```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
```

```
from google.colab import files
uploaded = files.upload()
```

Choose Files student_scores.csv

- **student_scores.csv**(text/csv) - 214 bytes, last modified: 5/17/2023 - 100% done

```
Saving student_scores.csv to student_scores.csv
```

```
import io
df= pd.read_csv(io.BytesIO(uploaded['student_scores.csv']))
```

```
print(df.head())
```

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

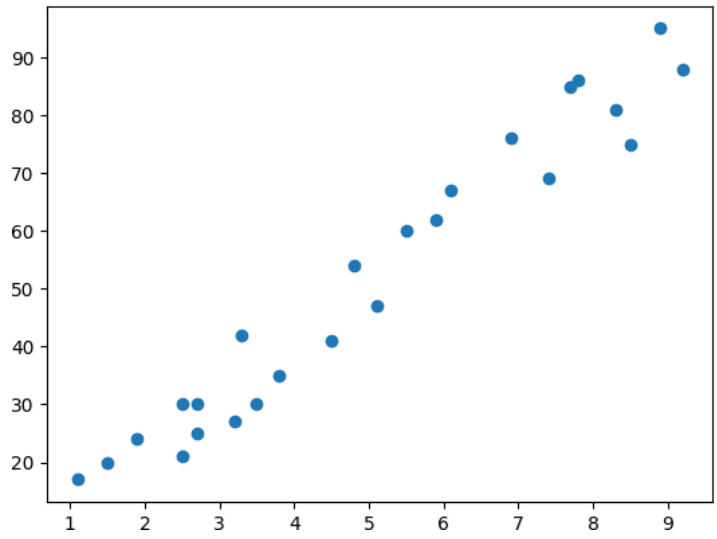
```
df.shape
```

```
(25, 2)
```

```
df.isnull().sum()
```

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

```
plt.scatter(df['Hours'], df['Scores'])
plt.show()
```

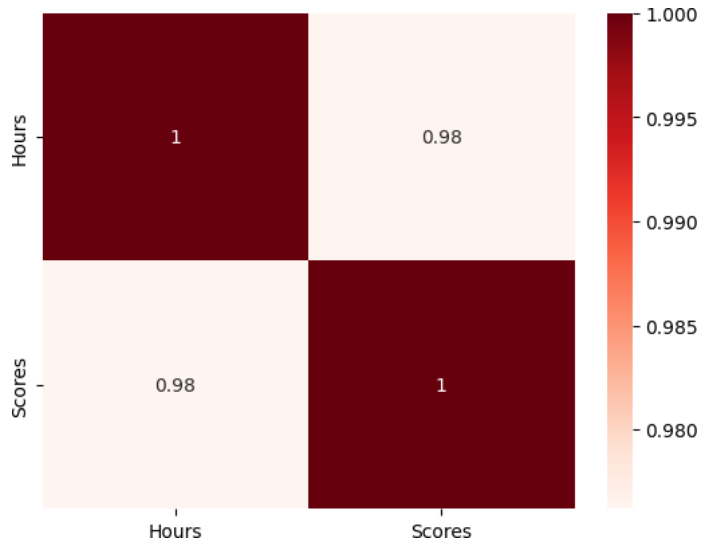



```
print(df.corr())
```

```
            Hours    Scores
Hours    1.000000  0.976191
Scores   0.976191  1.000000
```

```
import seaborn as sns
```

```
sns.heatmap(df[['Hours','Scores']].corr(), annot=True, cmap = 'Reds')
plt.show()
```




```
print(df.describe())
```

```
            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
```

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

```
print('X shape:', X.shape)
print('y shape:', y.shape)
print('X shape type:', type(X))
print('y shape type:', type(y))
```

```
X shape: (25, 1)
y shape: (25,)
X shape type: <class 'pandas.core.frame.DataFrame'>
y shape type: <class 'pandas.core.series.Series'>
```

```
print('X_wrong shape:', X_wrong.shape , type(X_wrong) )
print('y_wrong shape:', y_wrong.shape , type(y_wrong))
```

```
X_wrong shape: (25,) <class 'pandas.core.series.Series'>
y_wrong shape: (25,) <class 'pandas.core.series.Series'>
```

```
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=42)
print(X_train)
```

```
    Hours
9     2.7
13    3.3
1     5.1
22    3.8
5     1.5
2     3.2
12    4.5
15    8.9
3     8.5
4     3.5
20    2.7
17    1.9
21    4.8
18    6.1
24    7.8
7     5.5
10    7.7
```

```
14    1.1
19    7.4
6     9.2
```

```
print(y_train)
```

```
9     25
13    42
1     47
22    35
5     20
2     27
12    41
15    95
3     75
4     30
20    30
17    24
21    54
18    67
24    86
7     60
10    85
14    17
19    69
6     88
Name: Scores, dtype: int64
```

```
from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
regressor.fit(X_train, y_train)
```

```
LinearRegression
LinearRegression()
```

```
regressor.intercept_
```

```
2.826892353899737
```

```
regressor.coef_
```

```
array([9.68207815])
```

```
print(regressor.coef_[0])
```

```
9.682078154455697
```

```
def calc(slope, intercept, hours):
    return slope*hours+intercept
```

```
score = calc(regressor.coef_, regressor.intercept_, 9.5)
print(score)
```

```
[94.80663482]
```

```
score = regressor.predict([[9.5]])
print(score)
```

```
[94.80663482]
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:439: UserWarning: X does not have valid feature names, but LinearRegression was
  warnings.warn(
```

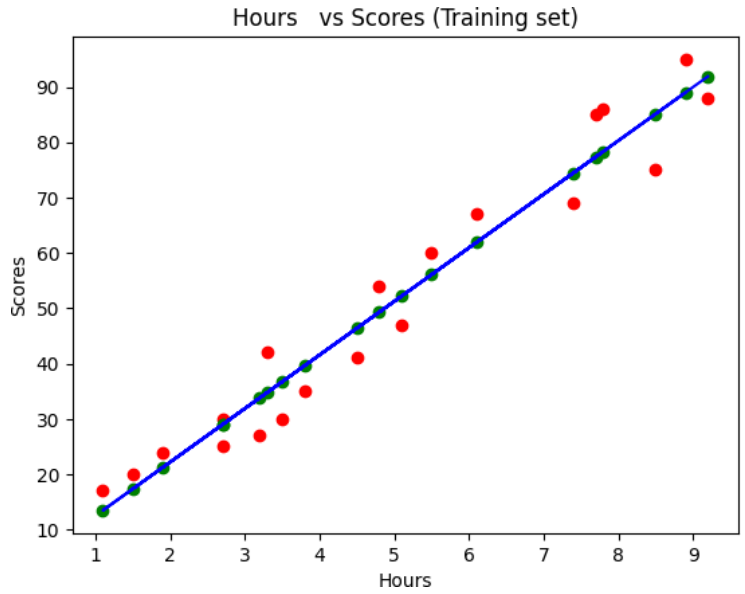
```
y_pred = regressor.predict(X_test)
```

```
df_preds = pd.DataFrame({'Actual': y_test, 'Predicted': y_pred})
print(df_preds)
```

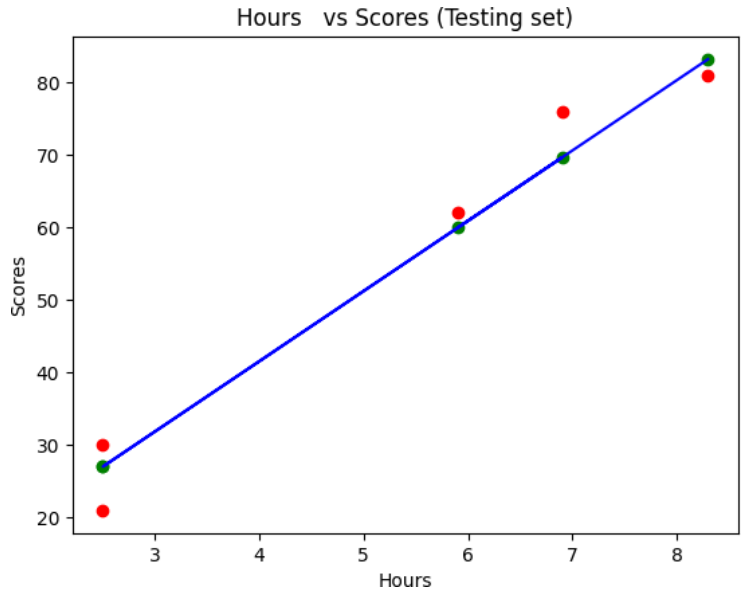
```
    Actual  Predicted
8       81  83.188141
16      30  27.032088
0       21  27.032088
```

```
23      76  69.633232
11      62  59.951153
```

```
plt.scatter(X_train, y_train, color = 'red')
plt.plot(X_train, regressor.predict(X_train), color = 'blue')
plt.scatter(X_train, regressor.predict(X_train), color = 'green')
plt.title('Hours   vs Scores (Training set)')
plt.xlabel('Hours')
plt.ylabel('Scores')
plt.show()
```




```
plt.scatter(X_test, y_test, color = 'red')
plt.plot(X_test, regressor.predict(X_test), color = 'blue')
plt.scatter(X_test, regressor.predict(X_test), color = 'green')
plt.title('Hours   vs Scores (Testing set)')
plt.xlabel('Hours')
plt.ylabel('Scores')
plt.show()
```




```
from sklearn.metrics import mean_absolute_error, mean_squared_error
mae = mean_absolute_error(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)
rmse = np.sqrt(mse)
```

✓0s completed at 10:23 AM