GRIP : THE SPARKS FOUNDATION

Data Science and Business Analytics Intern

Submitted by : Deepanjali Rao

Task 1 : Prediction using Supervised ML

In this task we have to predict the precentage score of a student based on the number of hours studied. The task has two variables where the feature is the no. of hours studied and the target value is the percentage score. This can be solved using simple linear regression.

In [1]:
```
# Importing required libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

Reading data from remote url

In [2]:
```
url = "https://raw.githubusercontent.com/AdiPersonalWorks/Random/master/student_scores%20-%20student_scores.csv"
data=pd.read_csv(url)
```

Exploring Data

In [3]:
```
print(data.shape)
data.head()
```

Out[3]:
```
(25, 2)
```

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

This means there are 25 rows and 2 columns.

In [4]:
```
data.describe()
```

Out[4]:

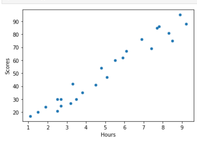
	Hours	Scores
count	25.000000	25.000000
mean	5.012000	51.480000
std	2.525094	25.286887
min	1.100000	17.000000
25%	2.700000	30.000000
50%	4.800000	47.000000
75%	7.400000	75.000000
max	9.200000	95.000000

In [5]:
```
data.info()
```

Out[5]:
```
<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]:
```
data.plot(kind='scatter',x='Hours',y='Scores');
plt.show()
```

Out[6]:



It shows that there is a linear relationship between the two variables. This can be validated with the help of Correlation Coefficient.

In [7]:
```
data.corr(method='pearson')
```

Out[7]:

	Hours	Scores
Hours	1.000000	0.976191
Scores	0.976191	1.000000

In [9]:
```
data.corr(method='spearman')
```

Out[9]:

	Hours	Scores
Hours	1.000000	0.971891
Scores	0.971891	1.000000

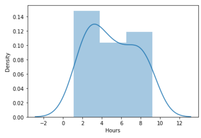
The correlation coefficient is 0.976 which is equal to 1 and positive value. So, this means that there is a positive linear relationship between two variables which is implies that if the number of hours increase then the score will also increase.

In [14]:
```
hour=data['Hours']
scores=data['Scores']
```

In [15]:
```
sns.distplot(hours)
```

```
C:\Users\dhruv\anaconda3\lib\site-packages\seaborn\distributions.py:2551: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

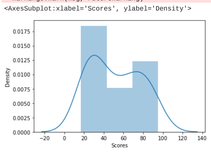
Out[15]:
```
<AxesSubplot:xlabel='Hours', ylabel='Density'>
```



In [16]:
```
sns.distplot(scores)
```

```
C:\Users\dhruv\anaconda3\lib\site-packages\seaborn\distributions.py:2551: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[16]:
```
<AxesSubplot:xlabel='Scores', ylabel='Density'>
```



Here, we are plotting the distribution plot of the two variables.

Variables are in particular range and there are no outliers in the variable.

Linear Regression

In [23]:
```
X = data.iloc[:, :-1].values
y = data.iloc[:, 1].values
```

In [21]:
```
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=50)
```

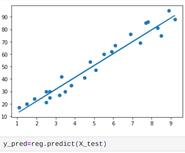
In [24]:
```
from sklearn.linear_model import LinearRegression
reg=LinearRegression()
reg.fit(X_train, y_train)
```

Out[24]:
```
LinearRegression()
```

We will use Model- Linear Regression.

First step is to divide the data into Train and Test datasets. 80% of dataset is into Train dataset and 20% into Test dataset.

In [25]:
```
m=reg.coef_
c=reg.intercept_
line=m*X+c
plt.scatter(X,y)
plt.plot(X,line);
plt.show()
```



In [26]:
```
y_pred=reg.predict(X_test)
```

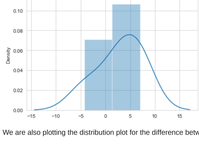
In [27]:
```
actual_predicted=pd.DataFrame({'Target':y_test,'Predicted':y_pred})
actual_predicted
```

Out[27]:

	Target	Predicted
0	95	88.211394
1	30	28.718453
2	76	69.020122
3	35	39.273652
4	17	13.365436

In [29]:
```
sns.set_style('whitegrid')
sns.distplot(np.array(y_test-y_pred))
plt.show()
```

```
C:\Users\dhruv\anaconda3\lib\site-packages\seaborn\distributions.py:2551: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```



We are also plotting the distribution plot for the difference between the target value and the predicted value.

So, this difference is also very close to 0 and it is in range from -5 to 5 which shows that a model is fitting the data well.

What would be the predicted score if a student studies for 9.25 hours/day?

In [30]:
```
h=9.25
s=reg.predict([[h]])
print("If a student for {} hours per day he/she will score {} % in exam.".format(h,s))
```

```
If a student for 9.25 hours per day he/she will score [91.56986604] % in exam.
```

Model Evaluation

In [31]:
```
from sklearn import metrics
from sklearn.metrics import r2_score
print('Mean Absolute Error:',metrics.mean_absolute_error(y_test, y_pred))
print('R2 score:', r2_score(y_test,y_pred))
```

```
Mean Absolute Error: 4.591649530063028
R2 score: 0.9710141413299442
```

We are using two performance metrics for the model evaluation. First one is Mean Absolute Error and second is R2 score.

We are getting higher R2 score of 0.97 which tells our model is predicting 97.1% of the data.