```
  The Sparks Foundation - Data Science & Business Analytics Internship
  ====================================================================
```

```
TASK 1 - Prediction using Supervised Machine Learning

In this task it is required to predict the percentage of a student on the basis of number
of hours studied using the Linear Regression supervised machine learning algorithm.

Author:THOTA VARUN KUMAR

STEP 1 - Importing the dataset
==============================
```

```
# Importing all the required libraries(pandas,numpy,seaborn)

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns

# To ignore the warnings
import warnings as wg
wg.filterwarnings("ignore")
```

```
# Reading data from remote link

url = "https://raw.githubusercontent.com/AdiPersonalWorks/Random/master/student_scores%20-
df = pd.read_csv(url)
print(df)
```

```
         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
     10    7.7      85
     11    5.9      62
     12    4.5      41
     13    3.3      42
     14    1.1      17
     15    8.9      95
     16    2.5      30
     17    1.9      24
     18    6.1      67
     19    7.4      69
     20    2.7      30
     21    4.8      54
     22    3.8      35
     23    6.9      76
     24    7.8      86
```

```
# To find more information about our dataset
df.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
```

```
df.describe()
```

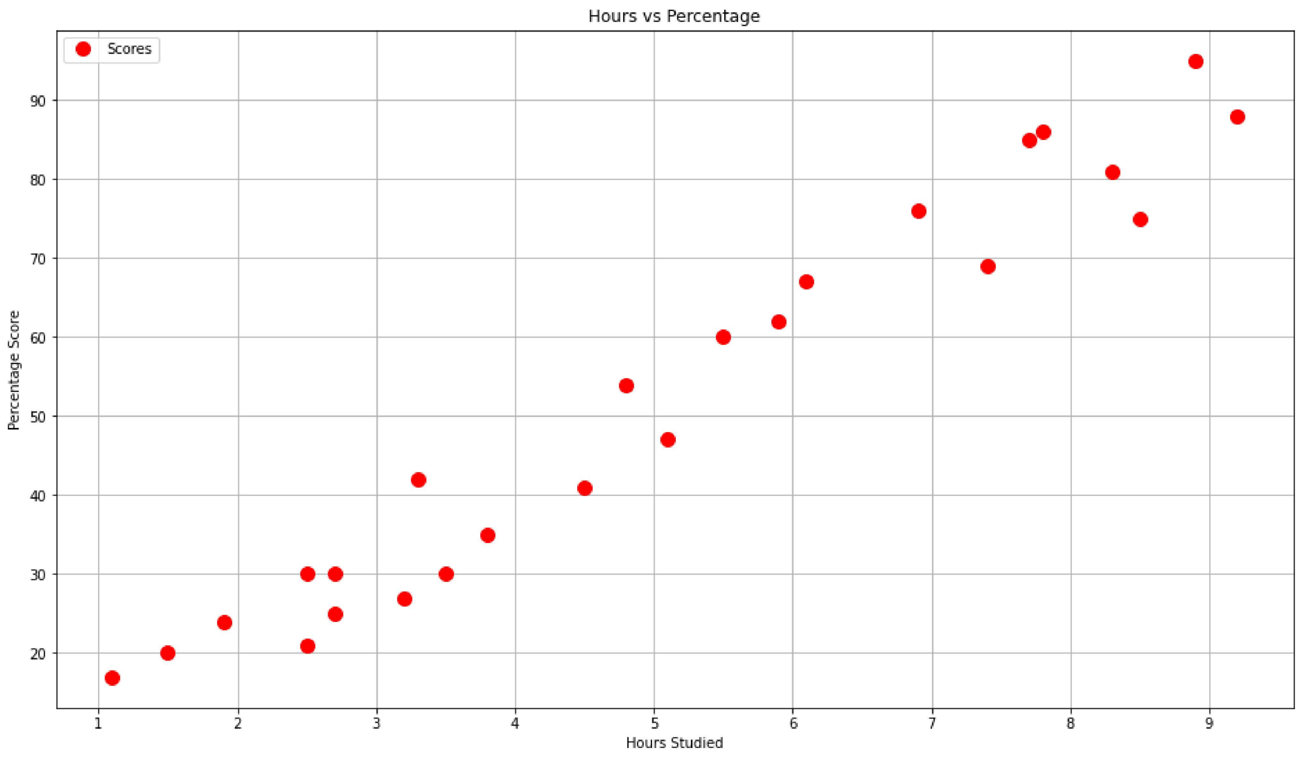
	Hours	**Scores**
count	25.000000	25.000000
mean	5.012000	51.480000
std	2.525094	25.286887
min	1.100000	17.000000
25%	2.700000	30.000000
50%	4.800000	47.000000
75%	7.400000	75.000000
max	9.200000	95.000000

```
# now we will check if our dataset contains null or missings values
df.isnull().sum()
```

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

```
STEP 2 - Visualizing the dataset
================================
```

```
# Plotting the dataset
plt.rcParams["figure.figsize"] = [16,9]
df.plot(x='Hours', y='Scores', style='.', color='red', markersize=20)
plt.title('Hours vs Percentage')
plt.xlabel('Hours Studied')
plt.ylabel('Percentage Score')
plt.grid()
plt.show()
print("From the graph below, we can observe that there is a linear relationship between ho
```



```
From the graph below, we can observe that there is a linear relationship between hour
```

```
# we can find corelation between between the variables using .corr
df.corr()
```

	Hours	Scores
Hours	1.000000	0.976191
Scores	0.976191	1.000000

```
STEP 3 - Data preparation
=========================
```

```
df.head()
```

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

```
# using iloc function we will divide the data
X = df.iloc[:, :1].values
y = df.iloc[:, 1:].values
```

```
X
```

```
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]])
```

```
y
```

```
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]])
```

```
# Splitting data into training and testing data

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)
```

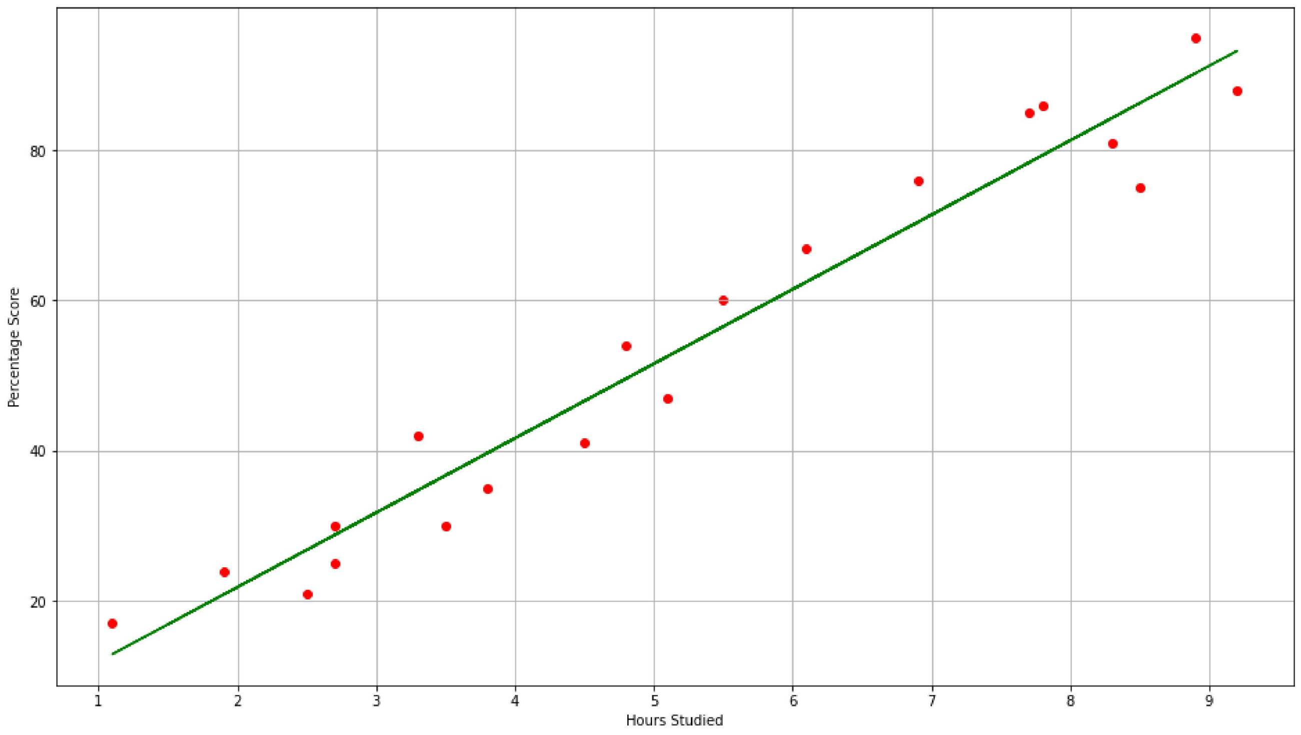
```
STEP 4 - Training the Algorithm
===============================
```

```
from sklearn.linear_model import LinearRegression

model = LinearRegression()
model.fit(X_train, y_train)
```

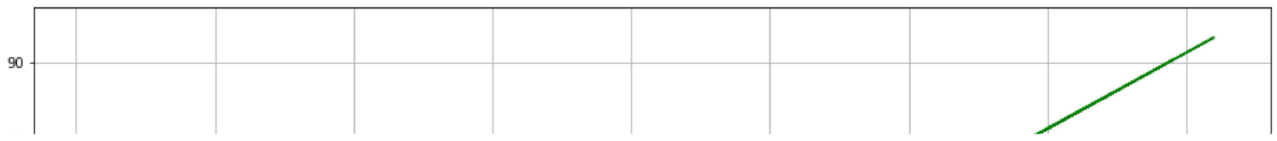
```
     LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
```

```
STEP 5 - Visualizing the model
==============================
```

```
line = model.coef_*X + model.intercept_

# Plotting for the training data
plt.rcParams["figure.figsize"] = [16,9]
plt.scatter(X_train, y_train, color='red')
plt.plot(X, line, color='green');
plt.xlabel('Hours Studied')
plt.ylabel('Percentage Score')
plt.grid()
plt.show()
```



```
# Plotting for the testing data
plt.rcParams["figure.figsize"] = [16,9]
plt.scatter(X_test, y_test, color='red')
plt.plot(X, line, color='green');
plt.xlabel('Hours Studied')
plt.ylabel('Percentage Score')
plt.grid()
plt.show()
```



```
STEP 6 - Making Predictions
===========================
```

```
print(X_test) # Testing data - In Hours
y_pred = model.predict(X_test) # Predicting the scores
```

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

```
# Comparing Actual vs Predicted

y_test
```

```
array([[20],
       [27],
       [69],
       [30],
       [62]])
```

```
y_pred
```

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

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

	Actual	Predicted
0	[[20], [27], [69], [30], [62]]	[[16.884144762398023], [33.732260779489835], [...

```
# Testing with your own data

hours = 9.25
own_pred = model.predict([[hours]])
print("The predicted score if a person studies for",hours,"hours is",own_pred[0])
```

```
The predicted score if a person studies for 9.25 hours is [93.69173249]
```

✓ 0s completed at 16:05