DAY-10

sleep_health

In [1]:
```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:
```python
df=pd.read_csv(r"C:\Users\user\Downloads\sleep_health.csv")[0:500]
df
```

Out[2]:

	Person ID	Gender	Age	Occupation	Sleep Duration	Quality of Sleep	Physical Activity Level	Stress Level	BMI Category	Blood Pressure	H F
0	1	Male	27	Software Engineer	6.1	6	42	6	Overweight	126/83	
1	2	Male	28	Doctor	6.2	6	60	8	Normal	125/80	
2	3	Male	28	Doctor	6.2	6	60	8	Normal	125/80	
3	4	Male	28	Sales Representative	5.9	4	30	8	Obese	140/90	
4	5	Male	28	Sales Representative	5.9	4	30	8	Obese	140/90	
...	...	...	...	...	...	...	...	...	...	...	
369	370	Female	59	Nurse	8.1	9	75	3	Overweight	140/95	
370	371	Female	59	Nurse	8.0	9	75	3	Overweight	140/95	
371	372	Female	59	Nurse	8.1	9	75	3	Overweight	140/95	
372	373	Female	59	Nurse	8.1	9	75	3	Overweight	140/95	
373	374	Female	59	Nurse	8.1	9	75	3	Overweight	140/95	

374 rows × 13 columns

In [3]: `df.head(10)`

Out[3]:

	Person ID	Gender	Age	Occupation	Sleep Duration	Quality of Sleep	Physical Activity Level	Stress Level	BMI Category	Blood Pressure	Hear Rat
0	1	Male	27	Software Engineer	6.1	6	42	6	Overweight	126/83	7
1	2	Male	28	Doctor	6.2	6	60	8	Normal	125/80	7
2	3	Male	28	Doctor	6.2	6	60	8	Normal	125/80	7
3	4	Male	28	Sales Representative	5.9	4	30	8	Obese	140/90	8
4	5	Male	28	Sales Representative	5.9	4	30	8	Obese	140/90	8
5	6	Male	28	Software Engineer	5.9	4	30	8	Obese	140/90	8
6	7	Male	29	Teacher	6.3	6	40	7	Obese	140/90	8
7	8	Male	29	Doctor	7.8	7	75	6	Normal	120/80	7
8	9	Male	29	Doctor	7.8	7	75	6	Normal	120/80	7
9	10	Male	29	Doctor	7.8	7	75	6	Normal	120/80	7

In [4]: `df.describe()`

Out[4]:

	Person ID	Age	Sleep Duration	Quality of Sleep	Physical Activity Level	Stress Level	Heart Rate	Daily Ste
count	374.000000	374.000000	374.000000	374.000000	374.000000	374.000000	374.000000	374.0000
mean	187.500000	42.184492	7.132086	7.312834	59.171123	5.385027	70.165775	6816.8449
std	108.108742	8.673133	0.795657	1.196956	20.830804	1.774526	4.135676	1617.9156
min	1.000000	27.000000	5.800000	4.000000	30.000000	3.000000	65.000000	3000.0000
25%	94.250000	35.250000	6.400000	6.000000	45.000000	4.000000	68.000000	5600.0000
50%	187.500000	43.000000	7.200000	7.000000	60.000000	5.000000	70.000000	7000.0000
75%	280.750000	50.000000	7.800000	8.000000	75.000000	7.000000	72.000000	8000.0000
max	374.000000	59.000000	8.500000	9.000000	90.000000	8.000000	86.000000	10000.0000

In [5]:
```python
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 374 entries, 0 to 373
Data columns (total 13 columns):
 #   Column                   Non-Null Count   Dtype
---  ------                   --------------   -----
 0   Person ID                374 non-null     int64
 1   Gender                   374 non-null     object
 2   Age                      374 non-null     int64
 3   Occupation               374 non-null     object
 4   Sleep Duration           374 non-null     float64
 5   Quality of Sleep         374 non-null     int64
 6   Physical Activity Level  374 non-null     int64
 7   Stress Level             374 non-null     int64
 8   BMI Category             374 non-null     object
 9   Blood Pressure           374 non-null     object
 10  Heart Rate               374 non-null     int64
 11  Daily Steps              374 non-null     int64
 12  Sleep Disorder           374 non-null     object
dtypes: float64(1), int64(7), object(5)
memory usage: 38.1+ KB
```

In [6]:
```python
df.columns
```

Out[6]:
```
Index(['Person ID', 'Gender', 'Age', 'Occupation', 'Sleep Duration',
       'Quality of Sleep', 'Physical Activity Level', 'Stress Level',
       'BMI Category', 'Blood Pressure', 'Heart Rate', 'Daily Steps',
       'Sleep Disorder'],
      dtype='object')
```

In [8]:
```python
x=df[['Person ID',  'Age',  'Sleep Duration',
       'Quality of Sleep', 'Physical Activity Level', 'Stress Level', 'Heart Rate']]
y=df['Daily Steps']
```

In [9]:
```python
#to split my dataset into traning and test data

from sklearn.model_selection import train_test_split

x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
```

In [10]:
```python
from sklearn.linear_model import LinearRegression

lr = LinearRegression()
lr.fit(x_train,y_train)
```

Out[10]:
```
LinearRegression()
```

In [11]:
```python
print(lr.intercept_)
```

```
13284.907088348442
```

In [12]:
```python
print(lr.score(x_test,y_test))
```

```
0.873258275734101
```

In [13]:
```python
lr.score(x_train,y_train)
```

Out[13]: 0.7772690323888585

Ridge Regression

In [14]:
```python
from sklearn.linear_model import Ridge,Lasso
```

In [15]:
```python
rr=Ridge(alpha=10)
rr.fit(x_train,y_train)
```

Out[15]: Ridge(alpha=10)

In [16]:
```python
rr.score(x_test,y_test)
```

Out[16]: 0.8711719409609766

Lasso Regression

In [17]:
```python
la=Lasso(alpha=10)
la.fit(x_train,y_train)
```

Out[17]: Lasso(alpha=10)

In [18]:
```python
la.score(x_test,y_test)
```

Out[18]: 0.8714956754922409

In [19]:
```python
from sklearn.linear_model import ElasticNet
en=ElasticNet()
en.fit(x_train,y_train)
```

Out[19]: ElasticNet()

In [20]:
```python
print(en.intercept_)
```

14239.060405396896

In [21]:
```python
predict=(en.predict(x_test))
```

In [22]:
```python
print(en.score(x_test,y_test))
```

0.8540481297557856

Evaluation Matrics

In [23]:
```python
from sklearn import metrics
```

In [24]:
```python
print("Mean Absolute Error:",metrics.mean_absolute_error(y_test,predict))
```

Mean Absolute Error: 471.55697987014383

In [25]:
```python
print("Mean Square Error:",metrics.mean_squared_error(y_test,predict))
```

Mean Square Error: 348797.88037529454

In [26]:
```python
print("Root Mean Square Error:",np.sqrt(metrics.mean_squared_error(y_test,predict))
```

Root Mean Square Error: 590.5911279178638

In [ ]: