DATA SCIENCE AND BUSINESS ANALYTICS INTERN AT THE SPARK FOUNDATION

Girija Kumaran

Task 1

Prediction Using supervised ML

Importing the libraries

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

loading the dataset

In [2]:
```
data_set=pd.read_csv("C:/Users/Girija/Downloads/student.data.csv")
print("The dataset is successfully loaded")
```

```
The dataset is successfully loaded
```

preprocessing the dataset

In [3]:
```
#first five row of dataset
data_set.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]:
```
#last five rows of the dataset
data_set.tail()
```

Out[4]:

	Hours	Scores
20	2.7	30
21	4.8	54
22	3.8	35
23	6.9	76
24	7.8	86

In [5]:
```
#basic info of data
data_set.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 [6]:
```
#statistical info of dataset
data_set.describe()
```

Out[6]:

	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 [7]:
```
#checking if there is null value in the dataset
data_set.isnull().sum()
```

Out[7]:
```
Hours     0
Scores    0
dtype: int64
```

In [8]:
```
#checking the datatypes of dataset
data_set.dtypes
```

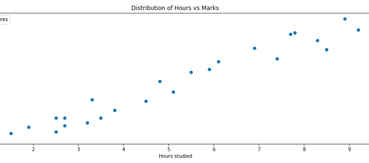
Out[8]:
```
Hours      float64
Scores       int64
dtype: object
```

In [9]:
```
#getting the shape of data set
data_set.shape
```

Out[9]:
```
(25, 2)
```

Data Visualization

In [10]:
```
data_set.plot(x="Hours",y="Scores",style="o",figsize=(15,5))
plt.title("Distribution of Hours vs Marks")
plt.xlabel("Hours studied")
plt.ylabel("Marks scored")
plt.legend()
plt.show()
```



From the above chart, we conclude that there is a linear relationship between the amount of hours studied and the marks scored.

In [11]:
```
x = data_set.iloc[:,:-1].values
y = data_set.iloc[:,1:].values

x.reshape(-1,1)
y.reshape(-1,1)
```

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)
```

Splitting dataset into testing and training set

In [12]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y)
```

In [13]:
```
from sklearn.linear_model import LinearRegression
lin_reg = LinearRegression()
lin_reg.fit(x_train,y_train)
```

Out[13]:
```
LinearRegression()
```

In [14]:
```
#visualising the training set
plt.scatter(x_train,y_train)
plt.title("Training test")
plt.plot(x_train,lin_reg.predict(x_train))
plt.xlabel("Hours studies")
plt.ylabel("Marks scored")
plt.show()
```


In [15]:
```
lin_reg.score(x_train,y_train)
```

Out[15]:
```
0.9530484855316504
```

linear regression equation

y=mx+c

In [16]:
```
line = lin_reg.coef_*x+lin_reg.intercept_
plt.scatter(x,y)
plt.title("Regression line")
plt.plot(x,line)
plt.xlabel("Hours studied")
plt.ylabel("Marks scored")
plt.show()
```


Making predictions

Now we trained our algorithm,we make predictions

In [17]:
```
print(x_test)
y_pred = lin_reg.predict(x_test)
```

```
[[3.5]
 [9.2]
 [1.9]
 [2.5]
 [3.2]
 [7.8]
 [7.4]]
```

In [18]:
```
y_pred
```

Out[18]:
```
array([[38.00792088],
       [92.85073124],
       [22.6134478 ],
       [28.3863752 ],
       [35.12145718],
       [79.38056729],
       [75.53194902]])
```

In [19]:
```
y_test
```

Out[19]:
```
array([[30],
       [88],
       [24],
       [21],
       [27],
       [86],
       [69]], dtype=int64)
```

In [20]:
```
hours = float(input("Enter a hours that students studied "))
```

```
Enter a hours that students studied 9.25
```

In [21]:
```
#the predicted score of students who have studied for 9.25 hours
own_prediction = lin_reg.predict([[hours]])
print("predicted score={}".format([own_prediction][0]))
```

```
predicted score=[93.33180853]
```

Hence the predicted score of student who have studied for 9.25 hour is 93.33

THANK YOU!