```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
df=pd.read_csv('/content/Sleep_health_and_lifestyle_dataset.csv')
df
```

```
df.head()
```

	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

Next steps: Generate code with df View recommended plots

```
df.tail()
```

```
df.isna().sum()
```

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

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

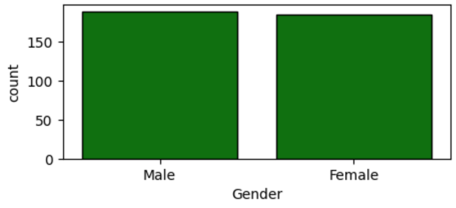
```
None           219
Sleep Apnea     78
Insomnia        77
Name: Sleep Disorder, dtype: int64
```

```
df.dtypes
```

```
Gender                      object
Age                          int64
Occupation                  object
Sleep Duration             float64
Quality of Sleep             int64
Physical Activity Level      int64
Stress Level                 int64
BMI Category                object
Blood Pressure              object
Heart Rate                   int64
Daily Steps                  int64
Sleep Disorder              object
dtype: object
```

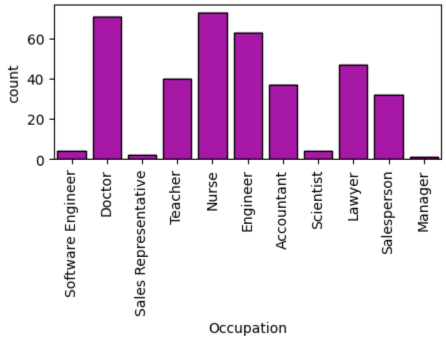
```
plt.figure(figsize=(5,2))
sns.countplot(x='Gender',data=df,color='g',edgecolor='k')
```

```
<Axes: xlabel='Gender', ylabel='count'>
```



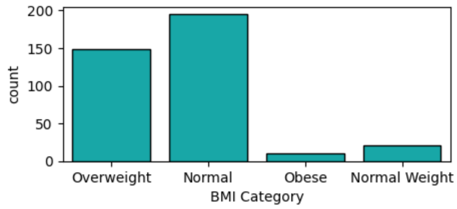
```
plt.figure(figsize=(5,2))
sns.countplot(x='Occupation',data=df,color='m',edgecolor='k')
plt.xticks(rotation='vertical')
```

```
([0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
 [Text(0, 0, 'Software Engineer'),
  Text(1, 0, 'Doctor'),
  Text(2, 0, 'Sales Representative'),
  Text(3, 0, 'Teacher'),
  Text(4, 0, 'Nurse'),
  Text(5, 0, 'Engineer'),
  Text(6, 0, 'Accountant'),
  Text(7, 0, 'Scientist'),
  Text(8, 0, 'Lawyer'),
  Text(9, 0, 'Salesperson'),
  Text(10, 0, 'Manager')])
```



```
plt.figure(figsize=(5,2))
sns.countplot(x='BMI Category',data=df,color='c',edgecolor='k')
```

```
<Axes: xlabel='BMI Category', ylabel='count'>
```



```
from sklearn.preprocessing import LabelEncoder
encode=LabelEncoder()
df['Gender']=encode.fit_transform(df['Gender'])
df['Occupation']=encode.fit_transform(df['Occupation'])
df['BMI Category']=encode.fit_transform(df['BMI Category'])
df['Blood Pressure']=encode.fit_transform(df['Blood Pressure'])
#df['Sleep Disorder']=encode.fit_transform(df['Sleep Disorder'])
df.dtypes
```

```
df.columns
```

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

```
corre=df.corr()
corre
```

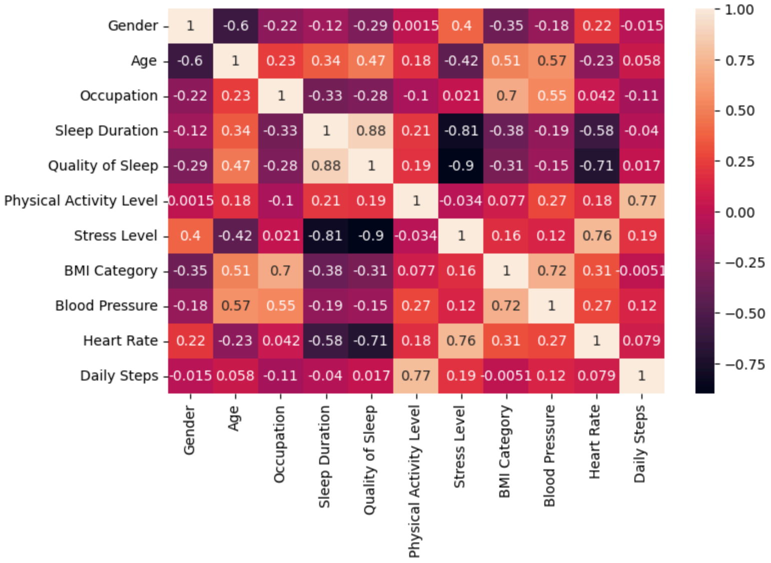
```
<ipython-input-261-3b7af5c18a19>:1: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid c
  corre=df.corr()
```

	Gender	Age	Occupation	Sleep Duration	Quality of Sleep	Physical Activity Level	Stress Level	BMI Category	Blood Pressure	Heart Rate	Daily Steps
Gender	1.000000	-0.596358	-0.219113	-0.121579	-0.291366	0.001454	0.396018	-0.352060	-0.176272	0.220026	-0.014509
Age	-0.596358	1.000000	0.231188	0.344709	0.473734	0.178993	-0.422344	0.511329	0.572813	-0.230374	0.057973
Occupation	-0.219113	0.231188	1.000000	-0.325775	-0.278071	-0.103660	0.021123	0.699504	0.546791	0.041571	-0.105877
Sleep Duration	-0.121579	0.344709	-0.325775	1.000000	0.883213	0.212360	-0.811023	-0.376358	-0.191704	-0.579284	-0.039533
Quality of Sleep	-0.291366	0.473734	-0.278071	0.883213	1.000000	0.192896	-0.898752	-0.312562	-0.147769	-0.709465	0.016791
Physical Activity Level	0.001454	0.178993	-0.103660	0.212360	0.192896	1.000000	-0.034134	0.077156	0.271913	0.176325	0.772723
Stress Level	0.396018	-0.422344	0.021123	-0.811023	-0.898752	-0.034134	1.000000	0.163895	0.117545	0.759798	0.186829
BMI Category	-0.352060	0.511329	0.699504	-0.376358	-0.312562	0.077156	0.163895	1.000000	0.724217	0.312890	-0.005059
Blood Pressure	-0.176272	0.572813	0.546791	-0.191704	-0.147769	0.271913	0.117545	0.724217	1.000000	0.272810	0.119455
Heart Rate	0.220026	-0.230374	0.041571	-0.579284	-0.709465	0.176325	0.759798	0.312890	0.272810	1.000000	0.078791
Daily Steps	-0.014509	0.057973	-0.105877	-0.039533	0.016791	0.772723	0.186829	-0.005059	0.119455	0.078791	1.000000

Next steps: Generate code with corre | View recommended plots

```
plt.figure(figsize=(8,5))
sns.heatmap(df.corr(),annot=True)
```

```
<ipython-input-259-f309de28f78d>:2: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid c
  sns.heatmap(df.corr(),annot=True)
<Axes: >
```



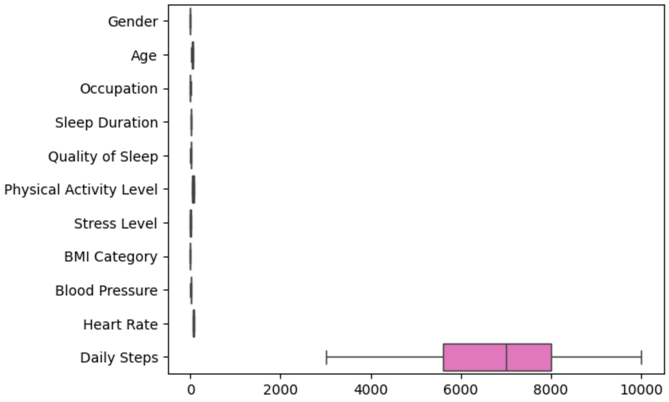
```
plt.figure(figsize=(5,5))
sns.boxplot( data = df, orient="h")
```

```
def iqr_rem(dfe,cols):
  for col in cols:
    q1=dfe[col].quantile(0.25)
    q3=dfe[col].quantile(0.75)
    IQR=q3-q1
    upper_bound=q3+(1.5*IQR)
    lower_bound=q1-(1.5*IQR)
    dfe[col]=dfe[col].clip(lower_bound,upper_bound)
```

```
featurs=df.columns[df.columns!='Sleep Disorder']
iqr_rem(df,featurs)
```

```
sns.boxplot(data=df,orient='h')
```

```
<Axes: >
```



```
x=df.iloc[:,:-1].values
x
```

```
y=df.iloc[:,-1].values
y
```

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.30,random_state=42)
x_train
y_train
```

```
from sklearn.preprocessing import StandardScaler
norm=StandardScaler()
norm.fit(x_train)
x_train=norm.transform(x_train)
x_test=norm.transform(x_test)
```

```
from sklearn.tree import DecisionTreeClassifier
model=DecisionTreeClassifier(criterion='entropy')
model.fit(x_train,y_train)
y_pred=model.predict(x_test)
y_pred
```

```
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix,ConfusionMatrixDisplay
score=accuracy_score(y_test,y_pred)
score
```

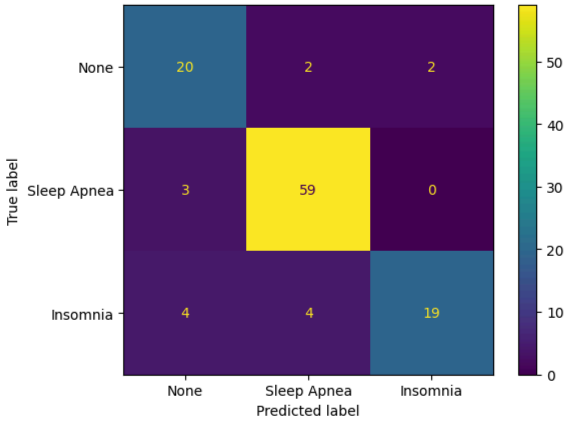
```
0.8672566371681416
```

```
cm=confusion_matrix(y_test,y_pred)
cm
```

```
array([[20,  2,  2],
       [ 3, 59,  0],
       [ 4,  4, 19]])
```

```
plt.figure(figsize=(5,5))
labels=['None', 'Sleep Apnea',  'Insomnia']
cmd=ConfusionMatrixDisplay(cm,display_labels=labels)
cmd.plot()
```

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7eab46baa500>
<Figure size 500x500 with 0 Axes>
```



```
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

    Insomnia       0.74      0.83      0.78        24
        None       0.91      0.95      0.93        62
 Sleep Apnea       0.90      0.70      0.79        27

    accuracy                           0.87       113
   macro avg       0.85      0.83      0.84       113
weighted avg       0.87      0.87      0.87       113
```

```
from sklearn import tree
plt.figure(figsize=(15,15))
tree.plot_tree(model,feature_names=['Gender', 'Age', 'Occupation', 'Sleep Duration', 'Quality of Sleep',
       'Physical Activity Level', 'Stress Level', 'BMI Category',
       'Blood Pressure', 'Heart Rate', 'Daily Steps', 'Sleep Disorder'],
               class_names=['None', 'Sleep Apnea',  'Insomnia'],filled=True)
```

```
#Hyper Parameter Tuning
model1=DecisionTreeClassifier()
```

```
from sklearn.model_selection import GridSearchCV
param={'criterion':['gini', 'entropy', 'log_loss'],'splitter':['best', 'random']}
clf=GridSearchCV(model1,param,cv=10,scoring='accuracy')
clf.fit(x_train,y_train)
```

```
▸            GridSearchCV
▸ estimator: DecisionTreeClassifier
     ▸ DecisionTreeClassifier
```

```
clf.best_params_
```

```
{'criterion': 'entropy', 'splitter': 'best'}
```

```
model2=DecisionTreeClassifier(criterion='log_loss',splitter='best')
model2.fit(x_train,y_train)
y_pred1=model2.predict(x_test)
y_pred1
```

```
print("Score after tuning",accuracy_score(y_test,y_pred))
```