# Anushka Joshi

## Data science and business analytics task 1

## predicting the percentage of a student based on study hours

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

In [2]:
```
url = "http://bit.ly/w-data"
df = pd.read_csv(url)
df.head()
```

Out[2]:

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

In [3]:
```
df.head(10)
```

Out[3]:

| | 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 |

In [4]:
```
df.shape
```

Out[4]:
```
(25, 2)
```

In [5]:
```
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
```

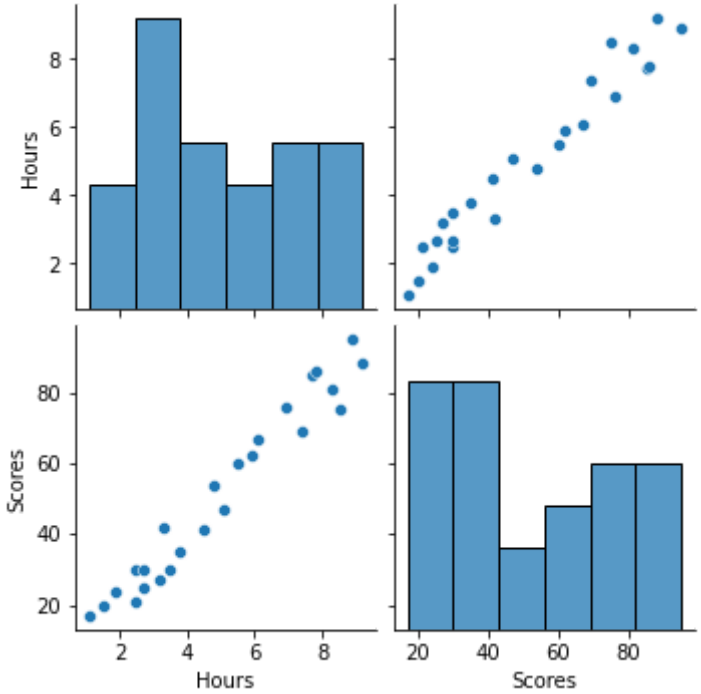
In [6]:
```
df.describe()
```

Out[6]:

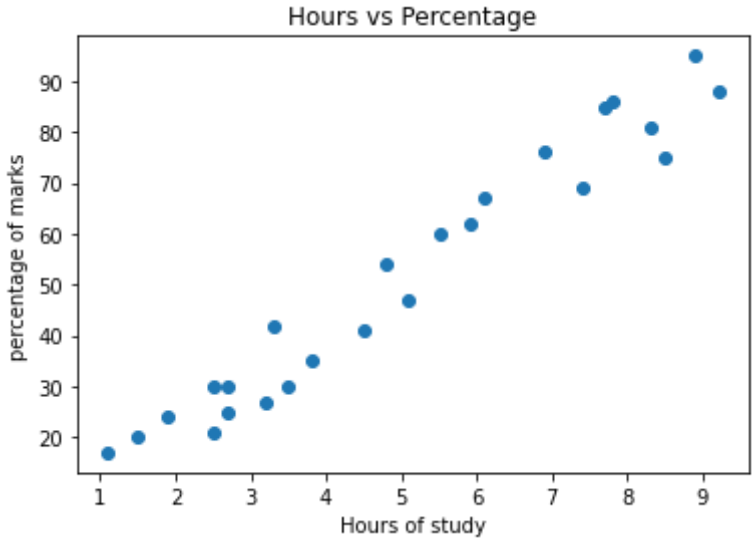
| | Hours | Scores |
|---|---|---|
| **count** | 25.000000 | 25.000000 |
| **mean** | 5.012000 | 51.480000 |
| **std** | 2.525094 | 25.286887 |
| **min** | 1.100000 | 17.000000 |
| **25%** | 2.700000 | 30.000000 |
| **50%** | 4.800000 | 47.000000 |
| **75%** | 7.400000 | 75.000000 |
| **max** | 9.200000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

In [7]:
```
sns.pairplot(df)
```

Out[7]:
```
<seaborn.axisgrid.PairGrid at 0x158782d7eb0>
```



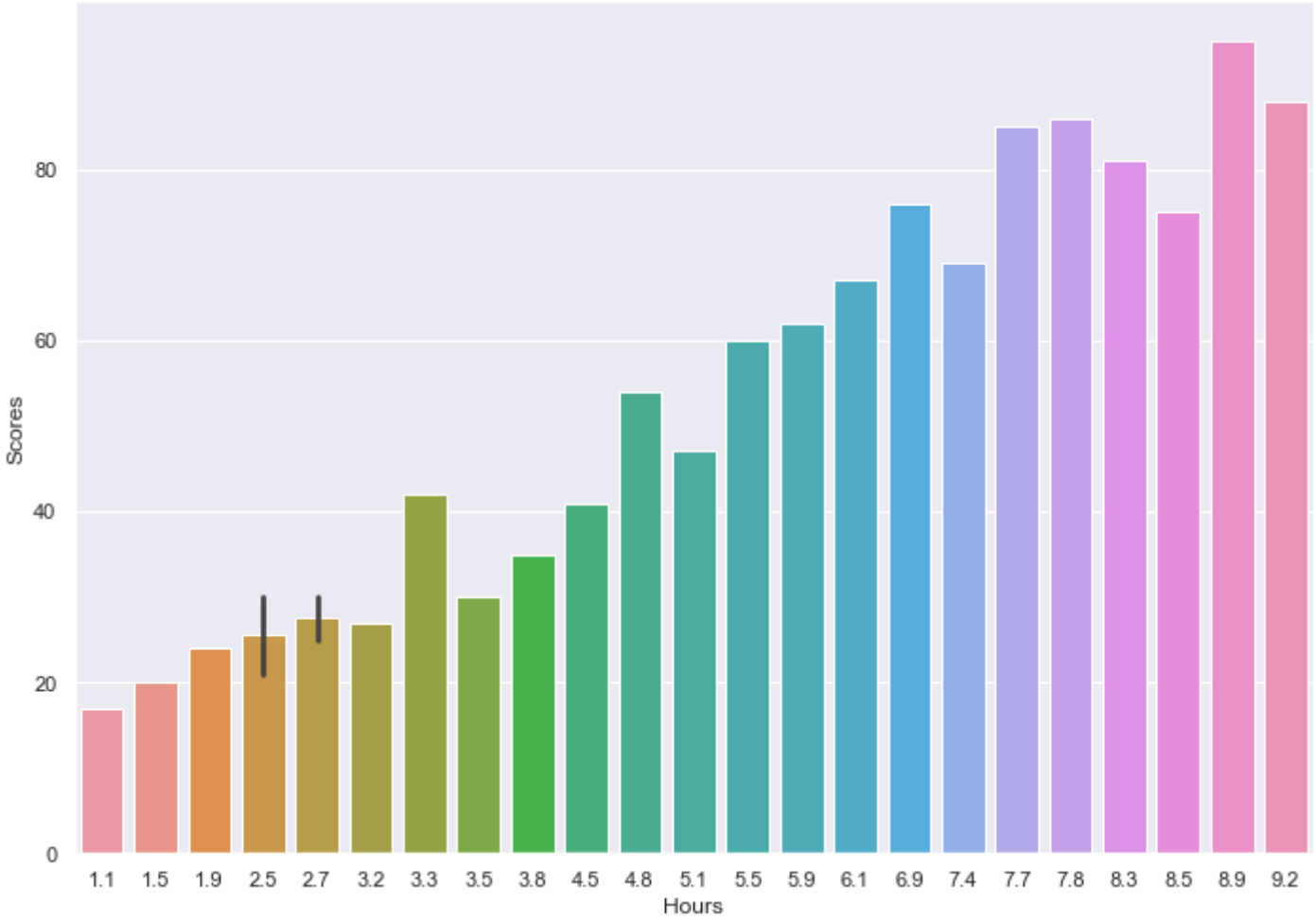
In [8]:
```
plt.scatter(df['Hours'], df['Scores'])
plt.title('Hours vs Percentage')
plt.xlabel('Hours of study')
plt.ylabel('percentage of marks')
plt.show()
```



In [9]:
```
sns.set(rc={'figure.figsize':(11.7,8.27)})


res = sns.barplot(df['Hours'],df['Scores'])
plt.show()
```

```
C:\Users\devj7\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variables as
keyword args: x, y. From version 0.12, the only valid positional argument will be `data`, and passing other argument
s without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```



In [10]:
```
X = df.iloc[:, :-1].values
y = df.iloc[:,-1].values
```

In [11]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y,
                            test_size=0.25, random_state=6)
```

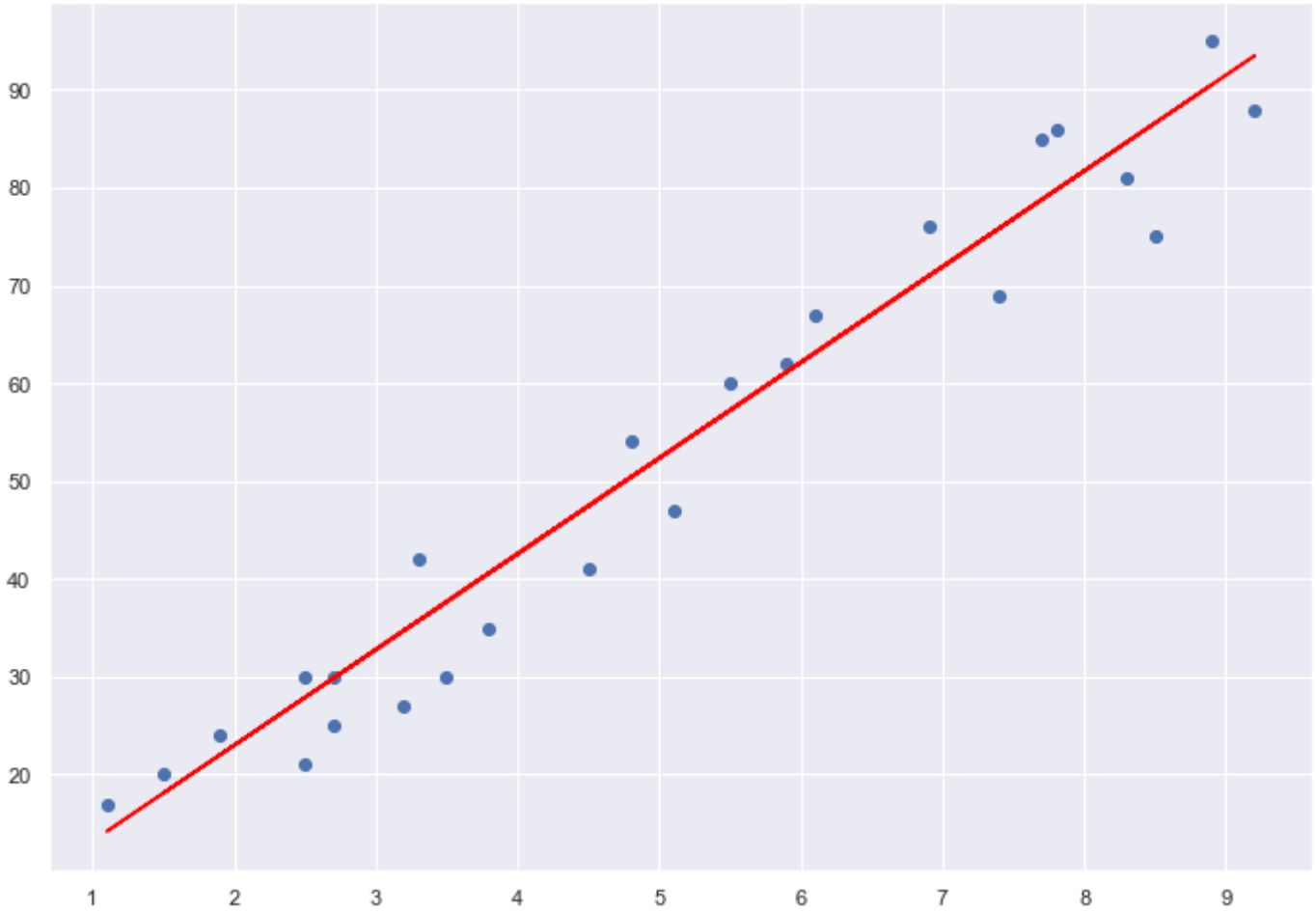
In [12]:
```
from sklearn.linear_model import LinearRegression
lr = LinearRegression()
```

In [13]:
```
lr.fit(X_train, y_train)
```

Out[13]:
```
▾ LinearRegression

LinearRegression()
```

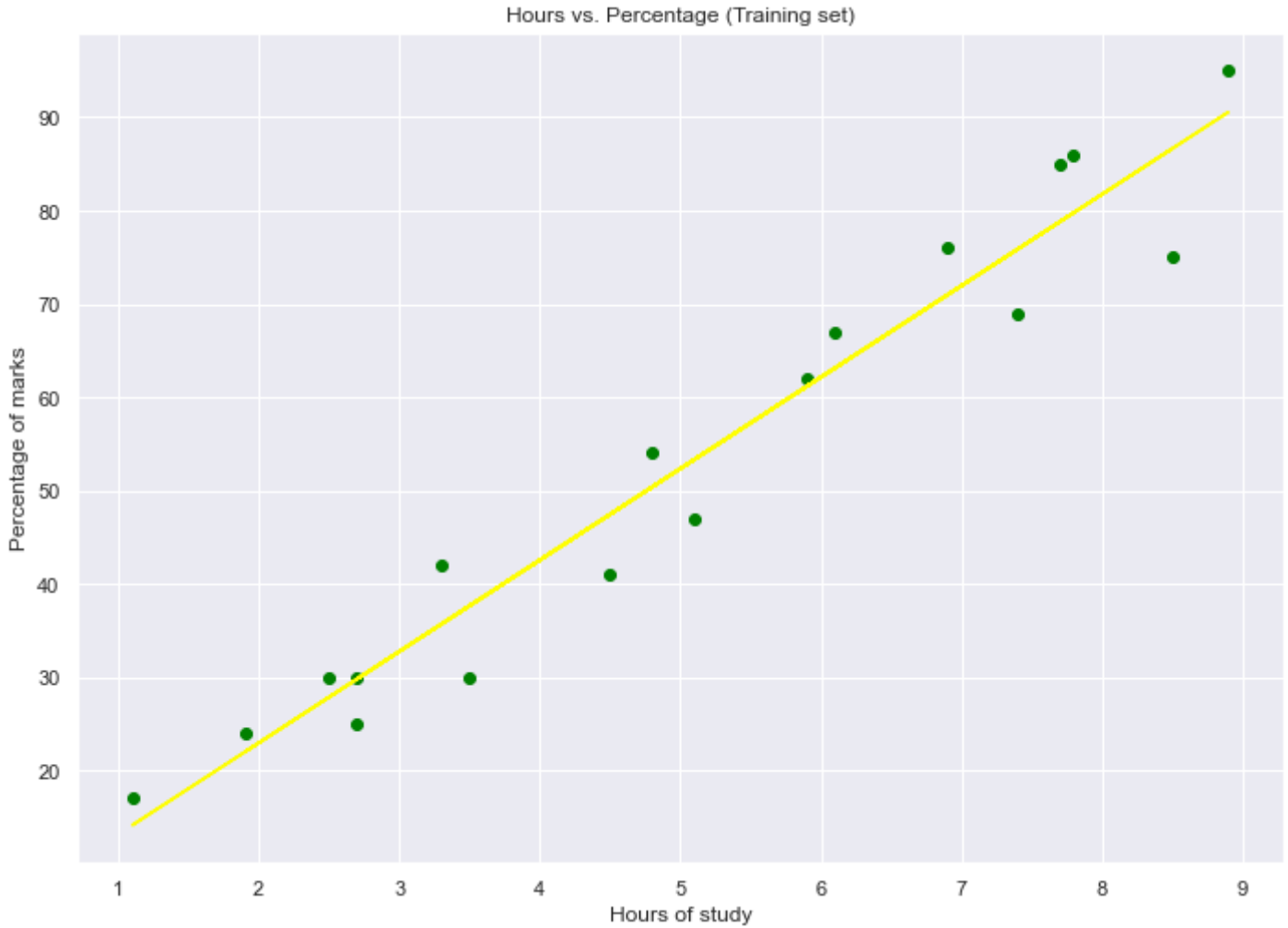
In [14]:
```
line = lr.coef_*X+lr.intercept_
plt.scatter(X, y)
plt.plot(X, line,color = 'red');
plt.show()
```



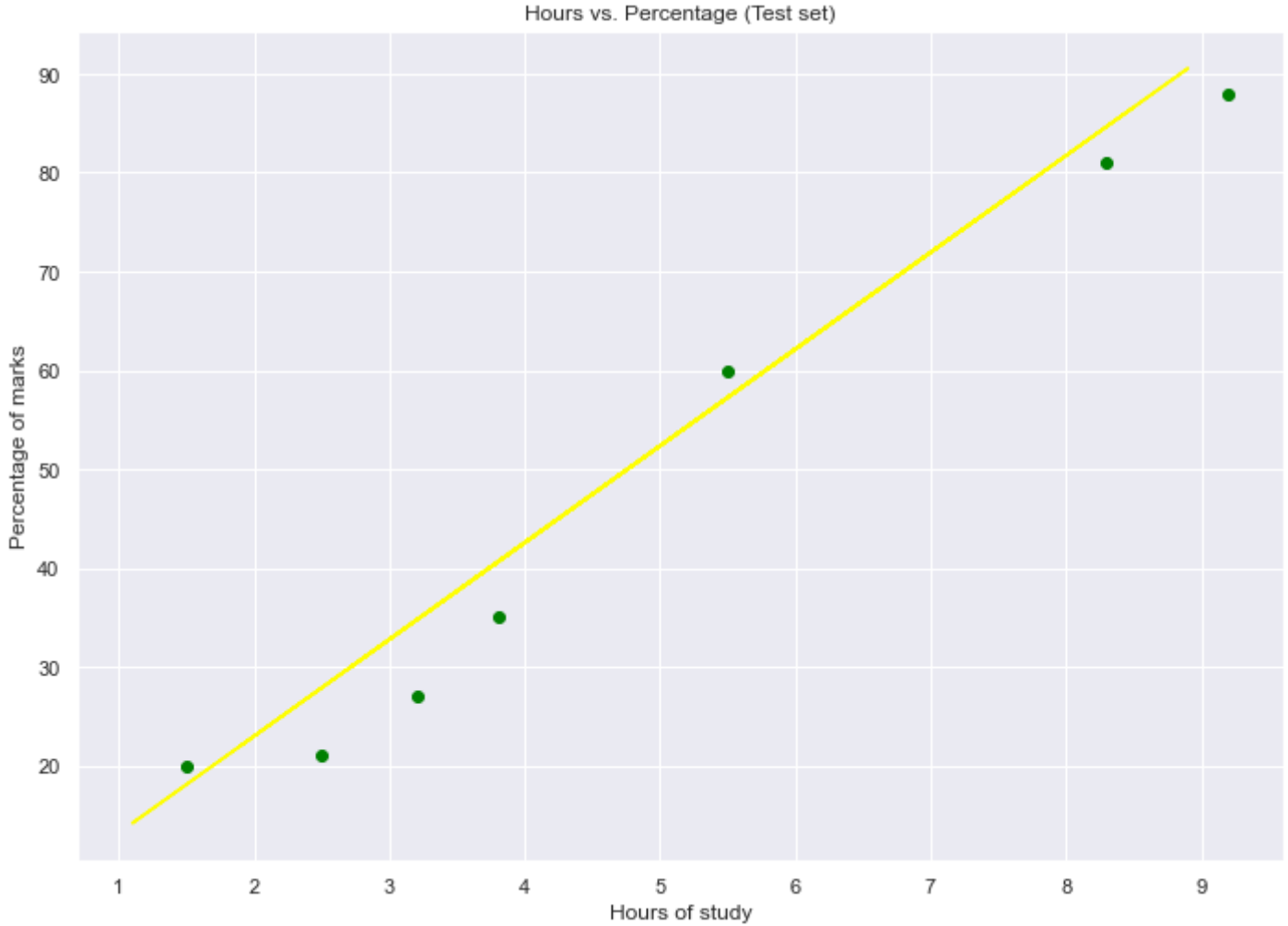

In [15]:
```
y_pred = lr.predict(X_test)
print(y_pred)
```

```
[40.59959726 57.24380354 18.08096524 93.46942896 84.65779035 34.72517152
 27.87167482]
```

In [16]:
```
plt.scatter(X_train, y_train, color = 'green')
plt.plot(X_train, lr.predict(X_train), color = 'yellow')
plt.title('Hours vs. Percentage (Training set)')
plt.xlabel('Hours of study')
plt.ylabel('Percentage of marks')
plt.show()
```



In [17]:
```
plt.scatter(X_test, y_test, color = 'green')
plt.plot(X_train, lr.predict(X_train), color = 'yellow')
plt.title('Hours vs. Percentage (Test set)')
plt.xlabel('Hours of study')
plt.ylabel('Percentage of marks')
plt.show()
```



In [18]:
```
dataset = pd.DataFrame({'Actual': y_test, 'Predicted': y_pred})
print(dataset.to_markdown())
```

```
|    |   Actual |   Predicted |
|---:|---------:|------------:|
|  0 |       35 |     40.5996 |
|  1 |       60 |     57.2438 |
|  2 |       20 |     18.081  |
|  3 |       88 |     93.4694 |
|  4 |       81 |     84.6578 |
|  5 |       27 |     34.7252 |
|  6 |       21 |     27.8717 |
```

In [19]:
```
dataset = np.array(7.25)
dataset = dataset.reshape(-1, 1)
pred = lr.predict(dataset)
print("If a student studies for 7.25 hours per day, the score is {}.".format(pred))
```

```
If a student studies for 7.25 hours per day, the score is [74.37754529].
```

# Mean Absolute Error (MAE)

# Mean Squared Error (MSE)

# Root Mean Squared Error (RMSE)

# R Squared (R2)

In [20]:
```
from sklearn.metrics import mean_absolute_error
print("MAE",mean_absolute_error(y_test,y_pred))
```

```
MAE 4.856984875410573
```

In [21]:
```
from sklearn.metrics import mean_squared_error
print("MSE",mean_squared_error(y_test,y_pred))
```

```
MSE 27.546725152852122
```

In [22]:
```
print("RMSE",np.sqrt(mean_squared_error(y_test,y_pred)))
```

```
RMSE 5.248497418581067
```

In [23]:
```
from sklearn.metrics import r2_score
r2 = r2_score(y_test,y_pred)
print(r2)
```

```
0.9610742434972386
```

# Fin.

In [ ]:

In [ ]: