TASK-1

Author: ISHWARYA D

THE SPARKS FOUNDATION INTERNSHIP

PREDECTION USING SUPERVISED ML

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.

Predict the percentage of an student based on the no. of study hours.

In [1]:
```
#import the libraries;
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
```

In [2]:
```
data = pd.read_csv('marks_csv')
data.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]:
```
data.describe()
```

Out[3]:

	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 [4]:
```
data.tail(9)
```

Out[4]:

	Hours	Scores
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

In [5]:
```
data.dtypes
```

Out[5]:
```
Hours      float64
Scores       int64
dtype: object
```

In [6]:
```
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
```

mean and std of Scores and Hours

In [7]:
```
data['Hours'].mean()
```

Out[7]:
```
5.012
```

In [8]:
```
data['Hours'].std()
```

Out[8]:
```
2.5250940576540906
```

In [9]:
```
data['Scores'].std()
```

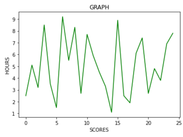
Out[9]:
```
25.286887824747802
```

In [10]:
```
data['Scores'].mean()
```

Out[10]:
```
51.48
```

PLOTTING THE GRAPH

In [11]:
```
plt.plot(data['Hours'],color='green')
plt.xlabel('SCORES')
plt.ylabel('HOURS')
plt.title("GRAPH")
plt.show()
```



CATEGORISING DATA USING PIECHART

In [12]:
```
marks=(27,60,45,89,90)
```

In [13]:
```
labels=('bad','good','excellent','fair','Very good')
```

In [14]:
```
plt.pie(marks,labels=labels)
```

Out[14]:
```
([<matplotlib.patches.Wedge at 0x1e37662af10>,
  <matplotlib.patches.Wedge at 0x1e376638400>,
  <matplotlib.patches.Wedge at 0x1e376638880>,
  <matplotlib.patches.Wedge at 0x1e376638d00>,
  <matplotlib.patches.Wedge at 0x1e3766441c0>],
 [Text(1.0593339257551518, 0.2963112175216724, 'bad'),
  Text(0.4477486355294599, 1.0047493017571631, 'good'),
  Text(-0.6581941915970181, 0.8813514657320018, 'excellent'),
  Text(-1.0924749692906918, -0.4528177734427245, 'fair'),
  Text(0.6758648206750973, -0.8678748436115766, 'Very good')])
```

PIVOT TABLE FOR HOURS AND SCORES

In [15]:
```
grades_mean=data.pivot_table(values='Scores',columns='Hours',aggfunc=np.mean)
grades_mean
```

Out[15]:

Hours	1.1	1.5	1.9	2.5	2.7	3.2	3.3	3.5	3.8	4.5	...	5.9	6.1	6.9	7.4	7.7	7.8	8.3	8.5	8.9	9.2
Scores	17.0	20.0	24.0	25.5	27.5	27.0	42.0	30.0	35.0	41.0	...	62.0	67.0	76.0	69.0	85.0	86.0	81.0	75.0	95.0	88.0

1 rows × 23 columns

Splitting the data

In [16]:
```
# from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(data['Hours'].values.reshape(-1,1), data['Scores'], test_size = 0.2, random_state = 42)
X_train.shape, y_train.shape, X_test.shape, y_test.shape
```

Out[16]:
```
((20, 1), (20,), (5, 1), (5,))
```

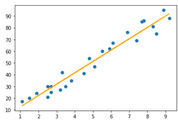
BEST-FIT LINEAR REGRESSION PLOT

In [17]:
```
from sklearn.linear_model import LinearRegression
model = LinearRegression()
model.fit(X_train, y_train)
```

Out[17]:
```
LinearRegression()
```

In [18]:
```
coefficient = model.coef_
intercept = model.intercept_
line = (data['Hours'].values * coefficient) + intercept
```

In [19]:
```
plt.scatter(data.Hours, data.Scores)
plt.plot(data.Hours, line,color='orange')
plt.show()
```



Actual Values Vs Predicted Values

In [20]:
```
pred = model.predict(X_test)
pred
```

Out[20]:
```
array([83.18814104, 27.03208774, 27.03208774, 69.63323162, 59.95115347])
```

In [21]:
```
pred_compare = pd.DataFrame({'Actual Values': y_test, 'Predicted Values':pred})
pred_compare
```

Out[21]:

	Actual Values	Predicted Values
8	81	83.188141
16	30	27.032088
0	21	27.032088
23	76	69.633232
11	62	59.951153

MODEL EVALUATION

In [22]:
```
from sklearn import metrics
print("Mean Absolute Error: ", metrics.mean_absolute_error(y_test, pred))
print("Mean Squared Error: ", metrics.mean_squared_error(y_test, pred))
print("Root Mean Squared Error: ", metrics.mean_squared_error(y_test, pred)**0.5)
print("R2 Score: ", metrics.r2_score(y_test, pred))
```

```
Mean Absolute Error:  3.9207511902099244
Mean Squared Error:  18.943211722315272
Root Mean Squared Error:  4.352380006653288
R2 Score:  0.9678055545167994
```

What will be predicted score if a student studies for 9.25 hrs/ day?

In [23]:
```
hours = np.asarray(9.25).reshape(-1,1)
print(f"{model.predict(hours)[0]} will be the predicted score if a student study for 9.25 hrs/day.")
```

```
92.38611528261494 will be the predicted score if a student study for 9.25 hrs/day.
```

In [ ]:
```

```