Prediction using Supervised

Using simple Linear Regression the model predict the marks scored by the student based on the number hours studied.

Name-> Vikrant Patil

In [2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

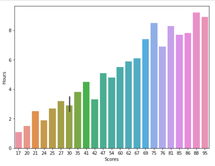
Loading the dataset

df=pd.read_csv("https://raw.githubusercontent.com/AdiPersonalWorks/Random/master/student_scores%20-%20student_scores.csv")

Data visualization

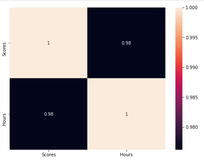
Here we see the hidden features and trends in the dataset.

In [4]:
```
sns.barplot(x=df['Scores'],y=df['Hours'])
fig=plt.gcf()
fig.set_size_inches(8,6)
plt.show()
```



In [5]:
```
numeric_value=['Scores','Hours']
```

In [29]:
```
sns.heatmap(df[numeric_value].corr(),annot=True)
fig=plt.gcf()
fig.set_size_inches(8,6)
plt.show()
```



In [7]:
```
df
```

Out[7]:

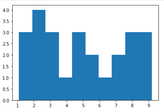
	Hours	Scores
0	2.5	21
1	5.1	47
2	3.2	27
3	8.5	75
4	3.5	30
5	1.5	20
6	9.2	88
7	5.5	60
8	8.3	81
9	2.7	25
10	7.7	85
11	5.9	62
12	4.5	41
13	3.3	42
14	1.1	17
15	8.9	95
16	2.5	30
17	1.9	24
18	6.1	67
19	7.4	69
20	2.7	30
21	4.8	54
22	3.8	35
23	6.9	76
24	7.8	86

In [8]:
```
df.head()
```

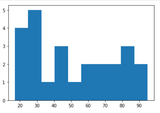
Out[8]:

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

In [12]:
```
plt.hist(df['Hours'])
plt.show()
```



In [10]:
```
plt.hist(df['Scores'])
plt.show()
```



Variable selection for the algorithm

x is the independent variable(Hours)

y is the dependent variable(Score)if

In [13]:
```
x=df.iloc[:,:-1].values
```

In [14]:
```
y=df.iloc[:,-1].values
```

In [15]:
```
x.shape
```

Out[15]:
```
(25, 1)
```

In [16]:
```
y.shape
```

Out[16]:
```
(25,)
```

In [17]:
```
x,y
```

Out[17]:
```
(array([[2.5],
        [5.1],
        [3.2],
        [8.5],
        [3.5],
        [1.5],
        [9.2],
        [5.5],
        [8.3],
        [2.7],
        [7.7],
        [5.9],
        [4.5],
        [3.3],
        [1.1],
        [8.9],
        [2.5],
        [1.9],
        [6.1],
        [7.4],
        [2.7],
        [4.8],
        [3.8],
        [6.9],
        [7.8]]),
 array([21, 47, 27, 75, 30, 20, 88, 60, 81, 25, 85, 62, 41, 42, 17, 95, 30,
        24, 67, 69, 30, 54, 35, 76, 86], dtype=int64))
```

Training and testing the model

Dividing the x and y data in testing and training

In [18]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3,random_state=False)
```

In [19]:
```
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
lr.fit(x_train,y_train)
```

Out[19]:
```
LinearRegression()
```

In [20]:
```
y_pred=lr.predict(x_test)
```

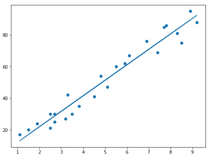
In [21]:
```
prediction=pd.DataFrame({"Actual_value":y_test,"Predicted_value":y_pred})
prediction
```

Out[21]:

	Actual_value	Predicted_value
0	20	17.053665
1	27	33.694229
2	69	74.806209
3	30	26.842232
4	62	60.123359
5	35	39.567369
6	24	20.969092
7	86	78.721636

Line of Regression

In [22]:
```
line = lr.coef_*x+lr.intercept_
plt.scatter(x, y)
plt.plot(x, line);
fig=plt.gcf()
fig.set_size_inches(8,6)
plt.show()
```



Predicting the output against the input.

In [23]:
```
hours=np.array([[9.25]])
score_predict=lr.predict(hours)
print(f"Number of hours studied is:{float(hours)}")
print(f"Predicted marks obtained are:{int(score_predict)}" )
```

```
Number of hours studied is:9.25
Predicted marks obtained are:92
```

Evaluating the model

Here we test how good out model is performing for the testing and predicted values.

In [24]:
```
from sklearn import metrics
MAE=metrics.mean_absolute_error(y_test,y_pred)
print(f'Mean absolute error is: {MAE}')
```

```
Mean absolute error is: 4.419727808027652
```

In [25]:
```
from math import sqrt

RMSE=sqrt(metrics.mean_squared_error(y_test,y_pred))
print(f'Root Mean Squred error is: {RMSE}') .
```

```
Root Mean Squred error is: 4.792191274636315
```

Thankyou !

In [ ]:
```

```