

THE SPARKS FOUNDATION INTERNSHIP

SWATI SHAH
Task_1

PREDICTION USING SUPERVISED MACHINE LEARNING

GRIP2021

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

```
In [56]: import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

READING DATA SET

```
In [57]: data_set=pd.read_csv('http://bit.ly/w-data')
```

```
In [58]: data_set.head()
```

Out[58]:

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

```
In [59]: data_set.isnull().sum()
```

Out[59]: Hours 0
Scores 0
dtype: int64

```
In [60]: data_set.shape
```

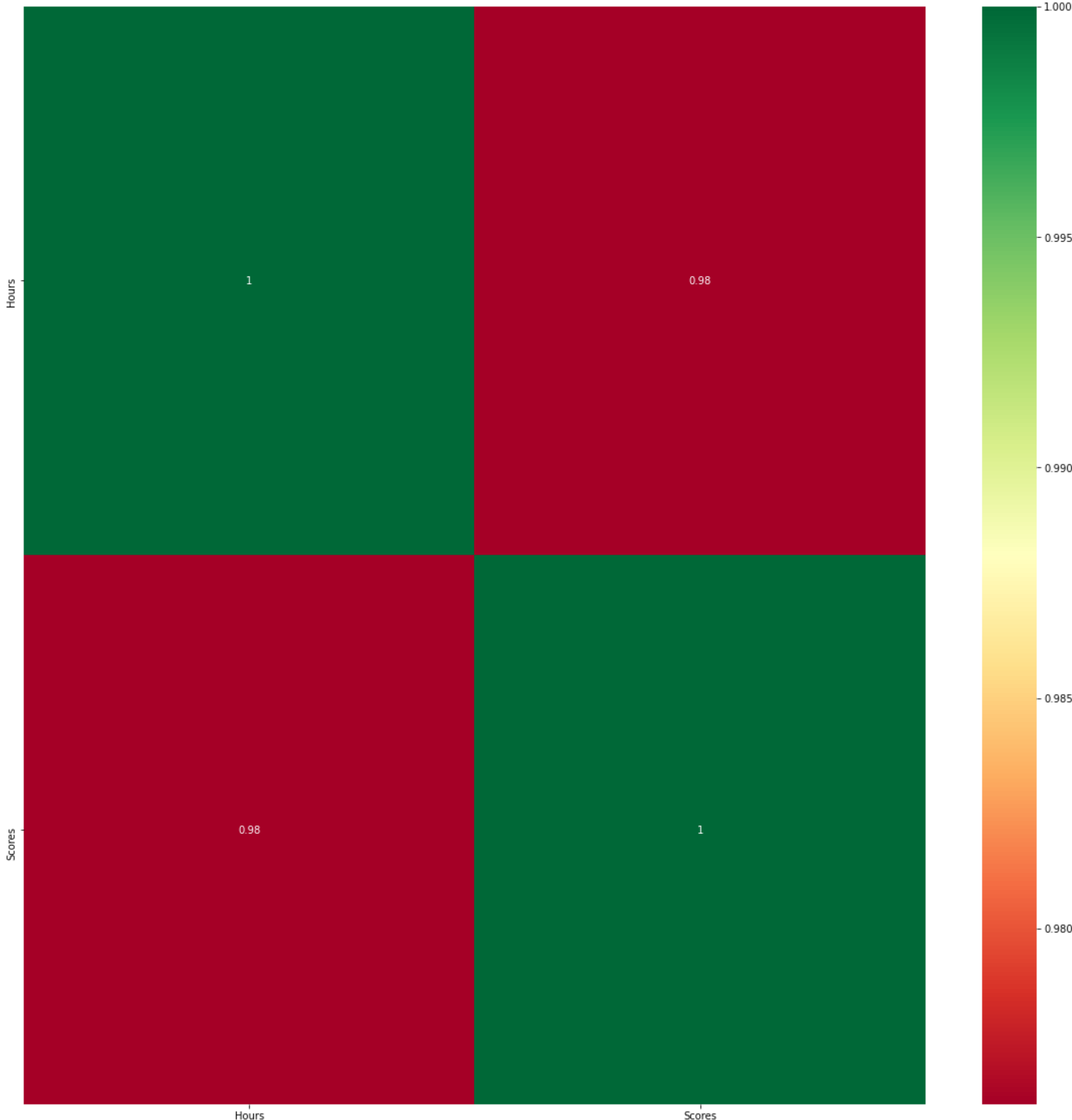
Out[60]: (25, 2)

```
In [61]: data_set.select_dtypes(include=['category'])
```

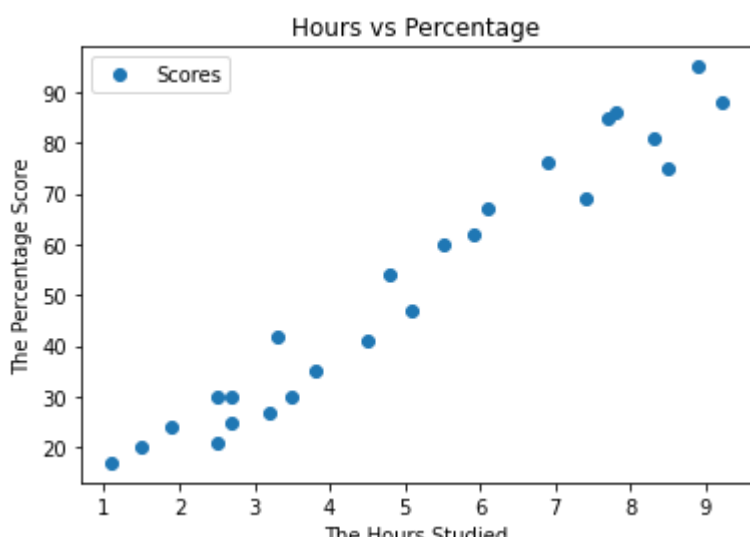
Out[61]:

```
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
```

```
In [62]: import seaborn as sns
#get correlations of each features in dataset
corrmat = data_set.corr()
top_corr_features = corrmat.index
plt.figure(figsize=(20,20))
#plot heat map
g=sns.heatmap(data_set[top_corr_features].corr(),annot=True,cmap="RdYlGn")
```



```
In [63]: data_set.plot(x='Hours', y='Scores',style='o' )
plt.title('Hours vs Percentage')
plt.xlabel('The Hours Studied')
plt.ylabel('The Percentage Score')
plt.show()
```



```
In [64]: independent=data_set.iloc[:, :-1].values
dependent=data_set.iloc[:, 1].values
```

```
In [65]: from sklearn.model_selection import train_test_split
independent_train, independent_test, dependent_train, dependent_test = train_test_split(independent, dependent,test_size=0.2, random_state=0)
```

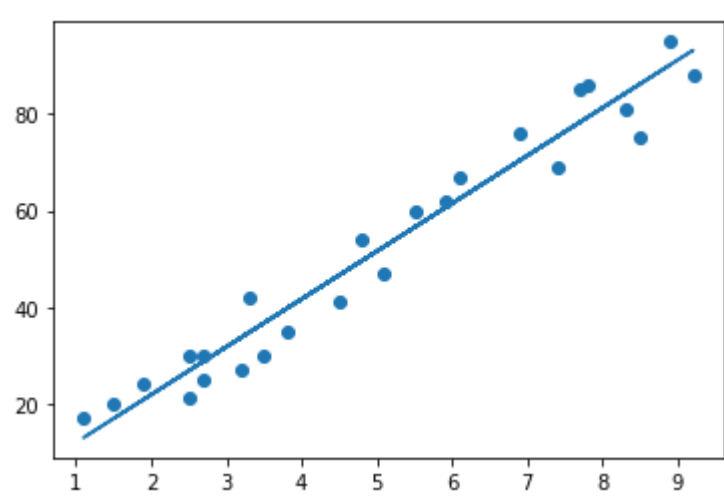
```
In [66]: from sklearn.linear_model import LinearRegression
regressor=LinearRegression()
regressor.fit(independent_train,dependent_train)
```

Out[66]: LinearRegression()

```
In [67]: LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
```

Out[67]: LinearRegression()

```
In [68]: line=regressor.coef_*independent+regressor.intercept_
plt.scatter(independent,dependent)
plt.plot(independent,line)
plt.show()
```



```
In [69]: print(independent_test)
```

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

```
In [70]: dependent_pred=regressor.predict(independent_test)
```

```
In [71]: model=pd.DataFrame({'Actual':dependent_test,'Predicted':dependent_pred})
```

```
In [72]: model
```

Out[72]:

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

```
In [73]: hours=[9.25]
own_pred=regressor.predict(hours)
print("Number of hours={}".format(hours))
if own_pred[0]>100:
    print("prediction score=100")
else:
    print("Prediction Score = {}".format(own_pred[0]))
```

Number of hours=[9.25]
Prediction Score = 93.69173248737538

```
In [74]: from sklearn import metrics
print('Mean Absolute Error:',
      metrics.mean_absolute_error(dependent_test, dependent_pred))
```

Mean Absolute Error: 4.183859899002975

```
In [75]: print('Variance score :%2f'% regressor.score(independent_test,dependent_test))
```

Variance score :0.945491

TASK 1 COMPLETED: