task-1

August 12, 2021

1 Janvi Singh

DATA SCIENCE AND BUSINESS ANALYTICS INTERN Under Graduate Rotational Internship Program (Grip August 2021) by The Sparks Foundation

Task-1 Prediction Using Supervised ML

The aim of the task is to be predict the percentage of a student based on the no of study hours . It is a simple linear regression task as it involves just two variables . I have used Python Language in this project . Further, different python language [Numpy , Pandas , Seaborn and Matplotlib] are imported for performing different data analytic techniques.

[1]:
```
!pip install jovian --upgrade --quiet
```

[2]:
```
import jovian
```

[3]:
```
# Execute this to save new versions of the notebook
jovian.commit(project="task-1")
```

```
<IPython.core.display.Javascript object>
```

```
[jovian] Updating notebook "janvi-singh-142000/task-1" on https://jovian.ai
[jovian] Committed successfully! https://jovian.ai/janvi-singh-142000/task-1
```

[3]:
```
'https://jovian.ai/janvi-singh-142000/task-1'
```

[4]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

The Dataset for the project is downloaded from “http://bit.ly/w-data”

[5]:
```
dataset=pd.read_csv('http://bit.ly/w-data')
```

Data Prepration and Data Cleaning

[6]:
```
print(dataset.shape)
```

```
(25, 2)
```

[7]:
```
dataset.head(15)
```

[7]:

	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
8	8.3	81
9	2.7	25
10	7.7	85
11	5.9	62
12	4.5	41
13	3.3	42
14	1.1	17

[8]:
```
dataset.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
```

[9]:
```
dataset.describe()
```

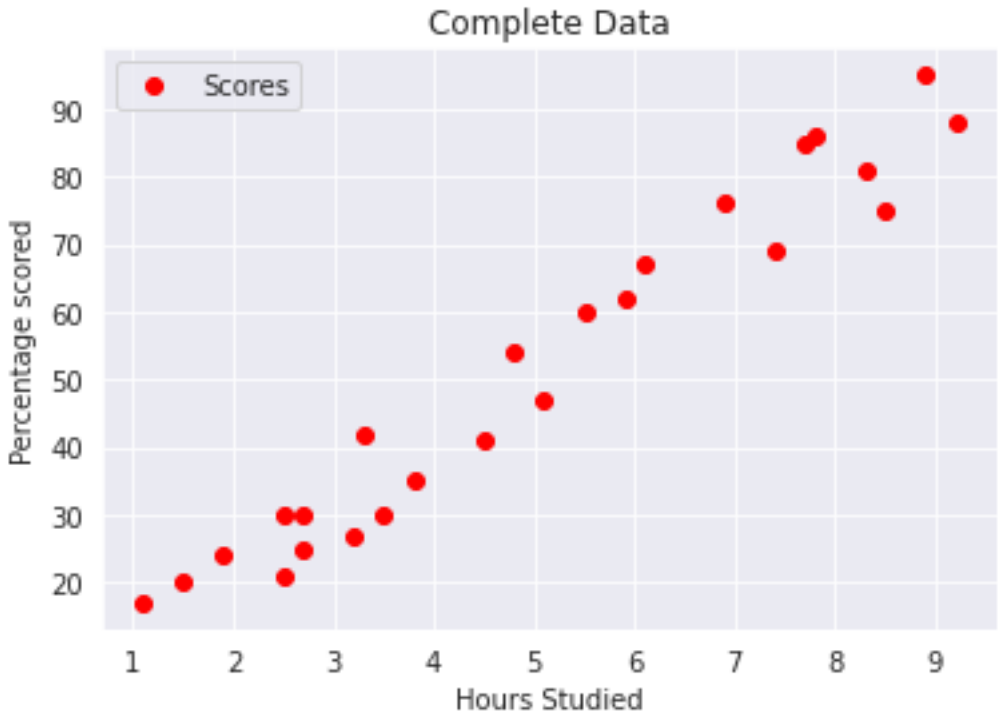
[9]:

	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

[10]:
```
sns.set_style("darkgrid")
```

[11]:
```
dataset.plot( x= 'Hours' , y='Scores' , style='or')
plt.title("Complete Data")
```

```
plt.xlabel("Hours Studied")
plt.ylabel("Percentage scored")
plt.figure(figsize=(12, 6))
plt.show()
```




```
<Figure size 864x432 with 0 Axes>
```

By the help of different functions we analyze , prepare and clean the data . Here,

- dataset.shape shows that there are 25 rows and 2 columns.
  - dataset.describe() describes the data .
    - dataset.info() gives information about the data . In this case there are 25 non-null values which depicts that the dataset does not have any null values and we don't need to take care for that .
      - plot() shows the linear relation between Hours and scores which implies that there will be an increase in scores for the higher hours of study .

Training Of Dataset

[12]:
```
X=dataset.iloc[:, :-1].values
y=dataset.iloc[:, 1].values
```

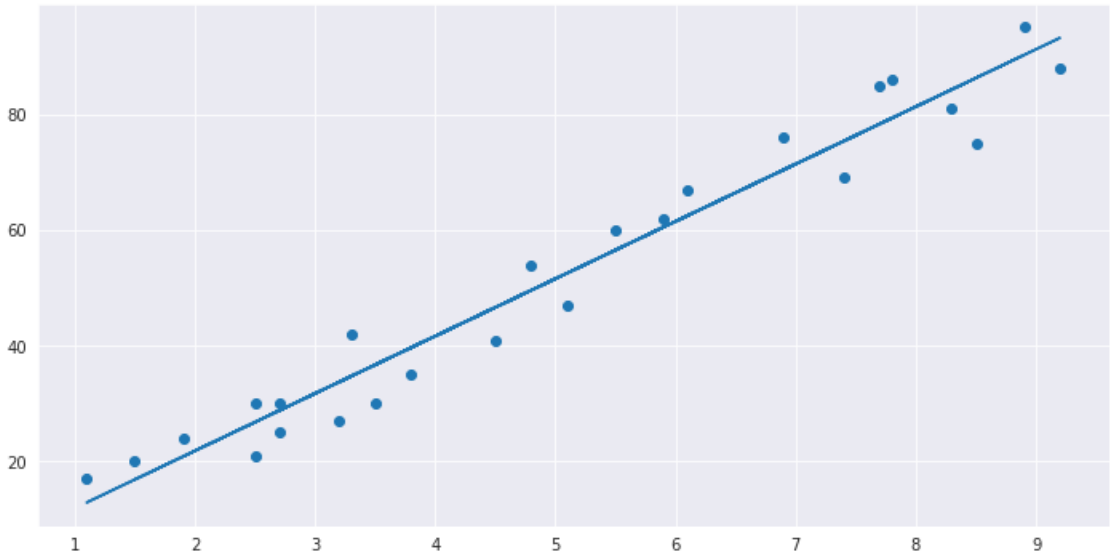
[13]:
```
from sklearn.model_selection import train_test_split
X_train, X_test ,y_train, y_test= train_test_split(X ,y ,test_size=0.2 ,␣
↪random_state=0)
```

[14]:
```
from sklearn.linear_model import LinearRegression
reg=LinearRegression()
reg.fit(X_train , y_train)
print("Successfully trained the model")
```

```
Successfully trained the model
```

Here, the dataset is first split into training and test series using scikit learn's bulletin method of train_test_split(). Further, The algorithm is trained using Linear Regression .

[15]:
```
plt.figure(figsize=(12, 6))
line=reg.coef_*X+reg.intercept_
plt.scatter(X,y)
plt.plot(X,line);
plt.show()
```



[16]:
```
print(X_test)
y_pred=reg.predict(X_test)
```

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

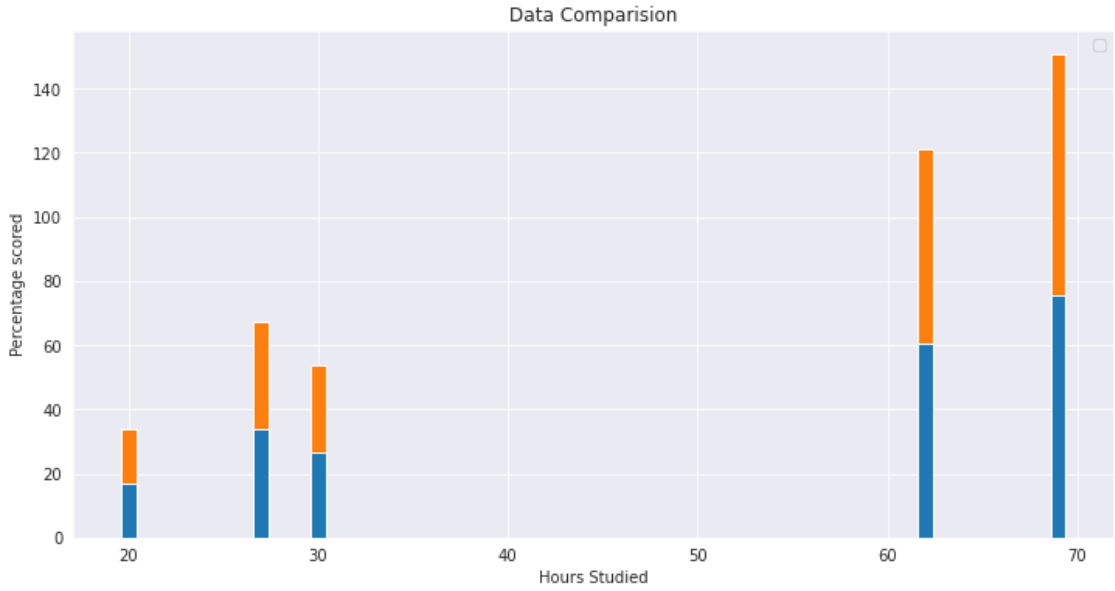
Now the Original dataset is compared to the predicted data to check that the model is fitted properly or not.

[17]:
```
res_df=pd.DataFrame({ 'Original Data': y_test , 'Predicted Data':y_pred})
res_df
```

[17]:

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

[24]:
```
plt.figure(figsize=(12, 6))
plt.title("Data Comparision")
plt.xlabel("Hours Studied")
plt.ylabel("Percentage scored")
plt.legend(['Original Data', 'Predicted Data'])
plt.bar(y_test,y_pred)
plt.bar(y_test,y_pred, bottom =y_pred);
```



On comparing the datasets we can say that the predicted data does not provide exact values but a close enough approach . Hence, we can say that model is fitted properly and can be used

TO Do Question

What will be predicted score if a student studies for 9.25 hrs/day?

[19]:
```
hours=[[9.25]]
mod_pred=reg.predict(hours)
print(" For {} hours Predicted Score is {}" .format(hours , mod_pred[0]))
```

```
 For [[9.25]] hours Predicted Score is 93.69173248737539
```

Conclusion

Since, the model predicts the percentage of student based on the no of study hours Successfully . Moreover, the To-Do question in the task predicts the value close enough to what would have been predicted by original value . Hence, we can conclude that our model predicts the value effectively .