Sleep Analysis

The purpose of the project

- **Problem** :Sleep quality is one of the main products of a healthy lifestyle. We spend about a third of our lives sleeping — or at least we should. All kinds of food that sparkles in food.Unfortunately, many of us fall short of that goal. According to the National Institutes of Health, insomnia affects about one-third of the general population, making it the most common sleep disorder in the United States. Here's what you need to know about sleep.
- **Solving the problem** : With the development we are witnessing from artificial intelligence, machine learning models can be used and then trained on a set of training data, then tested on a set of test data, and the classifier predicts whether a person has sleep disorder or not based on the data to be entered.

Data features

- **Person ID**: An identifier for each individual.
- **Gender**: The gender of the person (Male/Female).
- **Age**: The age of the person in years.
- **Occupation**: The occupation or profession of the person.
- **Sleep Duration (hours)**: The number of hours the person sleeps per day.
- **Quality of Sleep (scale**: 1-10): A subjective rating of the quality of sleep, ranging from 1 to 10.
- **Physical Activity Level (minutes/day)**: The number of minutes the person engages in physical activity daily.
- **Stress Level (scale: 1-10)**: A subjective rating of the stress level experienced by the person, ranging from 1 to 10.
- **BMI Category**: The BMI category of the person (e.g., Underweight, Normal, Overweight).
- **Blood Pressure (systolic/diastolic)**: The blood pressure measurement of the person, indicated as systolic pressure over diastolic pressure.
- **Heart Rate (bpm)**: The resting heart rate of the person in beats per minute.
- **Daily Steps**: The number of steps the person takes per day.
- **Sleep Disorder**: The presence or absence of a sleep disorder in the person (None, Insomnia, Sleep Apnea).

1. Import libraries

In [ ]:

```
# Reading data
import pandas as pd

# Fixings warnings
```

```
import warnings
warnings.filterwarnings('ignore')

# For mathematical operations
import numpy as np

# Visualisation
import seaborn as sns
import plotly.express as px
from termcolor import colored
import matplotlib.pyplot as plt
import plotly.graph_objects as go
import plotly.figure_factory as ff

# Data spliting
from sklearn.model_selection import train_test_split

# For converting non-numeric data (String or Boolean) into numbers
from sklearn.preprocessing import LabelEncoder
```

2. Reading Data

In [ ]:

```
sleep_data = pd.read_csv('Sleep_health_and_lifestyle_dataset.csv')

sleep_data['Sleep Disorder'].fillna("None", inplace = True)
sleep_data
```

Out[ ]:

	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
0	1	Male	27	Software Engineer	6.1	6	42	6	Overweight	126/83	77	4200	None
1	2	Male	28	Doctor	6.2	6	60	8	Normal	125/80	75	10000	None
2	3	Male	28	Doctor	6.2	6	60	8	Normal	125/80	75	10000	None
3	4	Male	28	Sales Representative	5.9	4	30	8	Obese	140/90	85	3000	Sleep Apnea
4	5	Male	28	Sales Representative	5.9	4	30	8	Obese	140/90	85	3000	Sleep Apnea
...	...	...	...	...	...	...	...	...	...	...	...	...	...
369	370	Female	59	Nurse	8.1	9	75	3	Overweight	140/95	68	7000	Sleep Apnea
370	371	Female	59	Nurse	8.0	9	75	3	Overweight	140/95	68	7000	Sleep Apnea
371	372	Female	59	Nurse	8.1	9	75	3	Overweight	140/95	68	7000	Sleep Apnea
372	373	Female	59	Nurse	8.1	9	75	3	Overweight	140/95	68	7000	Sleep Apnea
373	374	Female	59	Nurse	8.1	9	75	3	Overweight	140/95	68	7000	Sleep Apnea

374 rows × 13 columns

3. Statistical information

- A visual and mathematical portrayal of information is statistics.**

- Data science is all about making calculations with data.**

- We make decisions based on that data using mathematical conditions known.**

In [ ]:

```
shape = sleep_data.shape
print('The dimention of data is :',shape)
```

```
The dimention of data is : (374, 13)
```

Observations

- Here **374 rows** , **13 coulmns**

In [ ]:

```
sleep_data.info() # for empty and type of values from 374 rows × 13 columns table
```

```
<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
```

- There are **5** columns are **string** the rest are **numeric** in terms of datatype.

- There aren't **null** values

pandas.io.formats.style.Styler.background_gradient

https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.io.formats.style.Styler. background_gradient.html

Colormaps in Matplotlib

https://matplotlib.org/stable/users/explain/colors/colormaps.html

In [ ]:

```
# for statistical info in number
sleep_data.describe().style.background_gradient(cmap='OrRd') #for colored output
```

Out[ ]:

	Person ID	Age	Sleep Duration	Quality of Sleep	Physical Activity Level	Stress Level	Heart Rate	Daily Steps
count	374.000000	374.000000	374.000000	374.000000	374.000000	374.000000	374.000000	374.000000
mean	187.500000	42.184492	7.132086	7.312834	59.171123	5.385027	70.165775	6816.844920
std	108.108742	8.673133	0.795657	1.196956	20.830804	1.774526	4.135676	1617.915679
min	1.000000	27.000000	5.800000	4.000000	30.000000	3.000000	65.000000	3000.000000
25%	94.250000	35.250000	6.400000	6.000000	45.000000	4.000000	68.000000	5600.000000
50%	187.500000	43.000000	7.200000	7.000000	60.000000	5.000000	70.000000	7000.000000
75%	280.750000	50.000000	7.800000	8.000000	75.000000	7.000000	72.000000	8000.000000
max	374.000000	59.000000	8.500000	9.000000	90.000000	8.000000	86.000000	10000.000000

In [ ]:

```
#  for statistical info including string values
sleep_data.describe(include='O')
```

Out[ ]:

	Gender	Occupation	BMI Category	Blood Pressure	Sleep Disorder
count	374	374	374	374	374
unique	2	11	4	25	3
top	Male	Nurse	Normal	130/85	None
freq	189	73	195	99	219

4. Exploratory Data Analysis (EDA)

- **EDA** is a step in the Data Analysis Process, where a number of techniques are used to better understand the dataset being used.
- **'Understanding the dataset'** can refer to a number of things , the relationships between them ,handling Missing values or human error and identifying outliers.

In [ ]:

```
# show names of all columns
columns_name = sleep_data.columns
columns_name
```

Out[ ]:

```
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 [ ]:

```
# number of values of each column
number_of_values = sleep_data.nunique()

print(number_of_values)
```

```
Person ID                  374
Gender                       2
Age                         31
Occupation                  11
Sleep Duration              27
Quality of Sleep             6
Physical Activity Level     16
Stress Level                 6
BMI Category                 4
Blood Pressure              25
Heart Rate                  19
Daily Steps                 20
Sleep Disorder               3
dtype: int64
```

```
In [ ]: display(plt.style.available)
```

```
['Solarize_Light2',
 '_classic_test_patch',
 '_mpl-gallery',
 '_mpl-gallery-nogrid',
 'bmh',
 'classic',
 'dark_background',
 'fast',
 'fivethirtyeight',
 'ggplot',
 'grayscale',
 'seaborn-v0_8',
 'seaborn-v0_8-bright',
 'seaborn-v0_8-colorblind',
 'seaborn-v0_8-dark',
 'seaborn-v0_8-dark-palette',
 'seaborn-v0_8-darkgrid',
 'seaborn-v0_8-deep',
 'seaborn-v0_8-muted',
 'seaborn-v0_8-notebook',
 'seaborn-v0_8-paper',
 'seaborn-v0_8-pastel',
 'seaborn-v0_8-poster',
 'seaborn-v0_8-talk',
 'seaborn-v0_8-ticks',
 'seaborn-v0_8-white',
 'seaborn-v0_8-whitegrid',
 'tableau-colorblind10']
```

Choosing color palettes

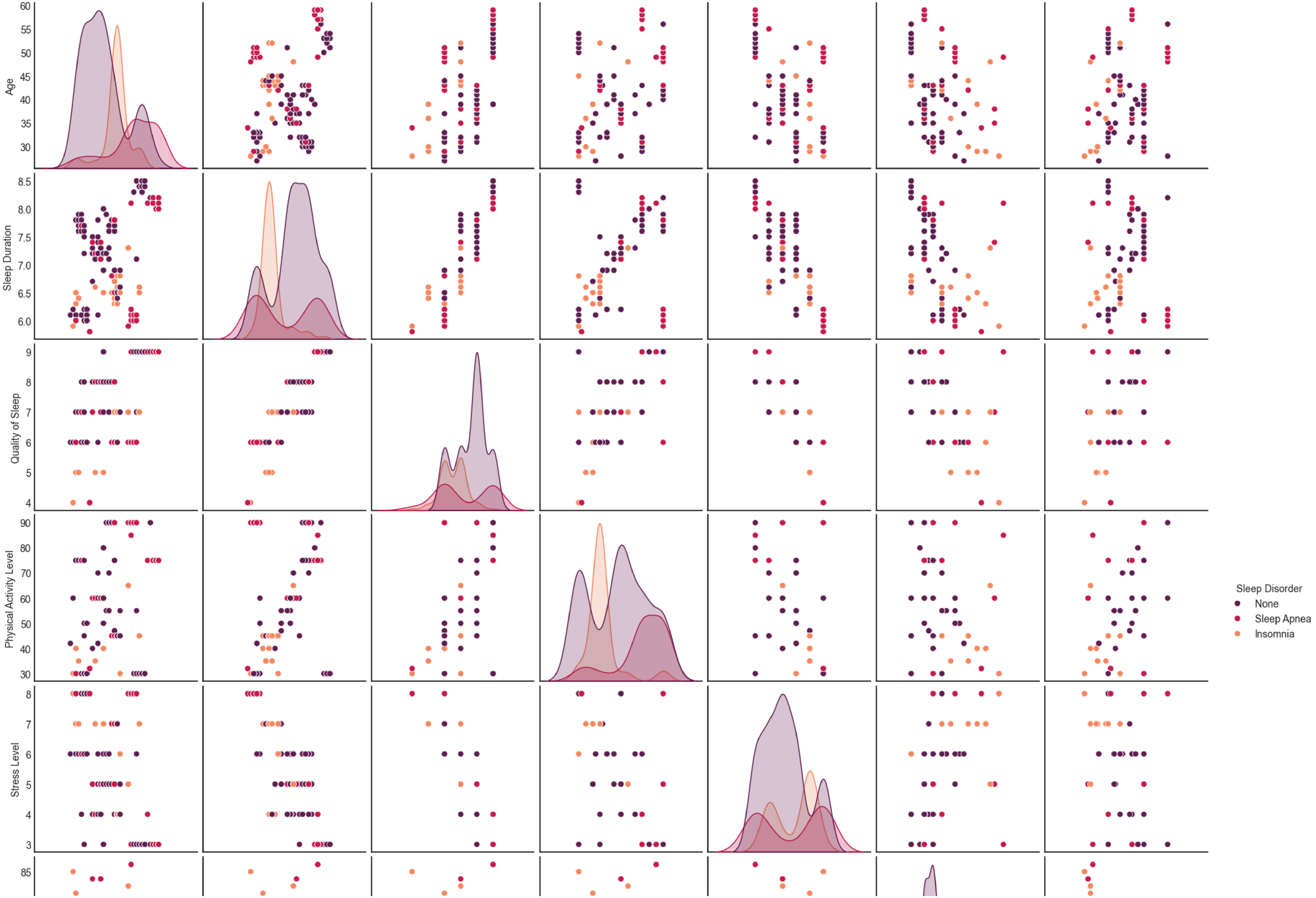
https://seaborn.pydata.org/tutorial/color_palettes.html

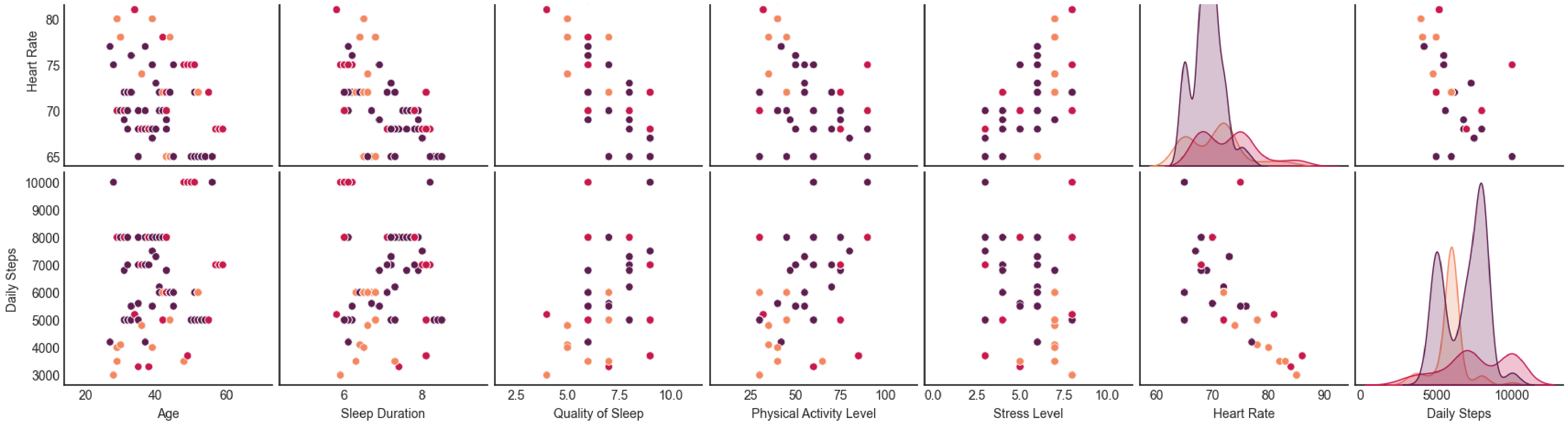
```
In [ ]: plt.style.use('seaborn-v0_8-white')
        sns.pairplot(data=sleep_data.drop('Person ID',axis=1),hue='Sleep Disorder',palette='rocket')

        # function add a legend
        plt.legend()
```

```
# function to show the plot
plt.show()
```

```
No artists with labels found to put in legend.  Note that artists whose label start with an underscore are ignored when legend() is called with no argument.
```


Age
60
55
50
45
40
35
30
Sleep Duration
8.5
8.0
7.5
7.0
6.5
6.0
Quality of Sleep
9
8
7
6
5
4
Physical Activity Level
90
80
70
60
50
40
30
Stress Level
8
7
6
5
4
3
85
Sleep Disorder
● None
● Sleep Apnea
● Insomnia



Percentage of persons have sleep disorder or not

In [ ]:

```
classes = sleep_data['Sleep Disorder'].unique()
print('The outputs from the classification are :',classes)
```

```
The outputs from the classification are : ['None' 'Sleep Apnea' 'Insomnia']
```

- **Sleep apnea** (โรคหยุดหายใจขณะหลับ) is a potentially serious sleep disorder in which breathing repeatedly stops and starts. If you snore loudly and feel tired even after a full night's sleep, you might have sleep apnea.
- **Insomnia** (โรคนอนไม่หลับ) s a sleep disorder in which you have trouble falling and/or staying asleep. With insomnia, you may have trouble falling asleep, staying asleep, or getting good quality sleep.

In [ ]:

```
sleep_data['Sleep Disorder'].value_counts()
```

Out[ ]:

```
Sleep Disorder
None           219
Sleep Apnea     78
Insomnia        77
Name: count, dtype: int64
```

Observations - Sleep Disorder

- It is clear that the proportion of normal people is more

In [ ]:

```
fig=px.histogram(sleep_data,x='Sleep Disorder',
                 barmode="group",color='Sleep Disorder',
                 color_discrete_sequence=['#aed581', '#f9bdbb','#e84e40'],
                 text_auto=True)


fig.update_layout(title='<b>Distribution of persons have sleep disorder or not</b>..',
                 title_font={'size':25},
                 paper_bgcolor='#fff8f7',
                 plot_bgcolor='#fff8f7',
                 showlegend=True)


fig.update_yaxes(showgrid=False)

fig.show()
```

Observations - Sleep Disorder & Sex

- It is clear that **Normal** men more than women
- It is clear that Men who suffer from **Insomnia** more than women
- It is clear that Women who suffer from **Sleep Apnea** more than women

In [ ]:

```
Gender = sleep_data['Gender'].unique()
print('The values of sex column are :', Gender)
```

```
The values of sex column are : ['Male' 'Female']
```

In [ ]:

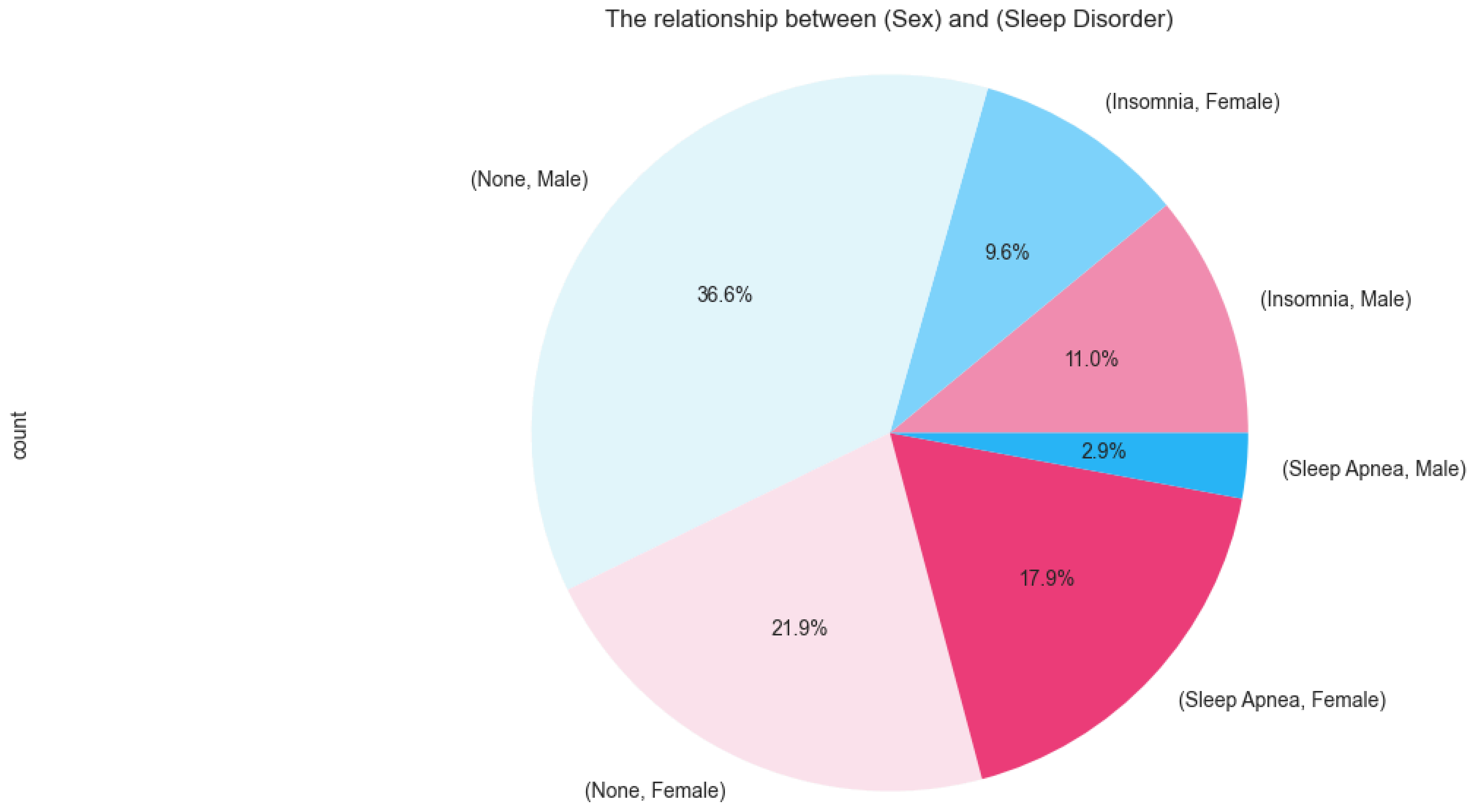
```
sleep_data.groupby('Sleep Disorder')['Gender'].value_counts()
```

Out[ ]:

```
Sleep Disorder  Gender
Insomnia        Male       41
                Female     36
None            Male      137
                Female     82
Sleep Apnea     Female     67
                Male       11
Name: count, dtype: int64
```

In [ ]:

```
sleep_data.groupby('Sleep Disorder')['Gender'].value_counts().plot.pie(autopct ='%1.1f%%',figsize=(15,7),
                                                                       colors=['#f48fb1','#81d4fa','#e1f5fe','#fce4ec','#ec407a','#29b6f6'])
plt.title('The relationship between (Sex) and (Sleep Disorder)')
plt.axis('equal')
plt.show()
```



Observations - Sleep Disorder & Occupation

- It is clear that in **Normal**, **Doctor** more than others
- It is clear that the people who suffer from **Insomnia**, **Salesperson** more than others
- It is clear that the people who suffer from **Sleep Apnea**, **Nurse** more than others

```
In [ ]: jobs = sleep_data['Occupation'].unique()
        print('The types of jobs that exist are :',jobs)
```

```
The types of jobs that exist are : ['Software Engineer' 'Doctor' 'Sales Representative' 'Teacher' 'Nurse'
 'Engineer' 'Accountant' 'Scientist' 'Lawyer' 'Salesperson' 'Manager']
```

```
In [ ]: sleep_data.groupby('Sleep Disorder')['Occupation'].value_counts()
```

Out[ ]:

```
Sleep Disorder  Occupation
Insomnia        Salesperson             29
                Teacher                 27
                Accountant               7
                Engineer                 5
                Nurse                    3
                Doctor                   3
                Lawyer                   2
                Software Engineer        1
None            Doctor                  64
                Engineer                57
                Lawyer                  42
                Accountant              30
                Nurse                    9
                Teacher                  9
                Software Engineer        3
                Salesperson              2
                Scientist                2
                Manager                  1
Sleep Apnea     Nurse                   61
                Doctor                   4
                Teacher                  4
                Lawyer                   3
                Sales Representative     2
                Scientist                2
                Engineer                 1
                Salesperson              1
Name: count, dtype: int64
```

In [ ]:

```python
fig = px.treemap(sleep_data.dropna(),path=[px.Constant('Jobs'),'Sleep Disorder','Occupation'],
               color='Sleep Disorder',
              color_discrete_sequence=['#e84e40','#aed581','#f9bdbb','#fde0dc'])

fig.update_layout(title='<b>The effect of job on sleep</b>',
                 title_font={'size':20})


fig.show()
```

Observations - Sleep Disorder & Quality of Sleep

In [ ]:

```python
sleep_data.pivot_table(index='Quality of Sleep',columns='Sleep Disorder',values='Sleep Duration',aggfunc='mean').style.background_gradient(cmap='OrRd')
```

Out[ ]:

Sleep Disorder	Insomnia	None	Sleep Apnea
Quality of Sleep			
4	5.900000	nan	5.850000
5	6.500000	nan	6.500000
6	6.371875	6.117500	6.118182
7	6.638235	7.540000	7.500000
8	7.520000	7.399010	7.366667
9	8.300000	8.365789	8.096875

In [ ]:

```
fig=px.sunburst(sleep_data.dropna(),path=[px.Constant('Sleep quality'),'Sleep Disorder','Quality of Sleep'],
               color='Sleep Disorder',values='Sleep Duration',
              color_discrete_sequence=['#aed581','#e84e40','#f9bdbb','#fde0dc'],
              hover_data=['Gender'])

fig.update_layout(title='<b>The effect of quality of sleep on sleep </b>',
                 title_font={'size':25})

fig.show()
```

Observations - Sleep Disorder & Physical Activity Level

In [ ]:

```
fig = px.violin(sleep_data, x="Sleep Disorder",y='Physical Activity Level',
                 color='Sleep Disorder',
                 color_discrete_sequence=['#aed581','#e84e40','#f9bdbb'],
                 violinmode='overlay')


fig.update_layout(title='<b>The effect of activities on sleep </b>..',
                 title_font={'size':25},
                 paper_bgcolor='#fff8f7',
                 plot_bgcolor='#fff8f7')

fig.update_yaxes(showgrid=False)
fig.show()
```

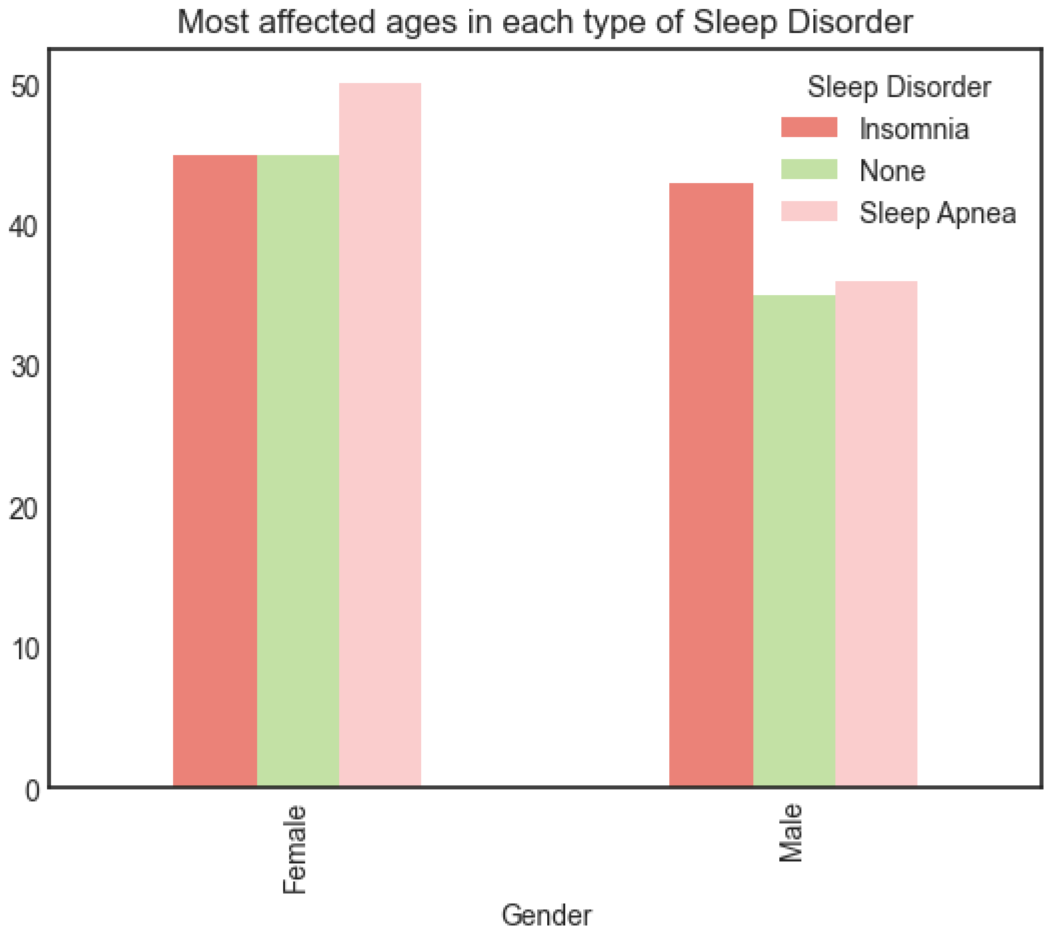
Observations - Sleep Disorder & Age

- ผญูที่อายุมากมีความเสี่ยงมากกว่าผู้ชายอายุมาก

In [ ]:

```
sleep_data.pivot_table(index='Gender',columns='Sleep Disorder',values='Age',aggfunc='median').plot(kind='bar',color=['#e84e40','#aed581','#f9bdbb'],
                                                                                                   title='Most affected ages in each type of Sleep Disorder',
                                                                                                    label='Age',alpha=.7)


plt.show()
```



- risk of Insomnia is higher than Sleep Apnea

```
In [ ]: fig=px.ecdf(sleep_data,x='Age',
                    color='Sleep Disorder',
                    color_discrete_sequence=['#aed581','#e84e40','#f9bdbb'])


        fig.update_layout(title='<b>The effect of ages on sleep </b>',
                          title_font={'size':25},
                          paper_bgcolor='#fff8f7',
                          plot_bgcolor='#fff8f7')


        fig.update_xaxes(showgrid=False)
        fig.update_yaxes(showgrid=False)
        fig.show()
```

Observations - Sleep Disorder & Sleep Duration

- Sleep duration of Sleep Apnea is higheer than Insomnia

```
In [ ]: fig = px.histogram(sleep_data,x='Sleep Disorder',y='Sleep Duration',
                          color='Sleep Disorder',color_discrete_sequence=['#aed581','#e84e40','#f9bdbb'],
                 text_auto=True)



fig.update_layout(title='<b>The effect of Sleep Duration on Sleep Disorder</b>',
                  titlefont={'size': 24,'family': 'Serif'},
                  showlegend=True,
                  paper_bgcolor='#fff8f7',
                  plot_bgcolor='#fff8f7')



fig.update_yaxes(showgrid=False)




fig.show()
```

Observations - Sleep Disorder & BMI Category, Blood Pressure, and Heart Rate

In [ ]:

```
fig=px.scatter_3d(sleep_data,x='BMI Category',y='Blood Pressure',z='Heart Rate',
                  color='Sleep Disorder',width=1000,height=900,
                  color_discrete_sequence=['#aed581','#e84e40','#f9bdbb'])


fig.update_layout(title='<b>The relationship between (BMI Category , Blood Pressure and Heart Rate) and their effect on  Sleep Disorder</b>',
                  titlefont={'size': 20,'family': 'Serif'},
                  showlegend=True)



fig.show()
```

Observations - Sleep Disorder & Stress Level

In [ ]:

```
sleep_data.pivot_table(index='Stress Level',columns='Sleep Disorder',aggfunc={'Sleep Disorder':'count'}).style.background_gradient(cmap='OrRd')
```

Out[ ]:

	Sleep Disorder		
Sleep Disorder	**Insomnia**	**None**	**Sleep Apnea**
Stress Level			
3	1	40	30
4	24	43	3
5	6	57	4
6	2	43	1
7	41	3	6
8	3	33	34

In [ ]:

```
fig=px.histogram(sleep_data,x='Sleep Disorder',
                 color='Sleep Disorder',
                 facet_col='Stress Level',
                 barmode='group',
                 color_discrete_sequence=['#aed581','#e84e40','#f9bdbb'],
                 opacity=.8)


fig.update_layout(title='<b>The effect of Stress Level on Sleep Disorder</b>',title_font={'size':30},
                  paper_bgcolor='#fff8f7',
                  plot_bgcolor='#fff8f7')



fig.update_yaxes(showgrid=False)
fig.show()
```

Observations - Sleep Disorder & BMI Category

- overweight people have the highest risk of sleep disorder

In [ ]:

```
BMI_Category = sleep_data['BMI Category'].unique()
print('The values of BMI Category column are :',BMI_Category)
```

```
The values of BMI Category column are : ['Overweight' 'Normal' 'Obese' 'Normal Weight']
```

In [ ]:

```
sleep_data.pivot_table(index='BMI Category',columns='Sleep Disorder',aggfunc={'Sleep Disorder':'count'}).style.background_gradient(cmap='OrRd')
```

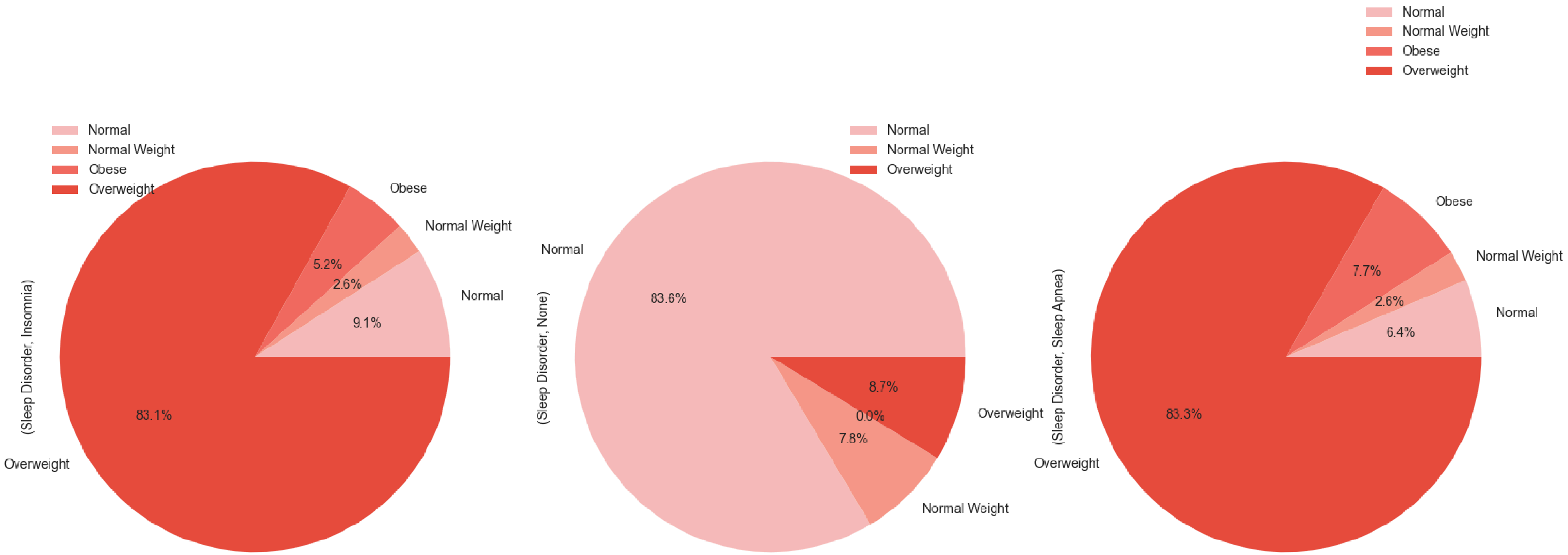
Out[ ]:

	Sleep Disorder		
Sleep Disorder	**Insomnia**	**None**	**Sleep Apnea**
BMI Category			
Normal	7.000000	183.000000	5.000000
Normal Weight	2.000000	17.000000	2.000000
Obese	4.000000	nan	6.000000
Overweight	64.000000	19.000000	65.000000

In [ ]:

```
sleep_data.pivot_table(index='BMI Category',columns='Sleep Disorder',aggfunc={'Sleep Disorder':'count'}).plot.pie(autopct ='%1.1f%%',
                                                                                                                  subplots=True,figsize=(20,10),
                                                                                                                  colors=['#f9bdbb','#f69988','#f36c60','#e84e40'])

plt.axis('equal')
plt.show()
```



5. Data preprocessing

- **Data preprocessing** refers to the technique of preparing (cleaning and organizing) the raw data to make it suitable for a building and training Machine Learning models.

Finding and cleaning aull values

- There is no null values

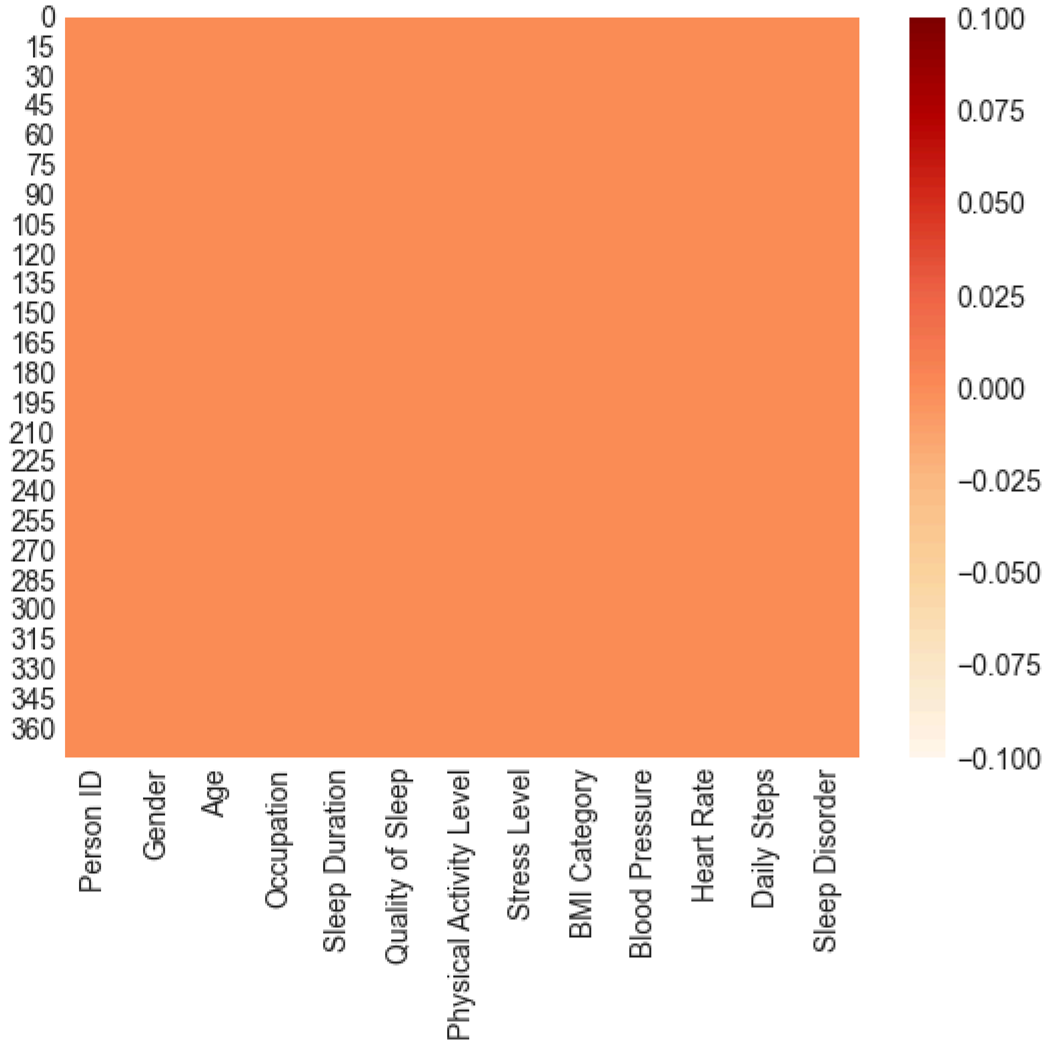
```
In [ ]: # checking null

        sleep_data.isna().sum()
```

```
Out[ ]: Person ID                  0
        Gender                     0
        Age                        0
        Occupation                 0
        Sleep Duration             0
        Quality of Sleep           0
        Physical Activity Level    0
        Stress Level               0
        BMI Category               0
        Blood Pressure             0
        Heart Rate                 0
        Daily Steps                0
        Sleep Disorder             0
        dtype: int64
```

```
In [ ]: sns.heatmap(sleep_data.isna(),cmap='OrRd')
```

```
Out[ ]: <Axes: >
```



Data Encoding

- turning all value into number

In [ ]:

```
sleep_data.columns
```

Out[ ]:

```
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 [ ]:

```
sleep_data['Blood Pressure'].unique()
```

Out[ ]:

```
array(['126/83', '125/80', '140/90', '120/80', '132/87', '130/86',
       '117/76', '118/76', '128/85', '131/86', '128/84', '115/75',
       '135/88', '129/84', '130/85', '115/78', '119/77', '121/79',
       '125/82', '135/90', '122/80', '142/92', '140/95', '139/91',
       '118/75'], dtype=object)
```

Note

- Ideal blood pressure **systolic (upper number)** : less than **120** , **diastolic (bottom number)** : less than **80**
- Normal **systolic (upper number)** : in range **(120 - 129)** , **diastolic (bottom number)** : in range **(80 - 84)**
- Otherwise, blood pressure is **high**

In [ ]:

```
sleep_table = sleep_data.copy()
sleep_table['Blood Pressure'] = sleep_table['Blood Pressure'].apply(lambda x:0 if x in ['120/80','126/83','125/80','128/84','129/84','117/76','118/76','115/75','125/82','122/80'] else 1)
# 0 = normal blood pressure
# 1 = abnormal blood pressure

sleep_table
```

Out[ ]:

	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
0	1	Male	27	Software Engineer	6.1	6	42	6	Overweight	0	77	4200	None
1	2	Male	28	Doctor	6.2	6	60	8	Normal	0	75	10000	None
2	3	Male	28	Doctor	6.2	6	60	8	Normal	0	75	10000	None
3	4	Male	28	Sales Representative	5.9	4	30	8	Obese	1	85	3000	Sleep Apnea
4	5	Male	28	Sales Representative	5.9	4	30	8	Obese	1	85	3000	Sleep Apnea
...	...	...	...	...	...	...	...	...	...	...	...	...	...
369	370	Female	59	Nurse	8.1	9	75	3	Overweight	1	68	7000	Sleep Apnea
370	371	Female	59	Nurse	8.0	9	75	3	Overweight	1	68	7000	Sleep Apnea
371	372	Female	59	Nurse	8.1	9	75	3	Overweight	1	68	7000	Sleep Apnea
372	373	Female	59	Nurse	8.1	9	75	3	Overweight	1	68	7000	Sleep Apnea
373	374	Female	59	Nurse	8.1	9	75	3	Overweight	1	68	7000	Sleep Apnea

374 rows × 13 columns

In [ ]:

```
sleep_table["Age"] = pd.cut(sleep_table["Age"],2)
sleep_table["Heart Rate"] = pd.cut(sleep_table["Heart Rate"],4)
sleep_table["Daily Steps"] = pd.cut(sleep_table["Daily Steps"],4)
sleep_table["Sleep Duration"] = pd.cut(sleep_table["Sleep Duration"],3)
sleep_table["Physical Activity Level"] = pd.cut(sleep_table["Physical Activity Level"],4)

sleep_table
```

Out[ ]:

	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
0	1	Male	(26.968, 43.0]	Software Engineer	(5.797, 6.7]	6	(29.94, 45.0]	6	Overweight	0	(75.5, 80.75]	(2993.0, 4750.0]	None
1	2	Male	(26.968, 43.0]	Doctor	(5.797, 6.7]	6	(45.0, 60.0]	8	Normal	0	(70.25, 75.5]	(8250.0, 10000.0]	None
2	3	Male	(26.968, 43.0]	Doctor	(5.797, 6.7]	6	(45.0, 60.0]	8	Normal	0	(70.25, 75.5]	(8250.0, 10000.0]	None
3	4	Male	(26.968, 43.0]	Sales Representative	(5.797, 6.7]	4	(29.94, 45.0]	8	Obese	1	(80.75, 86.0]	(2993.0, 4750.0]	Sleep Apnea
4	5	Male	(26.968, 43.0]	Sales Representative	(5.797, 6.7]	4	(29.94, 45.0]	8	Obese	1	(80.75, 86.0]	(2993.0, 4750.0]	Sleep Apnea
...	...	...	...	...	...	...	...	...	...	...	...	...	...
369	370	Female	(43.0, 59.0]	Nurse	(7.6, 8.5]	9	(60.0, 75.0]	3	Overweight	1	(64.979, 70.25]	(6500.0, 8250.0]	Sleep Apnea
370	371	Female	(43.0, 59.0]	Nurse	(7.6, 8.5]	9	(60.0, 75.0]	3	Overweight	1	(64.979, 70.25]	(6500.0, 8250.0]	Sleep Apnea
371	372	Female	(43.0, 59.0]	Nurse	(7.6, 8.5]	9	(60.0, 75.0]	3	Overweight	1	(64.979, 70.25]	(6500.0, 8250.0]	Sleep Apnea
372	373	Female	(43.0, 59.0]	Nurse	(7.6, 8.5]	9	(60.0, 75.0]	3	Overweight	1	(64.979, 70.25]	(6500.0, 8250.0]	Sleep Apnea
373	374	Female	(43.0, 59.0]	Nurse	(7.6, 8.5]	9	(60.0, 75.0]	3	Overweight	1	(64.979, 70.25]	(6500.0, 8250.0]	Sleep Apnea

374 rows × 13 columns

In [ ]:

```python
# converting non-numeric data (String or Boolean) into numbers

LE = LabelEncoder()

categories = ['Gender','Age','Occupation','Sleep Duration','Physical Activity Level','BMI Category','Heart Rate','Daily Steps','Sleep Disorder']
for label in categories:
    sleep_table[label] = LE.fit_transform(sleep_table[label])

sleep_table
```

Out[ ]:

	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
0	1	1	0	9	0	6	0	6	3	0	2	0	1
1	2	1	0	1	0	6	1	8	0	0	1	3	1
2	3	1	0	1	0	6	1	8	0	0	1	3	1
3	4	1	0	6	0	4	0	8	2	1	3	0	2
4	5	1	0	6	0	4	0	8	2	1	3	0	2
...	...	...	...	...	...	...	...	...	...	...	...	...	...
369	370	0	1	5	2	9	2	3	3	1	0	2	2
370	371	0	1	5	2	9	2	3	3	1	0	2	2
371	372	0	1	5	2	9	2	3	3	1	0	2	2
372	373	0	1	5	2	9	2	3	3	1	0	2	2
373	374	0	1	5	2	9	2	3	3	1	0	2	2

374 rows × 13 columns

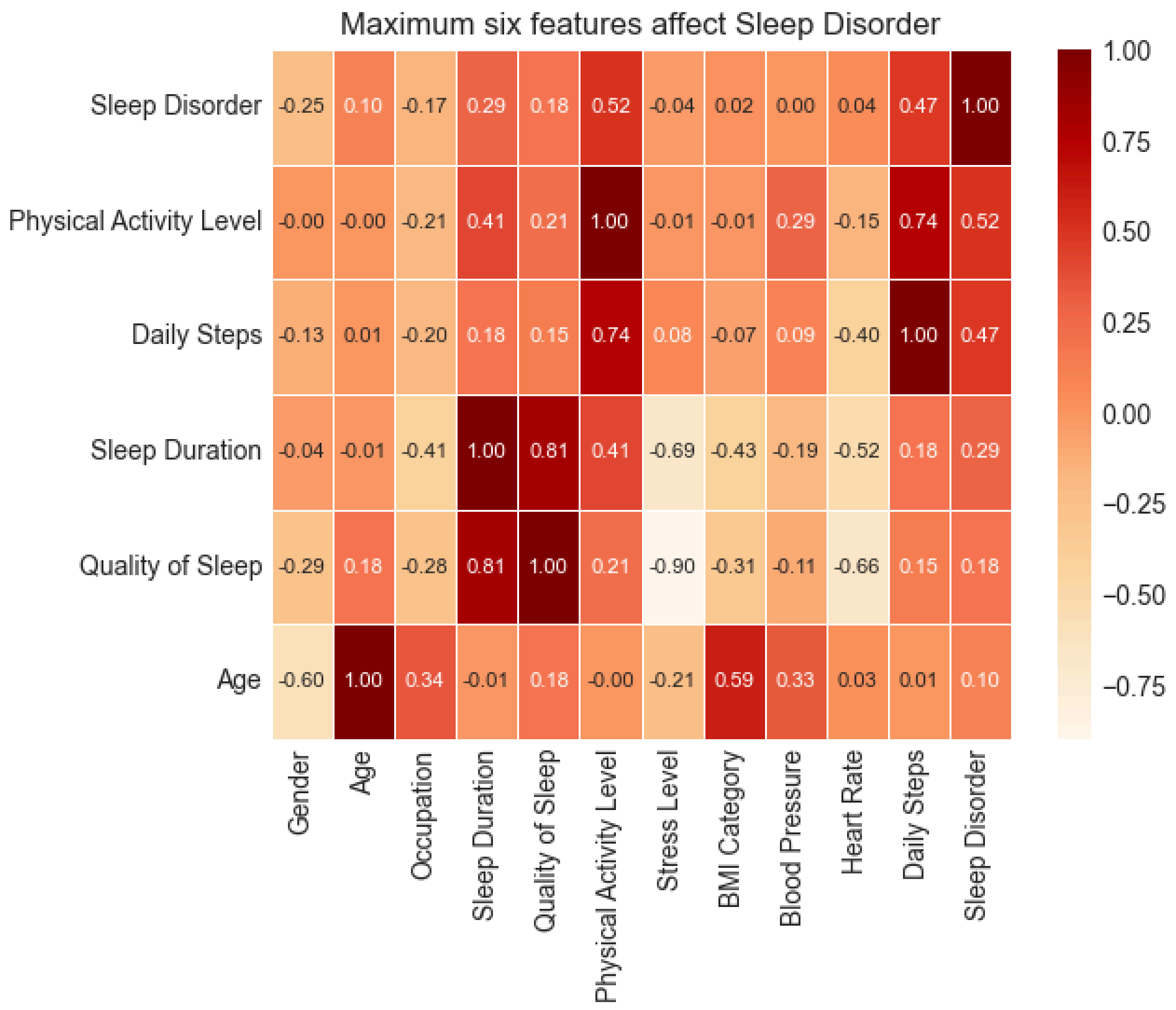
In [ ]:

```
# drop ID column

sleep_table.drop(['Person ID'], axis=1, inplace=True)
```

In [ ]:

```
correlation = sleep_table.corr()
max_6_corr = correlation.nlargest(6,"Sleep Disorder")
sns.heatmap(max_6_corr,annot=True,fmt=".2F",annot_kws={"size":8},linewidths=0.5,cmap='OrRd')
plt.title('Maximum six features affect Sleep Disorder')
plt.show()
```



Data Spliting

In [ ]:

```
x = sleep_table.iloc[:,:-1] # all rows, all columns except the last one
y = sleep_table.iloc[:,-1] # all rows, only the last column

x_shape = x.shape
y_shape = y.shape
print('The dimensions of x is : ',x_shape)
print('The dimensions of y is : ',y_shape)
```

```
The dimensions of x is :  (374, 11)
The dimensions of y is :  (374,)
```

In [ ]:

```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=.33, random_state=32, shuffle=True)
```

In [ ]:

```
x_train_shape=  x_train.shape
x_test_shape = x_test.shape
y_train_shape = y_train.shape
y_test_shape = y_test.shape

print("x train dimensions :",x_train_shape)
print("x test dimensions: ",x_test_shape)
print("y train dimensions :",y_train_shape)
print("y test dimensions :",y_test_shape)
```

```
x train dimensions : (250, 11)
x test dimensions:  (124, 11)
y train dimensions : (250,)
y test dimensions : (124,)
```

In [ ]:
```
x_train
```

Out[ ]:

	Gender	Age	Occupation	Sleep Duration	Quality of Sleep	Physical Activity Level	Stress Level	BMI Category	Blood Pressure	Heart Rate	Daily Steps
17	1	0	1	0	6	0	8	0	0	0	2
160	1	0	3	1	8	1	5	0	1	0	2
115	0	0	0	1	8	1	4	0	0	0	2
260	0	1	10	0	7	0	4	3	1	0	1
8	1	0	1	2	7	2	6	0	0	0	2
...	...	...	...	...	...	...	...	...	...	...	...
252	0	1	10	0	7	0	4	3	1	0	1
88	1	0	2	1	8	1	4	0	0	0	1
310	0	1	0	0	7	0	7	3	1	1	1
43	1	0	1	2	7	2	6	0	0	0	2
215	1	0	2	2	8	3	5	0	1	0	2

250 rows × 11 columns

In [ ]:
```
x_test
```

Out[ ]:

	Gender	Age	Occupation	Sleep Duration	Quality of Sleep	Physical Activity Level	Stress Level	BMI Category	Blood Pressure	Heart Rate	Daily Steps
107	1	0	2	2	8	2	4	1	0	0	2
56	1	0	1	2	7	2	6	0	0	0	2
358	0	1	5	2	9	2	3	3	1	0	2
60	1	0	1	0	6	0	8	0	0	1	1
271	0	1	5	0	6	3	8	3	1	1	3
...	...	...	...	...	...	...	...	...	...	...	...
282	0	1	5	0	6	3	8	3	1	1	3
291	0	1	5	0	6	3	8	3	1	1	3
289	0	1	5	0	6	3	8	3	1	1	3
238	1	1	7	0	6	0	7	3	1	1	1
348	0	1	5	2	9	2	3	3	1	0	2

124 rows × 11 columns

6. Data Modeling

6.1 LogisticRegression Model

In [ ]:

```python
# LogisticRegression for Data Modeling
from sklearn.linear_model import LogisticRegression

LR = LogisticRegression().fit(x_train,y_train)
```

In [ ]:

```python
LR_training_score = round(LR.score(x_train,y_train)*100,2)
LR_testing_score = round(LR.score(x_test,y_test)*100,2)

print(f"LR training score :",LR_training_score)
print("LR testing score :",LR_testing_score)
```

```
LR training score : 90.8
LR testing score : 91.94
```

In [ ]:

```python
LR_y_pred = LR.predict(x_test)
```

6.2 XGBClassifier Model

In [ ]:

```python
import xgboost
from xgboost import XGBClassifier
xgb = xgboost.XGBClassifier()
xgb.fit(x_train,y_train)
```

Out[ ]:

```
▾ XGBClassifier
XGBClassifier(base_score=None, booster=None, callbacks=None,
              colsample_bylevel=None, colsample_bynode=None,
              colsample_bytree=None, device=None, early_stopping_rounds=None,
              enable_categorical=False, eval_metric=None, feature_types=None,
              gamma=None, grow_policy=None, importance_type=None,
              interaction_constraints=None, learning_rate=None, max_bin=None,
              max_cat_threshold=None, max_cat_to_onehot=None,
              max_delta_step=None, max_depth=None, max_leaves=None,
              min_child_weight=None, missing=nan, monotone_constraints=None,
```

In [ ]:

```
xgb_training_score = round(xgb.score(x_train,y_train)*100,2)
xgb_testing_score = round(xgb.score(x_test,y_test)*100,2)
print("xgb training score :",xgb_training_score)
print("xgb testing score :",xgb_testing_score)
```

```
xgb training score : 93.2
xgb testing score : 91.13
```

In [ ]:

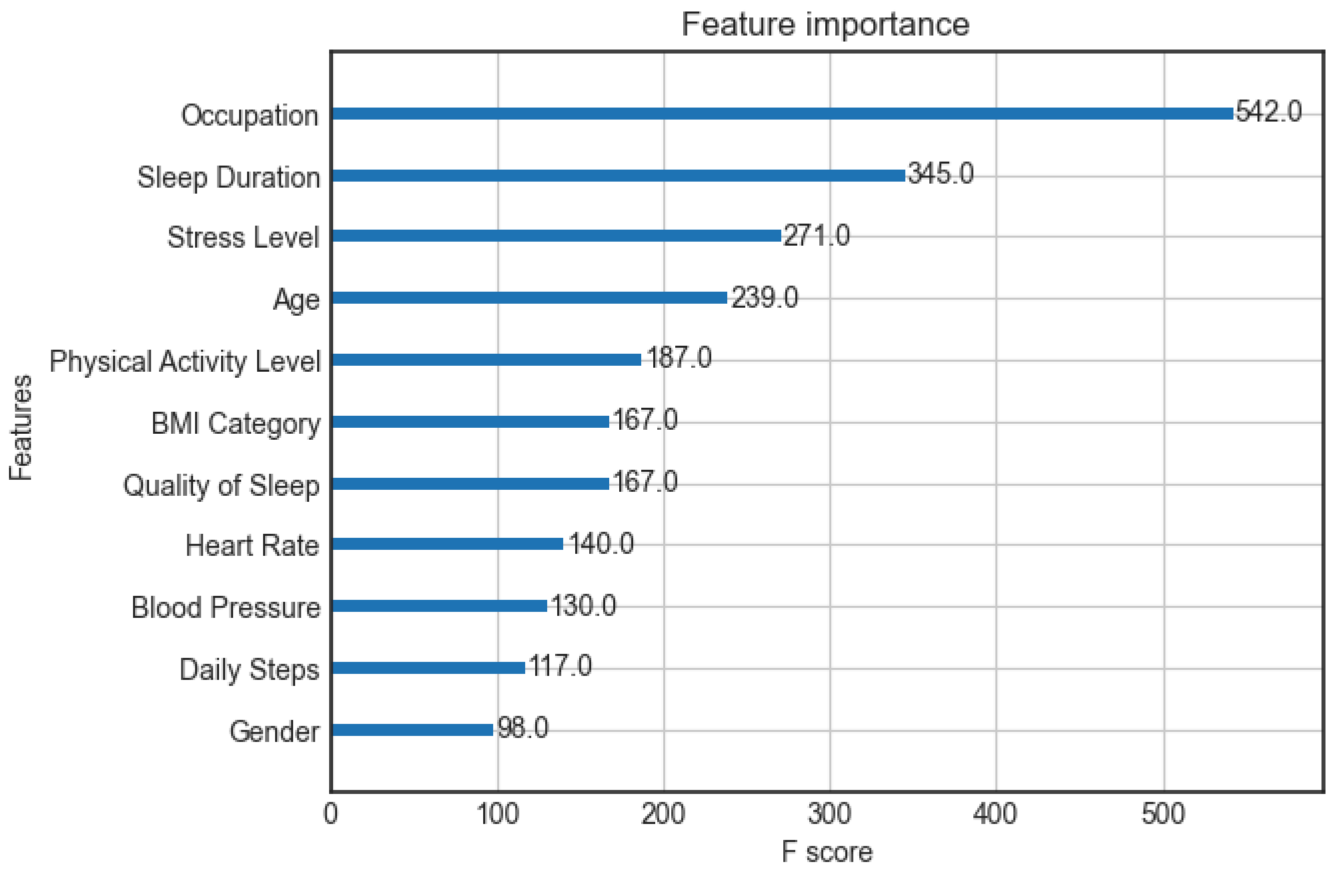
```
xgb_y_pred = xgb.predict(x_test)
```

In [ ]:

```
xgboost.plot_importance(xgb)
```

Out[ ]:

```
<Axes: title={'center': 'Feature importance'}, xlabel='F score', ylabel='Features'>
```



6.3 CatBoostClassifier Model

In [ ]:
```
from catboost import CatBoostClassifier
CBC = CatBoostClassifier(verbose=False).fit(x_train,y_train)
```

In [ ]:
```
CBC_training_score = round(CBC.score(x_train,y_train)*100,2)
CBC_testing_score = round(CBC.score(x_test,y_test)*100,2)

print("CBC training score :",CBC_training_score)
print("CBC testing score :",CBC_testing_score)
```

```
CBC training score : 93.2
CBC testing score : 91.13
```

In [ ]:
```
CBC_y_pred = CBC.predict(x_test)
```

6.4 GradientBoostingClassifier Model

In [ ]:
```
from sklearn.ensemble import GradientBoostingClassifier
GBC = GradientBoostingClassifier().fit(x_train,y_train)
```

In [ ]:
```
GBC_training_score = round(GBC.score(x_train,y_train)*100,2)
GBC_testing_score = round(GBC.score(x_test,y_test)*100,2)

print("GBC training score :",GBC_training_score)
print("GBC testing score :",GBC_testing_score)
```

```
GBC training score : 93.2
GBC testing score : 91.13
```

In [ ]:
```
GBC_y_pred = GBC.predict(x_test)
```

6.5 SVC Model

In [ ]:
```
from sklearn.svm import SVC
svc = SVC().fit(x_train,y_train)
```

In [ ]:
```
svc_training_score = round(svc.score(x_train,y_train)*100,2)
svc_testing_score = round(svc.score(x_test,y_test)*100,2)

print("svc training score :",svc_training_score)
print("svc testing score :",svc_testing_score)
```

```
svc training score : 88.8
svc testing score : 87.9
```

In [ ]:
```
svc_y_pred = svc.predict(x_test)
```

7. Models Evaluation

In [ ]:
```
from sklearn.metrics import confusion_matrix
```

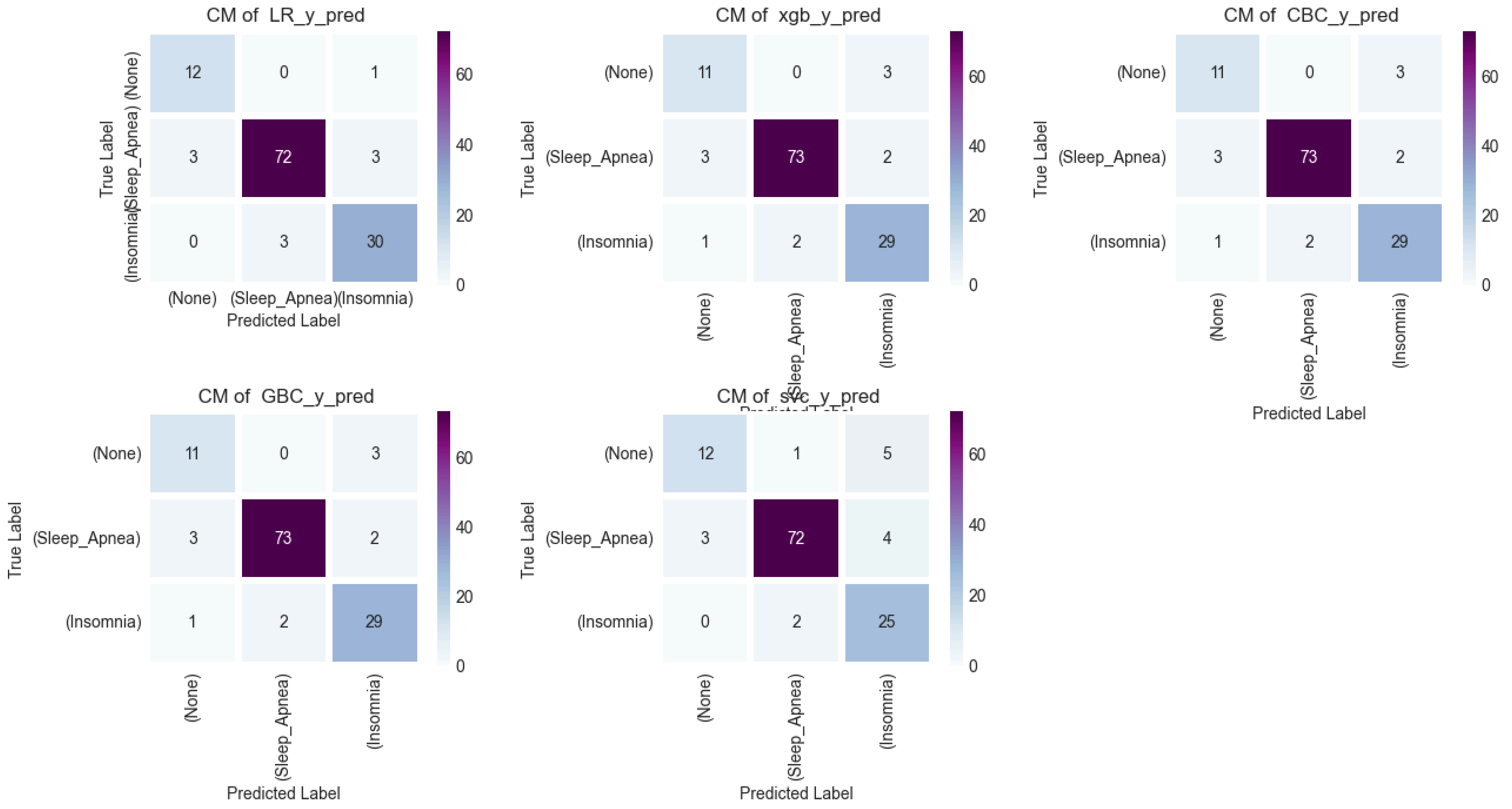
```
models_predictions = [LR_y_pred,xgb_y_pred,CBC_y_pred,GBC_y_pred,svc_y_pred]
model = {1:'LR_y_pred',2:'xgb_y_pred',3:'CBC_y_pred',4:'GBC_y_pred',5:'svc_y_pred'}


plt.figure(figsize=(15,7))
for i,y_pred in enumerate(models_predictions,1) :

    cm = confusion_matrix(y_pred,y_test)

    plt.subplot(2,3,i)
    sns.heatmap(cm,cmap='BuPu',linewidth=3,fmt='',annot=True,
                xticklabels=['(None)','(Sleep_Apnea)','(Insomnia)'],
                yticklabels=['(None)','(Sleep_Apnea)','(Insomnia)'])


    plt.title(' CM of  '+ model[i])
    plt.xlabel('Predicted Label')
    plt.ylabel('True Label')
    plt.subplots_adjust(hspace=0.5,wspace=0.5)
```



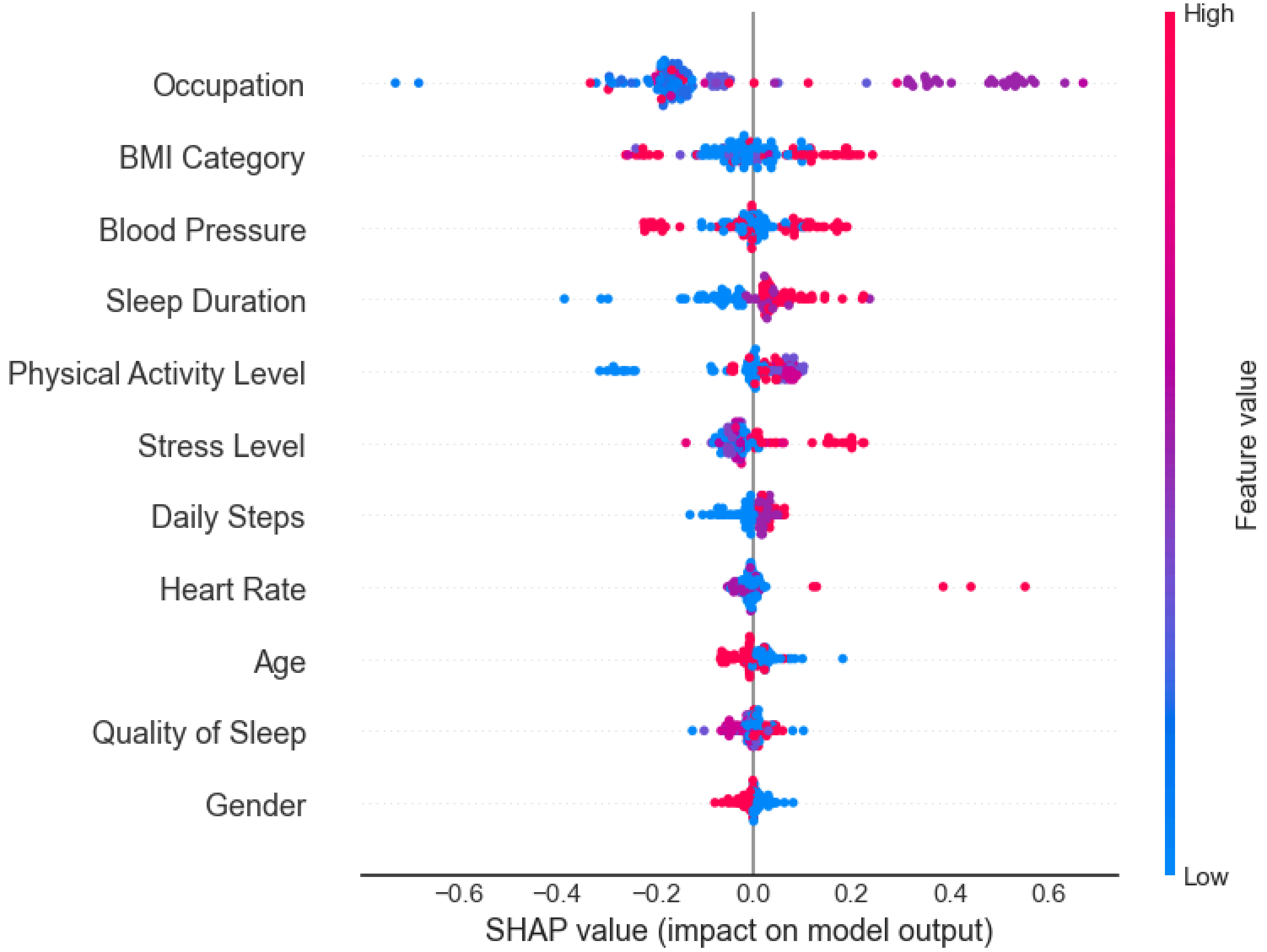
8. Interpretation one model

- XGBClassifier Model

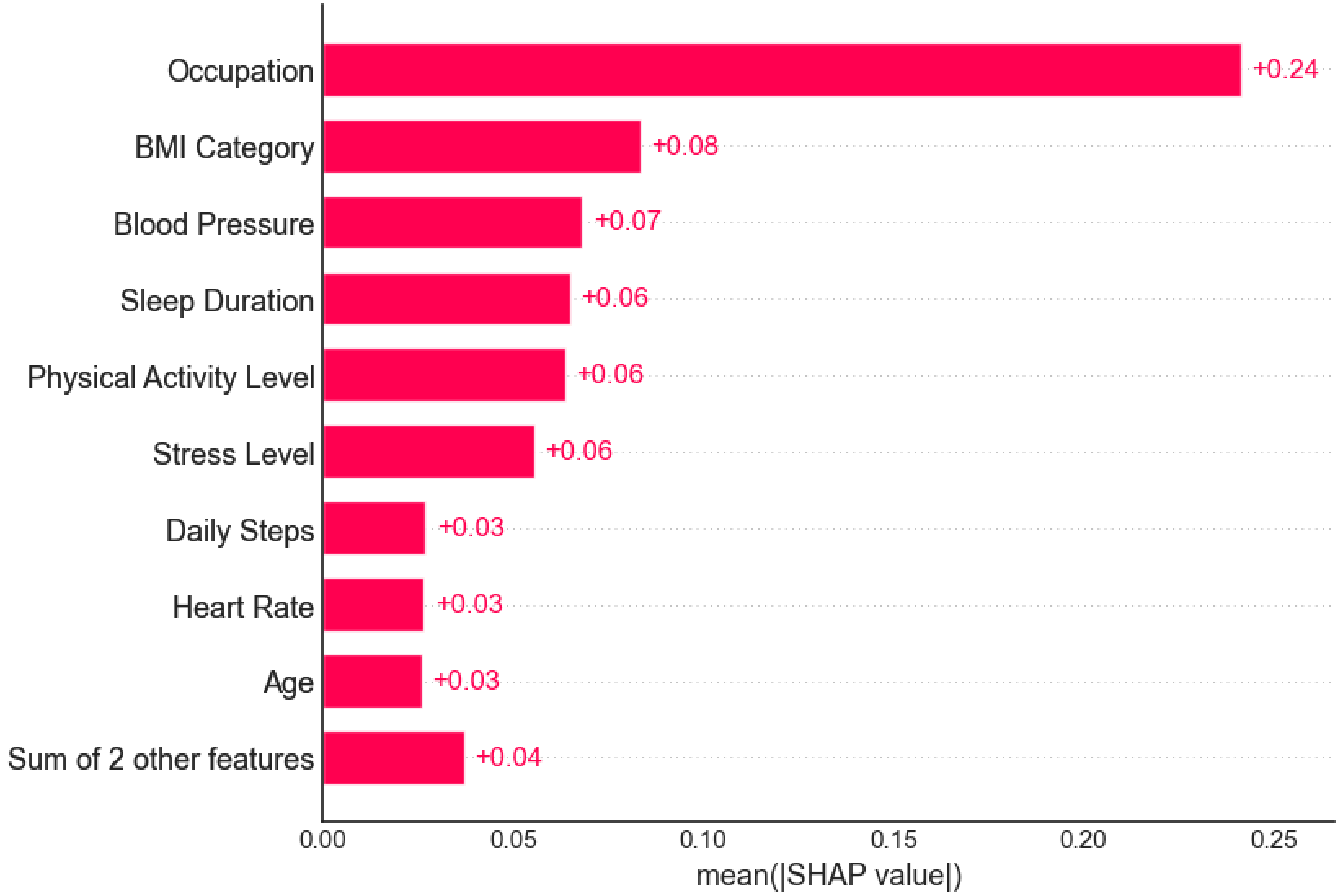
```
In [ ]: import shap

explainer = shap.Explainer(xgb.predict, x_test)
shap_values = explainer(x_test)
shap.summary_plot(shap_values, x_test,class_names=['None','Sleep_Apnea','Insomnia'])
```

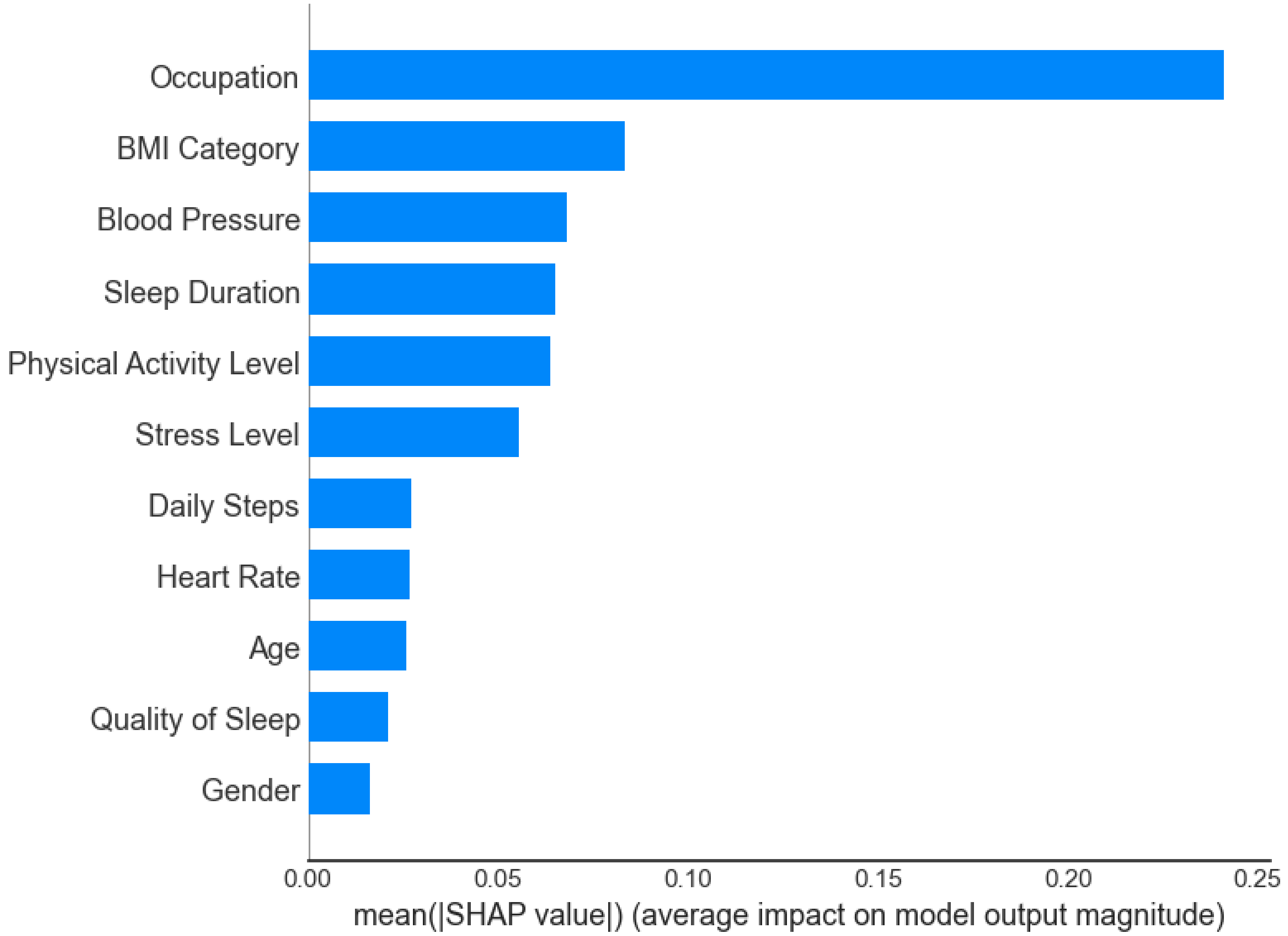
```
PermutationExplainer explainer: 125it [00:36,  2.40it/s]
```



```
In [ ]: shap.plots.bar(shap_values)
```



```
In [ ]: shap.summary_plot(shap_values, x_test,class_names=['None','Sleep_Apnea','Insomnia'],plot_type='bar')
```


Occupation
BMI Category
Blood Pressure
Sleep Duration
Physical Activity Level
Stress Level
Daily Steps
Heart Rate
Age
Quality of Sleep
Gender
0.00
0.05
0.10
0.15
0.20
0.25
mean(|SHAP value|) (average impact on model output magnitude)