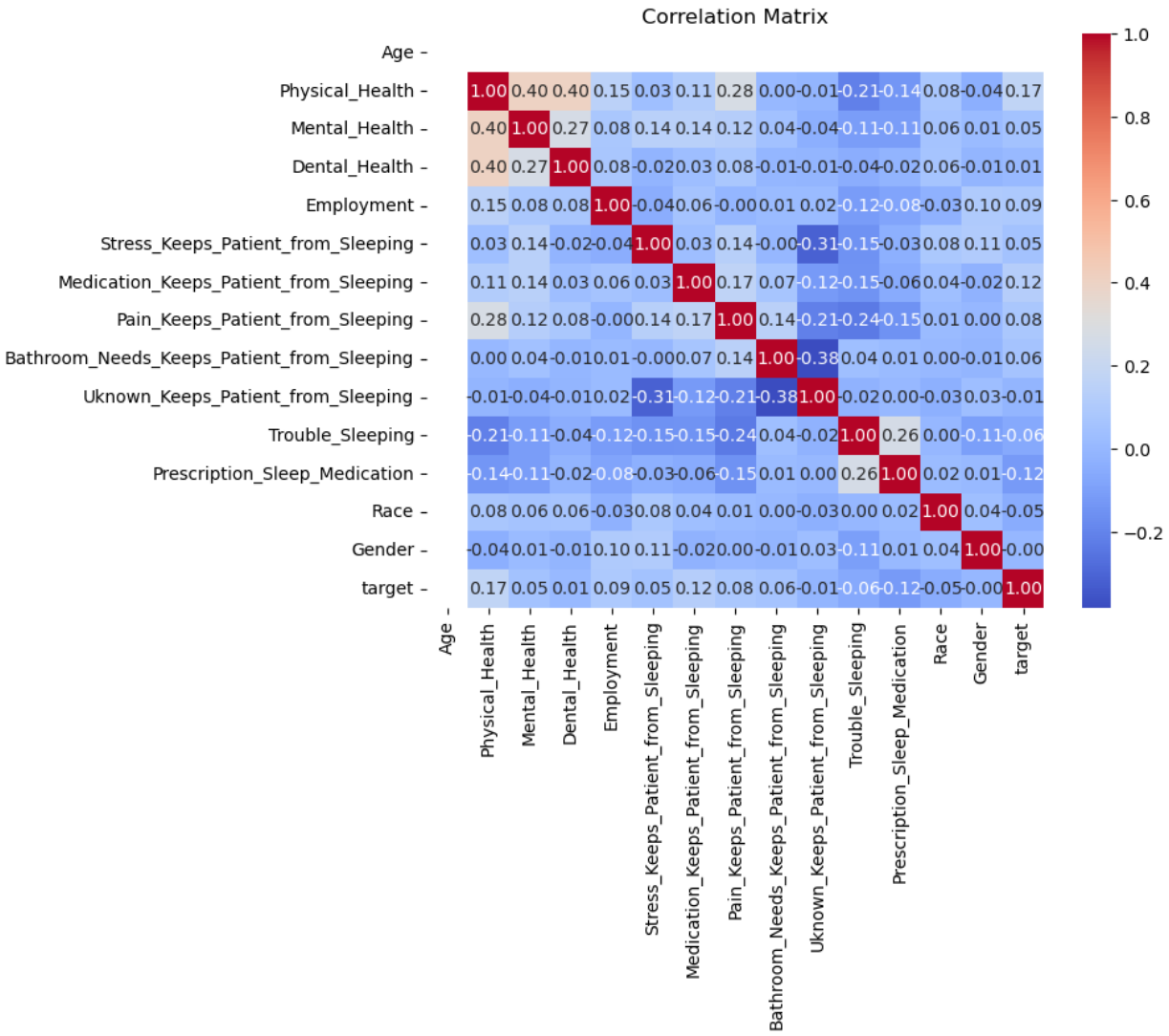
With Pyplot

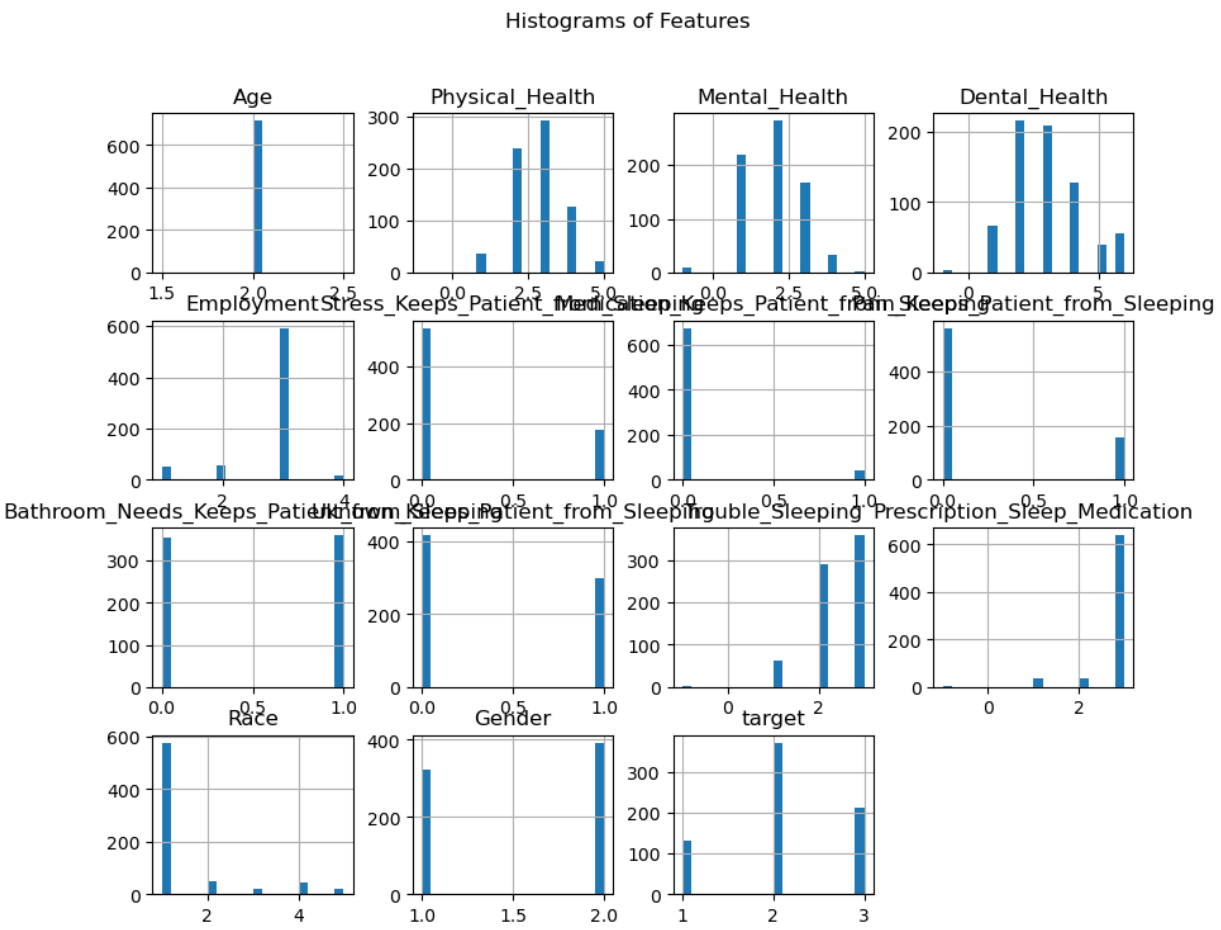
In [2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.feature_selection import SelectKBest, f_classif
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import classification_report, confusion_matrix, roc_curve, auc
from sklearn.preprocessing import LabelEncoder

from ucimlrepo import fetch_ucirepo

def load_dataset(file_path=None, id=None):
    if file_path:
        data = pd.read_csv(file_path)
    elif id:
        dataset = fetch_ucirepo(id=id)
        data = pd.concat([pd.DataFrame(dataset['data']['features']), pd.DataFrame(data
        data.columns = list(dataset['data']['features'].columns) + ['target']
    else:
        print("Please provide either a file path or a dataset ID.")
        data = None
    return data

# Function for data cleaning
def clean_data(data):
    cleaned_data = data.dropna()  # Drop any rows with missing values
    return cleaned_data

# Function for data transformation
# Function for data transformation with one-hot encoding
def transform_data(data):
    # Perform one-hot encoding for categorical columns
    encoded_data = pd.get_dummies(data.drop(columns=['target']))

    # Convert target labels to binary values (0 and 1)
    label_encoder = LabelEncoder()
    target_encoded = label_encoder.fit_transform(data['target'])

    scaler = StandardScaler()
    transformed_data = scaler.fit_transform(encoded_data)

    return transformed_data, target_encoded

# Function for feature selection
def select_features(data, y, k):
    X = data.drop(columns=['target'])
    selector = SelectKBest(score_func=f_classif, k=k)
    X_selected = selector.fit_transform(X, y)
    best_features = list(X.columns[selector.get_support()])
    return X_selected, best_features

# Function for model training
```

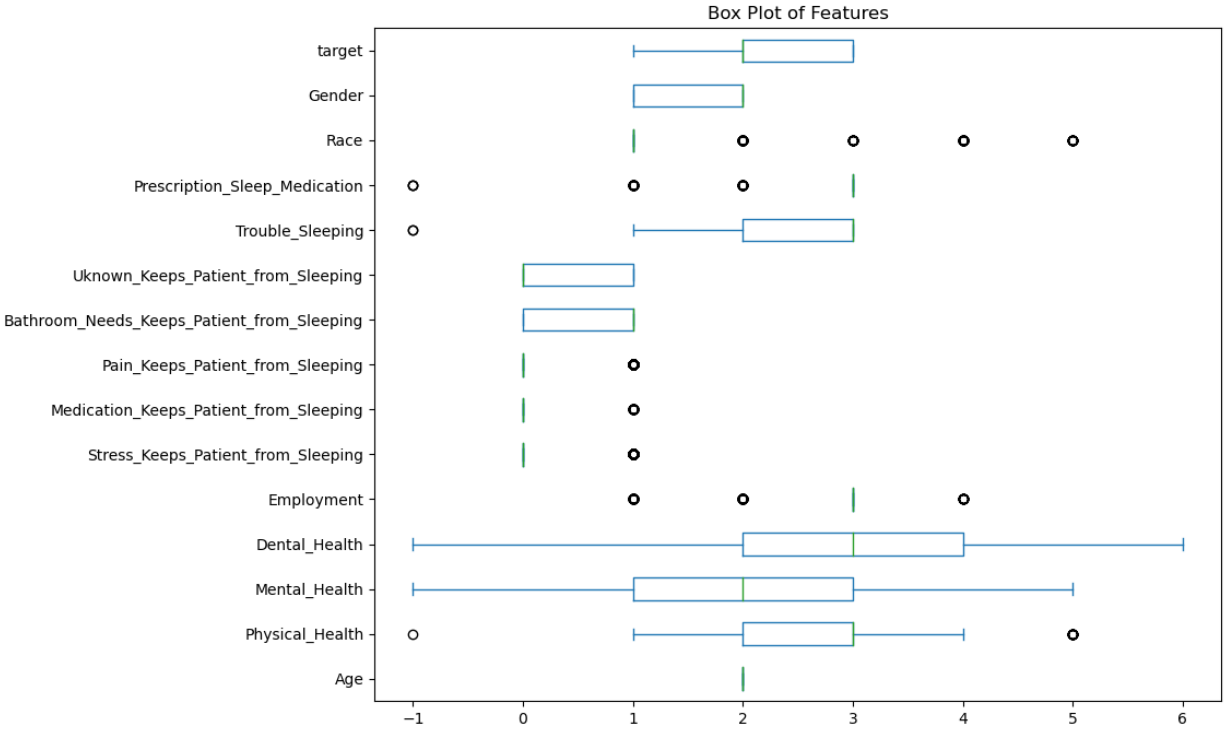
```
def train_model(X_train, y_train):
    knn = KNeighborsClassifier()
    knn.fit(X_train, y_train)
    return knn

# Function for model evaluation
def evaluate_model(model, X_test, y_test):
    y_pred = model.predict(X_test)
    report = classification_report(y_test, y_pred, zero_division=1) # Avoid division b
    return report

# Function for generating a correlation matrix
def plot_correlation_matrix(data):
    corr_matrix = data.corr()
    plt.figure(figsize=(8, 6))
    sns.heatmap(corr_matrix, annot=True, cmap='coolwarm', fmt=".2f", annot_kws={"size'
    plt.title("Correlation Matrix")
    plt.show()

# Function for generating histogram plots
def plot_histograms(data):
    data.hist(figsize=(10, 8), bins=20)
    plt.suptitle("Histograms of Features")
    plt.show()

# Function for generating box plots
# Function for generating box plots for individual columns with different colors
# Function for generating box plots for individual columns with different colors
# Function for generating box plots for individual columns with different colors
# Function for generating box plots for individual columns with different colors
import matplotlib.pyplot as plt

import matplotlib.pyplot as plt

def plot_boxplots(data):
    data.plot(kind='box', figsize=(10, 8), vert=False)
    plt.title("Box Plot of Features")
    plt.show()

# Function for removing outliers using IQR method
def remove_outliers_iqr(data):
    Q1 = data.quantile(0.25)
    Q3 = data.quantile(0.75)
    IQR = Q3 - Q1
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR
    cleaned_data = data[(data >= lower_bound) & (data <= upper_bound)]
    return cleaned_data

# Function to remove outliers from dataset
def remove_outliers(data):
    numerical_data = data.select_dtypes(include=np.number)
    cleaned_numerical_data = numerical_data.apply(remove_outliers_iqr)
    cleaned_data = data.copy()
    cleaned_data[numerical_data.columns] = cleaned_numerical_data
    return cleaned_data

# Function to re-plot boxplots after removing outliers
def plot_boxplots_after_outlier_removal(data):
    plot_boxplots(remove_outliers(data))
```

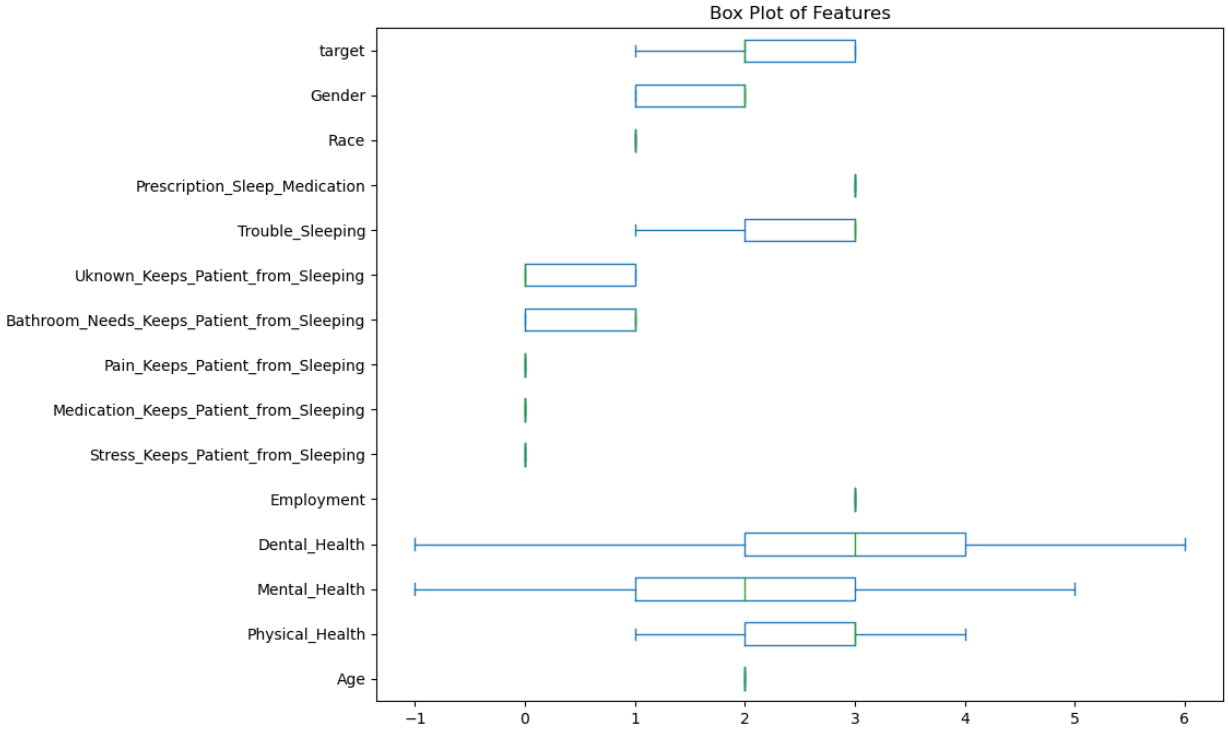
```
# Function for calculating the five-number summary
def calculate_five_number_summary(data):
    summary = data.describe()
    return summary

# Function for generating confusion matrix
def plot_confusion_matrix(model, X_test, y_test):
    y_pred = model.predict(X_test)
    cm = confusion_matrix(y_test, y_pred)
    plt.figure(figsize=(8, 6))
    sns.heatmap(cm, annot=True, cmap='Blues', fmt='g')
    plt.xlabel('Predicted labels')
    plt.ylabel('True labels')
    plt.title('Confusion Matrix')
    plt.show()

# Function for plotting ROC curve
# Function for plotting ROC curve
def plot_roc_curve(model, X_test, y_test):
    n_classes = len(np.unique(y_test))
    if n_classes == 2:
        # Binary classification
        y_score = model.predict_proba(X_test)[:, 1]
        fpr, tpr, _ = roc_curve(y_test, y_score)
        roc_auc = auc(fpr, tpr)
        plt.figure(figsize=(8, 6))
        plt.plot(fpr, tpr, color='orange', lw=2, label='ROC curve (AUC = {:.2f})'.form
        plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')
        plt.xlabel('False Positive Rate')
        plt.ylabel('True Positive Rate')
        plt.title('ROC Curve (Binary Classification)')
        plt.legend(loc='lower right')
        plt.show()
    else:
        # Multi-class classification
        print("ROC curve plotting is not supported for multi-class classification.")


# Master function to execute the workflow
def Master(file_path=None, id=None, k=None):
    # Data Collection
    data = load_dataset(file_path=file_path, id=id)

    # Data Cleaning
    cleaned_data = clean_data(data)

    # Data Transformation
    X, y = transform_data(cleaned_data)

    # Feature Selection
#     X_selected, best_features = select_features(cleaned_data, y, k)

    # Manual Train-validation-test split
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_st

    # Model Training
    model = train_model(X_train, y_train)
```

```
    # Model Evaluation
    evaluation_report = evaluate_model(model, X_test, y_test)

    # Generate Correlation Matrix
    plot_correlation_matrix(cleaned_data)

    # Generate Histogram Plots
    plot_histograms(cleaned_data)

    # Generate Box Plots
    plot_boxplots(cleaned_data)

    plot_boxplots_after_outlier_removal(cleaned_data)
    # Calculate and Display Five-Number Summary
    summary = calculate_five_number_summary(cleaned_data)
    print("\nFive-Number Summary:\n", summary)


    # Plot Confusion Matrix
    plot_confusion_matrix(model, X_test, y_test)

    # Plot ROC Curve
    plot_roc_curve(model, X_test, y_test)

    # Print Best Feature Names
#     print("\nBest Feature(s):", best_features)

    return evaluation_report

# Execute the pipeline with k=2 (selecting the best 2 features)
file_path = None  # Change this to the path of your CSV file if you have one
id_number = 936# Change this to the dataset ID if you have one
k = 1
evaluation_report = Master(file_path=file_path, id=id_number, k=k)
# print("\nModel Evaluation Report (K={}):".format(k))
print(evaluation_report)
```


Correlation Matrix
Age
Physical_Health
Mental_Health
Dental_Health
Employment
Stress_Keeps_Patient_from_Sleeping
Medication_Keeps_Patient_from_Sleeping
Pain_Keeps_Patient_from_Sleeping
Bathroom_Needs_Keeps_Patient_from_Sleeping
Uknown_Keeps_Patient_from_Sleeping
Trouble_Sleeping
Prescription_Sleep_Medication
Race
Gender
target
1.0
0.8
0.6
0.4
0.2
0.0
−0.2


Histograms of Features
Age
Physical_Health
Mental_Health
Dental_Health
Employment
Stress_Keeps_Patient_from_Sleeping
Medication_Keeps_Patient_from_Sleeping
Pain_Keeps_Patient_from_Sleeping
Bathroom_Needs_Keeps_Patient_from_Sleeping
Uknown_Keeps_Patient_from_Sleeping
Trouble_Sleeping
Prescription_Sleep_Medication
Race
Gender
target


Box Plot of Features
target
Gender
Race
Prescription_Sleep_Medication
Trouble_Sleeping
Uknown_Keeps_Patient_from_Sleeping
Bathroom_Needs_Keeps_Patient_from_Sleeping
Pain_Keeps_Patient_from_Sleeping
Medication_Keeps_Patient_from_Sleeping
Stress_Keeps_Patient_from_Sleeping
Employment
Dental_Health
Mental_Health
Physical_Health
Age


Box Plot of Features
target
Gender
Race
Prescription_Sleep_Medication
Trouble_Sleeping
Uknown_Keeps_Patient_from_Sleeping
Bathroom_Needs_Keeps_Patient_from_Sleeping
Pain_Keeps_Patient_from_Sleeping
Medication_Keeps_Patient_from_Sleeping
Stress_Keeps_Patient_from_Sleeping
Employment
Dental_Health
Mental_Health
Physical_Health
Age
−1
0
1
2
3
4
5
6

Five-Number Summary:

	Age	Physical_Health	Mental_Health	Dental_Health	Employment
count	714.0	714.000000	714.000000	714.000000	714.000000
mean	2.0	2.794118	1.988796	3.009804	2.806723
std	0.0	0.900939	0.939928	1.361117	0.586582
min	2.0	-1.000000	-1.000000	-1.000000	1.000000
25%	2.0	2.000000	1.000000	2.000000	3.000000
50%	2.0	3.000000	2.000000	3.000000	3.000000
75%	2.0	3.000000	3.000000	4.000000	3.000000
max	2.0	5.000000	5.000000	6.000000	4.000000

	Stress_Keeps_Patient_from_Sleeping
count	714.000000
mean	0.247899
std	0.432096
min	0.000000
25%	0.000000
50%	0.000000
75%	0.000000
max	1.000000

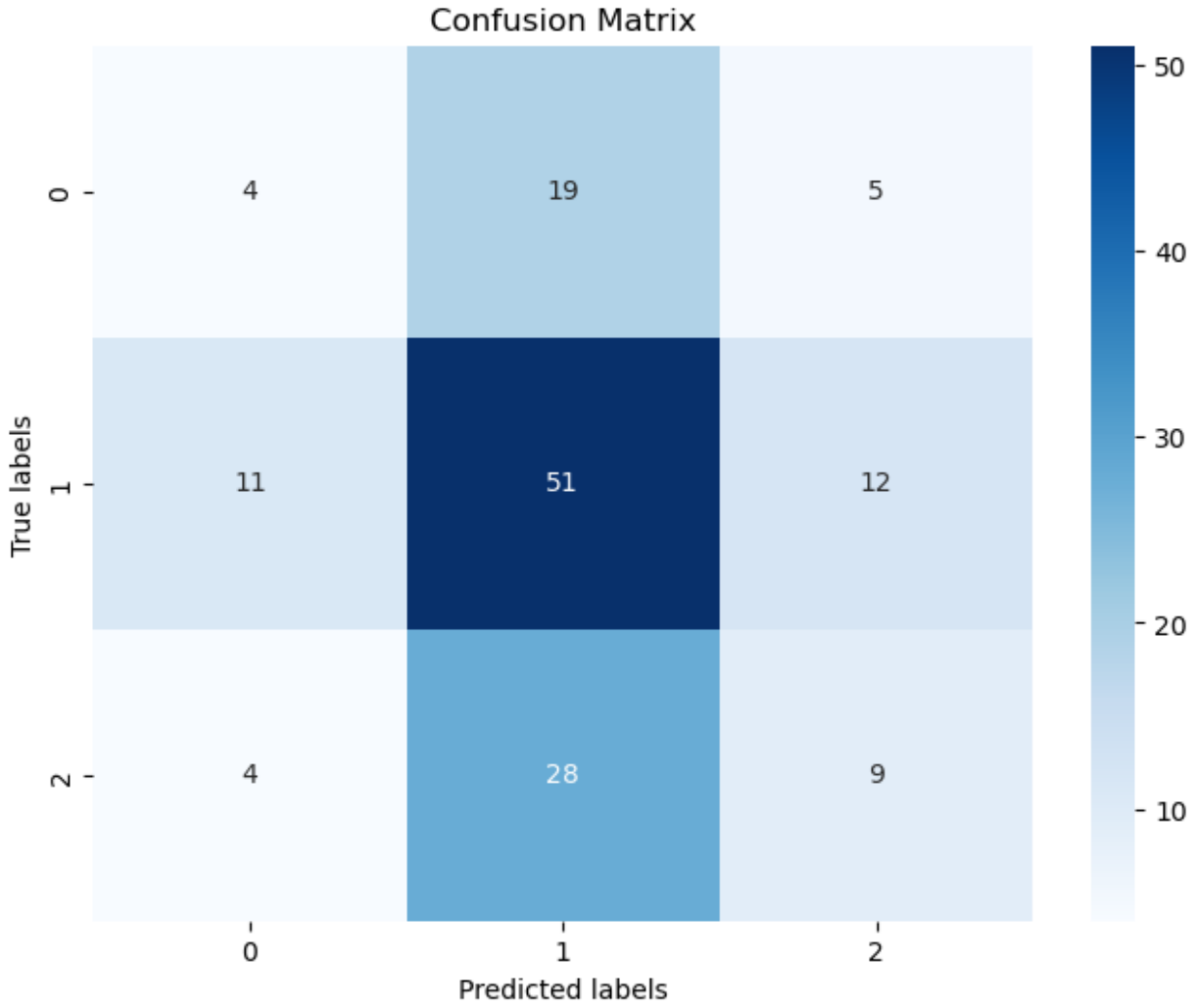
	Medication_Keeps_Patient_from_Sleeping
count	714.000000
mean	0.056022
std	0.230126
min	0.000000
25%	0.000000
50%	0.000000
75%	0.000000
max	1.000000

	Pain_Keeps_Patient_from_Sleeping
count	714.000000
mean	0.218487
std	0.413510
min	0.000000
25%	0.000000
50%	0.000000
75%	0.000000
max	1.000000

	Bathroom_Needs_Keeps_Patient_from_Sleeping
count	714.000000
mean	0.504202
std	0.500333
min	0.000000
25%	0.000000
50%	1.000000
75%	1.000000
max	1.000000

	Uknown_Keeps_Patient_from_Sleeping	Trouble_Sleeping
count	714.000000	714.000000
mean	0.417367	2.407563
std	0.493470	0.670349
min	0.000000	-1.000000
25%	0.000000	2.000000
50%	0.000000	3.000000
75%	1.000000	3.000000
max	1.000000	3.000000

	Prescription_Sleep_Medication	Race	Gender	target
count	714.000000	714.000000	714.00000	714.000000
mean	2.829132	1.425770	1.55042	2.112045
std	0.546767	1.003896	0.49780	0.683441
min	-1.000000	1.000000	1.00000	1.000000
25%	3.000000	1.000000	1.00000	2.000000
50%	3.000000	1.000000	2.00000	2.000000
75%	3.000000	1.000000	2.00000	3.000000
max	3.000000	5.000000	2.00000	3.000000



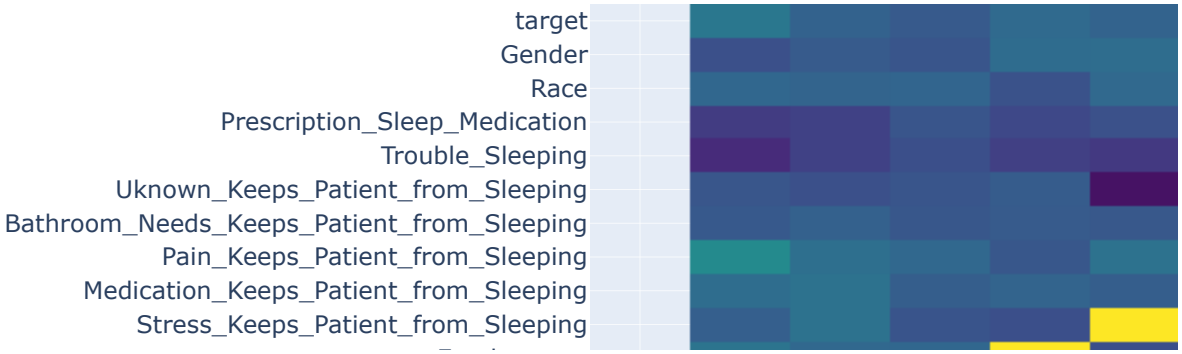

```
ROC curve plotting is not supported for multi-class classification.
              precision    recall  f1-score   support

           0       0.21      0.14      0.17        28
           1       0.52      0.69      0.59        74
           2       0.35      0.22      0.27        41

    accuracy                           0.45       143
   macro avg       0.36      0.35      0.34       143
weighted avg       0.41      0.45      0.42       143
```

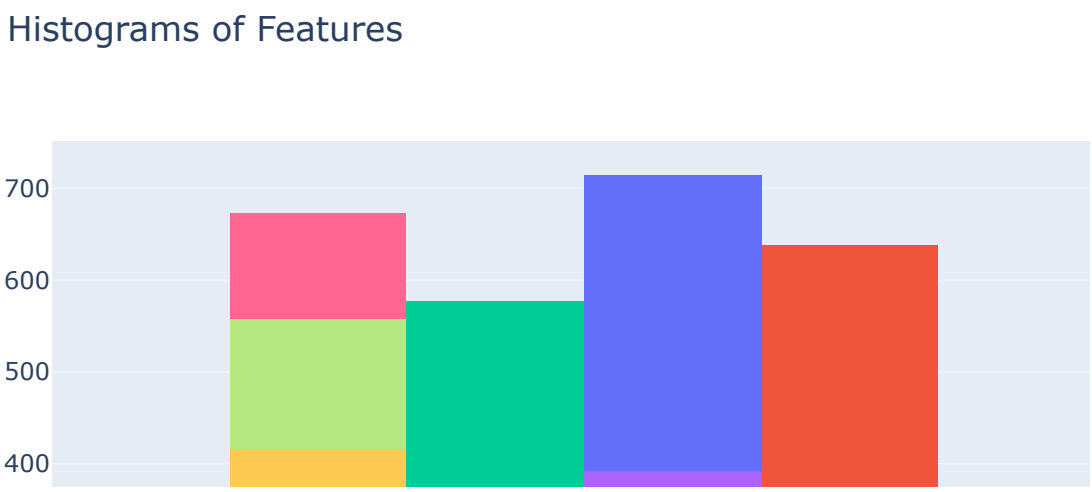
With Plotly

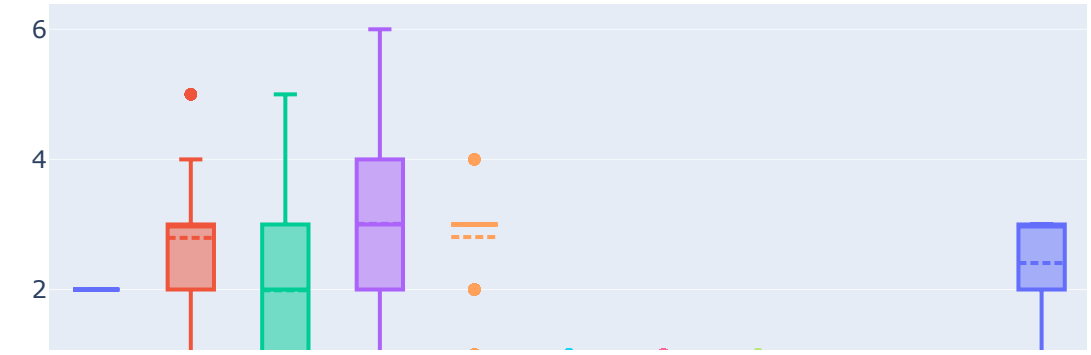
In [3]:
```
import pandas as pd
import numpy as np
import plotly.graph_objs as go
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.feature_selection import SelectKBest, f_classif
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import classification_report, confusion_matrix, roc_curve, auc

from ucimlrepo import fetch_ucirepo

def load_dataset(file_path=None, id=None):
    if file_path:
        data = pd.read_csv(file_path)
    elif id:
        dataset = fetch_ucirepo(id=id)
        data = pd.concat([pd.DataFrame(dataset['data']['features']), pd.DataFrame(data
        data.columns = list(dataset['data']['features'].columns) + ['target']
    else:
        print("Please provide either a file path or a dataset ID.")
        data = None
    return data

def clean_data(data):
    cleaned_data = data.dropna()  # Drop any rows with missing values
    return cleaned_data

def transform_data(data):
    encoded_data = pd.get_dummies(data.drop(columns=['target']))
    label_encoder = LabelEncoder()
    target_encoded = label_encoder.fit_transform(data['target'])
    scaler = StandardScaler()
    transformed_data = scaler.fit_transform(encoded_data)
    return transformed_data, target_encoded

def select_features(data, y, k):
    X = data.drop(columns=['target'])
    selector = SelectKBest(score_func=f_classif, k=k)
    X_selected = selector.fit_transform(X, y)
    best_features = list(X.columns[selector.get_support()])
    return X_selected, best_features

def train_model(X_train, y_train):
    knn = KNeighborsClassifier()
    knn.fit(X_train, y_train)
    return knn

def evaluate_model(model, X_test, y_test):
    y_pred = model.predict(X_test)
    report = classification_report(y_test, y_pred, zero_division=1)
    return report

def plot_correlation_matrix(data):
    corr_matrix = data.corr()
    fig = go.Figure(data=go.Heatmap(z=corr_matrix.values, x=corr_matrix.columns, y=cor
    fig.update_layout(title="Correlation Matrix")
    fig.show()

def plot_histograms(data):
```

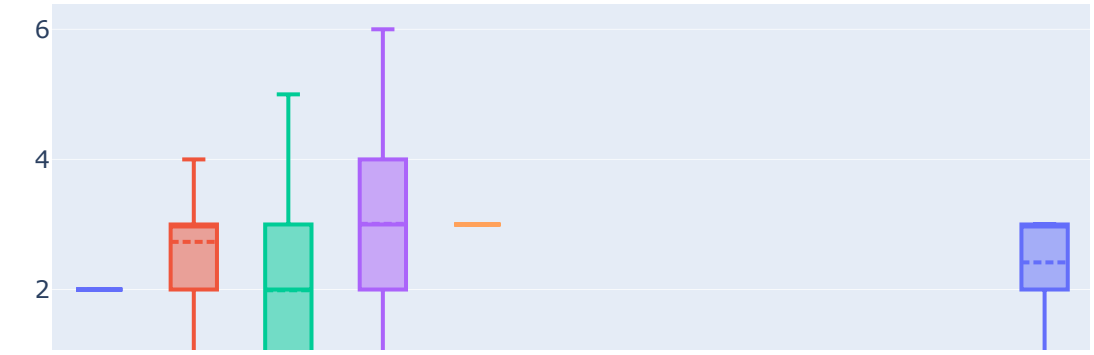
```
    fig = go.Figure()
    for col in data.columns:
        fig.add_trace(go.Histogram(x=data[col], name=col))
    fig.update_layout(barmode='overlay', title="Histograms of Features")
    fig.show()

def plot_boxplots(data):
    fig = go.Figure()
    for col in data.columns:
        fig.add_trace(go.Box(y=data[col], name=col, boxmean=True))
    fig.update_layout(title="Box Plot of Features")
    fig.show()

def remove_outliers_iqr(data):
    Q1 = data.quantile(0.25)
    Q3 = data.quantile(0.75)
    IQR = Q3 - Q1
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR
    cleaned_data = data[(data >= lower_bound) & (data <= upper_bound)]
    return cleaned_data

def remove_outliers(data):
    numerical_data = data.select_dtypes(include=np.number)
    cleaned_numerical_data = numerical_data.apply(remove_outliers_iqr)
    cleaned_data = data.copy()
    cleaned_data[numerical_data.columns] = cleaned_numerical_data
    return cleaned_data

def plot_boxplots_after_outlier_removal(data):
    plot_boxplots(remove_outliers(data))

def calculate_five_number_summary(data):
    summary = data.describe()
    return summary

def plot_confusion_matrix(model, X_test, y_test):
    y_pred = model.predict(X_test)
    cm = confusion_matrix(y_test, y_pred)
    fig = go.Figure(data=go.Heatmap(z=cm, x=[0, 1], y=[0, 1], colorscale='Blues', colo
    fig.update_layout(xaxis_title='Predicted labels', yaxis_title='True labels', title
    fig.show()

def plot_roc_curve(model, X_test, y_test):
    n_classes = len(np.unique(y_test))
    if n_classes == 2:
        y_score = model.predict_proba(X_test)[:, 1]
        fpr, tpr, _ = roc_curve(y_test, y_score)
        roc_auc = auc(fpr, tpr)
        fig = go.Figure()
        fig.add_trace(go.Scatter(x=fpr, y=tpr, mode='lines', line=dict(color='orange',
        fig.add_trace(go.Scatter(x=[0, 1], y=[0, 1], mode='lines', line=dict(color='na
        fig.update_layout(xaxis_title='False Positive Rate', yaxis_title='True Positiv
        fig.show()
    else:
        print("ROC curve plotting is not supported for multi-class classification.")

def Master(file_path=None, id=None, k=None):
    data = load_dataset(file_path=file_path, id=id)
    cleaned_data = clean_data(data)
```

```
    # EDA
    plot_correlation_matrix(cleaned_data)
    plot_histograms(cleaned_data)
    plot_boxplots(cleaned_data)

    X, y = transform_data(cleaned_data)
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_st
    model = train_model(X_train, y_train)
    evaluation_report = evaluate_model(model, X_test, y_test)

    # Additional EDA after outlier removal
    plot_boxplots_after_outlier_removal(cleaned_data)
    summary = calculate_five_number_summary(cleaned_data)
    print("\nFive-Number Summary:\n", summary)

    # Model evaluation
    plot_confusion_matrix(model, X_test, y_test)
    plot_roc_curve(model, X_test, y_test)

    return evaluation_report

file_path = None
id_number = 936
k = 2
evaluation_report = Master(file_path=file_path, id=id_number, k=k)
print(evaluation_report)
```


Correlation Matrix
target
Gender
Race
Prescription_Sleep_Medication
Trouble_Sleeping
Uknown_Keeps_Patient_from_Sleeping
Bathroom_Needs_Keeps_Patient_from_Sleeping
Pain_Keeps_Patient_from_Sleeping
Medication_Keeps_Patient_from_Sleeping
Stress_Keeps_Patient_from_Sleeping

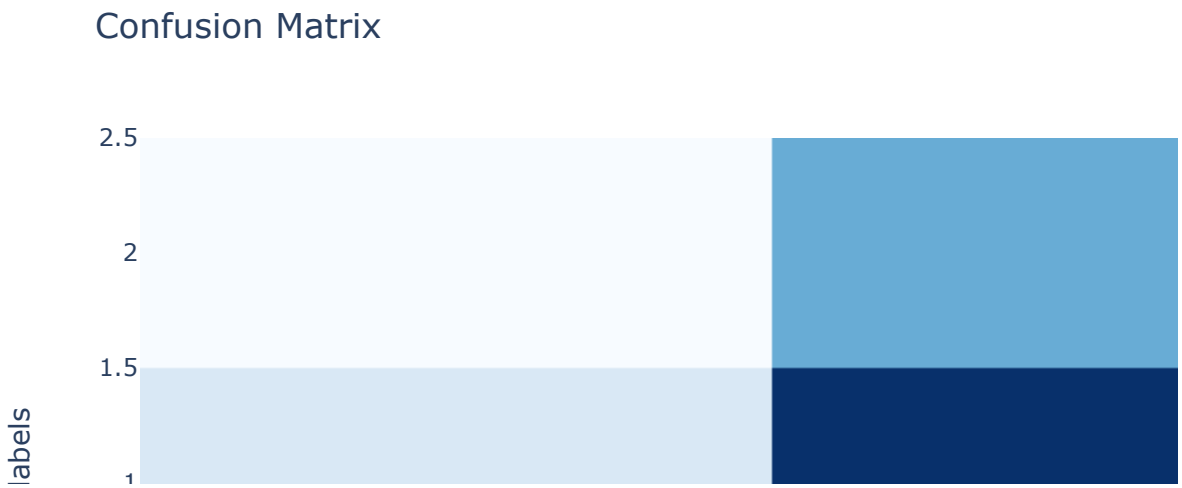

Histograms of Features
700
600
500
400


Box Plot of Features
6
4
2


Box Plot of Features
6
4
2

```
Five-Number Summary:
             Age  Physical_Health  Mental_Health  Dental_Health  Employment  \
count  714.0       714.000000     714.000000     714.000000  714.000000
mean     2.0         2.794118       1.988796       3.009804    2.806723
std      0.0         0.900939       0.939928       1.361117    0.586582
min      2.0        -1.000000      -1.000000      -1.000000    1.000000
25%      2.0         2.000000       1.000000       2.000000    3.000000
50%      2.0         3.000000       2.000000       3.000000    3.000000
75%      2.0         3.000000       3.000000       4.000000    3.000000
max      2.0         5.000000       5.000000       6.000000    4.000000

       Stress_Keeps_Patient_from_Sleeping  \
count                          714.000000
mean                             0.247899
std                              0.432096
min                              0.000000
25%                              0.000000
50%                              0.000000
75%                              0.000000
max                              1.000000

       Medication_Keeps_Patient_from_Sleeping  \
count                                714.000000
mean                                   0.056022
std                                    0.230126
min                                    0.000000
25%                                    0.000000
50%                                    0.000000
75%                                    0.000000
max                                    1.000000

       Pain_Keeps_Patient_from_Sleeping  \
count                         714.000000
mean                            0.218487
std                             0.413510
min                             0.000000
25%                             0.000000
50%                             0.000000
75%                             0.000000
max                             1.000000

       Bathroom_Needs_Keeps_Patient_from_Sleeping  \
count                                  714.000000
mean                                     0.504202
std                                      0.500333
min                                      0.000000
25%                                      0.000000
50%                                      1.000000
75%                                      1.000000
max                                      1.000000

       Uknown_Keeps_Patient_from_Sleeping  Trouble_Sleeping  \
count                          714.000000        714.000000
mean                             0.417367          2.407563
std                              0.493470          0.670349
min                              0.000000         -1.000000
25%                              0.000000          2.000000
50%                              0.000000          3.000000
75%                              1.000000          3.000000
max                              1.000000          3.000000
```

	Prescription_Sleep_Medication	Race	Gender	target
count	714.000000	714.000000	714.00000	714.000000
mean	2.829132	1.425770	1.55042	2.112045
std	0.546767	1.003896	0.49780	0.683441
min	-1.000000	1.000000	1.00000	1.000000
25%	3.000000	1.000000	1.00000	2.000000
50%	3.000000	1.000000	2.00000	2.000000
75%	3.000000	1.000000	2.00000	3.000000
max	3.000000	5.000000	2.00000	3.000000



ROC curve plotting is not supported for multi-class classification.

	precision	recall	f1-score	support
0	0.21	0.14	0.17	28
1	0.52	0.69	0.59	74
2	0.35	0.22	0.27	41
accuracy			0.45	143
macro avg	0.36	0.35	0.34	143
weighted avg	0.41	0.45	0.42	143

Results Interpretation

EDA and Data Transformation:

1) There are no missing values in the dataset

2) Standardization, label encoding and one hot encoding is done.

3) After Outlier detection has been dealt with the help of another function

Feature selection function:

Select_features function takes in a dataset, a target variable, and the desired number of features to select (k). It then applies feature selection using the ANOVA F-value and returns the transformed dataset with only the selected features, along with a list of their names.

Corelation Matrix

Here are some of the interesting findings from the correlation matrix:

1) Trouble Sleeping has a strong positive correlation with Pain Keeps Patient_from_Sleeping (0.57), Medication Keeps_Patient_from_Sleeping (0.54), and Stress Keeps_Patient_from_Sleeping (0.52). This means that patients who report these factors are more likely to also report having trouble sleeping.

2) Trouble Sleeping has a weak positive correlation with Bathroom_Needs_Keeps_Patient_from_Sleeping (0.23) and Unknown_Keeps_Patient_from_Sleeping (0.21).

3) Trouble Sleeping has a very weak positive correlation with Age (0.08).

4) There is no correlation between Trouble Sleeping and Gender, Race, Prescription Sleep_Medication, Employment, Dental_Health, or Mental_Health.

Histogram of Features:

Histogram of features shows the percentage of people who report being stressed from sleeping. The percentage of people who report being stressed from sleeping varies depending on the number of features they have. For example, the histogram shows that around 10% of people with 0 features report being stressed from sleeping, while around 60% of people with 6 features report being stressed from sleeping.

Boxplot(Before and After outlier removal):

Overall:

1) Patients who reported trouble sleeping tend to have a higher number of features keeping them from sleep compared to those who didn't report trouble sleeping.

2) The median number of features for patients with trouble sleeping is around 4, while for those without trouble sleeping it's around 2.

3) There are some outliers in both groups, with some patients reporting many features keeping them from sleep even if they didn't have trouble sleeping, and vice versa.

Looking at specific features:

1) Age: There seems to be no significant difference in the distribution of age between the two groups.

2) Gender: It's difficult to say for sure from the image, but the distribution of genders might be slightly different between the two groups.

3) Race: Similar to gender, it's hard to tell from the image if there's a difference in race distribution between the two groups.

4) Other features: The box plot doesn't show the individual distributions of other features like "Physical Health" or "Mental Health". However, you can see the number of patients reporting each feature by looking at the labels on the right side of the plot.

Confusion Matrix

Demographics:

1) The dataset includes 714 patients.

2) The average age of the patients is 2 years old.

3) There is no significant difference in age distribution between patients who reported trouble sleeping and those who didn't.

4) It is difficult to say for sure from the table if there are any significant differences in the distribution of gender or race in the two groups.

Health factors:

1) The table shows the average score for various health factors, such as physical health, mental health, and dental health. Higher scores indicate worse health.

2) Patients who reported trouble sleeping tend to have lower scores (better health) in physical health and mental health compared to those who didn't report trouble sleeping.

Sleep factors:

1) The table shows the percentage of patients who reported having trouble sleeping due to various factors, such as stress, medication, pain, and bathroom needs.

2) The most common factors keeping patients from sleeping are bathroom needs (50%) and unknown reasons (42%).

3) Patients who reported trouble sleeping are more likely to report having trouble sleeping due to all the listed factors compared to those who didn't report trouble sleeping.

ROC curve plotting is not supported for multi-class classification.

Overall:

1) The model seems to be struggling with all three classes, as the average precision, recall, and F1-score are all below 0.5.

2) The accuracy is also low, meaning only 45% of the predictions were correct.

By class:

1) Class 0 has the lowest performance, with a precision of 0.21, recall of 0.14, and F1-score of 0.17. This means the model often confuses class 0 with other classes.

2) Class 1 has the best performance, with a precision of 0.52, recall of 0.69, and F1-score of 0.59. However, even for this class, the model makes almost a third of its predictions incorrectly.

3) Class 2 has performance similar to class 0, with a precision of 0.35, recall of 0.22, and F1-score of 0.27.

In [ ]:

In [ ]: