


TASK GIVEN:

Prediction using Supervised ML (Level – Beginner)

- 
- A decorative graphic on the left side of the slide. It features a large blue hexagon with the text "#1" in white. Surrounding it are several smaller hexagons and icons: a lightbulb, a thumbs up, a smartphone, a magnifying glass, a gear, and a speech bubble.
- Predict the percentage of an student based on the no. of study hours.
 - This is a simple linear regression task as it involves just 2 variables.
 - You can use R, Python, SAS Enterprise Miner or any other tool
 - Data can be found at <http://bit.ly/w-data>
 - What will be predicted score if a student studies for 9.25 hrs/ day?
 - Sample Solution : <https://bit.ly/2HxiGGI>
 - Task submission:
 1. Host the code on GitHub Repository (public). Record the code and output in a video. Post the video on YouTube
 2. Share links of code (GitHub) and video (YouTube) as a post on **YOUR LinkedIn profile**, not TSF Network.
 3. Submit the LinkedIn link in Task Submission Form when shared.

ALLOTTED DATA:

Hours	Scores
2.5	21
5.1	47
3.2	27
8.5	75
3.5	30
1.5	20
9.2	88
5.5	60
8.3	81
2.7	25
7.7	85
5.9	62
4.5	41
3.3	42
1.1	17
8.9	95
2.5	30
1.9	24
6.1	67
7.4	69
2.7	30
4.8	54
3.8	35
6.9	76
7.8	86

ANALYSIS:

Linear Regression with Python Scikit Learn

In this section we will see how the Python Scikit-Learn library for machine learning can be used to implement regression functions. We will start with simple linear regression involving two variables.

Simple Linear Regression

In this regression task we will predict the percentage of marks that a student is expected to score based upon the number of hours they studied. This is a simple linear regression task as it involves just two variables.

Author - Abhishek Vishwakarma

Step 1: Importing Libraries

In [4]:

```
import pandas as pd
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error
import matplotlib.pyplot as plt
```

Step 2: Reading Data

In [5]:

```
data = pd.read_csv("http://bit.ly/w-data")
data.head(10)
```

Out[5]:

	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

Step 3: Data Preprocessing

In [3]:

```
#Checking for missing value
data.isnull().sum()
```

Out[3]:

```
Hours      0
Scores     0
dtype: int64
```

In [4]:

```
#Getting more info
data.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
```

Step 4: Setting Dependent and Independent Variable

In [5]:

```
x = data['Hours']
y = data['Scores']
x = x.values.reshape(len(x),1)
y = y.values.reshape(len(y),1)
```

Step 5: Building Model

In [6]:

```
#Splitting data into training and test set
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=0)

#Making instance
lr = LinearRegression()

#Fitting Model
lr.fit(x_train, y_train)
```

Out[6]:

```
LinearRegression()
```

Step 6: Evaluating Mean Squared Error

In [7]:

```
y_pred = lr.predict(x_test)

mse = mean_squared_error(y_test,y_pred)

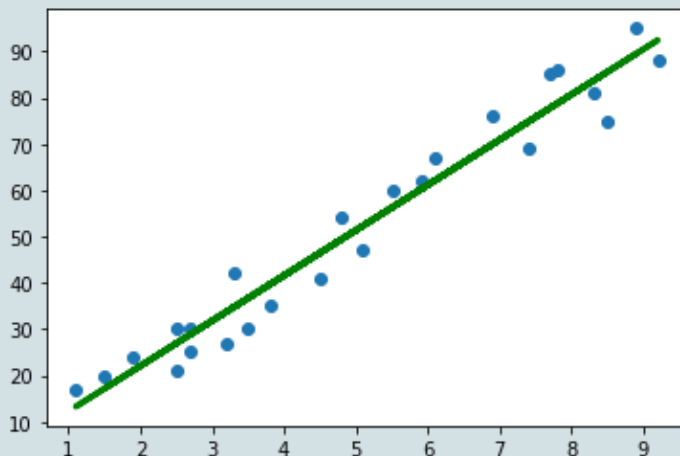
print("Mean Square Error = ", mse)
```

Mean Square Error = 21.5987693072174

Step 7: Plotting Best Fit Line

In [16]:

```
line = lr.intercept_ + lr.coef_ * X
plt.scatter(X, y)
plt.plot(X, line, color='green', linewidth=3);
plt.show()
```



Step 8: Making Prediction

In [17]:

```
#Predicting for x = 9.25
Y = lr.intercept_ + lr.coef_ * 9.25

print("Predicted Score if a student studies for 9.25 hrs/day = ", Y)
```

Predicted Score if a student studies for 9.25 hrs/day = [[92.90985477]]

THANK

YOU