Pranjal Jain

The Sparks Foundation

Prediction of Scores Using Linear Regression

Task 1

In [0]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
```

In [10]:
```
%pwd
```

Out[10]:
```
'C:\\Users\\hp'
```

In [12]:
```
data = pd.read_csv("C:\\Users\\hp\\Desktop\\data.csv")
data.head()
```

Out[12]:

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

In [6]:
```
data.shape
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-6-047ed65ff157> in <module>
----> 1 data.shape

NameError: name 'data' is not defined
```

In [18]:
```
data.dtypes
```

Out[18]:
```
Hours      float64
Scores       int64
dtype: object
```

In [19]:
```
data.info()
```

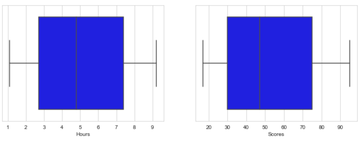
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 25 entries, 0 to 24
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   Hours   25 non-null     float64
 1   Scores  25 non-null     int64  
dtypes: float64(1), int64(1)
memory usage: 528.0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

In [20]:
```
data.describe()
```

Out[20]:

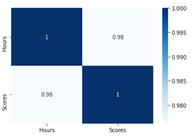
	Hours	Scores
count	25.000000	25.000000
mean	5.012000	51.480000
std	2.525094	25.286887
min	1.100000	17.000000
25%	2.700000	30.000000
50%	4.800000	47.000000
75%	7.400000	75.000000
max	9.200000	95.000000

In [54]:
```
fig,axes = plt.subplots(1,2,figsize=(12,4))
sns.set_style('whitegrid')
sns.boxplot(ax=axes[0],x=data['Hours'],color='blue')
sns.boxplot(ax=axes[1],x=data['Scores'],color='blue')
plt.show()
```



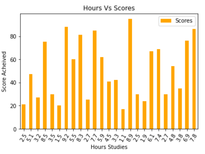
In [51]:
```
corr=data.corr()
print(corr)
sns.heatmap(corr,annot=True,cmap="Blues")
```

Out[51]:
```
            Hours    Scores
Hours   1.000000  0.976191
Scores  0.976191  1.000000
<AxesSubplot:>
```

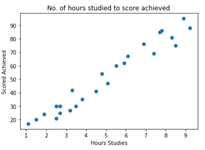


In [45]:
```
data.plot(kind = 'bar',x ='Hours',y ='Scores',style="o",color='orange',rot=60)
plt.title("Hours Vs Scores")
plt.xlabel("Hours Studies")
plt.ylabel("Score Acheived")
```

Out[45]:
```
Text(0, 0.5, 'Score Acheived')
```



In [9]:
```
plt.scatter(data['Hours'],data['Scores'])
plt.title("No. of hours studied to score achieved")
plt.xlabel("Hours Studies")
plt.ylabel("Scored Achieved")
```



In [13]:
```
x=data['Hours'].values.reshape(-1,1)
y=data['Scores'].values.reshape(-1,1)
```

In [14]:
```
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)
print("Training set X :",x_train.shape)
print("Training set Y :",y_train.shape)
print("Test set X :",x_test.shape)
print("Test set Y :",y_test.shape)
```

```
Training set X : (20, 1)
Training set Y : (20, 1)
Test set X : (5, 1)
Test set Y : (5, 1)
```

In [15]:
```
from sklearn.linear_model import LinearRegression
```

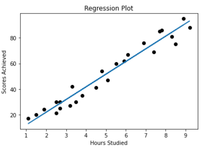
In [18]:
```
regression_model = LinearRegression()
regression_model.fit(x_train,y_train)
print("Coefficient :",regression_model.coef_)
print("Intercept :",regression_model.intercept_)
```

```
Coefficient : [[9.91065648]]
Intercept : [2.01816004]
```

In [21]:
```
line = regression_model.coef_*x + regression_model.intercept_

plt.scatter(x,y,c="Black")
plt.title("Regression Plot")
plt.xlabel("Hours Studied")
plt.ylabel("Scores Achieved")
plt.plot(x,line)
```

Out[21]:
```
[<matplotlib.lines.Line2D at 0x27a90ab2af0>]
```



In [29]:
```
y_pred = regression_model.predict(x_test)
y_pred
```

Out[29]:
```
array([[16.88414476],
       [33.73226078],
       [75.357018  ],
       [26.79480124],
       [60.49103328]])
```

In [26]:
```
from sklearn.metrics import mean_absolute_error ,r2_score
```

In [30]:
```
print("Mean Absolute Error :",mean_absolute_error(y_test, y_pred))
print("R2 Score :",r2_score(y_test, y_pred))
```

```
Mean Absolute Error : 4.183859899002975
R2 Score : 0.9454906892105356
```

In [25]:
```
Hour = np.array([9.25]).reshape(-1,1)
Percentage = regression_model.predict(Hour)
Percentage = np.round(Percentage,decimals=1)
Percentage
```

Out[25]:
```
array([[93.7]])
```

In [ ]:
```

```