Task 1: Prediction using Supervised ML

Noel S John

Aim

To predict the percentage of an student based on the no. of study hours.

Simple linear regression

Simple linear regression is a statistical method that allows us to summarize and study relationships between two continuous (quantitative) variables: One variable, denoted x, is regarded as the predictor, explanatory, or independent variable. The other variable, denoted y, is regarded as the response, outcome, or dependent variable.

Supervised ML

Supervised learning is the types of machine learning in which machines are trained using well "labelled" training data, and on basis of that data, machines predict the output. The labelled data means some input data is already tagged with the correct output.In supervised learning, the training data provided to the machines work as the supervisor that teaches the machines to predict the output correctly.

Procedure & Analysis

In [23]:
```
# Importing the libraries required
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

In [39]:
```
# Importing the data
url = "http://bit.ly/w-data"
st_data = pd.read_csv(url)
st_data.head(5)
```

Out[39]:

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

About the Data

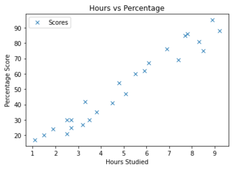
Here we are using the data which consists of 2 variables,i.e no. of study hours taken by a student and their respective scores obtained. This is a simple linear regression task as it involves just two variables.As we are following simple linear regression, we are taking scores as the dependent variable and hours as the independent variable.

In [25]:
```
st_data.describe()
```

Out[25]:

	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 [26]:
```
# Plotting the distribution of scores w.r. to hours studied
st_data.plot(x='Hours', y='Scores', style='x')
plt.title('Hours vs Percentage')
plt.xlabel('Hours Studied')
plt.ylabel('Percentage Score')
plt.show()
```



In [27]:
```
#obtaining the correlation matrix of hours and scores
st_data.corr()
```

Out[27]:

	Hours	Scores
Hours	1.000000	0.976191
Scores	0.976191	1.000000

From the plot and the correlation matrix obtained we can understand that there is a positive correlation between hours taken to study and scores of students.

Preparing the data

In [28]:
```
X = st_data.iloc[:, :-1].values#obtaining an array of our independent variable x, i.e.,hours
y = st_data.iloc[:, 1].values#obtaining an array of our dependent variable y, i.e.,scores
```

Here, we have obtained our attributes and labels.

In [29]:
```
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)
```

Here, we have split this data into training and test sets from the obtained attributes and levels by using the Scikit-Learn's built-in train_test_split() method.

Training the Algorithm

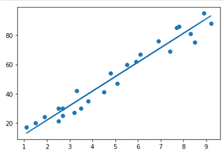
In [30]:
```
from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
regressor.fit(X_train, y_train)

print("Training of the model is complete.")
```

```
Training of the model is complete.
```

Here we have trained our algorithm using the training and test sets.

In [31]:
```
# to plot the regression line of the obtained model
line = regressor.coef_*X+ regressor.intercept_
# to plot the test data
plt.scatter(X, y)
plt.plot(X, line);
plt.show()
```



In [32]:
```
print(X_test) # Testing data - In Hours
```

```
[[1.5]
 [3.2]
 [7.4]
 [2.5]
 [5.9]]
```

In [33]:
```
y_pred = regressor.predict(X_test) # Predicting the scores
y_pred # predicted values
```

Out[33]:
```
array([16.88414476, 33.73226078, 75.357018  , 26.79480124, 60.49103328])
```

Here we have obtained the array of the predicted scores.

In [34]:
```
# Comparing Actual score vs Predicted score.
df = pd.DataFrame({'Actual': y_test, 'Predicted': y_pred})
df
```

Out[34]:

	Actual	Predicted
0	20	16.884145
1	27	33.732261
2	69	75.357018
3	30	26.794801
4	62	60.491033

Predicting score of a student who studies for 9.25 hrs/ day

In [37]:
```
hours = [[9.25]]
own_pred = regressor.predict(hours)
df1 = pd.DataFrame({'Hours': 9.25, 'Predicted Score': own_pred })
df1
```

Out[37]:

	Hours	Predicted Score
0	9.25	93.691732

Hence, the predicted score of a student who studies for 9.25 hrs/ day is 93.69

Evaluating the model

In [38]:
```
from sklearn import metrics
print('Mean Absolute Error:',
      metrics.mean_absolute_error(y_test, y_pred))
```

```
Mean Absolute Error: 4.183859899002975
```

Here we have evaluated the model using the mean square error and it has been observed that the model has an error of 4.183

Conclusion

Basically, in the analysis, firstly we followed a simple linear regression as the data contained just two variables.We found that there is a high correlation between the hours of study and scores of the students.Later using the Python Scikit-Learn library for machine learning we, implemented regression functions and after the training of the model, we predicted the scores of the students. Hence we predicted that a student studying 9.25 hrs/ day, the score will be 93.69.