Deena 20104016

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as pp
```

Problem Statement

LINEAR REGRESSION

In [2]:
```
a = pd.read_csv("16_Sleep_health_and_lifestyle_dataset.csv")
```

Out[2]:

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

374 rows × 13 columns

HEAD

In [3]:

Out[3]:

	Person ID	Gender	Age	Occupation	Sleep Duration	Quality of Sleep	Physical Activity Level	Stress Level	BMI Category	Bloo Pressur
0	1	Male	27	Software Engineer	6.1	6	42	6	Overweight	126/8
1	2	Male	28	Doctor	6.2	6	60	8	Normal	125/8
2	3	Male	28	Doctor	6.2	6	60	8	Normal	125/8
3	4	Male	28	Sales Representative	5.9	4	30	8	Obese	140/9
4	5	Male	28	Sales Representative	5.9	4	30	8	Obese	140/9

Data Cleaning and Preprocessing

In [4]:

Out[4]:

	Person ID	Gender	Age	Occupation	Sleep Duration	Quality of Sleep	Physical Activity Level	Stress Level	BMI Category	Bloo Pressur
0	1	Male	27	Software Engineer	6.1	6	42	6	Overweight	126/8
1	2	Male	28	Doctor	6.2	6	60	8	Normal	125/8
2	3	Male	28	Doctor	6.2	6	60	8	Normal	125/8
3	4	Male	28	Sales Representative	5.9	4	30	8	Obese	140/9
4	5	Male	28	Sales Representative	5.9	4	30	8	Obese	140/9

In [5]:

Out[5]:

	Person ID	Age	Sleep Duration	Quality of Sleep	Physical Activity Level	Stress Level	Heart Rate	D
count	374.000000	374.000000	374.000000	374.000000	374.000000	374.000000	374.000000	3
mean	187.500000	42.184492	7.132086	7.312834	59.171123	5.385027	70.165775	68
std	108.108742	8.673133	0.795657	1.196956	20.830804	1.774526	4.135676	16
min	1.000000	27.000000	5.800000	4.000000	30.000000	3.000000	65.000000	30
25%	94.250000	35.250000	6.400000	6.000000	45.000000	4.000000	68.000000	56
50%	187.500000	43.000000	7.200000	7.000000	60.000000	5.000000	70.000000	70
75%	280.750000	50.000000	7.800000	8.000000	75.000000	7.000000	72.000000	80
max	374.000000	59.000000	8.500000	9.000000	90.000000	8.000000	86.000000	100

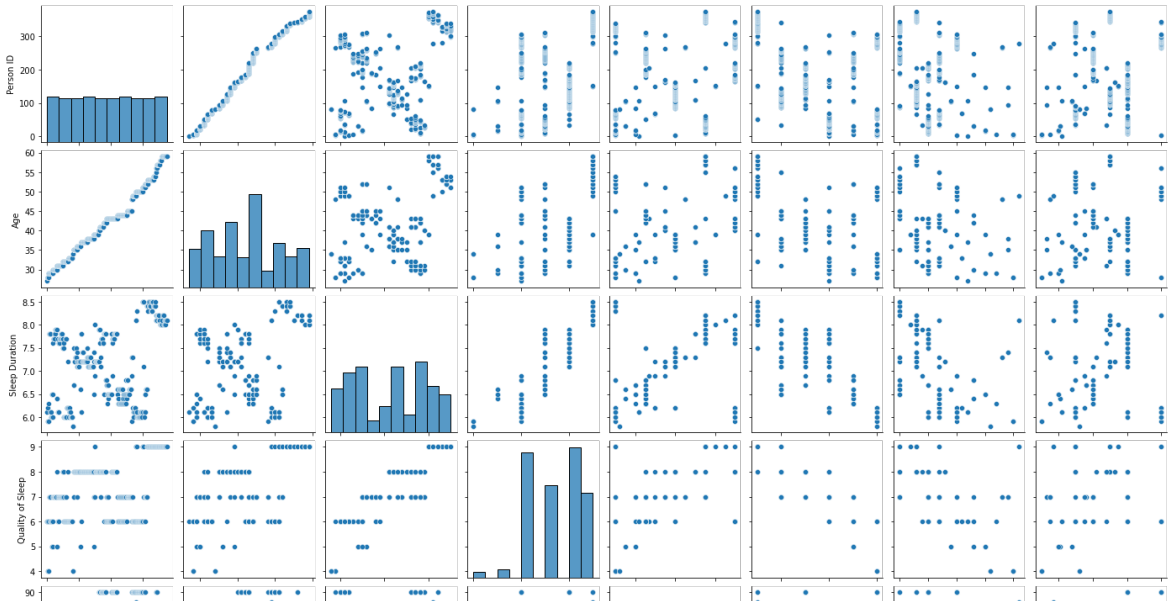
To display heading

In [6]:

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')
```

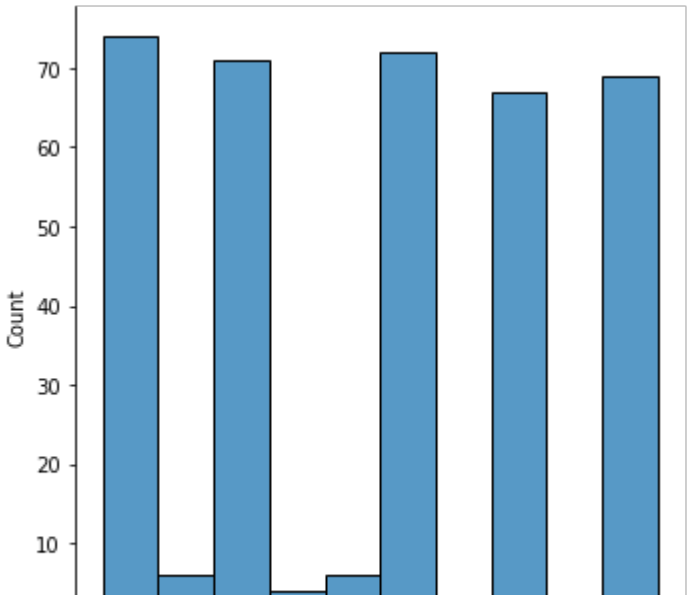
In [7]:

Out[7]: <seaborn.axisgrid.PairGrid at 0x1eb743a27f0>



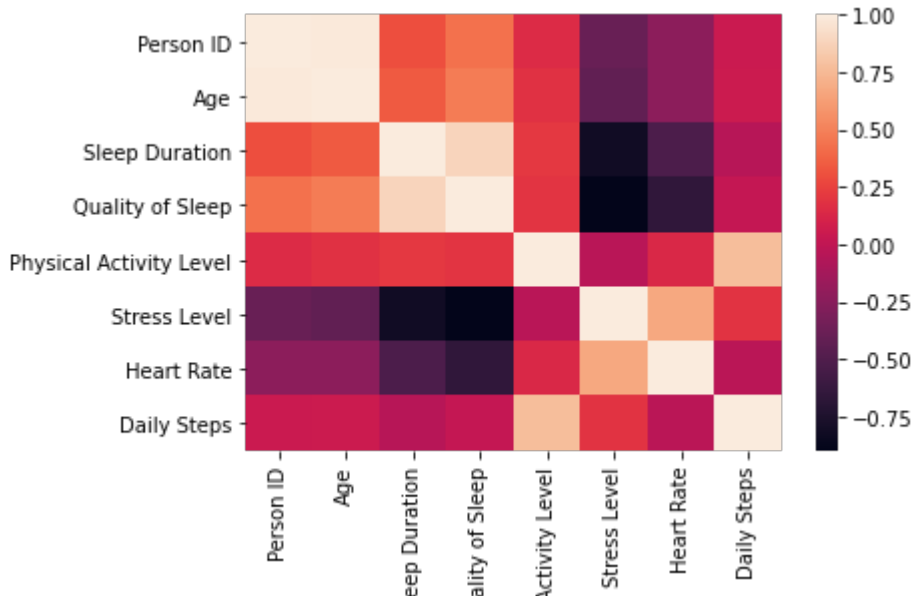
In [8]:

Out[8]: <seaborn.axisgrid.FacetGrid at 0x1eb770d8fd0>



In [9]:

Out[9]: <AxesSubplot:>



TO TRAIN THE MODEL - MODEL BUILDING

In [10]:
```
x = a[['Physical Activity Level']]
```

In [11]:
```
# to split my dataset into training and test data
from sklearn.model_selection import train_test_split
```

In [12]:
```
from sklearn.linear_model import LinearRegression
lr = LinearRegression()
```

Out[12]: LinearRegression()

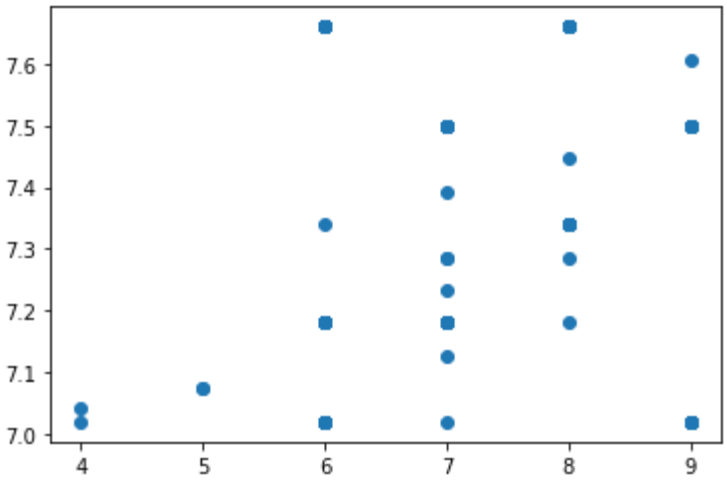
In [13]:
```
coeff = pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
```

Out[13]:

	Co-efficient
Physical Activity Level	0.010658

In [14]:
```
prediction= lr.predict(x_test)
```

Out[14]: <matplotlib.collections.PathCollection at 0x1eb7993e4f0>



In [15]:

Out[15]: 0.044168430888826826

RIDGE & LASSO

In [16]:
```
from sklearn.linear_model import Ridge,Lasso
rr=Ridge(alpha=10)
```

Out[16]: Ridge(alpha=10)

In [17]:

Out[17]: 0.044167566151201765

In [18]:
```
la=Lasso(alpha=10)
```

Out[18]: Lasso(alpha=10)

In [19]:

Out[19]: -0.003250264199169184

In [20]:
```
from sklearn.linear_model import ElasticNet
a=ElasticNet()
```

Out[20]: ElasticNet()

In [21]:
```
print(a.coef_)
print(a.intercept_)
print(a.score(x_test,y_test))
```

```
[0.00944102]
6.772913229464308
0.042603672087692
[7.05614368 7.62260459 7.48098936 7.33937414 7.05614368 7.48098936
 7.33937414 7.19775891 7.05614368 7.33937414 7.05614368 7.33937414
 7.62260459 7.62260459 7.48098936 7.10334876 7.62260459 7.19775891
 7.19775891 7.62260459 7.19775891 7.33937414 7.38657921 7.62260459
 7.05614368 7.33937414 7.10334876 7.62260459 7.05614368 7.62260459
 7.62260459 7.07502571 7.29216906 7.19775891 7.33937414 7.05614368
 7.48098936 7.05614368 7.24496398 7.19775891 7.62260459 7.43378429
 7.19775891 7.48098936 7.62260459 7.05614368 7.05614368 7.48098936
 7.19775891 7.33937414 7.19775891 7.62260459 7.19775891 7.48098936
 7.15055383 7.62260459 7.62260459 7.62260459 7.33937414 7.29216906
 7.62260459 7.48098936 7.05614368 7.48098936 7.48098936 7.19775891
 7.05614368 7.05614368 7.48098936 7.48098936 7.33937414 7.05614368
 7.19775891 7.05614368 7.62260459 7.48098936 7.33937414 7.48098936
 7.33937414 7.33937414 7.62260459 7.05614368 7.57539951 7.48098936
 7.33937414 7.33937414 7.33937414 7.19775891 7.48098936 7.62260459
 7.19775891 7.05614368 7.29216906 7.05614368 7.62260459 7.05614368
 7.19775891 7.48098936 7.05614368 7.05614368 7.19775891 7.05614368
 7.05614368 7.19775891 7.19775891 7.19775891 7.48098936 7.48098936
 7.48098936 7.62260459 7.05614368 7.62260459 7.19775891]
```

In [22]:
```
from sklearn import metrics
print(" Mean Absolute Error :",metrics.mean_absolute_error(y_test,prediction))
print(" Mean Squared Error :",metrics.mean_squared_error(y_test,prediction))
```

```
 Mean Absolute Error : 0.9655021576621973
 Mean Squared Error : 1.3536892548990882
 Root Mean Absolute Error : 0.9825996934979154
```