Task 1: Prediction using supervised ML

Importing the Dependencies

In [20]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import sklearn
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
```

Data Collection from link

In [21]:
```
url = "http://bit.ly/w-data"
ds = pd.read_csv(url)
print("Data is Imported")
```

```
Data is Imported
```

In [22]:
```
ds.shape
```

Out[22]: (25, 2)

Print Few Rows of Dataset using head function

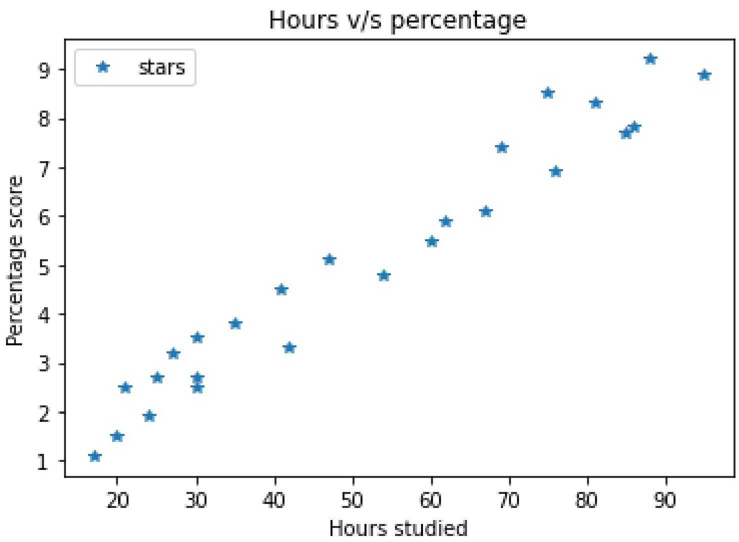
In [23]:
```
ds.head()
```

Out[23]:

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

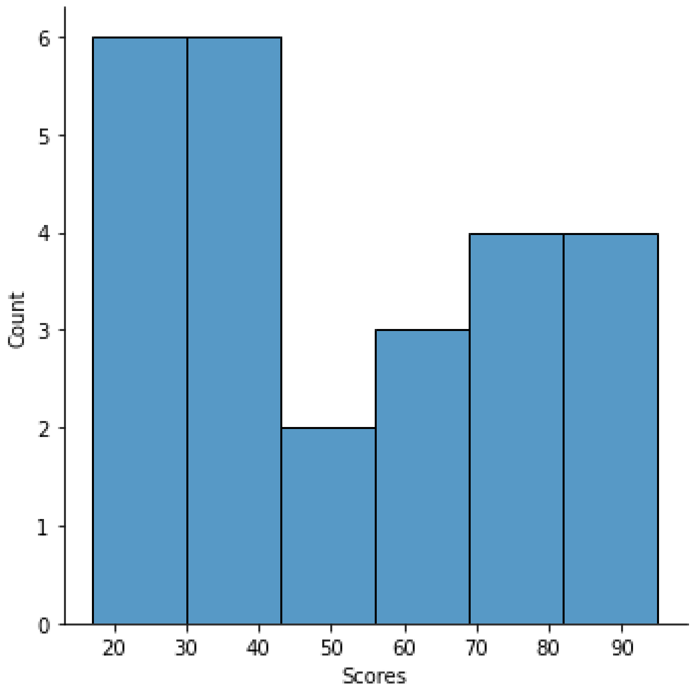
Print a plot and visualized it with respect of x and y axis. Here, x represent scores and y represent hours .

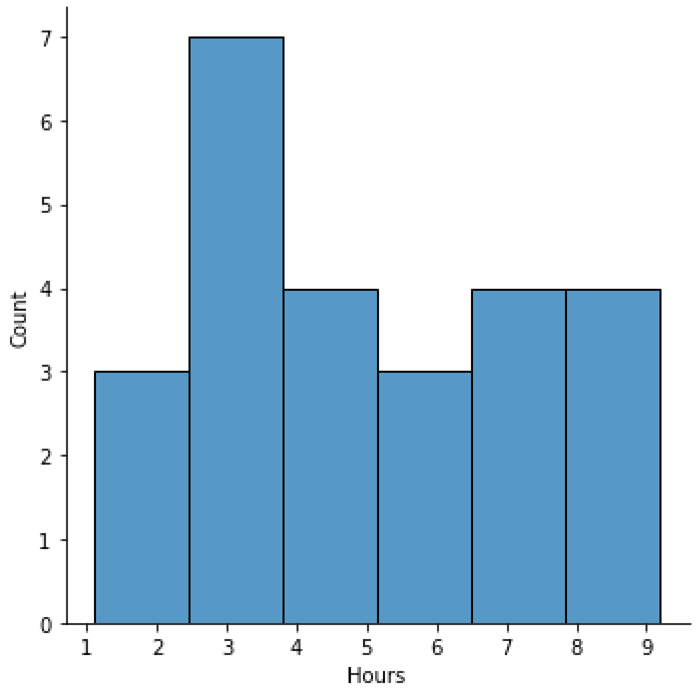
In [24]:
```
#Plotting the distribution of scores

ds.plot(x = "Scores", y="Hours", label="stars", style="*")
plt.title("Hours v/s percentage")
plt.xlabel("Hours studied")
plt.ylabel("Percentage score")

plt.show()
```



Vizualisation through Bar Graph

In [25]:
```
sns.displot(ds["Scores"])
sns.displot(ds["Hours"])
plt.show()
```





In [26]:

```
ds.isnull().sum()
```

Out[26]:

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

Selecting Row & column by their position

In [27]:

```
X= ds.iloc[:,:-1].values
Y= ds.iloc[:,1].values
Y
```

Out[27]:

```
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], dtype=int64)
```

Training the Model and plotting the Regression line

In [28]:

```
#Splitting the data
Y_train, Y_test, X_train, X_test = train_test_split(Y,X,test_size=0.2, random_state=
#The train-test split is a technique for evaluating the performance of machine learn
```

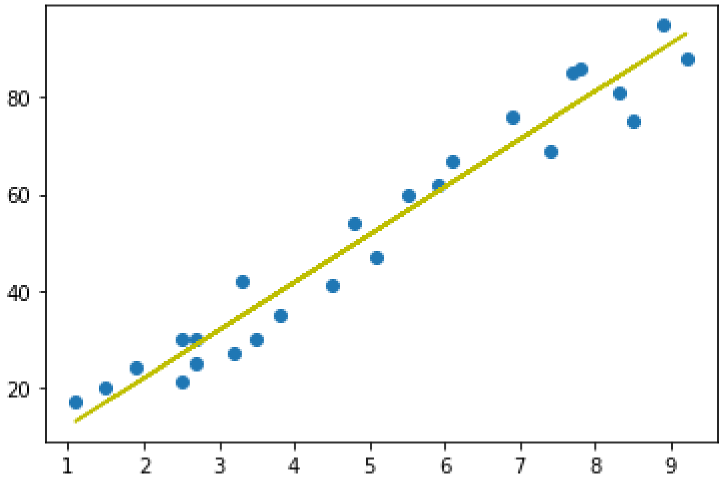
In [29]:

```
regressor = LinearRegression()
regressor.fit(X_train,Y_train)
print("Training Completed..")
```

```
Training Completed..
```

In [30]:

```
#Plotting the regressor line
line = regressor.coef_*X+regressor.intercept_
plt.scatter(X,Y)
plt.plot(X,line, color='y');
plt.show()
```



In [31]:

```
#Prediction Making
print(X_test)
Y_predict = regressor.predict(X_test)
```

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

Compair the prediction model result

In [33]:

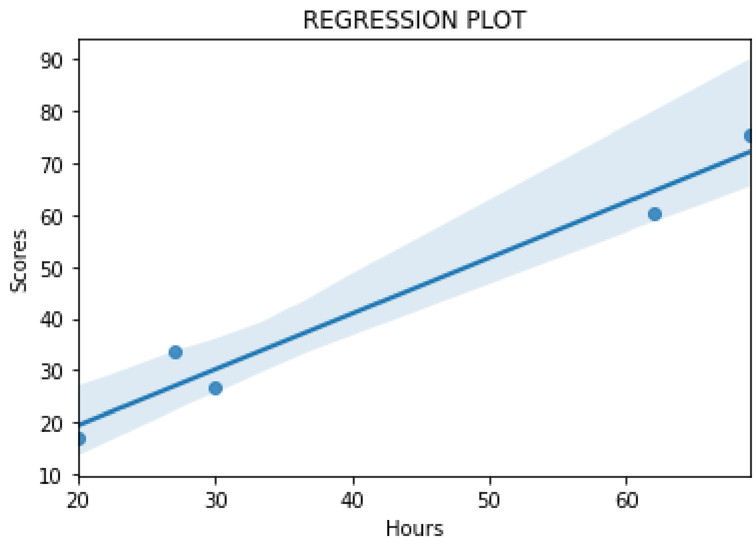
```
ds =pd.DataFrame({'Actual': Y_test, 'Predicted':Y_predict})
ds
```

Out[33]:

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

In [35]:

```
sns.regplot(x=Y_test, y=Y_predict)
plt.xlabel("Hours")
plt.ylabel("Scores")
plt.title("REGRESSION PLOT")
plt.show()
```



Check the performance of training nd testing score

In [37]:

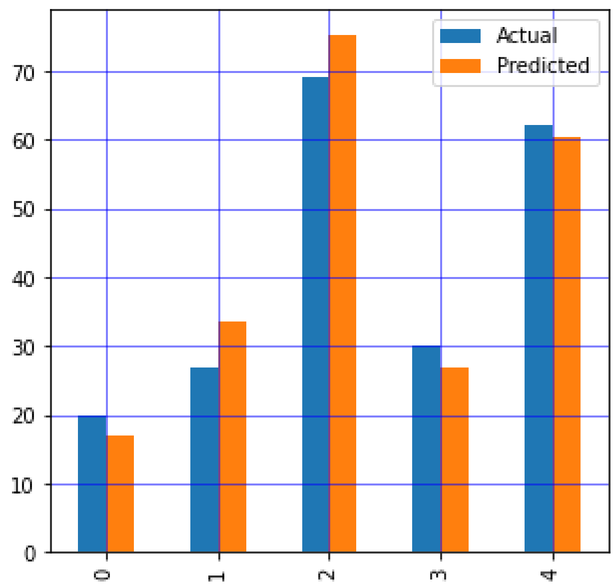
```
print("Training Score:",regressor.score(X_train,Y_train))
print("Test Score:",regressor.score(X_test,Y_test))
```

```
Training Score: 0.9515510725211552
Test Score: 0.9454906892105356
```

Plotting a bar graph to depict the difference b/w actual and predicted value

In [38]:

```
ds.plot(kind='bar',figsize=(5,5))
plt.grid(which='major',linewidth='0.5',color='blue')
plt.grid(which='minor',linewidth='0.5',color='red')
plt.show()
```



Prediction of score when student studies for 9.25hrs/day

In [40]:

```
Hours=9.25
prediction = regressor.predict([[9.25]])
print("Hours = {}".format(Hours))
print("Score:",prediction[0])
```

```
Hours = 9.25
Score: 93.69173248737538
```

Mean Absolute Error

In [42]:

```
from sklearn import metrics
print("Mean Absolute Error:", metrics.mean_absolute_error(Y_test,Y_predict))
```

```
Mean Absolute Error: 4.183859899002975
```