The Spark Foundation

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

Step 1: Importing Python lIibrary

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

Step 2 : Importing data-set

In [2]:
```
data = pd.read_csv("data.csv")
print("Importing Data Successfully")
```

```
Importing Data Successfully
```

Step 3 : check whether Data imported successfully or not ?

In [3]:
```
print("For this we print first 10 data of Data-set ")
data.head(10)
```

```
For this we print first 10 data of Data-set
```

Out[3]:

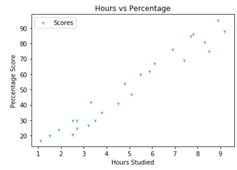
	Hours	Scores
0	2.5	21
1	5.1	47
2	3.2	27
3	8.5	75
4	3.5	30
5	1.5	20
6	9.2	88
7	5.5	60
8	8.3	81
9	2.7	25

In [4]:
```
print("You have correctly imported Data-set")
```

```
You have correctly imported Data-set
```

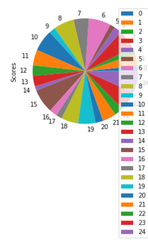
Step 4 : Plot the graph 📊 ,for detail analysis of Data-set

In [5]:
```
data.plot(x='Hours',y='Scores',style='1')
plt.title('Hours vs Percentage')
plt.xlabel('Hours Studied')
plt.ylabel('Percentage Score')
plt.show()
```



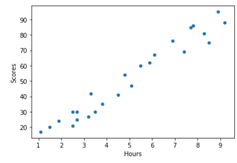
In [6]:
```
data.plot.pie(x='Hours',y='Scores')
```

Out[6]:
```
<matplotlib.axes._subplots.AxesSubplot at 0xbba650>
```



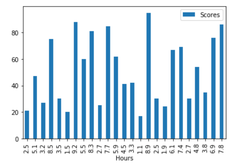
In [7]:
```
data.plot.scatter(x='Hours',y='Scores')
```

Out[7]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x4a67730>
```



In [8]:
```
data.plot.bar(x='Hours',y='Scores')
```

Out[8]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x4a677b0>
```



After ploting different graph , we have observed that as Study Hours increases , score is also increase. Which is good sign of correct data. In our daily life , we have observe the same phenomenon.

Step 5 : Now, we have prepared tha data for our model

In [9]:
```
X = data.iloc[:, :-1].values
y = data.iloc[:, 1].values
```

Step 6 : Now , we have divide the data for tarining & testing the model

In [10]:
```
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)
```

Training the Algorithm

In [11]:
```
from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
regressor.fit(X_train, y_train)
print("Training complete.")
```

```
Training complete.
```

Step 7 : Now, our Model is ready . Its time to test it .

In [12]:
```
print(X_test)
print("Predection of Score")
y_pred = regressor.predict(X_test)
print(y_pred)
```

```
[[1.5]
 [3.2]
 [7.4]
 [2.5]
 [5.9]]
Predection of Score
[16.88414476 33.73226078 75.357018   26.79480124 60.49103328]
```

Step 8 : Now, Checking the accuracy of our model

In [13]:
```
df = pd.DataFrame({'Actual': y_test, 'Predicted': y_pred})

df
```

Out[13]:

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

Step 9 : Now, Its times to prediction with custom input

In [14]:
```
hours = [[9.25]]
pred = regressor.predict(hours)
print(pred)
```

```
[93.69173249]
```

Evaluating the model

In [15]:
```
from sklearn import metrics
print('Mean Absolute Error:',
      metrics.mean_absolute_error(y_test, y_pred))
```

```
Mean Absolute Error: 4.183859899002975
```

In [ ]: