PRATEEK MALL

TASK 2

TO EXPLORE SUPERVISED MACHINE LEARNING

In [10]:
```
# IMPORTING LIBRARIES
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

In [ ]:
```
```

In [39]:
```
# data loading
url = "http://bit.ly/w-data"
sdata = pd.read_csv(url)
print("data imported successfully")
sdata.head(8)
```

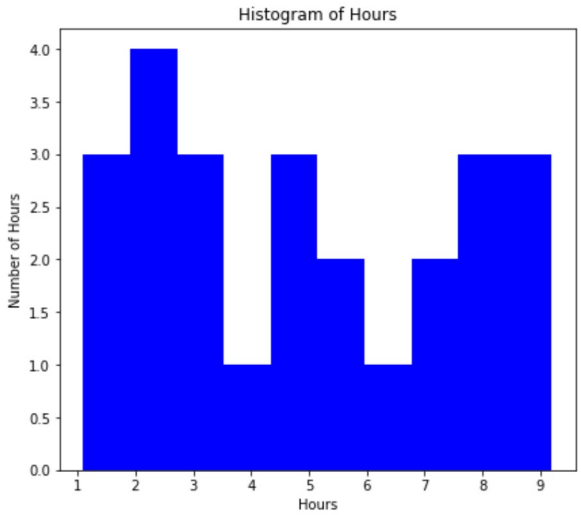
```
data imported successfully
```

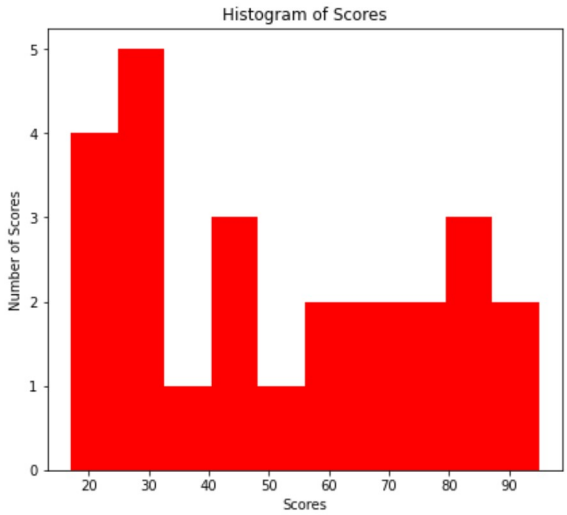
Out[39]:

	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

In [56]:
```
#Lets see the distribution of scores and hours individually

d, axes = plt.subplots(ncols=2, figsize=(15, 6))
ax = sdata['Hours'].plot(kind='hist',ax=axes[0], color="b")
ax1 = sdata['Scores'].plot(kind='hist',ax=axes[1], color="r")

ax.set_title("Histogram of Hours")
ax.set_xlabel('Hours')
ax.set_ylabel('Number of Hours')

ax1.set_title("Histogram of Scores")
ax1.set_xlabel('Scores')
ax1.set_ylabel('Number of Scores')
```

Out[56]:
```
Text(0, 0.5, 'Number of Scores')
```





In [12]:
```
sdata.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: 464.0 bytes
```

In [40]:
```
sdata.describe()
```

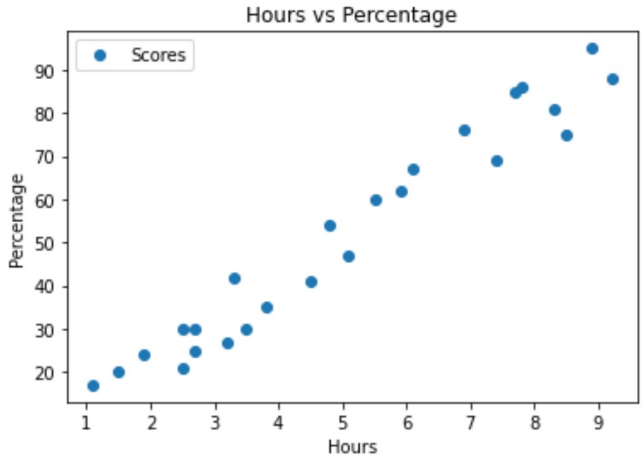
Out[40]:

	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

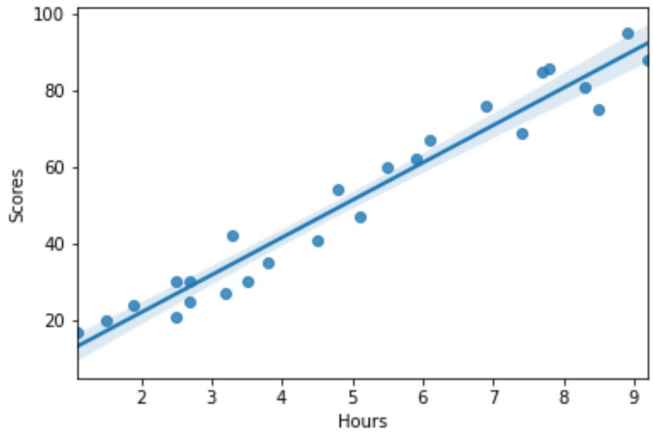
In [41]:
```
# checking for nulls
for feature in sdata.columns:
    if sdata[feature].isnull().sum()>1:
        print("{} Features has {}% Missing values".format(feature,round(sdata[featur
e].isnull().mean()*100,1)))
    else:
        print("No null values found")
```

```
No null values found
No null values found
```

In [44]:
```
# Plotting the distribution of scores
sdata.plot(x ='Hours',y='Scores',style='o')
plt.title('Hours vs Percentage')
plt.xlabel('Hours')
plt.ylabel('Percentage')
plt.show()
```



In [15]:
```
# data visualisation
import seaborn as sns
sns.regplot(x="Hours",y="Scores",data = sdata);
```



In [46]:
```
# Training Model
# Spllitting data into X and Y
x = sdata.iloc[:,:-1].values
y = sdata.iloc[:, -1].values
```

In [47]:
```
y
```

Out[47]:
```
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)
```

In [25]:
```
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)
```

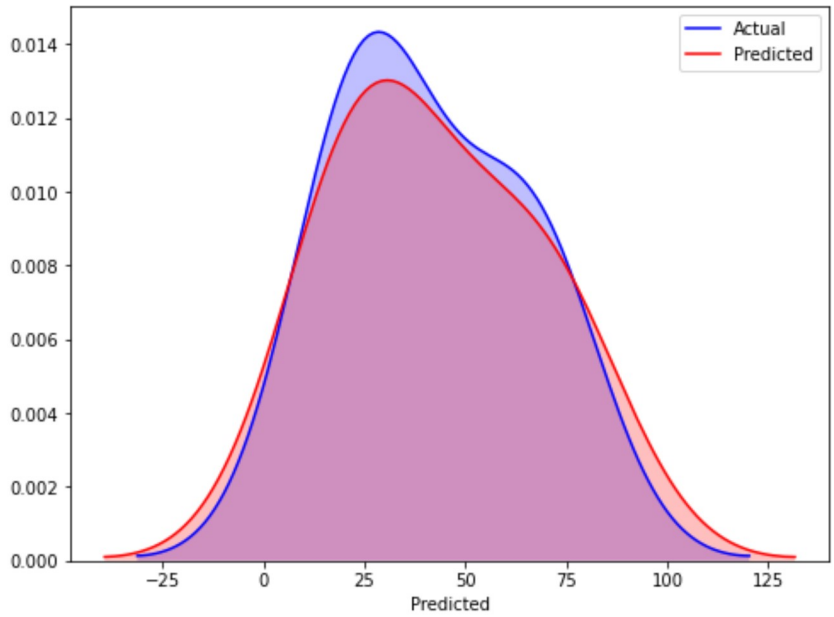
In [58]:
```
# Training model
from sklearn.linear_model import LinearRegression
regr = LinearRegression()
regr.fit(x_train,y_train)
y_predict = regr.predict(x_test)
print("Training complete.")
```

```
Training complete.
```

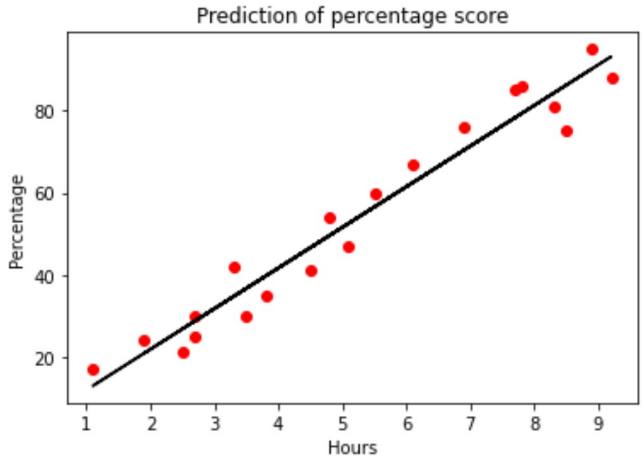
In [60]:
```
#creating new dataframe with actual and predicted results
results = pd.DataFrame({'Actual':y_test,'Predicted':y_predict})
```

In [61]:
```
#We will check how accurately the the linear Regression model predicted the results by
plotting a dist plot
fig = plt.figure(figsize=(8,6))
ax1 = sns.distplot(results.Actual,hist=False,color='b',label='Actual',kde_kws={"shad
e": True})
sns.distplot(results.Predicted,hist=False,color='r',label='Predicted',ax=ax1,kde_kws=
{"shade": True})
```

Out[61]:
```
<matplotlib.axes._subplots.AxesSubplot at 0xa96beb0>
```



In [52]:
```
#Plottting training dataset
plt.scatter(x_train,y_train,color="red")
plt.plot(x_train,regr.predict(x_train),color  ="black")
plt.title("Prediction of percentage score")
plt.xlabel("Hours")
plt.ylabel("Percentage")
plt.show()
```



In [32]:
```
#predicting the scores
y_pred = regr.predict(x_test)
```

In [53]:
```
#calculating actual and predicted score
score = pd.DataFrame({"Actual":y_test,"Predicted":y_pred})
score
```

Out[53]:

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

In [54]:
```
#Predicting scores for 9.25 hrs
hrs = 9.25
print("Number of hours the student studied:",hrs)
print("Predicted score for the student:",regr.predict(np.array(hrs).reshape(1,-1))[0])
```

```
Number of hours the student studied: 9.25
Predicted score for the student: 93.69173248737539
```

In [35]:
```
# Evaluvating the model
from sklearn import metrics
print("Mean absolute error:",metrics.mean_absolute_error(y_test,y_pred))
```

```
Mean absolute error: 4.183859899002982
```

In [ ]: