Sleep Disorder Prediction

By : Sampurna Dey

Linkedin: www.linkedin.com/in/sampurna-dey-060573129

Github: https://github.com/sampurna-project

```
#importing the libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

```
#loading the dataset
Sleephealth = pd.read_csv('/content/Sleep_health_and_lifestyle_dataset.csv')
Sleephealth.head()
```

	Person ID	Gender	Age	Occupation	Sleep Duration	Quality of Sleep	Physical Activity Level	Stress Level	B Catego
0	1	Male	27	Software Engineer	6.1	6	42	6	Overweig
1	2	Male	28	Doctor	6.2	6	60	8	Norm
2	3	Male	28	Doctor	6.2	6	60	8	Norm
3	4	Male	28	Sales Representative	5.9	4	30	8	Obe
4	5	Male	28	Sales Representative	5.9	4	30	8	Obe

Data Preprocessing

```
#checking for missing values
Sleephealth.isnull().sum()
```

```
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
```

```
#replacing the null values with 'None' in the column 'Sleep Disorder'
Sleephealth['Sleep Disorder'].fillna('None', inplace=True)
```

The nan/None value in sleep disorder stands for no sleep disorder, so it is not a missing value

```
#drop column Person ID
Sleephealth.drop('Person ID', axis=1, inplace=True)
```

```
#checking the number of unique values in each column
print("Unique values in each column are:")
for col in Sleephealth.columns:
    print(col,Sleephealth[col].nunique())
```

```
Unique values in each column are:
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
```

Splitting the blood pressure into systolic and diastolic

```
#spliting the blood pressure into two columns
Sleephealth['systolic_bp'] = Sleephealth['Blood Pressure'].apply(lambda x: x.split('/')[0])
Sleephealth['diastolic_bp'] = Sleephealth['Blood Pressure'].apply(lambda x: x.split('/')[1])
#droping the blood pressure column
Sleephealth.drop('Blood Pressure', axis=1, inplace=True)
```

```
#replacing normal weight with normal in BMI column
Sleephealth['BMI Category'] = Sleephealth['BMI Category'].replace('Normal Weight', 'Normal')
```

```
Sleephealth.head()
```

	Gender	Age	Occupation	Sleep Duration	Quality of Sleep	Physical Activity Level	Stress Level	BMI Category	Hear Rat
0	Male	27	Software Engineer	6.1	6	42	6	Overweight	7
1	Male	28	Doctor	6.2	6	60	8	Normal	7
2	Male	28	Doctor	6.2	6	60	8	Normal	7
3	Male	28	Sales Representative	5.9	4	30	8	Obese	8
4	Male	28	Sales Representative	5.9	4	30	8	Obese	8

Checking the unique values from each categorical column

```
#unique values from categorical columns
print(Sleephealth.Occupation.unique())
print('\n')
print(Sleephealth['BMI Category'].unique())
print('\n')
print(Sleephealth['Sleep Disorder'].unique())
```

```
['Software Engineer' 'Doctor' 'Sales Representative' 'Teacher' 'Nurse'
 'Engineer' 'Accountant' 'Scientist' 'Lawyer' 'Salesperson' 'Manager']


['Overweight' 'Normal' 'Obese']


['None' 'Sleep Apnea' 'Insomnia']
```

The EDA is divided into two phases:

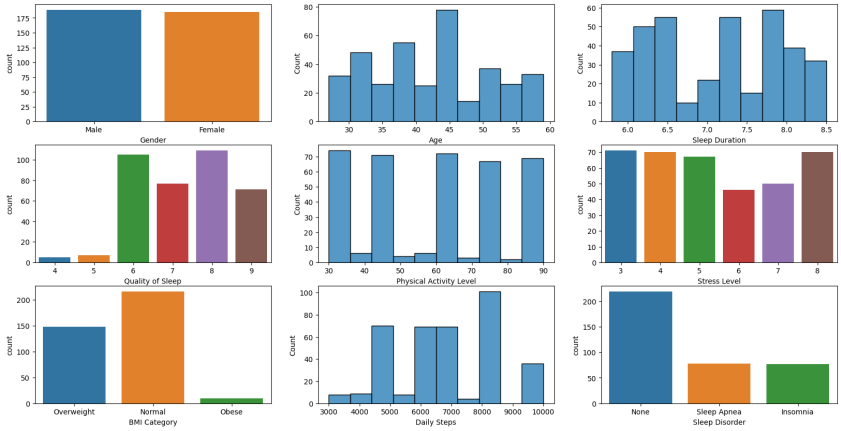
Phase 1: Understanding the data by plotting its variables

Phase 2: Understanding the correlation between the variables

Phase 1

```
fig,ax = plt.subplots(3,3,figsize=(20,10))
sns.countplot(x = 'Gender', data = Sleephealth, ax = ax[0,0])
sns.histplot(x = 'Age', data = Sleephealth, ax = ax[0,1], bins = 10)
sns.histplot(x = 'Sleep Duration', data = Sleephealth, ax = ax[0,2], bins = 10)
sns.countplot(x = 'Quality of Sleep', data = Sleephealth, ax = ax[1,0])
sns.histplot(x = 'Physical Activity Level', data = Sleephealth, ax = ax[1,1], bins = 10)
sns.countplot(x = 'Stress Level', data = Sleephealth, ax = ax[1,2])
sns.countplot(x = 'BMI Category', data = Sleephealth, ax = ax[2,0])
sns.histplot(x = 'Daily Steps', data = Sleephealth, ax = ax[2,1], bins = 10)
sns.countplot(x = 'Sleep Disorder', data = Sleephealth, ax = ax[2,2])
```

```
<Axes: xlabel='Sleep Disorder', ylabel='count'>
```



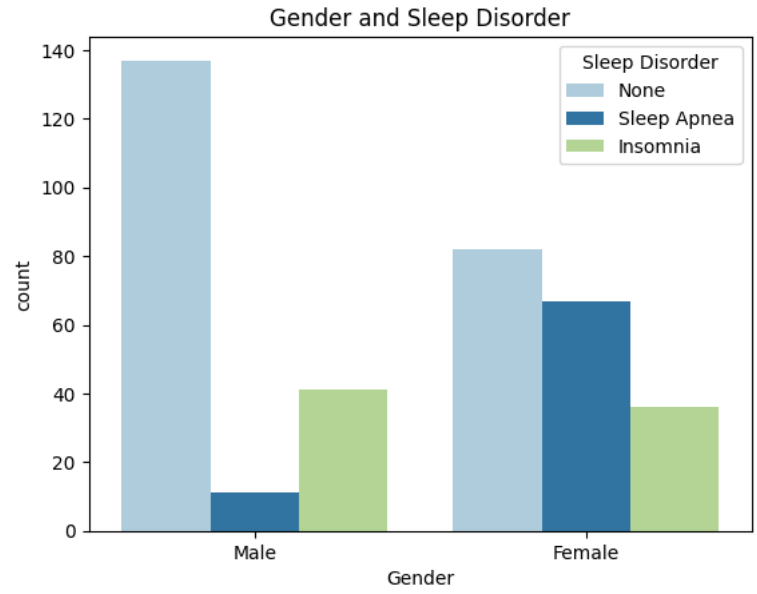
The number of males and females is almost equal, out of which majority of the people have age between 30-45 years. Most of the people have sleep quality greater than 5 which means there are getting sufficient sleep. Moreover, most of the people have normal BMI whci directly relates with the distribution of sleep disorder which shows equal number of people with and without sleep disorder.

Phase 2

Gender and Sleep Disorder

```
#Gender count plot
sns.countplot(x = 'Gender', data = Sleephealth, palette = 'Paired', hue = 'Sleep Disorder').set_title('Gender and Sleep Disorder')
```

```
Text(0.5, 1.0, 'Gender and Sleep Disorder')
```

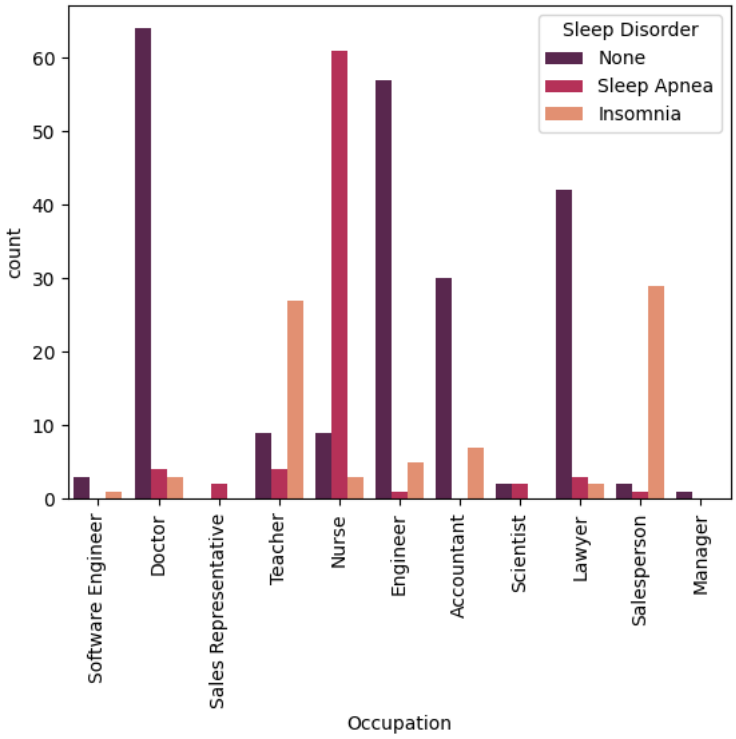


Most of the males and females are not suffering from any sleep disorder. However females tend to have more sleep disorder as compared to males. The number of females suffering from Sleep Apnea is quite high as compared to males. But in contrast to that, greater number of males are suffering from Insomia as compared to females.

Effect of Occupation on Sleep Disorder

```
ax = sns.countplot(x = 'Occupation', data = Sleephealth,palette = 'rocket', hue = 'Sleep Disorder')
ax.set_xticklabels(ax.get_xticklabels(), rotation = 90)
```

```
[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')]
```



From the graph it is clear that the occupation has huge impact on the sleep disorder. Nurses are more subjected to have Sleep Apenea as compared to other occupations and very few of them have no sleep disorder. After nurses, the next most affected occupation is the Salesperson, which counts for the most suffering from Insomia followed by teachers. However there are some occupations where most of the people have very few instance of Sleep Apenea and Insomia such as Engineers, Doctors, Accountants, Lawyers. The Software ENgineers and Managers are so less in number so I cannot say much about that, But the occupation Sales Representative has shown only Sleep Apenea and no Insomia or No sleep disorder.

BMI and Sleep Disorder

```
sns.countplot(x = 'BMI Category', hue = 'Sleep Disorder', data = Sleephealth, palette = 'icefire').set_title('BMI Category and Sleep Disc
```

```
Text(0.5, 1.0, 'BMI Category and Sleep Disorder')
```

People with normal BMI are less likely to suffer from any sleep disorder. However, this is opposite in case of Overweight and Obese people. Overweight are more likely to suffer more from sleep disordera than Obese people.

Data Preprocessing Part 2

Label Encoding for categorical variables

```
from sklearn import preprocessing
label_encoder = preprocessing.LabelEncoder()
```

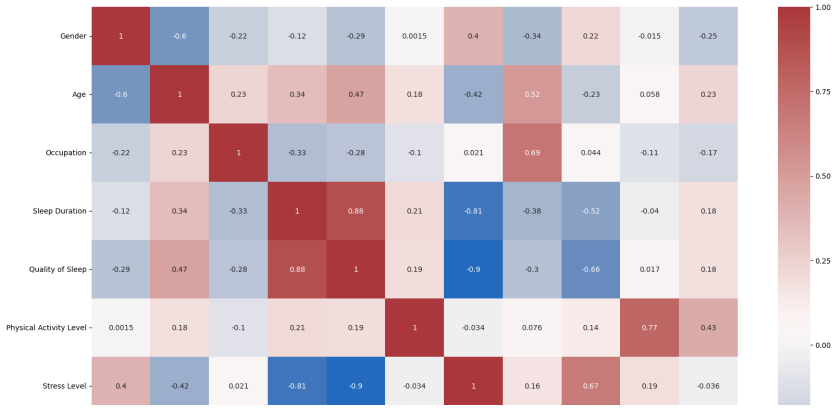
```
vars = ['Gender', 'Occupation','BMI Category','Sleep Disorder']
for i in vars:
    label_encoder.fit(Sleephealth[i].unique())
    Sleephealth[i] = label_encoder.transform(Sleephealth[i])
    print(i,':' ,Sleephealth[i].unique())
```

```
 Gender : [1 0]
 Occupation : [ 9  1  6 10  5  2  0  8  3  7  4]
 BMI Category : [2 0 1]
 Sleep Disorder : [1 2 0]
```

Correlation Matrix Heatmap

```
#Correlation Matrix Heatmap
plt.figure(figsize=(20, 16))
sns.heatmap(Sleephealth.corr(), annot = True, cmap = 'vlag')
```

```
<ipython-input-25-d6865e006031>:3: FutureWarning: The default value of numeric_onl
  sns.heatmap(Sleephealth.corr(), annot = True, cmap = 'vlag')
<Axes: >
```



Train Test Split


```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(Sleephealth.drop('Sleep Disorder',axis=1), Sleephealth['Sleep Disorder'], test_size=0
```

Model Building

For predictiong the sleep disorder thriugh classification algorithms I will use the following algorithms:

1. Decision Tree Classifier
2. Random Forest Classifier

Decision Tree Classifier

```
from sklearn.tree import DecisionTreeClassifier
dtree = DecisionTreeClassifier()
dtree
```

```
▾ DecisionTreeClassifier
DecisionTreeClassifier()
```

Training the model with train dataset

```
dtree.fit(X_train, y_train)
```

```
▾ DecisionTreeClassifier
DecisionTreeClassifier()
```

```
#training accuracy
print("Training Accuracy:",dtree.score(X_train,y_train))
```

```
Training Accuracy: 0.9348659003831418
```

Decision Tree Model Evalution

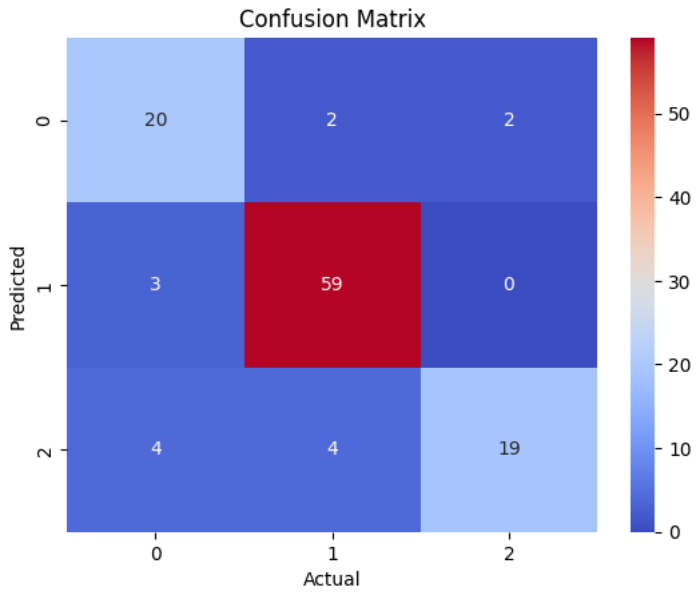
```
d_pred = dtree.predict(X_test)
d_pred
```

```
array([1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 2, 1, 0, 1, 1, 1, 1,
       1, 0, 0, 1, 0, 0, 0, 2, 1, 1, 1, 2, 2, 1, 1, 1, 1, 1, 0, 2, 0, 0,
       1, 1, 1, 1, 2, 1, 2, 2, 2, 1, 0, 2, 0, 2, 2, 1, 1, 0, 1, 1, 0, 1,
       0, 1, 1, 1, 1, 0, 1, 2, 2, 0, 1, 1, 2, 0, 1, 2, 1, 1, 1, 2, 0, 2,
       1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 0, 2, 0, 1, 1, 0, 2, 1, 1,
       2, 1, 0])
```

Using Confusion matrix heatmap to visualize the model accuracy

```
from sklearn.metrics import confusion_matrix
sns.heatmap(confusion_matrix(y_test, d_pred), annot=True, cmap='coolwarm', fmt='g')
plt.title('Confusion Matrix')
```

```
plt.xlabel('Actual')
plt.ylabel('Predicted')
plt.show()
```



The diagonal boxes show the count of true positive results, i.e correct predictions made by the model. The off-diagonal boxes show the count of false positive results, i.e incorrect predictions made by the model.

Distribution plot for predicted and actual values

```
ax = sns.distplot(y_test, hist=False, color="g", label="Actual Value")
sns.distplot(d_pred, hist=False, color="y", label="Fitted Values" , ax=ax)
plt.title('Actual vs Fitted Values for Sleep Disorder Prediction')
plt.xlabel('Sleep Disorder')
plt.ylabel('Proportion of People')
plt.show()
```

```
<ipython-input-35-b376c0a1bef5>:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `kdeplot` (an axes-level function for kernel density plots
```

The actual values are represented with red and the predicted ones with blue. As shown in the graph, the model's prediction are able to follow the curve of actual values but the predicted values are still different from actual ones. Therefore the model is not able to predict the values accurately.

Classification Report

```
Please adapt your code to use either `displot` (a figure-level function with
```

```
from sklearn.metrics import classification_report
print(classification_report(y_test, d_pred))
```

```
                precision    recall  f1-score   support

             0       0.74      0.83      0.78        24
             1       0.91      0.95      0.93        62
             2       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
```

The model gives pretty decent results with an accuracy of 87% and an average F1 score of 0.83. The model is able to predict the sleep disorder with a good accuracy.

Random Forest Classifier

```
from sklearn.ensemble import RandomForestClassifier
rfc = RandomForestClassifier(n_estimators=100, random_state=42)
```

```
rfc.fit(X_train, y_train)
```

```
RandomForestClassifier(random_state=42)
```

```
#Training accuracy
print("Training accuracy: ",rfc.score(X_train,y_train))
```

```
Training accuracy:  0.9348659003831418
```

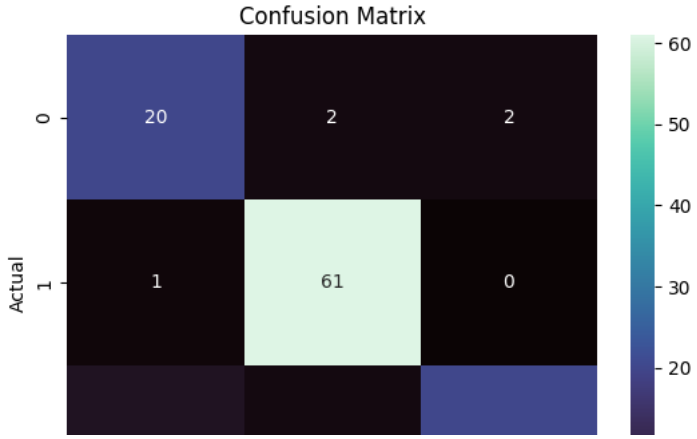
Random Forest Classifier Evaluation

```
rfc_pred = rfc.predict(X_test)
rfc_pred
```

```
array([1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 2, 1, 0, 1, 1, 1, 1,
       1, 0, 0, 1, 0, 0, 0, 2, 1, 1, 1, 2, 2, 1, 1, 1, 1, 1, 0, 2, 0, 0,
       1, 1, 1, 1, 2, 1, 2, 0, 2, 1, 0, 2, 0, 2, 2, 1, 1, 0, 1, 1, 0, 1,
       0, 1, 1, 1, 1, 0, 1, 2, 2, 0, 1, 1, 2, 0, 1, 2, 1, 1, 1, 2, 1, 2,
       1, 1, 2, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 1, 2, 0, 1, 2, 0, 2, 1, 1,
       2, 1, 0])
```

Using confusion matrix heatmap to visualize the model accuracy

```
#confusion matrix heatmap
sns.heatmap(confusion_matrix(y_test, rfc_pred), annot=True, cmap='mako')
plt.title('Confusion Matrix')
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.show()
```

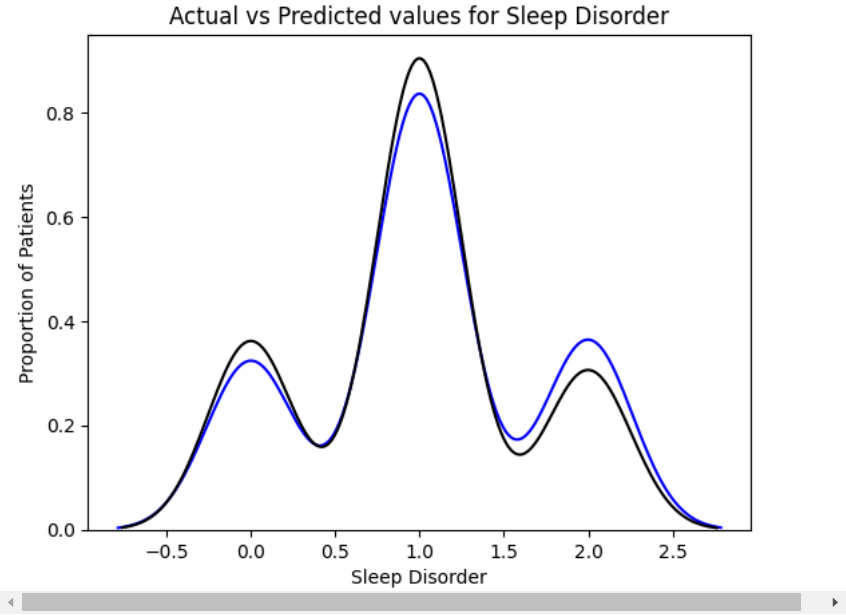


The Random Forest Classifier model has greater accuracy than the Decision Tree Classifier model. The diagonal boxes count for the True Positives i.e correct predictions, whereas the off-diagonal boxes show the count of false positive results, i.e incorrect predictions made by the model. Since the number of false positve value is less, it shows that the model is good at predicting the correct results.

Distribution plot for predicted and acutal values

```
ax = sns.distplot(y_test, hist=False, color="b", label="Actual Value")
sns.distplot(rfc_pred, hist=False, color="k", label="Predicted Values" , ax=ax)
plt.title('Actual vs Predicted values for Sleep Disorder')
plt.xlabel('Sleep Disorder')
plt.ylabel('Proportion of Patients')
plt.show()
```

```
<ipython-input-43-9a3f6bbc3928>:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `kdeplot` (an axes-level function for kernel density plots

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  ax = sns.distplot(y_test, hist=False, color="b", label="Actual Value")
<ipython-input-43-9a3f6bbc3928>:2: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `kdeplot` (an axes-level function for kernel density plots

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(rfc_pred, hist=False, color="k", label="Predicted Values" , ax=ax)
```



The Random forest classifier has improved accuracy as compared to the Decision Tree which is shown with the gap between the actual and predcited values which was wider incase of Descision Tree Classifier.

Classification Report

```
print(classification_report(y_test, rfc_pred))
```

```
                  precision    recall  f1-score   support

               0       0.77      0.83      0.80        24
               1       0.94      0.98      0.96        62
               2       0.91      0.74      0.82        27

        accuracy                           0.89       113
       macro avg       0.87      0.85      0.86       113
    weighted avg       0.90      0.89      0.89       113
```

The Random Forest Classifier model has an accuracy of 89% and an avergae F1 score of 0.86. From the metrics it is quite clear that the model is able to predict the sleep disorder quite effectively, with increased accuracy than Decision Tree Classifer.

Conclusion

From the exploratory data analysis, I have concluded that the sleep orders depends upon three main factors that are gender, occupation and BMI of the patient. The males have more instance of Insomia whereas femlaes have more instances of Sleep Apnea. In addition the that people with occupation such as nurses are more prone to sleep disorders. The BMI of the patient also plays a vital role in the prediction of sleep disorders. The patients who are either Obese or overweight are more prone to sleep disorders.

Coming to the classfication models, both the models performed pretty good, however the Random Forest Classifier have excellent results with 89% accuracy.

END