The Sparks Foundation

gripfeb2021

Task 1- Prediction using Supervised Machine Learning

By- Sugandha kumari

Problem Statement -What will be the predicted score if a student studies for 9.25 hrs/ day?

Tools used:Python,numpy,pandas,matplotlib,pyplot

Simple Regression- Simple regression is a linear regression model that has a single explanatory variable.That is it uses two variables one is independent and the other is dependent variable.

Steps to solve problem statement:

```
Import dataset
Data processing
Explanatory data analysis
Create Machine learning model
Evaluate the model
```

Import Library

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

Import Dataset

In [2]:
```
df=pd.read_csv(r'C:\Users\rashm\Desktop\Scores.csv')
```

In [3]:
```
df.head()
```

Out[3]:

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

In [4]:
```
#Exploring Data
df.isnull()
```

Out[4]:

	Hours	Scores
0	False	False
1	False	False
2	False	False
3	False	False
4	False	False
5	False	False
6	False	False
7	False	False
8	False	False
9	False	False
10	False	False
11	False	False
12	False	False
13	False	False
14	False	False
15	False	False
16	False	False
17	False	False
18	False	False
19	False	False
20	False	False
21	False	False
22	False	False
23	False	False
24	False	False

In [5]:
```
df.describe()
```

Out[5]:

	Hours	Scores
count	25.000000	25.000000
mean	5.012400	51.480000
std	2.531185	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 [6]:
```
df.shape
```

Out[6]:
```
(25, 2)
```

Data Visualisation

In [7]:
```
df.plot(kind='scatter',x='Hours',y='Scores')
```

Out[7]:
```
<AxesSubplot:xlabel='Hours', ylabel='Scores'>
```

In [8]:
```
#define dependent and independent variable
x=df.iloc[:,[0]].values
```

In [9]:
```
y=df.iloc[:,1].values
```

In [10]:
```
x
```

Out[10]:
```
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]])
```

In [11]:
```
y
```

Out[11]:
```
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)
```

Split data into train and test set

In [12]:
```
from sklearn.model_selection import train_test_split
```

In [13]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
```

In [14]:
```
x_train
```

Out[14]:
```
array([[2.7],
       [3.8],
       [8.5],
       [1.9],
       [5.9],
       [3.2],
       [4.8],
       [7.8],
       [4.5],
       [6.9],
       [7.7],
       [9.2],
       [2.5],
       [5.1],
       [3.5],
       [5.5],
       [7.4]])
```

In [15]:
```
x_test
```

Out[15]:
```
array([[8.9],
       [3.3],
       [1.1],
       [2.7],
       [1.5],
       [6.4],
       [8.3],
       [2.5]])
```

In [16]:
```
y_train
```

Out[16]:
```
array([25, 35, 75, 24, 62, 27, 54, 86, 41, 76, 85, 88, 30, 47, 30, 60, 69],
      dtype=int64)
```

In [17]:
```
y_test
```

Out[17]:
```
array([95, 42, 17, 30, 20, 67, 81, 21], dtype=int64)
```

In [18]:
```
from sklearn.linear_model import LinearRegression
```

In [19]:
```
lr=LinearRegression()
```

In [20]:
```
lr.fit(x_train,y_train)
```

Out[20]:
```
LinearRegression()
```

In [21]:
```
y_pred=lr.predict(x_test)
```

In [22]:
```
y_pred
```

Out[22]:
```
array([89.03626555, 33.44268144, 11.60234482, 27.48622599, 15.57331511,
       64.21770122, 83.07981811, 25.50074885])
```

In [23]:
```
plt.plot(x_test,y_pred)
plt.scatter(x_test,y_test)
plt.xlabel('Hours')
plt.ylabel('Scores')
plt.title('Actual vs Predicted')
```

Out[23]:
```
Text(0.5, 1.0, 'Actual vs Predicted')
```

In [24]:
```
cm=pd.DataFrame({'Target':y_test,'Predicted':y_pred})
```

In [25]:
```
cm
```

Out[25]:

	Target	Predicted
0	95	89.036266
1	42	33.442681
2	17	11.602345
3	30	27.486226
4	20	15.573315
5	67	64.217701
6	81	83.079810
7	21	25.500741

Solution of Problem Statement

In [26]:
```
lr.predict([[9.25]])
```

Out[26]:
```
array([92.51086456])
```

Evaluation of Model

In [27]:
```
from sklearn import metrics
```

In [28]:
```
print('MAE=',metrics.mean_absolute_error(y_test,y_pred))
```

```
MAE= 4.527752103862475 5
```