Name - ARYAN BAJAJ

Task 1 - Prediction using Supervised ML (Level - Beginner)

In [1]:

```
# Importing all the libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
from sklearn import metrics
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
```

In [2]:

```
# Reading data from the link
url = "http://bit.ly/w-data"
data = pd.read_csv(url)
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

Exploratory Data Analysis

In [3]:

```
data.shape
```

Out[3]:

```
(25, 2)
```

In [4]:

```
data.nunique()
```

Out[4]:

```
Hours     23
Scores    23
dtype: int64
```

In [5]:

```
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
```

In [6]:

```
data.isnull().sum()
```

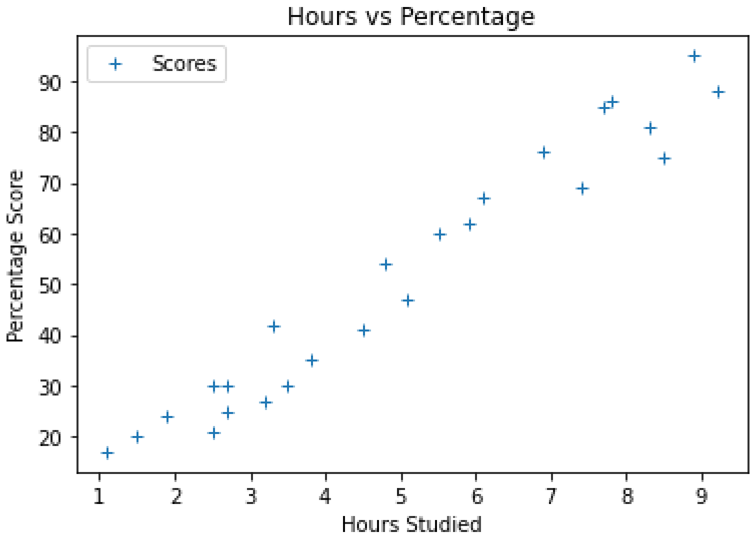
Out[6]:

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

Visualizing the Data

In [7]:

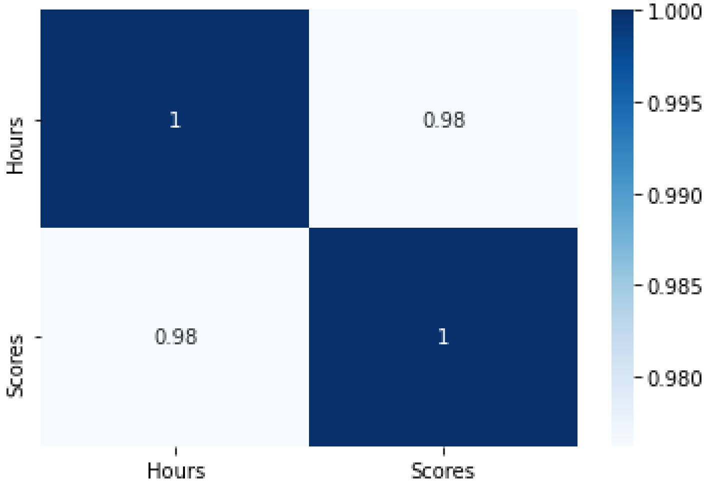
```
# Plotting the distribution of scores
data.plot(x='Hours', y='Scores', style='+')
plt.title('Hours vs Percentage')
plt.xlabel('Hours Studied')
plt.ylabel('Percentage Score')
plt.show()
```



From the graph above, we can clearly see that there is a positive linear relation between the number of hours studied and percentage of score.

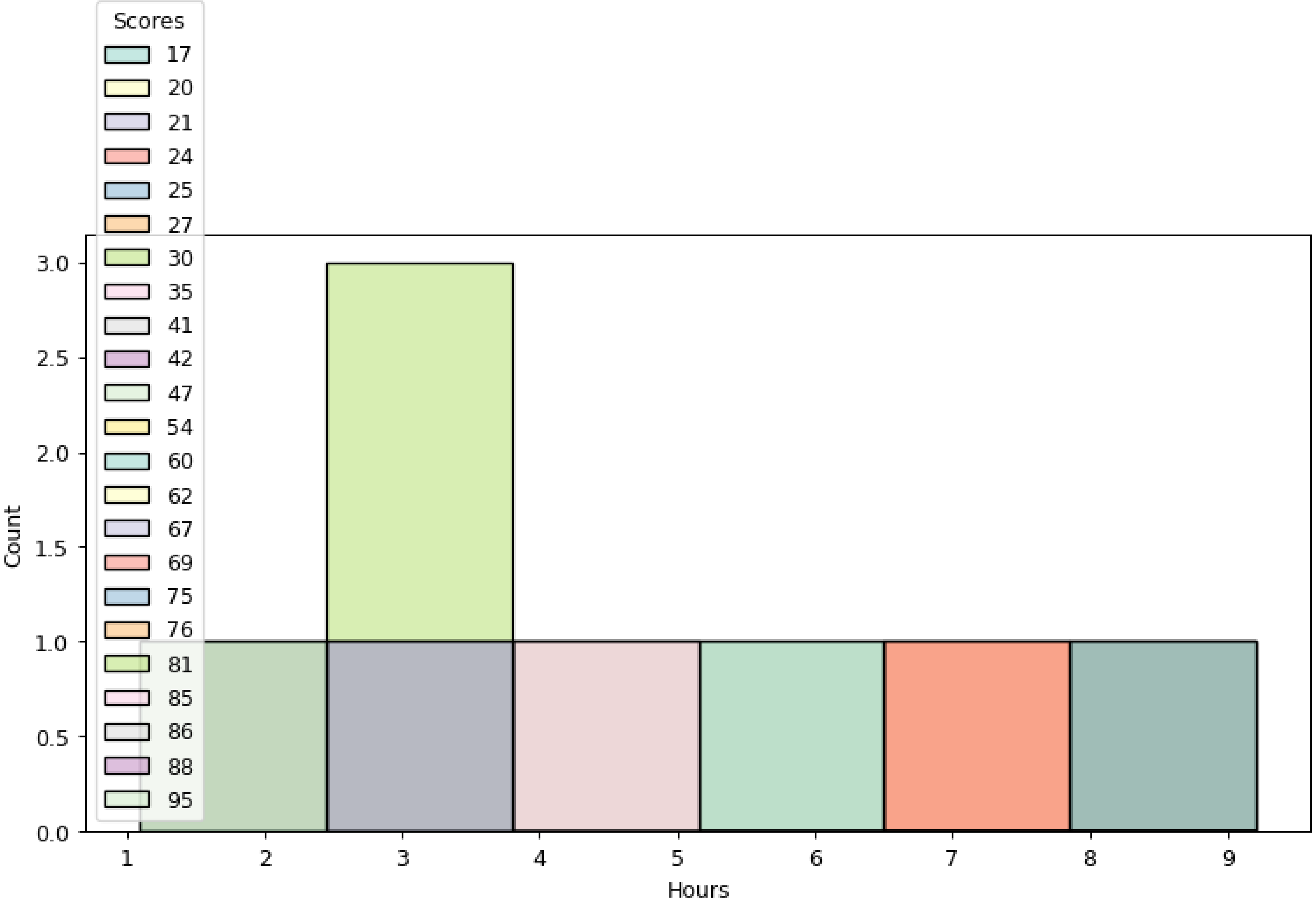
In [8]:
```
sns.heatmap(data.corr(),annot=True,cmap='Blues')
```

Out[8]:
```
<AxesSubplot:>
```



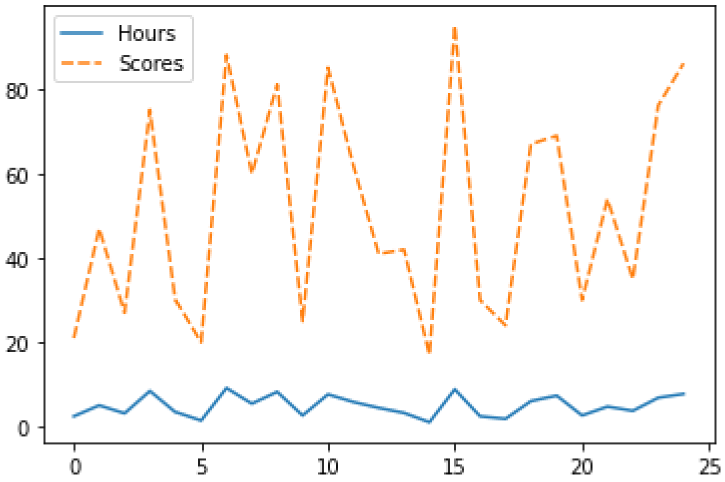
In [9]:
```
plt.figure(figsize=(10 , 5),dpi = 90)
sns.histplot(x='Hours' , hue='Scores' ,data=data  ,palette="Set3" , edgecolor='black')
```

Out[9]:
```
<AxesSubplot:xlabel='Hours', ylabel='Count'>
```



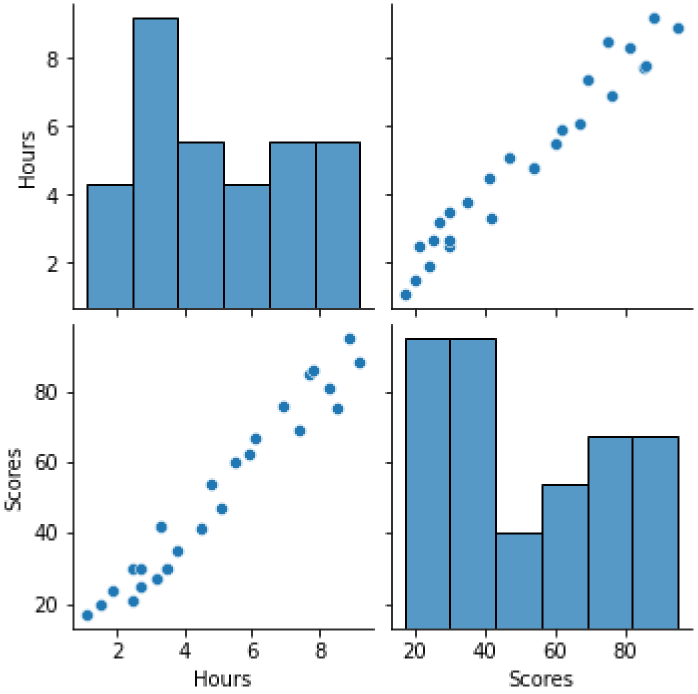
In [10]:
```
sns.lineplot(data=data)
```

Out[10]:
```
<AxesSubplot:>
```



```
In [11]: sns.pairplot(data)
```

```
Out[11]: <seaborn.axisgrid.PairGrid at 0x1df4bc778e0>
```



Preparing the data

The next step is to divide the data into inputs & outputs.

In [12]:

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

Now that we have our attributes and labels, the next step is to split this data into training and test sets.

In [13]:

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=0)
```

Test & Train the MODEL

We have split our data into training and testing sets, and now is finally the time to train our algorithm.

We'll use Linear Regression as the Dataset is interval based.

In [14]:

```
LR = LinearRegression()
LR.fit(X_train, y_train)
```

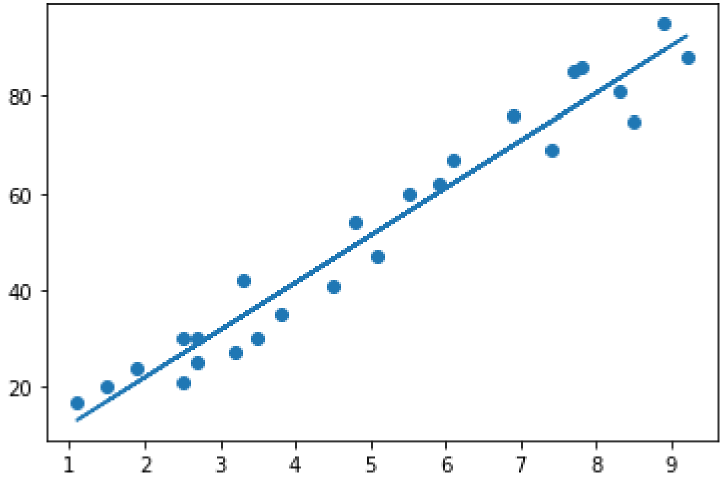
Out[14]:

```
LinearRegression()
```

In [15]:

```
# Plotting the regression line
line = LR.coef_*X+LR.intercept_

# Plotting for the test data
plt.scatter(X, y)
plt.plot(X, line);
plt.show()
```



Predictions

Now that we have trained our Model, it's time to make some predictions.

In [16]:
```
print(X_test) # Testing data - In Hours
y_pred = LR.predict(X_test) # Predicting the scores
```

```
[[1.5]
 [3.2]
 [7.4]
 [2.5]
 [5.9]
 [3.8]
 [1.9]
 [7.8]]
```

In [17]:
```
# Comparison
df = pd.DataFrame({'Actual': y_test, 'Predicted': y_pred})
df
```

Out[17]:

	Actual	Predicted
0	20	17.053665
1	27	33.694229
2	69	74.806209
3	30	26.842232
4	62	60.123359
5	35	39.567369
6	24	20.969092
7	86	78.721636

In [18]:

```
# Now performing the First Task given by TSF
hours = 9.25
my_pred = LR.predict([[hours]])
print("No of Hours = {}".format(hours))
print("Predicted Score = {}".format(my_pred[0]))
```

```
No of Hours = 9.25
Predicted Score = 92.91505723477056
```

Evaluating the model

The final step is to evaluate the performance of algorithm.

In [19]:

```
print('Mean Absolute Error by using Linear Regression is:', metrics.mean_absolute_error(y_test, y_pred),'%')
```

```
Mean Absolute Error by using Linear Regression is: 4.419727808027652 %
```

Categorical Prediction

In [20]:

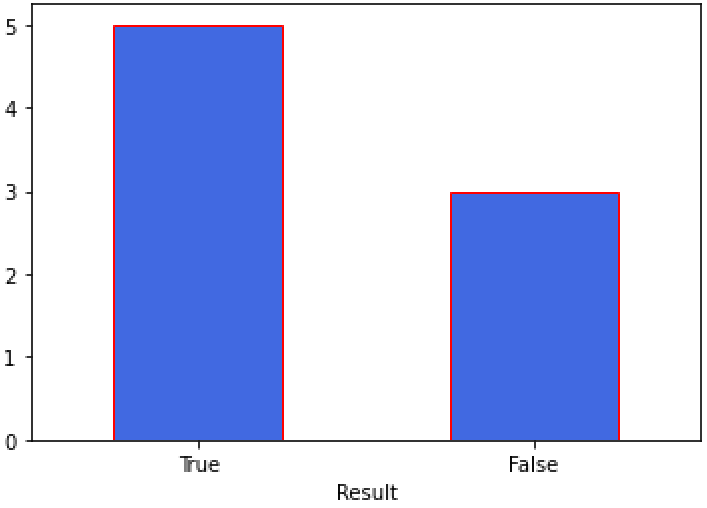
```
cutoff = 33
df['Result']=df['Predicted']>=cutoff
df
```

Out[20]:

	Actual	Predicted	Result
0	20	17.053665	False
1	27	33.694229	True
2	69	74.806209	True
3	30	26.842232	False
4	62	60.123359	True
5	35	39.567369	True
6	24	20.969092	False
7	86	78.721636	True

In [21]:
```
CP=df.value_counts('Result')
CP.plot(kind='bar', rot=0, color=['royalblue'],edgecolor='red')
print('--------------------------------------------')
print('      Passing Rate in Percentage (%) :      ')
print('--------------------------------------------')
print(df.value_counts('Result')/df.value_counts('Result').sum()*100)
```

```
--------------------------------------------
      Passing Rate in Percentage (%) :      
--------------------------------------------
Result
True     62.5
False    37.5
dtype: float64
```



Hence, it can be said that, according to this data more students will pass this time

THANK - YOU ^_^