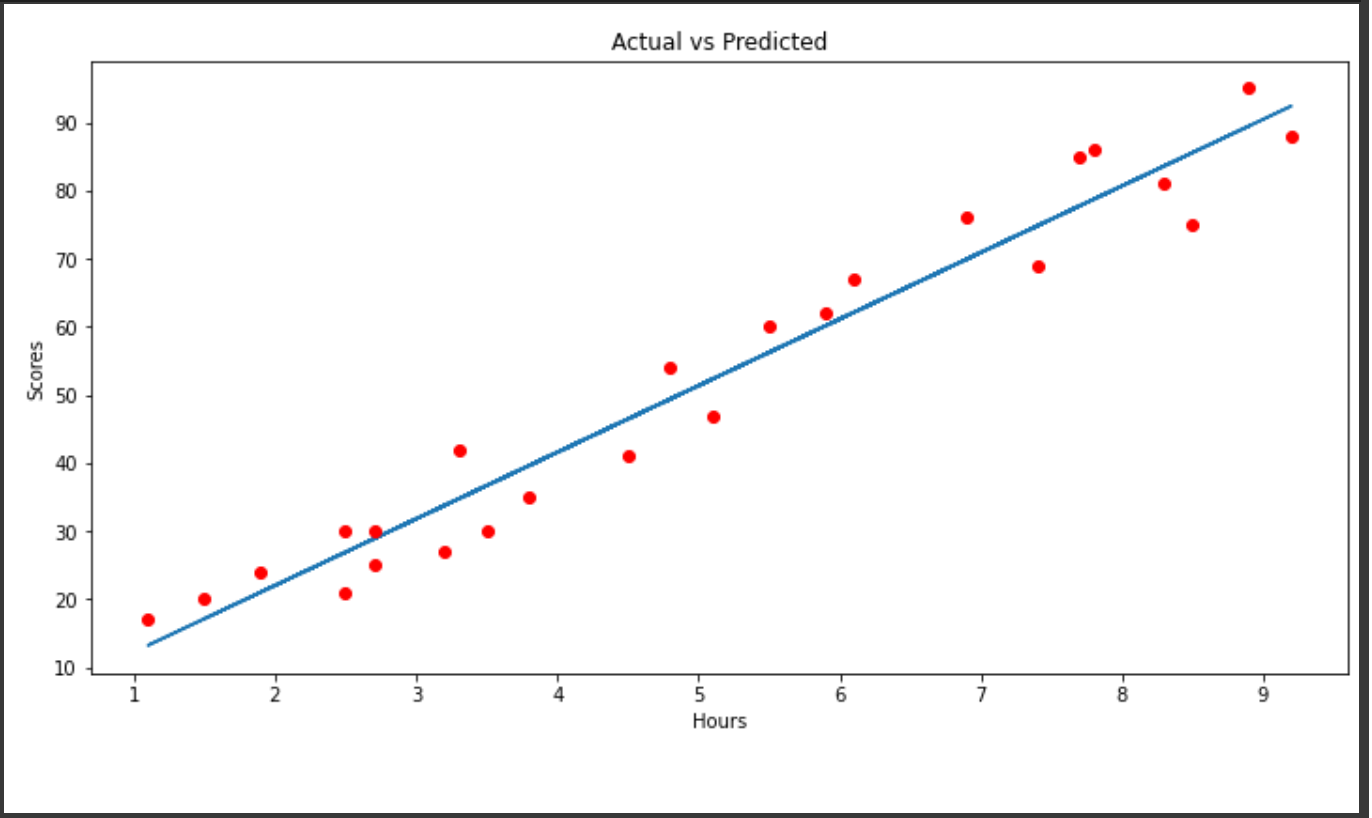
```
0.10439521325937494
```

```
1 import statsmodels.formula.api as smf
2 model=smf.ols('Scores ~ Hours',data=data)
3 model=model.fit()
```

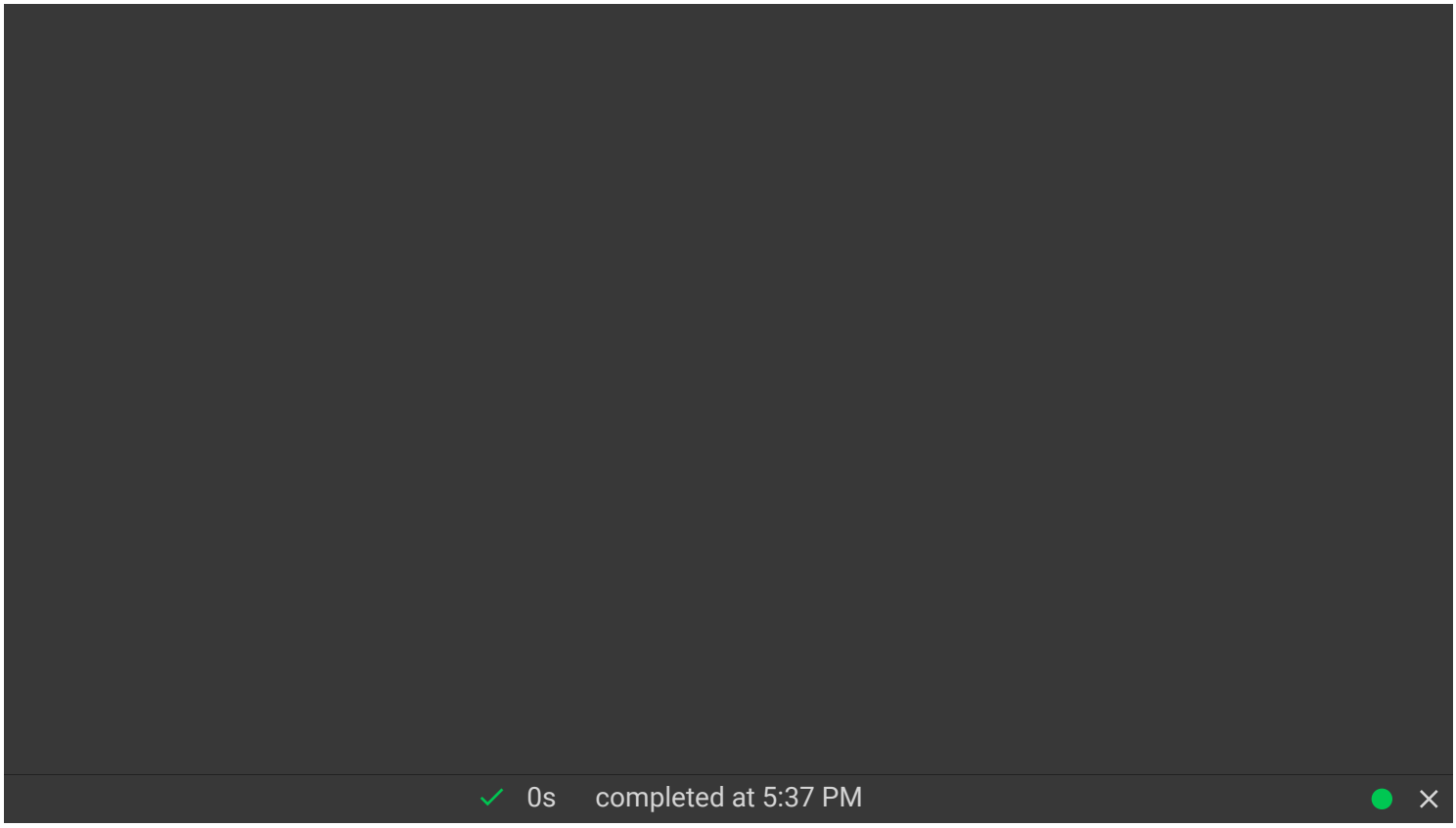
```
/usr/local/lib/python3.7/dist-packages/statsmodels/tools/_testing.py:19: FutureWarning:
  import pandas.util.testing as tm
```

```
1 data['pred_ols']=model.predict(data['Hours'])
```

```
1 plt.figure(figsize=(12,6))
2 plt.plot(data['Hours'],data['pred_ols']) #Regression line
3 plt.plot(data['Hours'],data['Scores'],'ro') #Scatter plot showing actual data
4 plt.title('Actual vs Predicted')
5 plt.xlabel('Hours')
6 plt.ylabel('Scores')
7 plt.show()
```



We can observe that the predicted value for 9.25 hours is around 92


0s
completed at 5:37 PM