Project Title	**OCD Patient Dataset: Demographics & Clinical Data** (regulatory affairs)
Tools	Python, ML, SQL, Excel
Domain	Data Analyst & Data scientist
Project Difficulties level	intermediate

Dataset : Dataset is available in the given link. You can download it at your convenience.

Click here to download data set

About Dataset

The "OCD Patient Dataset: Demographics & Clinical Data" is a comprehensive collection of information pertaining to 1500 individuals diagnosed with Obsessive-Compulsive Disorder (OCD). This dataset encompasses a wide range of parameters, providing a detailed insight into the demographic and clinical profiles of these individuals.

Included in this dataset are key demographic details such as age, gender, ethnicity, marital status, and education level, offering a comprehensive overview

of the sample population. Additionally, clinical information like the date of OCD diagnosis, duration of symptoms, and any previous psychiatric diagnoses are recorded, providing context to the patients' journeys.

The dataset also delves into the specific nature of OCD symptoms, categorizing them into obsession and compulsion types. Severity of these symptoms is assessed using the Yale-Brown Obsessive-Compulsive Scale (Y-BOCS) scores for both obsessions and compulsions. Furthermore, it documents any co-occurring mental health conditions, including depression and anxiety diagnoses.

Notably, the dataset outlines the medications prescribed to patients, offering valuable insights into the treatment approaches employed. It also records whether there is a family history of OCD, shedding light on potential genetic or environmental factors.

Overall, this dataset serves as a valuable resource for researchers, clinicians, and mental health professionals seeking to gain a deeper understanding of OCD and its manifestations within a diverse patient population.

NOTE :

1. this project is only for your guidance, not exactly the same you have to create. Here I am trying to show the way or idea of what steps you can follow and how your projects look. Some projects are very advanced (because it will be made with the help of flask, nlp, advance ai, advance DL and some advanced things) which you can not understand .

2. You can make or analyze your project with yourself, with your idea, make it more creative from where we can get some information and understand about our business. make sure what overall things you have created all things you understand very well.

Example: You can get the basic idea how you can create a project from here

Project Title:

OCD Patient Dataset: Demographics & Clinical Data Analysis

1. Objective

The goal of this project is to perform an exploratory data analysis (EDA) on a dataset containing demographic and clinical data of OCD patients. The analysis will focus on understanding the relationships between various demographic factors and clinical outcomes.

2. Dataset Overview

The dataset includes the following columns:

- `Patient ID`: Unique identifier for each patient.
- `Age`: Age of the patient.
- `Gender`: Gender of the patient.
- `Ethnicity`: Ethnicity of the patient.
- `Marital Status`: Marital status of the patient.
- `Education Level`: Level of education attained by the patient.
- `OCD Diagnosis Date`: Date when OCD was diagnosed.
- `Duration of Symptoms (months)`: Duration for which the patient has been experiencing symptoms.
- `Previous Diagnoses`: Any previous diagnoses before OCD.
- `Family History of OCD`: Whether the patient has a family history of OCD.

- `Obsession Type`: Type of obsessions experienced by the patient.
- `Compulsion Type`: Type of compulsions experienced by the patient.
- `Y-BOCS Score (Obsessions)`: Y-BOCS score related to obsessions.
- `Y-BOCS Score (Compulsions)`: Y-BOCS score related to compulsions.
- `Depression Diagnosis`: Whether the patient has been diagnosed with depression.
- `Anxiety Diagnosis`: Whether the patient has been diagnosed with anxiety.
- `Medications`: Medications the patient is currently taking.

3. Step-by-Step Guide

Step 1: Importing Libraries and Dataset

Start by importing the necessary Python libraries and loading the dataset.

```
import pandas as pd

import numpy as np

import seaborn as sns

import matplotlib.pyplot as plt


# Load the dataset

df = pd.read_csv('ocd_patient_dataset.csv')
```

Step 2: Initial Data Exploration

Explore the dataset to understand its structure, check for missing values, and get an overview of the data.

```
# Display the first few rows of the dataset
print(df.head())

# Get a summary of the dataset
print(df.info())

# Check for missing values
print(df.isnull().sum())
```

Step 3: Descriptive Statistics

Calculate basic statistics to understand the distribution of numerical columns.

```
# Summary statistics for numerical columns
print(df.describe())
```

```
# Summary statistics for categorical columns
print(df.describe(include=['O']))
```

Step 4: Visualizing Demographic Data

Create visualizations to explore the demographic data (`Age`, `Gender`, `Ethnicity`, etc.).

```
# Age distribution
sns.histplot(df['Age'], bins=20, kde=True)
plt.title('Age Distribution of Patients')
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.show()

# Gender distribution
sns.countplot(x='Gender', data=df)
plt.title('Gender Distribution')
```

```
plt.xlabel('Gender')

plt.ylabel('Count')

plt.show()



# Ethnicity distribution

sns.countplot(y='Ethnicity', data=df)

plt.title('Ethnicity Distribution')

plt.xlabel('Count')

plt.ylabel('Ethnicity')

plt.show()
```

Step 5: Clinical Data Analysis

Analyze the clinical data (`Duration of Symptoms`, `Y-BOCS Scores`, etc.) to uncover patterns and insights.

```
# Distribution of symptom duration

sns.histplot(df['Duration of Symptoms (months)'], bins=20, kde=True)
```

```
plt.title('Distribution of Symptom Duration')

plt.xlabel('Duration (months)')

plt.ylabel('Frequency')

plt.show()


# Boxplot of Y-BOCS Scores by Gender

sns.boxplot(x='Gender', y='Y-BOCS Score (Obsessions)', data=df)

plt.title('Y-BOCS Obsession Scores by Gender')

plt.xlabel('Gender')

plt.ylabel('Y-BOCS Score (Obsessions)')

plt.show()


# Relationship between Obsession and Compulsion Y-BOCS Scores

sns.scatterplot(x='Y-BOCS Score (Obsessions)', y='Y-BOCS Score (Compulsions)', hue='Gender', data=df)

plt.title('Relationship between Y-BOCS Scores (Obsessions vs Compulsions)')

plt.xlabel('Y-BOCS Score (Obsessions)')
```

```
plt.ylabel('Y-BOCS Score (Compulsions)')

plt.show()
```

Step 6: Correlation Analysis

Examine the correlation between numerical variables, such as `Age`, `Duration of Symptoms`, `Y-BOCS Scores`, etc.

```
# Correlation matrix

corr_matrix = df[['Age', 'Duration of Symptoms (months)', 'Y-BOCS Score (Obsessions)', 'Y-BOCS Score (Compulsions)']].corr()

sns.heatmap(corr_matrix, annot=True, cmap='coolwarm')

plt.title('Correlation Matrix')

plt.show()
```

Step 7: Key Insights and Reporting

Based on the analysis, identify key insights and patterns in the data. For example:

- Are there differences in OCD severity based on gender or age?
- Is there a correlation between the duration of symptoms and Y-BOCS scores?
- What are the common medications used, and how do they relate to the severity

of OCD symptoms?

4. Output and Interpretation

The visualizations and statistical outputs will help in interpreting the dataset. Discuss the key findings and their implications for understanding OCD in patients.

This structured approach, with corresponding code and output, provides a comprehensive EDA of the OCD patient dataset.

Example: You can get the basic idea how you can create a project from here

Sample code and output

Import Libraries

In [1]:

```
from scipy import stats

import pandas as pd

import numpy as np

import base64,os,random,gc

import seaborn as sns

import matplotlib.pyplot as plt

import missingno as msno
```

```python
import matplotlib.pyplot as plotter
import matplotlib.pyplot as plt
import plotly.express as px

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
import optuna
import xgboost as xgb
from xgboost import XGBClassifier
import catboost
from catboost import CatBoostClassifier
import lightgbm as lgbm
from lightgbm import LGBMClassifier
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import StratifiedKFold
from sklearn.base import BaseEstimator, TransformerMixin,
```

```
ClassifierMixin, clone

from sklearn.model_selection import KFold

from scipy import stats

from scipy.stats import norm, skew

from sklearn.model_selection import train_test_split

from sklearn.metrics import confusion_matrix, accuracy_score

from sklearn.feature_selection import SelectFromModel



from sklearn import datasets

optuna.logging.set_verbosity(optuna.logging.WARNING)

from lightgbm import *

pd.set_option("display.max_columns", None)



from sklearn.model_selection import train_test_split

from sklearn.ensemble import RandomForestClassifier

import eli5

from eli5.sklearn import PermutationImportance
```

```python
import warnings

warnings.filterwarnings('ignore')
```

Read Dataset

In [2]:

```python
train =
pd.read_csv('/kaggle/input/ocd-patient-dataset-demographics-and
-clinical-data/ocd_patient_dataset.csv')

display(train.head())
```

	Patient ID	Age	Gender	Ethnicity	Marital Status	Education Level	OCD Diagnosis Date	Duration of Symptoms (months)	Previous Diagnoses	Family History of O	Obsession Type	Compulsion Type	Y-BOCS Score (Obsessions)	Y-BOCS Score (Compulsions)	Depression Diagnosis	Anxiety Diagnosis	Medications

										CD							
0	1018	32	Female	African	Single	Some College	2016-07-15	203	MDD	No	Harm-related	Checking	17	10	Yes	Yes	SNRI
1	2406	69	Male	African	Divorced	Some College	2017-04-28	180	NaN	Yes	Harm-related	Washing	21	25	Yes	Yes	SSRI
2	1188	57	Male	Hispanic	Divorced	College Degree	2018-02-02	173	MDD	No	Contamination	Checking	3	4	No	No	Benzodiazepine
3	6200	27	Female	Hispanic	Married	College Degree	2014-08-25	126	PTSD	Yes	Symmetry	Washing	14	28	Yes	Yes	SSRI

4	5824	56	Female	Hispanic	Married	HighSchool	2022-02-20	168	PTSD	Yes	Hoarding	Ordering	39	18	No	No	NaN

EDA

In [3]:

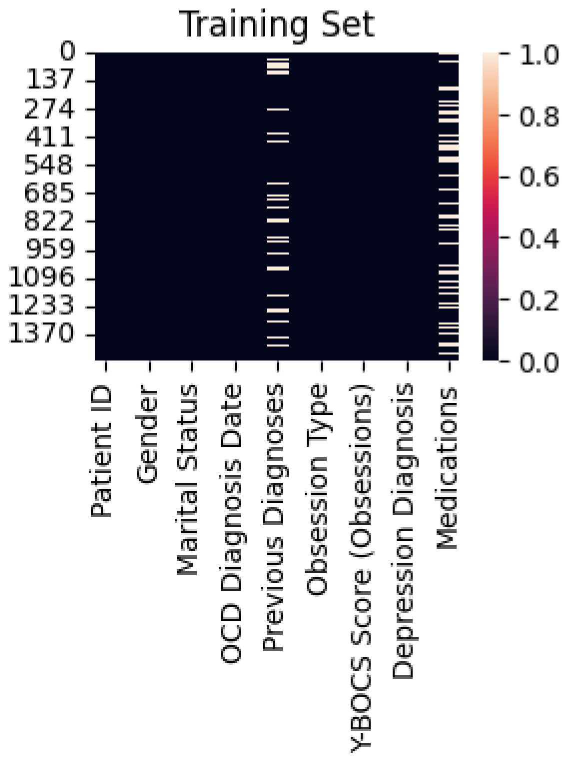
```
print('train')
display(train.isnull().sum())

plt.figure(figsize = (10, 2))

plt.subplot(1, 3, 1)
plt.title("Training Set")
sns.heatmap(train.isnull())
```

```
train
```

```
Patient ID                        0
Age                               0
Gender                            0
Ethnicity                         0
Marital Status                    0
Education Level                   0
OCD Diagnosis Date                0
Duration of Symptoms (months)     0
Previous Diagnoses              248
Family History of OCD             0
Obsession Type                    0
Compulsion Type                   0
Y-BOCS Score (Obsessions)         0
Y-BOCS Score (Compulsions)        0
Depression Diagnosis              0
Anxiety Diagnosis                 0
Medications                     386
dtype: int64
```

Out[3]:

<Axes: title={'center': 'Training Set'}>

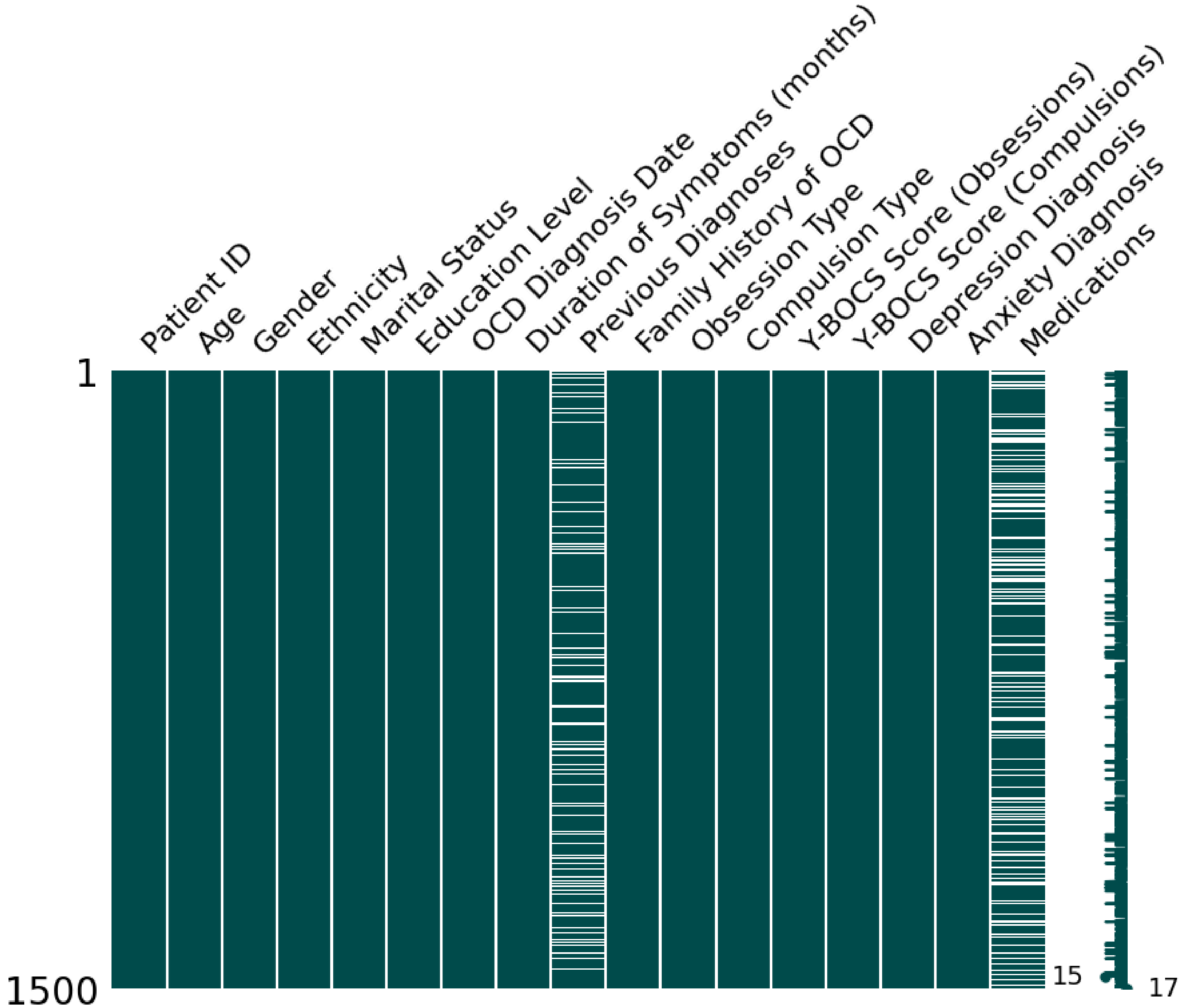


In [4]:

```
msno.matrix(df=train, figsize=(10,6), color=(0,.3,.3))
```

Out[4]:

<Axes: >



In [5]:

```
#train['Medications'] = train['Medications'].fillna('Unknown')

#train
```

In [6]:

```
print('train')
display(train.info())
```

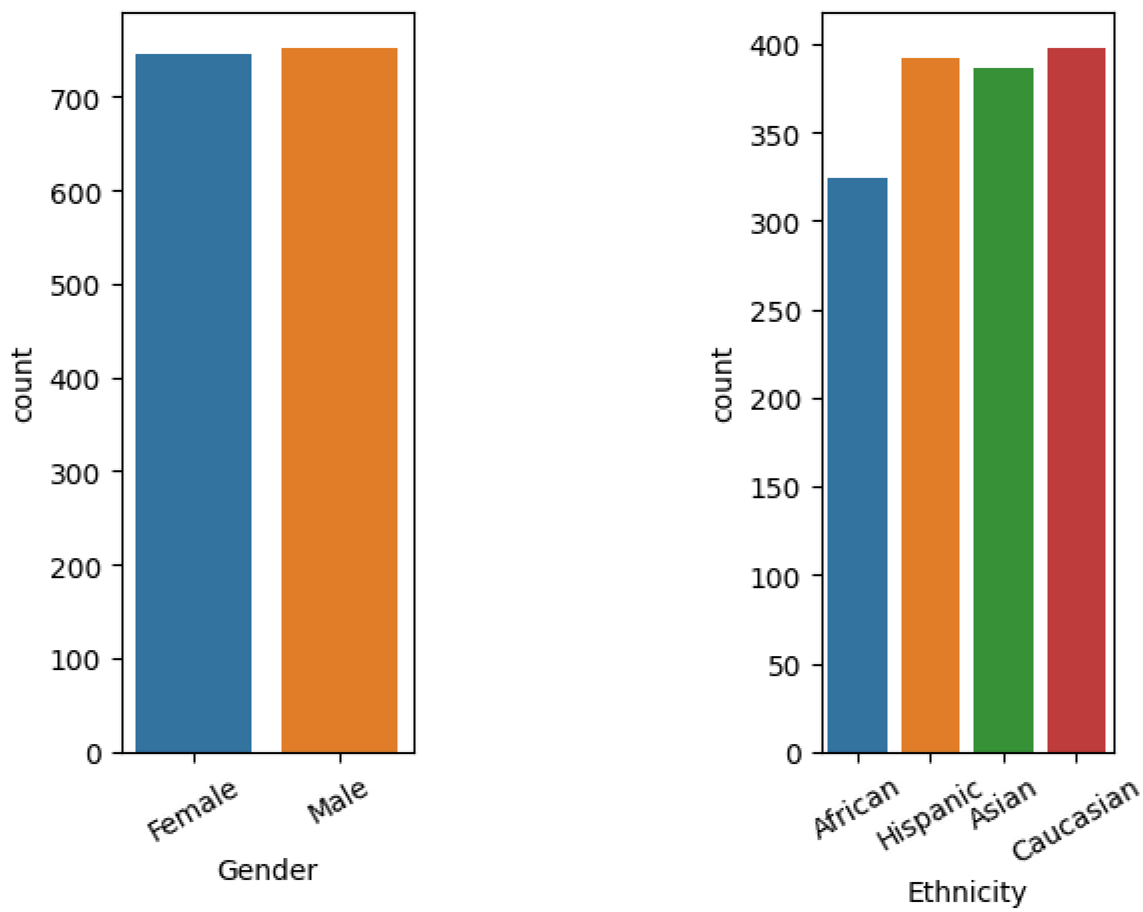
```
train
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1500 entries, 0 to 1499
Data columns (total 17 columns):
 #   Column                        Non-Null Count  Dtype
---  ------                        --------------  -----
 0   Patient ID                    1500 non-null   int64
 1   Age                           1500 non-null   int64
 2   Gender                        1500 non-null   object
 3   Ethnicity                     1500 non-null   object
 4   Marital Status                1500 non-null   object
 5   Education Level               1500 non-null   object
 6   OCD Diagnosis Date            1500 non-null   object
 7   Duration of Symptoms (months) 1500 non-null   int64
```

```
 8   Previous Diagnoses              1252 non-null   object
 9   Family History of OCD           1500 non-null   object
 10  Obsession Type                  1500 non-null   object
 11  Compulsion Type                 1500 non-null   object
 12  Y-BOCS Score (Obsessions)       1500 non-null   int64
 13  Y-BOCS Score (Compulsions)      1500 non-null   int64
 14  Depression Diagnosis            1500 non-null   object
 15  Anxiety Diagnosis               1500 non-null   object
 16  Medications                     1114 non-null   object
dtypes: int64(5), object(12)
memory usage: 199.3+ KB

None
```

In [7]:

```
plt.subplot(1, 3, 1)
sns.countplot(x = train["Gender"])
plotter.xticks(rotation = 30);
```

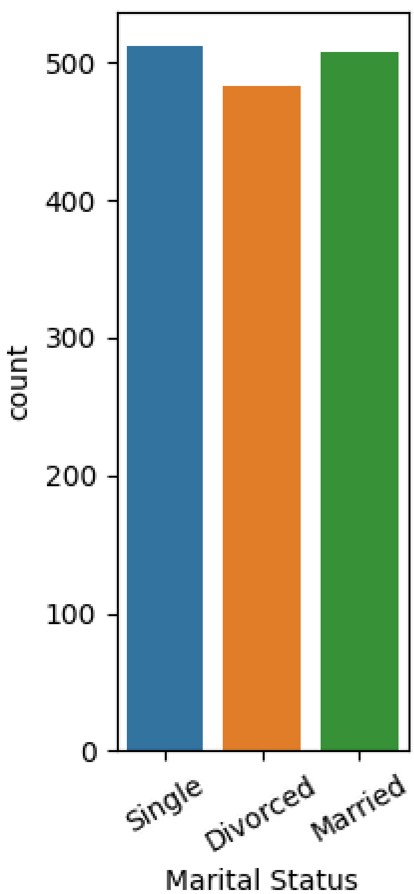
```
plt.subplot(1, 3, 3)

sns.countplot(x = train["Ethnicity"])

plotter.xticks(rotation = 30);
```

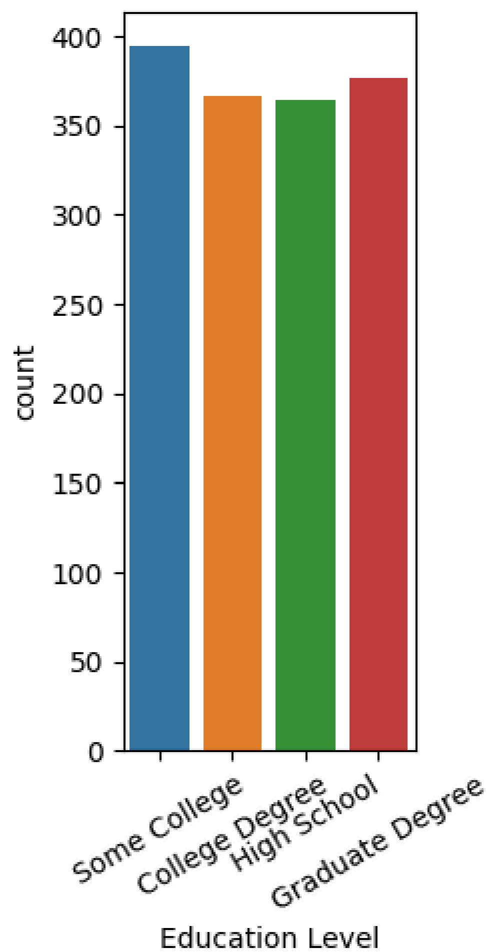


In [8]:

```
plt.subplot(1, 3, 1)

sns.countplot(x = train["Marital Status"])
```

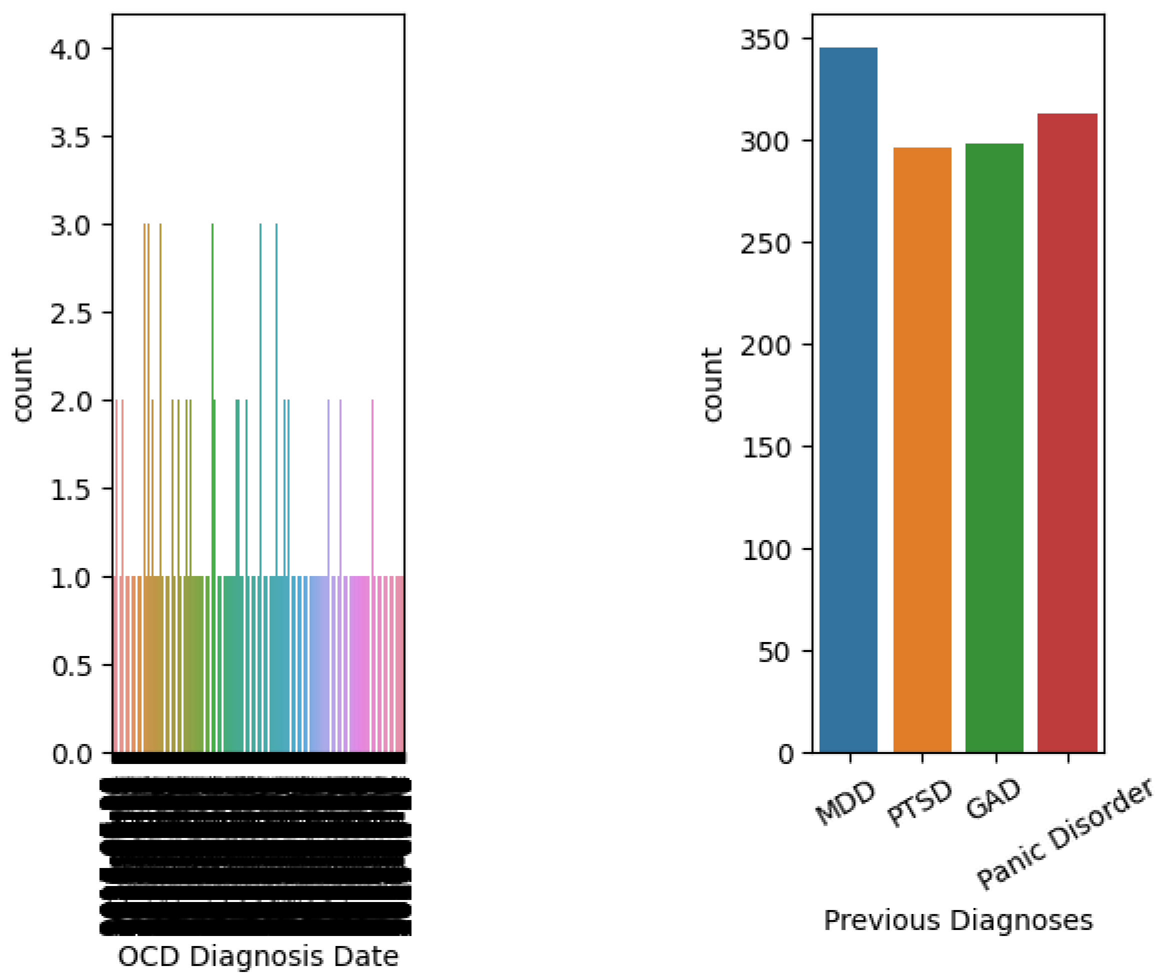
```
plotter.xticks(rotation = 30);



plt.subplot(1, 3, 3)

sns.countplot(x = train["Education Level"])

plotter.xticks(rotation = 30);
```





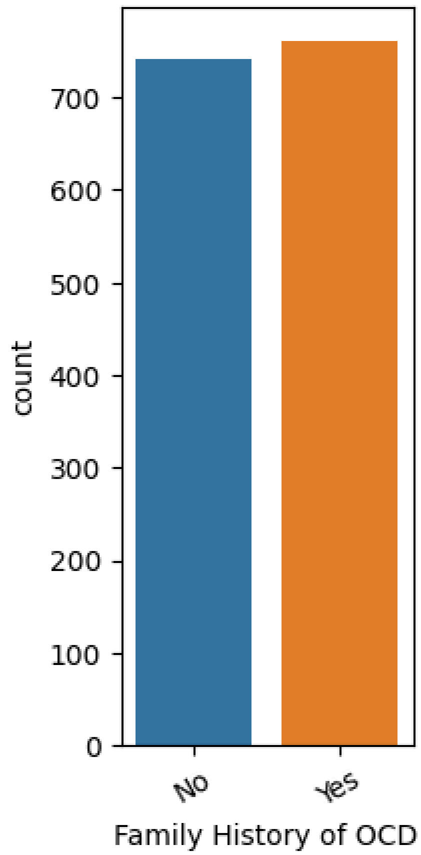
In [9]:

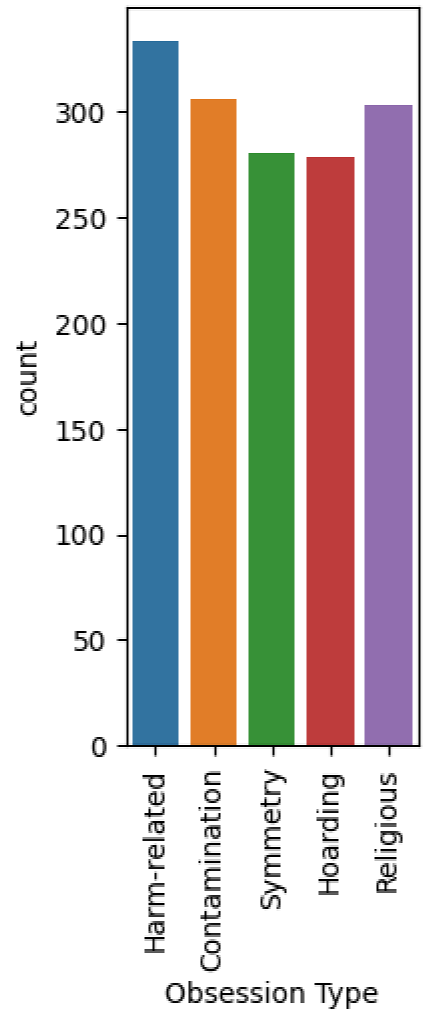
```python
plt.subplot(1, 3, 1)

sns.countplot(x = train["OCD Diagnosis Date"])

plotter.xticks(rotation = 90);



plt.subplot(1, 3, 3)

sns.countplot(x = train["Previous Diagnoses"])

plotter.xticks(rotation = 30);
```



In [10]:

```
plt.subplot(1, 3, 1)

sns.countplot(x = train["Family History of OCD"])

plotter.xticks(rotation = 30);



plt.subplot(1, 3, 3)

sns.countplot(x = train["Obsession Type"])

plotter.xticks(rotation = 90);
```

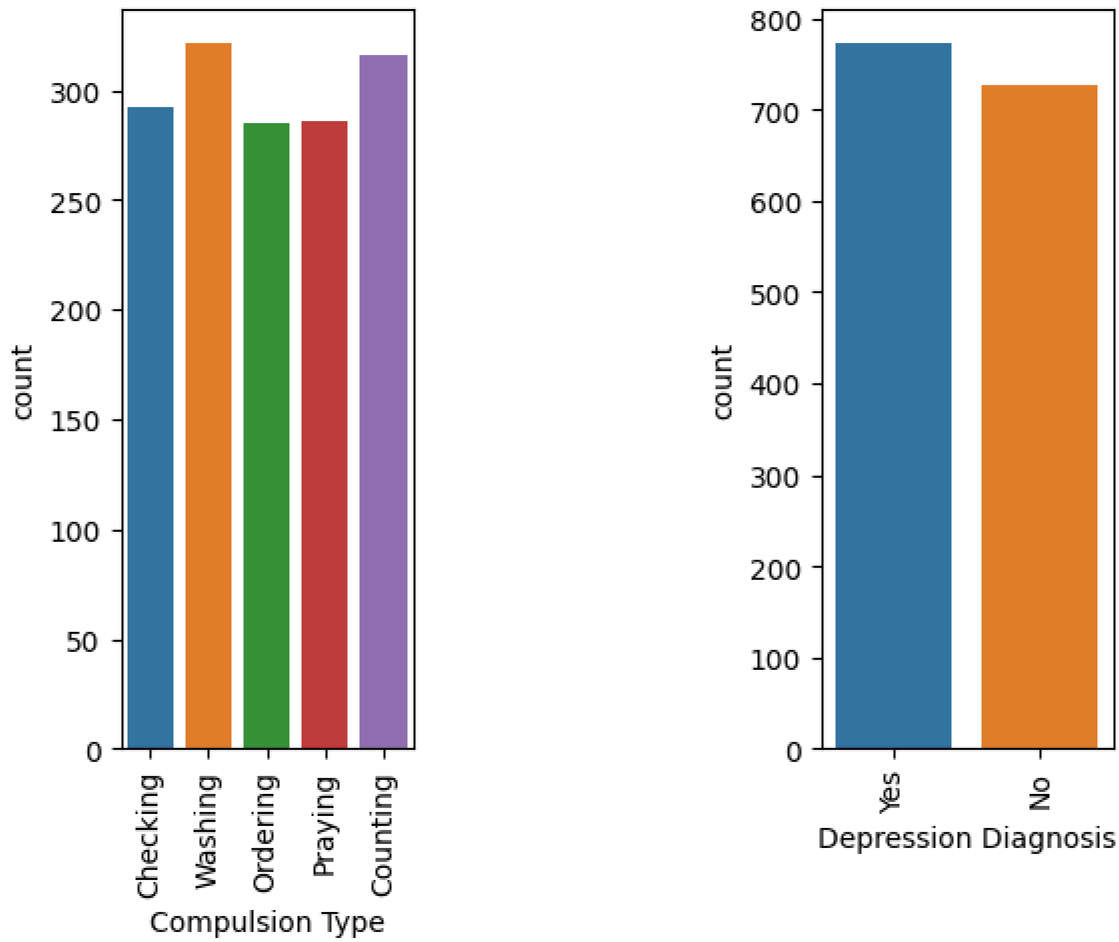




In [11]:

```
plt.subplot(1, 3, 1)

sns.countplot(x = train["Compulsion Type"])

plotter.xticks(rotation = 90);



plt.subplot(1, 3, 3)

sns.countplot(x = train["Depression Diagnosis"])
```

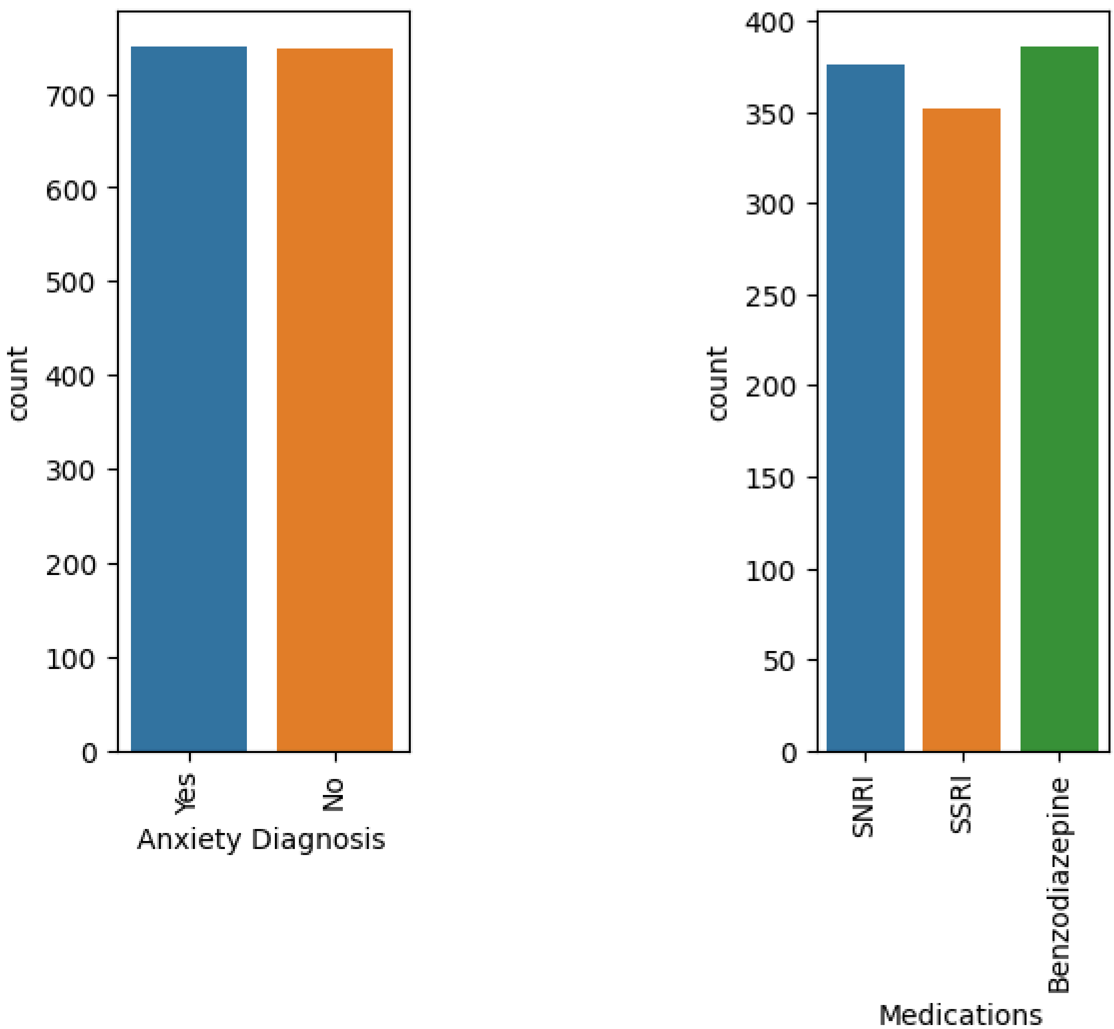
```
plotter.xticks(rotation = 90);
```



In [12]:

```
plt.subplot(1, 3, 1)

sns.countplot(x = train["Anxiety Diagnosis"])

plotter.xticks(rotation = 90);
```

```
plt.subplot(1, 3, 3)

sns.countplot(x = train["Medications"])

plotter.xticks(rotation = 90);
```



Preprocessing

In [13]:

```
train["Gender"] =
train["Gender"].replace({'Female':1,'Male':2})

train["Ethnicity"] =
train["Ethnicity"].replace({'African':1,'Hispanic':2,'Asian':3,
'Caucasian':4})

train["Marital Status"] = train["Marital
Status"].replace({'Single':1,'Divorced':2,'Married':3})

train["Education Level"] = train["Education
Level"].replace({'Some College':1,'College Degree':2,


'High School':3,'Graduate Degree':4})

train=train.drop(columns=['OCD Diagnosis Date'],axis=1)

train["Previous Diagnoses"] = train["Previous
Diagnoses"].replace({'MDD':1,'PTSD':2,'GAD':3,'Panic
Disorder':4})

train["Family History of OCD"] = train["Family History of
OCD"].replace({'No':1,'Yes':2})

train["Obsession Type"] = train["Obsession
Type"].replace({'Harm-related':1,'Contamination':2,'Symmetry':3
,
```

```
'Hoarding':4,'Religious':5})

train["Compulsion Type"] = train["Compulsion Type"].replace({'Checking':1,'Washing':2,'Ordering':3,'Praying':4,


'Counting':5})

train["Depression Diagnosis"] = train["Depression Diagnosis"].replace({'No':1,'Yes':2})

train["Anxiety Diagnosis"] = train["Anxiety Diagnosis"].replace({'No':1,'Yes':2})

train["Medications"] = train["Medications"].replace({'SNRI':0,'SSRI':1,'Benzodiazepine':2})

display(train)
```

	Patient ID	Age	Gender	Ethnicity	Marital Status	Education Level	Duration of Symptoms (mont	Previous Diagnoses	Family History of	Obsession Type	Compulsion Type	Y-BOCS Score (Obsessions)	Y-BOCS Score (Compulsions)	Depression Diagnosis	Anxiety Diagnosis	Medications

							hs)		OCD							
0	1018	32	1	1	1	1	203	1.0	1	1	1	17	10	2	2	0.0
1	2406	69	2	1	2	1	180	NaN	2	1	2	21	25	2	2	1.0
2	1188	57	2	2	2	2	173	1.0	1	2	1	3	4	1	1	2.0
3	6200	27	1	2	3	2	126	2.0	2	3	2	14	28	2	2	1.0
4	5824	56	1	2	3	3	168	2.0	2	4	3	39	18	1	1	NaN
..	...	.	...	...	...	...	...	...	...	...	...	...	...	...	...	...

.		..														
1495	5374	38	2	2	2	2	53	1.0	1	2	2	21	33	2	2	1.0
1496	5013	19	1	2	2	4	160	3.0	2	4	4	25	16	2	2	1.0
1497	6089	40	2	3	3	1	100	NaN	2	2	5	2	15	2	2	2.0
1498	3808	37	1	4	3	1	210	3.0	2	2	2	16	7	2	1	2.0
1499	2221	18	2	4	1	3	91	NaN	2	4	3	22	34	2	1	0.0

1500 rows × 16 columns

In [14]:

```
#Previous Diagnoses
print("Skewness: %f" % train['Previous Diagnoses'].skew())
print("Kurtosis: %f" % train['Previous Diagnoses'].kurt())
```

```
Skewness: 0.040787
Kurtosis: -1.409371
```

In [15]:

```
from sklearn.impute import SimpleImputer
num_cols = ['Previous Diagnoses']
num_imp = SimpleImputer(strategy='mean')
train[num_cols] =
pd.DataFrame(num_imp.fit_transform(train[num_cols]),columns=num
_cols)
train
```

Out[15]:

	Patient ID	Age	Gender	Ethnicity	Marital Status	Education Level	Duration of Symptoms (months)	Previous Diagnoses	Family History of OCD	Obsession Type	Compulsion Type	Y-BOCS Score (Obsessions)	Y-BOCS Score (Compulsions)	Depression Diagnosis	Anxiety Diagnosis	Medications
0	1018	32	1	1	1	1	203	1.000000	1	1	1	17	10	2	2	0.0
1	2406	69	2	1	2	1	180	2.466246	2	1	2	21	25	2	2	1.0
2	1188	57	2	2	2	2	173	1.000000	1	2	1	3	4	1	1	2.0
3	620	27	1	2	3	2	126	2.000	2	3	2	14	28	2	2	1.0

	0							00								
4	5824	56	1	2	3	3	168	2.000000	2	4	3	39	18	1	1	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1495	53774	38	2	2	2	2	53	1.000000	1	2	2	21	33	2	2	1.0
1496	50113	19	1	2	2	4	160	3.000000	2	4	4	25	16	2	2	1.0
1497	60089	40	2	3	3	1	100	2.462646	2	2	5	2	15	2	2	2.0
1498	38008	37	1	4	3	1	210	3.000000	2	2	2	16	7	2	1	2.0

1499	2221	18	2	4	1	3	91	2.462246	2	4	3	22	34	2	1	0.0

1500 rows × 16 columns

In [16]:

```
train=train.dropna(axis=0,how='any')

train
```

Out[16]:

	Patient ID	Age	Gender	Ethnicity	Marital Status	Education Level	Duration of Symptoms (months)	Previous Diagnoses	Family History of OCD	Obsession Type	Compulsion Type	Y-BOCS Score (Obsessions)	Y-BOCS Score (Compulsions)	Depression Diagnosis	Anxiety Diagnosis	Medications

0	1018	32	1	1	1	1	203	1.00000	1	1	1	17	10	2	2	0.0
1	2406	69	2	1	2	1	180	2.46246	2	1	2	21	25	2	2	1.0
2	1188	57	2	2	2	2	173	1.00000	1	2	1	3	4	1	1	2.0
3	6200	27	1	2	3	2	126	2.00000	2	3	2	14	28	2	2	1.0
5	6946	32	1	3	3	2	46	3.00000	1	4	3	26	11	2	2	1.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...

1495	5374	38	2	2	2	2	53	1.000000	1	2	2	21	33	2	2	1.0
1496	5013	19	1	2	2	4	160	3.000000	2	4	4	25	16	2	2	1.0
1497	6089	40	2	3	3	1	100	2.462446	2	2	5	2	15	2	2	2.0
1498	3808	37	1	4	3	1	210	3.000000	2	2	2	16	7	2	1	2.0
1499	2221	18	2	4	1	3	91	2.462446	2	4	3	22	34	2	1	0.0

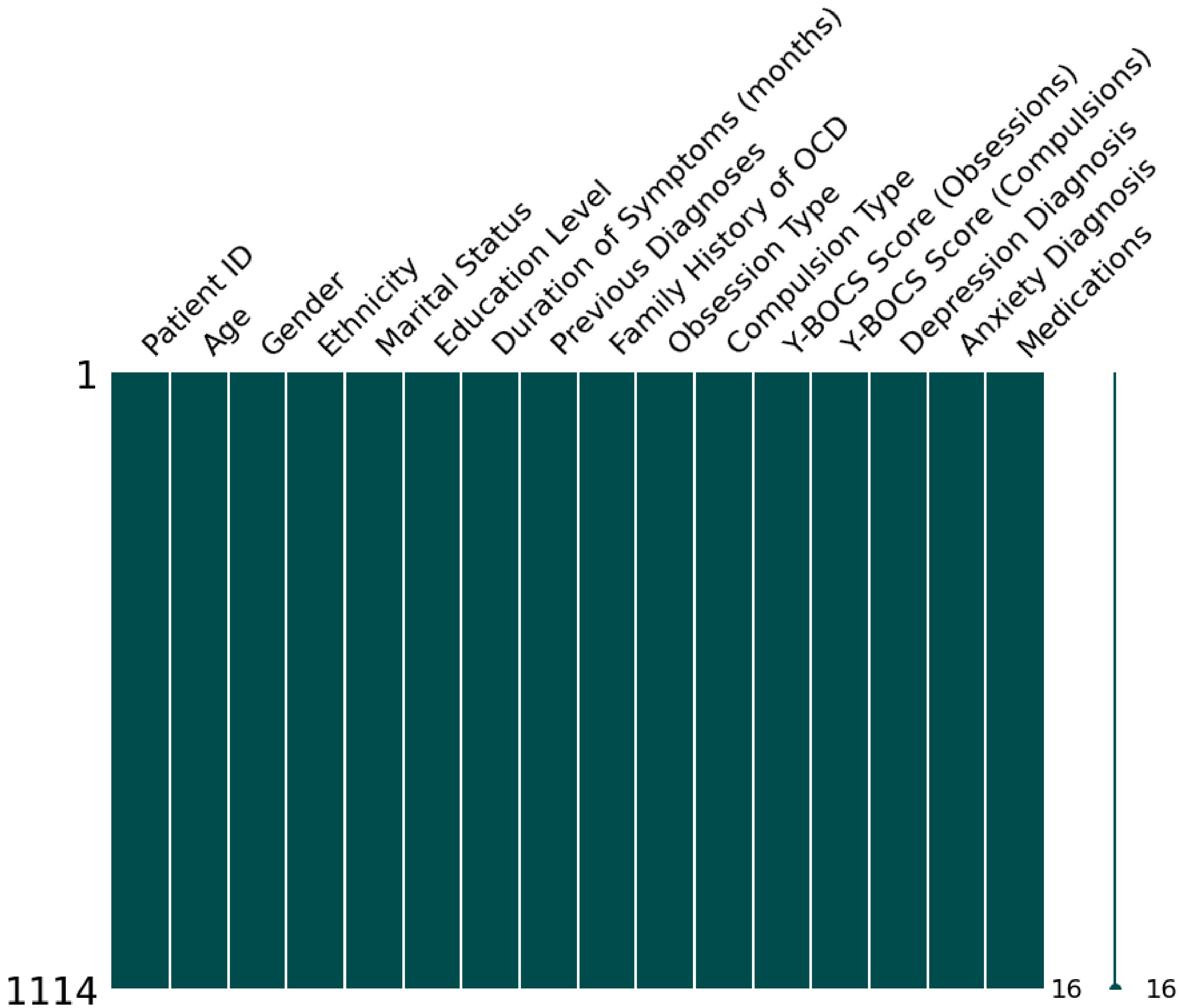
1114 rows × 16 columns

In [17]:

```
msno.matrix(df=train, figsize=(10,6), color=(0,.3,.3))
```

Out[17]:

```
<Axes: >
```



In [18]:

```
train_feature = train.columns.drop('Medications').tolist()

train_feature
```

Out[18]:

```
['Patient ID',
 'Age',
 'Gender',
 'Ethnicity',
 'Marital Status',
 'Education Level',
 'Duration of Symptoms (months)',
 'Previous Diagnoses',
 'Family History of OCD',
 'Obsession Type',
 'Compulsion Type',
 'Y-BOCS Score (Obsessions)',
 'Y-BOCS Score (Compulsions)',
 'Depression Diagnosis',
```

```
'Anxiety Diagnosis']
```

In [19]:

```
train[train_feature].describe().T\
        .style.bar(subset=['mean'],
color=px.colors.qualitative.G10[0])\
        .background_gradient(subset=['std'], cmap='BuPu')\
        .background_gradient(subset=['50%'], cmap='Reds')
```

Out[19]:

	count	mean	std	min	25%	50%	75%	max
Patient ID	1114.000000	5546.394973	2568.490997	1017.000000	3334.750000	5607.000000	7757.500000	9995.000000
Age	1114.000000	46.660682	16.889784	18.000000	32.000000	47.000000	61.000000	75.000000
Gender	1114.000000	1.500898	0.500224	1.000000	1.000000	2.000000	2.000000	2.000000
Ethnicity	1114.000000	2.582585	1.091049	1.000000	2.000000	3.000000	4.000000	4.000000

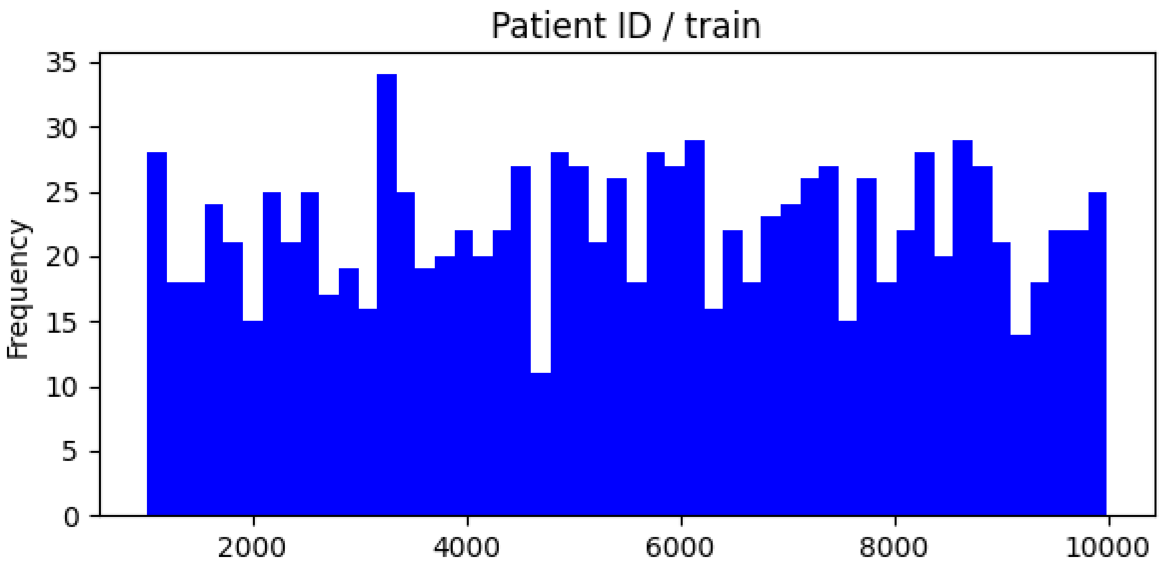
Marital Status	1114.000000	1.987433	0.820790	1.000000	1.000000	2.000000	3.000000	3.000000
Education Level	1114.000000	2.484740	1.129623	1.000000	1.000000	2.000000	3.000000	4.000000
Duration of Symptoms (months)	1114.000000	123.126571	67.473845	6.000000	65.000000	123.000000	179.000000	239.000000
Previous Diagnoses	1114.000000	2.441118	1.033862	1.000000	2.000000	2.462460	3.000000	4.000000
Family History of OCD	1114.000000	1.512567	0.500067	1.000000	1.000000	2.000000	2.000000	2.000000
Obsession Type	1114.000000	2.891382	1.439981	1.000000	2.000000	3.000000	4.000000	5.000000
Compulsion	1114.000000	3.038600	1.422872	1.000000	2.000000	3.000000	4.000000	5.000000

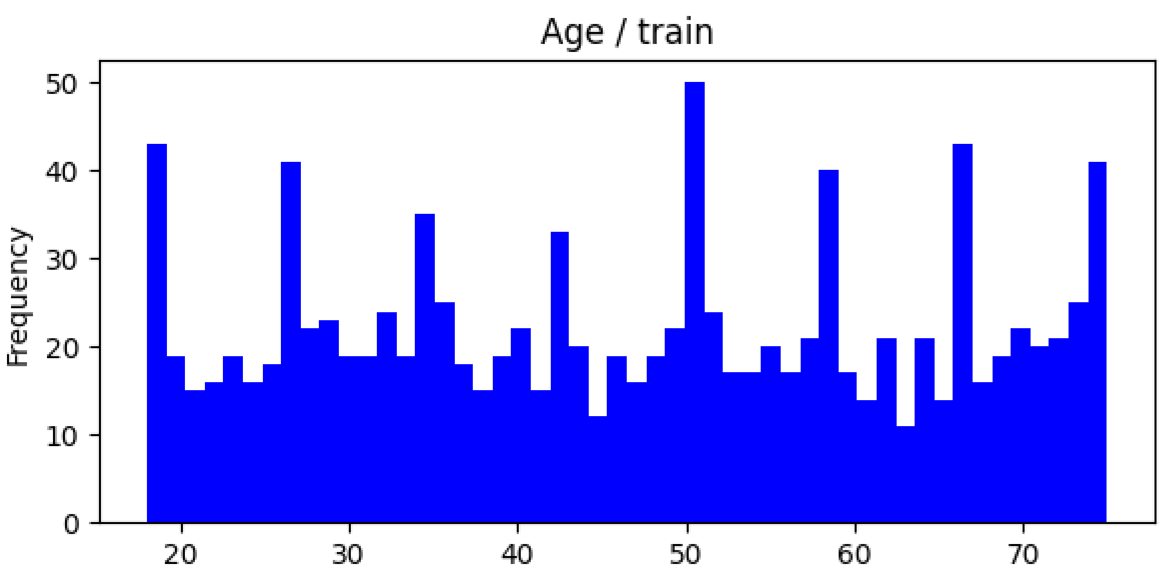
Type								
Y-BOCS Score (Obsessions)	1114.000000	20.073609	11.755367	0.000000	10.000000	20.000000	30.000000	40.000000
Y-BOCS Score (Compulsions)	1114.000000	19.598743	11.837308	0.000000	9.000000	20.000000	29.000000	40.000000
Depression Diagnosis	1114.000000	1.539497	0.498661	1.000000	1.000000	2.000000	2.000000	2.000000
Anxiety Diagnosis	1114.000000	1.487433	0.500067	1.000000	1.000000	1.000000	2.000000	2.000000

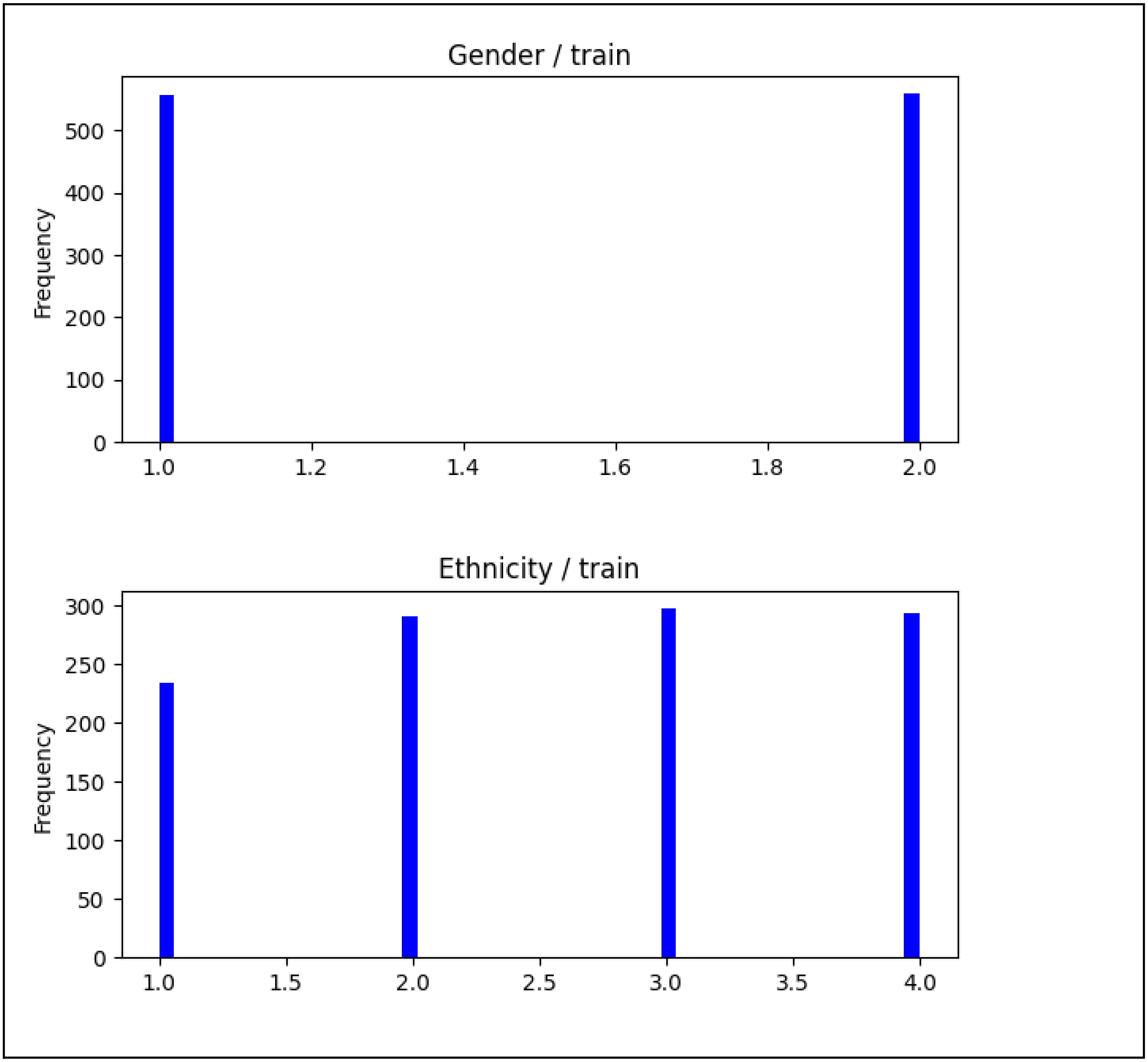
In [20]:

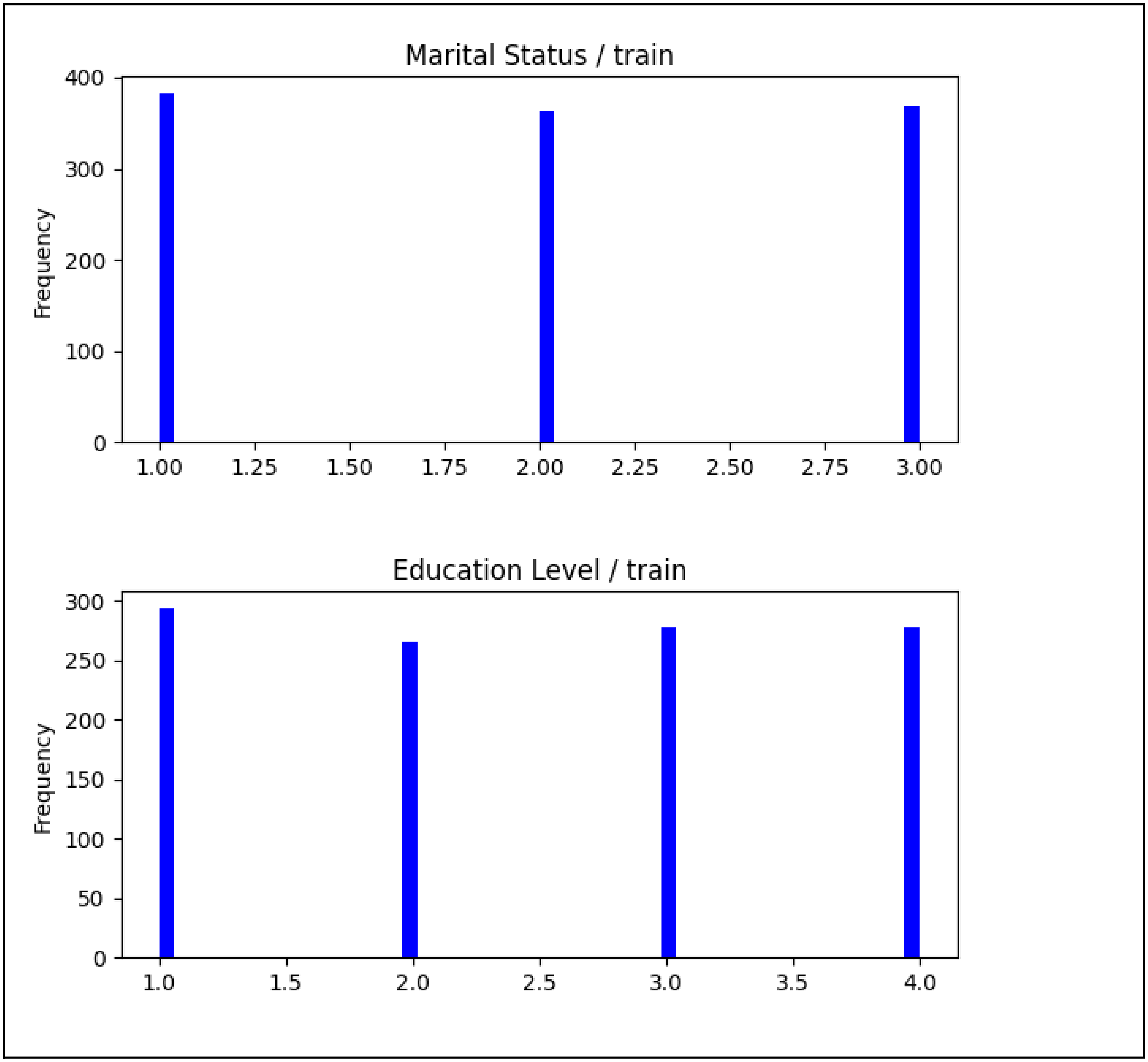
```
for feat in train_feature:
    plt.figure(figsize=(15,3))
    ax1 = plt.subplot(1,2,1)
    train[feat].plot(kind='hist', bins=50, color='blue')
```

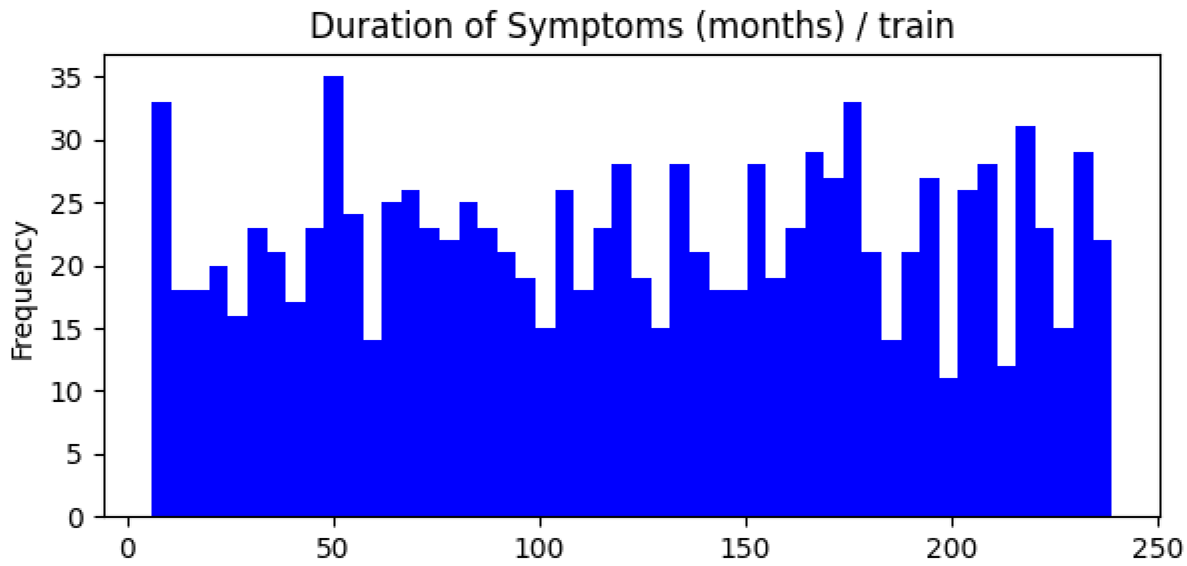
```python
    plt.title(feat + ' / train')

    plt.show()
```

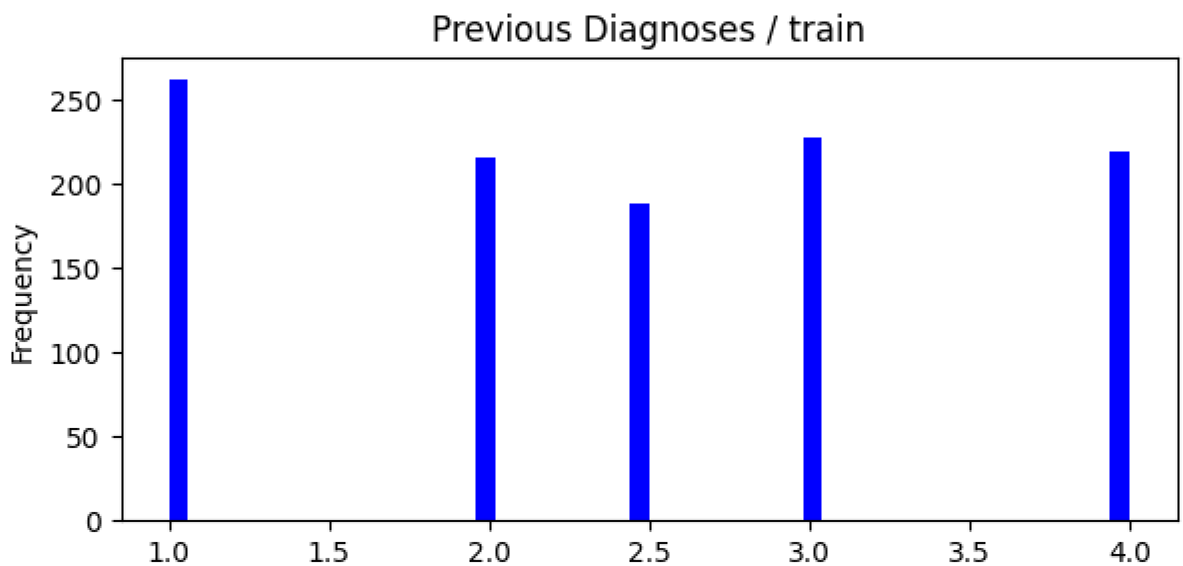


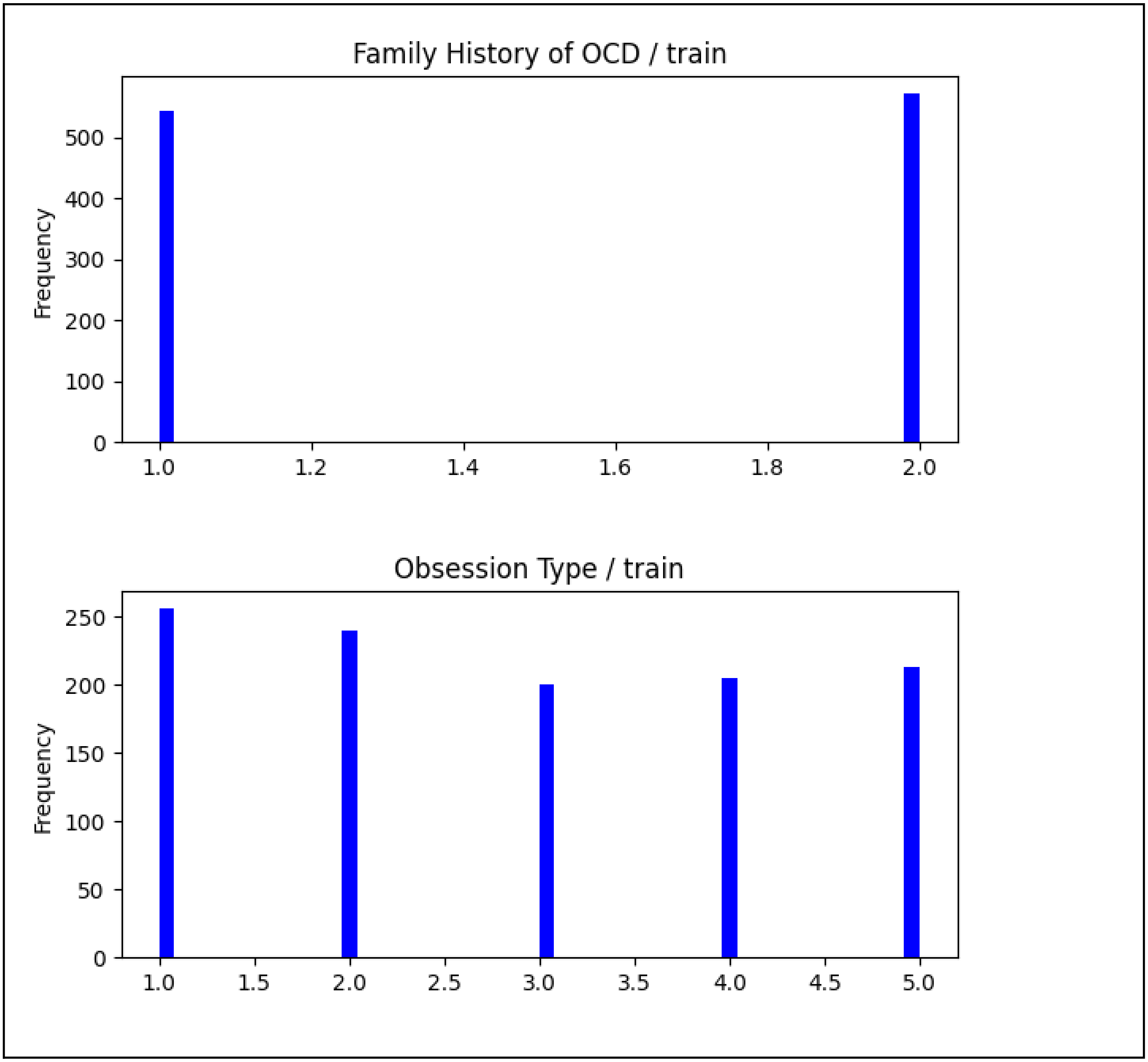


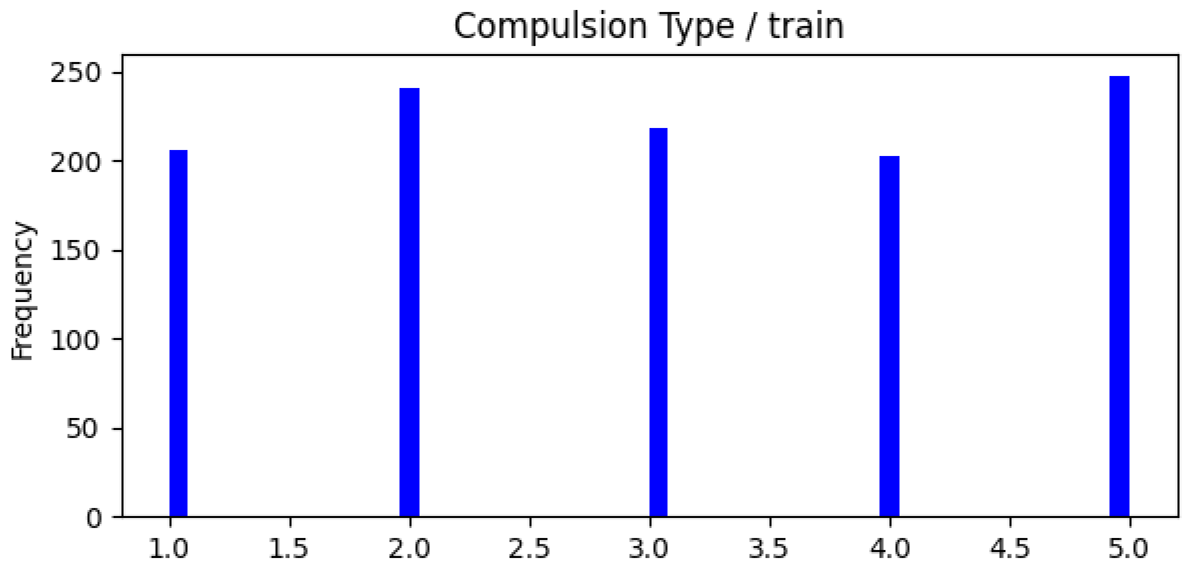

Gender / train
Frequency
0
100
200
300
400
500
1.0
1.2
1.4
1.6
1.8
2.0
Ethnicity / train
Frequency
0
50
100
150
200
250
300
1.0
1.5
2.0
2.5
3.0
3.5
4.0

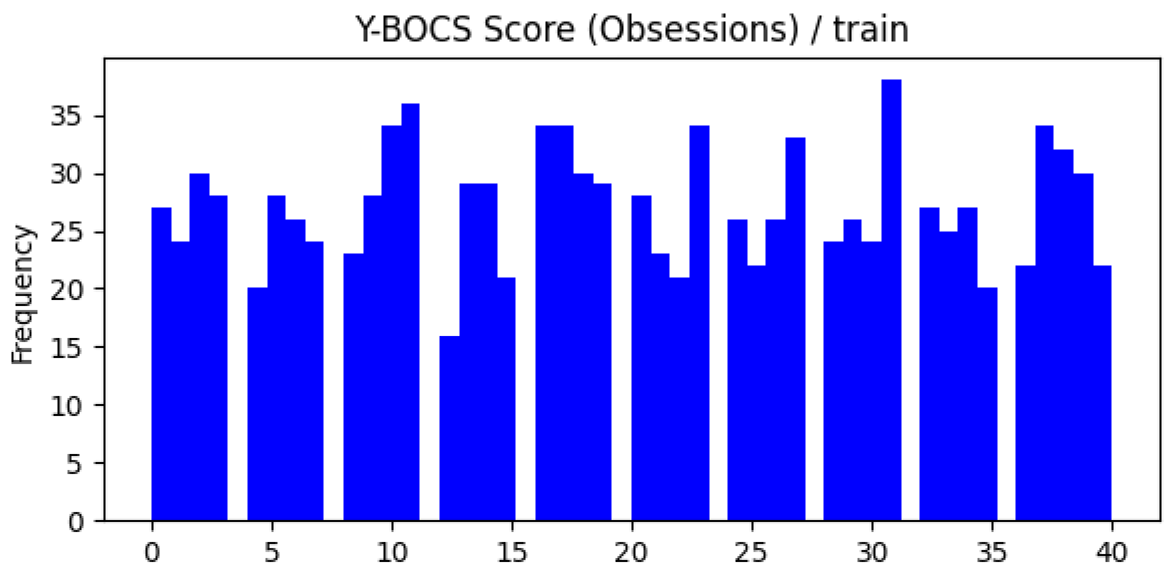

Marital Status / train
Frequency
0
100
200
300
400
1.00
1.25
1.50
1.75
2.00
2.25
2.50
2.75
3.00
Education Level / train
Frequency
0
50
100
150
200
250
300
1.0
1.5
2.0
2.5
3.0
3.5
4.0

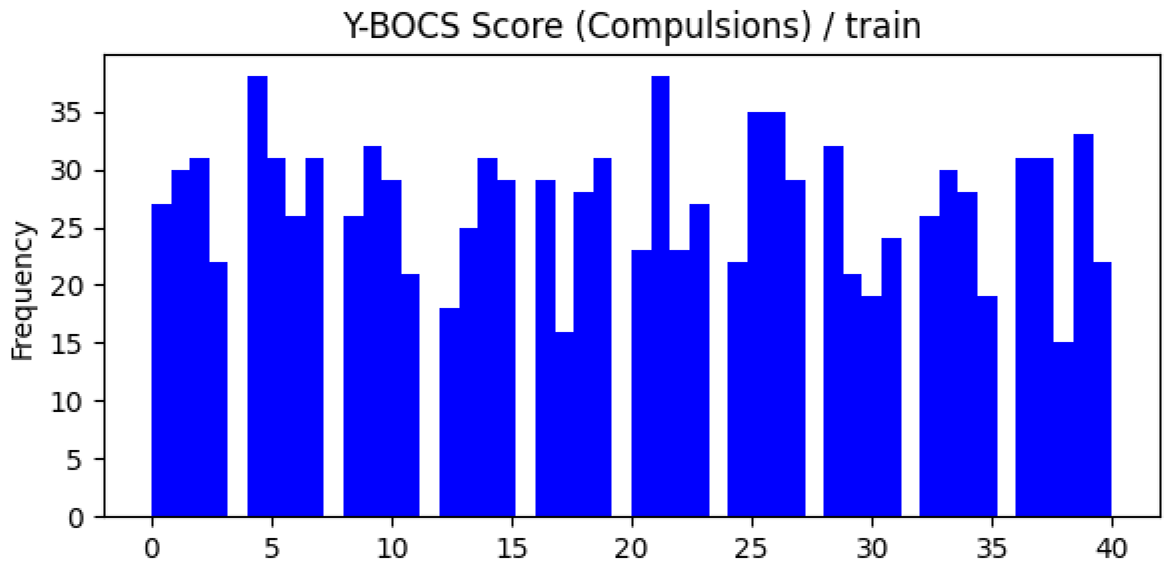

Duration of Symptoms (months) / train
Frequency
0
5
10
15
20
25
30
35
0
50
100
150
200
250

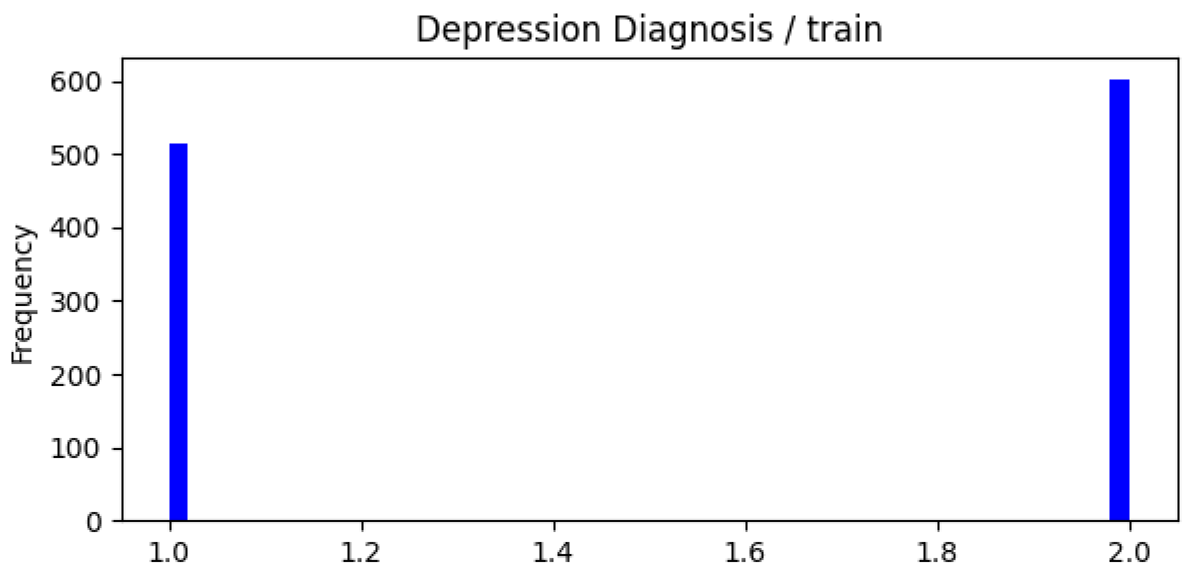

Previous Diagnoses / train
Frequency
0
50
100
150
200
250
1.0
1.5
2.0
2.5
3.0
3.5
4.0

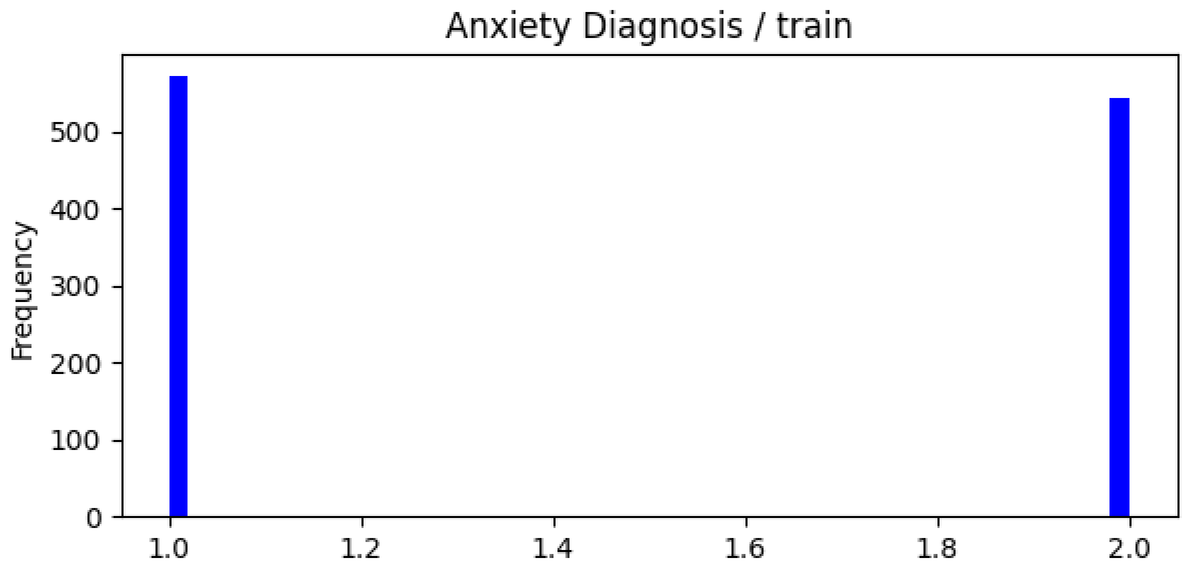

Family History of OCD / train
Frequency
0
100
200
300
400
500
1.0
1.2
1.4
1.6
1.8
2.0
Obsession Type / train
Frequency
0
50
100
150
200
250
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0


Compulsion Type / train
Frequency
250
200
150
100
50
0
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0


Y-BOCS Score (Obsessions) / train
Frequency
35
30
25
20
15
10
5
0
0
5
10
15
20
25
30
35
40


Y-BOCS Score (Compulsions) / train
Frequency
0
5
10
15
20
25
30
35
0
5
10
15
20
25
30
35
40


Depression Diagnosis / train
Frequency
0
100
200
300
400
500
600
1.0
1.2
1.4
1.6
1.8
2.0



In [21]:

linkcode

```
#skew & kurt

print("Skewness: %f" % train['Age'].skew())

print("Kurtosis: %f" % train['Age'].kurt())

print("Skewness: %f" % train['Gender'].skew())

print("Kurtosis: %f" % train['Gender'].kurt())

print("Skewness: %f" % train['Ethnicity'].skew())

print("Kurtosis: %f" % train['Ethnicity'].kurt())

print("Skewness: %f" % train['Marital Status'].skew())

print("Kurtosis: %f" % train['Marital Status'].kurt())

print("Skewness: %f" % train['Education Level'].skew())
```

```
print("Kurtosis: %f" % train['Education Level'].kurt())

print("Skewness: %f" % train['Duration of Symptoms
(months)'].skew())

print("Kurtosis: %f" % train['Duration of Symptoms
(months)'].kurt())

print("Skewness: %f" % train['Previous Diagnoses'].skew())

print("Kurtosis: %f" % train['Previous Diagnoses'].kurt())

print("Skewness: %f" % train['Family History of OCD'].skew())

print("Kurtosis: %f" % train['Family History of OCD'].kurt())

print("Skewness: %f" % train['Obsession Type'].skew())

print("Kurtosis: %f" % train['Obsession Type'].kurt())

print("Skewness: %f" % train['Compulsion Type'].skew())

print("Kurtosis: %f" % train['Compulsion Type'].kurt())

print("Skewness: %f" % train['Y-BOCS Score
(Obsessions)'].skew())

print("Kurtosis: %f" % train['Y-BOCS Score
(Obsessions)'].kurt())

print("Skewness: %f" % train['Y-BOCS Score
(Compulsions)'].skew())
```

```
print("Kurtosis: %f" % train['Y-BOCS Score (Compulsions)'].kurt())

print("Skewness: %f" % train['Depression Diagnosis'].skew())

print("Kurtosis: %f" % train['Depression Diagnosis'].kurt())

print("Skewness: %f" % train['Anxiety Diagnosis'].skew())

print("Kurtosis: %f" % train['Anxiety Diagnosis'].kurt())
```

```
Skewness: 0.006356

Kurtosis: -1.204796

Skewness: -0.003596

Kurtosis: -2.003587

Skewness: -0.089318

Kurtosis: -1.291406

Skewness: 0.023231

Kurtosis: -1.515368

Skewness: 0.009850

Kurtosis: -1.386177

Skewness: -0.014410
```

```
Kurtosis: -1.201152

Skewness: 0.054362

Kurtosis: -1.067981

Skewness: -0.050353

Kurtosis: -2.001060

Skewness: 0.112878

Kurtosis: -1.338223

Skewness: 0.008749

Kurtosis: -1.318512

Skewness: 0.000971

Kurtosis: -1.182194

Skewness: 0.023188

Kurtosis: -1.203450

Skewness: -0.158698

Kurtosis: -1.978370

Skewness: 0.050353

Kurtosis: -2.001060
```

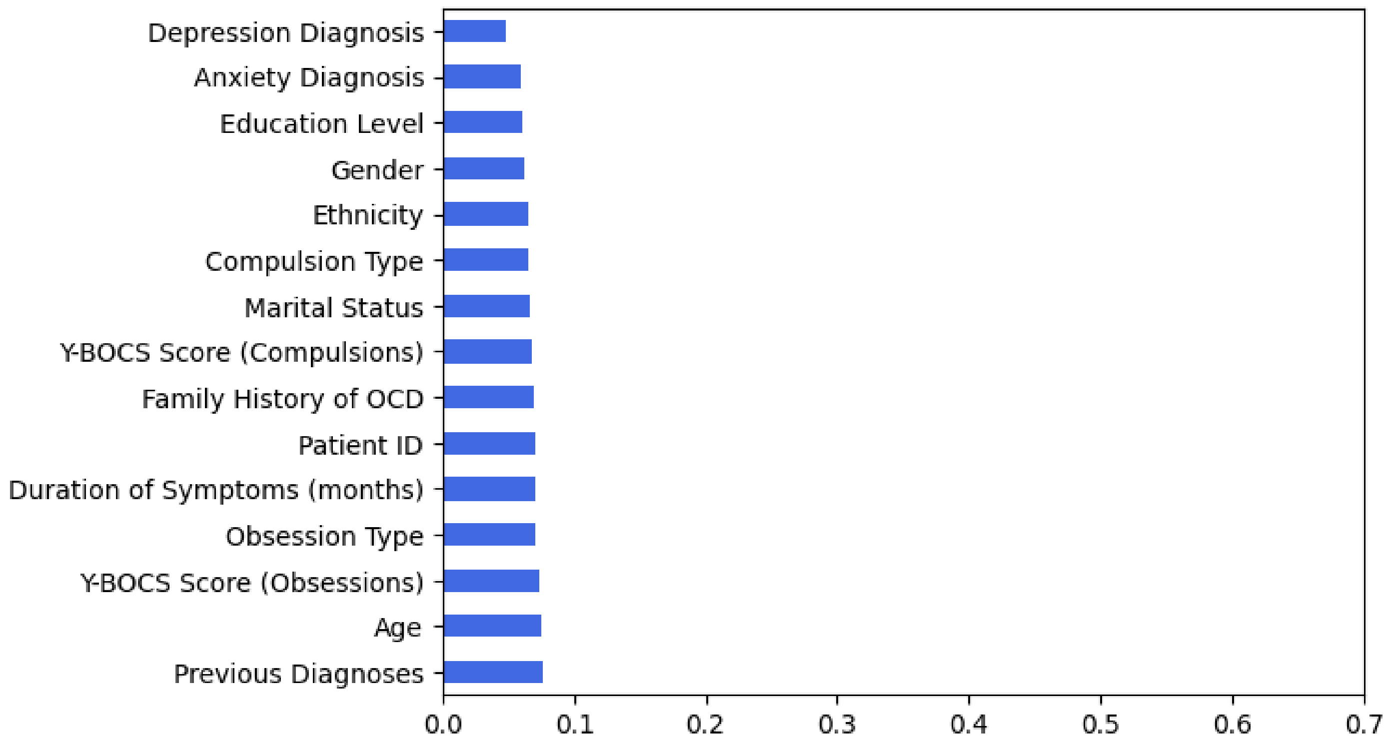
Feature Selection

In [22]:

```
X_data_feature= train.drop(columns=['Medications'],axis=1)

y_data_feature= train['Medications']


model = [XGBClassifier()]


model = [model[i].fit(X_data_feature,y_data_feature) for i in
range(len(model))]


num_chr = [12, 12, 10]


for i in range(len(model)):

    print(str(model[i])[:num_chr[i]] + ': \n',

          model[i].feature_importances_)

    feat_importances = pd.Series(model[i].feature_importances_,

                                 index=X_data_feature.columns)
```

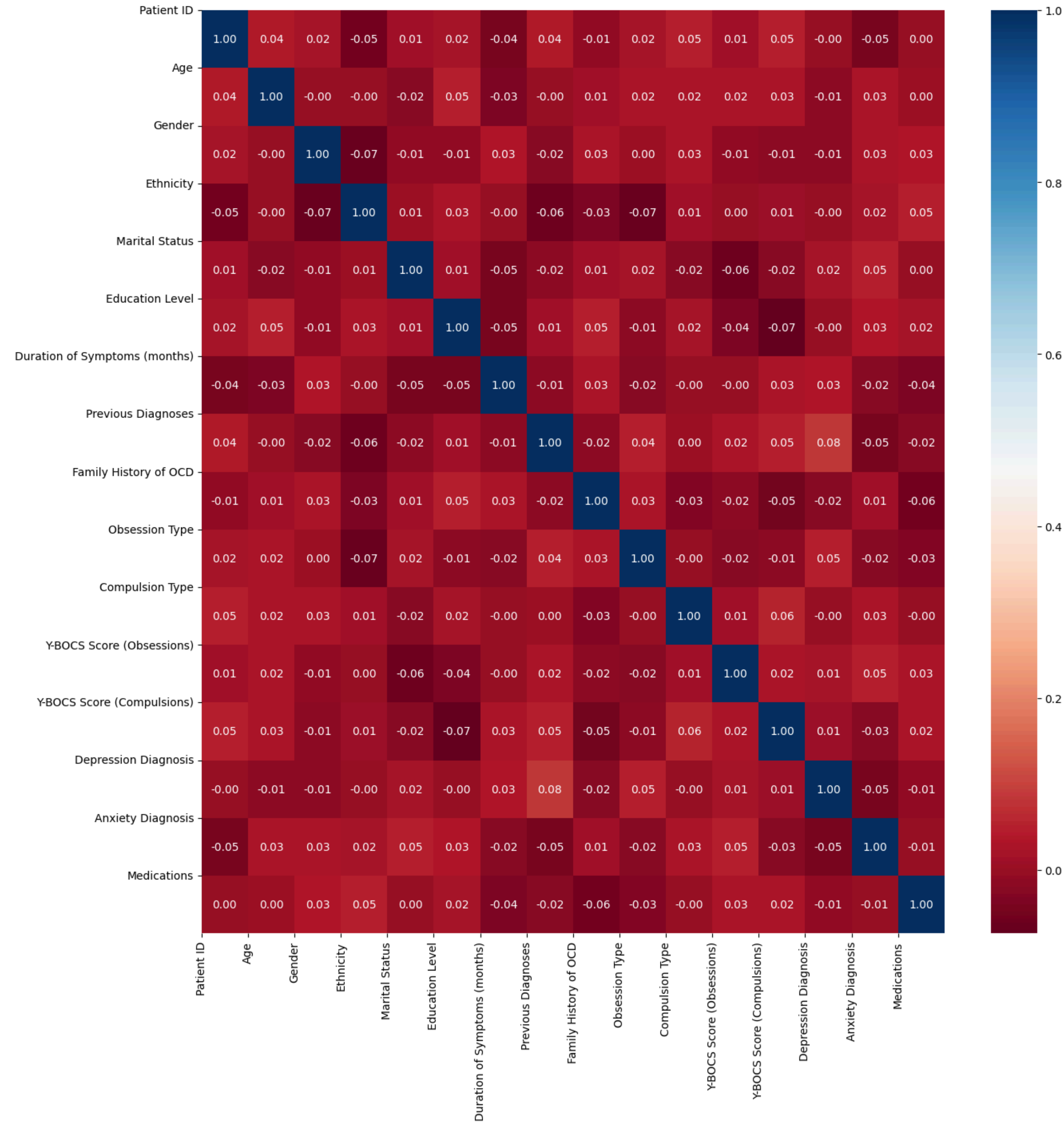
```
    feat_importances.nlargest(15).plot(kind='barh', color='royalblue')

    plt.xlim(0, 0.7)

    plt.show()
```

```
XGBClassifie:

 [0.07036036 0.07522406 0.06250991 0.06455468 0.06716412 0.06086183

 0.07048298 0.07587978 0.06881509 0.07055759 0.06507026 0.073146

 0.06800257 0.04821378 0.05915698]
```



In [23]:

```
corr = train.corr(method='pearson')

fig, ax = plt.subplots(figsize=(15, 15))

sns.heatmap(corr, cmap='RdBu', annot=True, fmt=".2f")

plt.xticks(range(len(corr.columns)), corr.columns);

plt.yticks(range(len(corr.columns)), corr.columns)

plt.show()
```



In [24]:

```
X= train.drop(columns=['Medications'],axis=1)
```

```
y= train['Medications']
```

In [25]:

```
X_train=X

y_train=y


from sklearn.preprocessing import MinMaxScaler

MinMaxScaler = MinMaxScaler()

X_train = MinMaxScaler.fit_transform(X_train)

X_train = pd.DataFrame(X_train)

X_train
```

Out[25]:

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
0	0.000111	0.245614	0.0	0.000000	0.0	0.000000	0.845494	0.000000	0.0	0.00	0.00	0.425	0.250	1.0	1.0

1	0.154712	0.894737	1.0	0.000000	0.5	0.000000	0.746781	0.487487	1.0	0.00	0.25	0.525	0.625	1.0	1.0
2	0.019047	0.684211	1.0	0.333333	0.5	0.333333	0.716738	0.000000	0.0	0.25	0.00	0.075	0.100	0.0	0.0
3	0.577300	0.157895	0.0	0.333333	1.0	0.333333	0.515021	0.333333	1.0	0.50	0.25	0.350	0.700	1.0	1.0
4	0.660392	0.245614	0.0	0.666667	1.0	0.333333	0.171674	0.666667	0.0	0.75	0.50	0.650	0.275	1.0	1.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1109	0.485297	0.350877	1.0	0.333333	0.5	0.333333	0.201717	0.000000	0.0	0.25	0.25	0.525	0.825	1.0	1.0
1110	0.445088	0.017544	0.0	0.333333	0.5	1.000000	0.660944	0.666667	1.0	0.75	0.75	0.625	0.400	1.0	1.0
11	0.56	0.38	1	0.66	1	0.00	0.40	0.48	1	0.	1.	0.0	0.3	1	1

11	4937	5965	.0	6667	.0	0000	3433	7487	.0	25	00	50	75	.0	.0
1112	0.310871	0.333333	0.0	1.000000	1.0	0.000000	0.875536	0.666667	1.0	0.25	0.25	0.400	0.175	1.0	0.0
1113	0.134106	0.000000	1.0	1.000000	0.0	0.666667	0.364807	0.487487	1.0	0.75	0.50	0.550	0.850	1.0	0.0

1114 rows × 15 columns

Modeling

In [26]:

```
X_train, X_eval, y_train, y_eval = train_test_split(X_train,
y_train, test_size=0.2,random_state=2019)

print("Shape of X_train: ",X_train.shape)

print("Shape of X_eval: ", X_eval.shape)

print("Shape of y_train: ",y_train.shape)

print("Shape of y_eval",y_eval.shape)
```

```
Shape of X_train:  (891, 15)

Shape of X_eval:  (223, 15)

Shape of y_train:  (891,)

Shape of y_eval (223,)
```

In [27]:

```
from sklearn.neighbors import KNeighborsClassifier

from sklearn.linear_model import
LogisticRegression,SGDClassifier,RidgeClassifier

from sklearn.ensemble import
RandomForestClassifier,ExtraTreesClassifier,HistGradientBoostin
gClassifier

from sklearn.naive_bayes import GaussianNB

from sklearn.dummy import DummyClassifier

from sklearn.svm import SVC

from sklearn.ensemble import VotingClassifier

clf1 = SVC()

clf2 = LGBMClassifier()
```

```
clf3 = LogisticRegression()

clf4 = SGDClassifier()

clf5 = XGBClassifier(objective='multi:softmax')

clf6 = KNeighborsClassifier()

clf7 = RandomForestClassifier()

clf8 = ExtraTreesClassifier()

clf9 = HistGradientBoostingClassifier()



eclf = VotingClassifier(estimators=[('svm', clf1), ('LGBM',
clf2), ('Log', clf3), ('SGD', clf4), ('XGBoost', clf5),
('KNeighbors', clf6), ('RandomForest', clf7), ('ExtraTrees',
clf8), ('HistGradientBoosting', clf9)],voting='hard')



for clf, label in zip([clf1, clf2, clf3, clf4,
clf5,clf6,clf7,clf8,clf9, eclf], ['SVC', 'LGBM',
'Log','SGD','XGBoost','KNeighbors','RandomForest','ExtraTrees',
'HistGradientBoosting', 'Ensemble']):

    scores = cross_val_score(clf, X_train, y_train,
scoring='accuracy', cv=5)

    print("Accuracy: %0.2f (+/- %0.2f) [%s]" % (scores.mean(),
```

```
scores.std(), label))
```

```
Accuracy: 0.32 (+/- 0.03) [SVC]
Accuracy: 0.32 (+/- 0.06) [LGBM]
Accuracy: 0.34 (+/- 0.02) [Log]
Accuracy: 0.32 (+/- 0.03) [SGD]
Accuracy: 0.34 (+/- 0.05) [XGBoost]
Accuracy: 0.32 (+/- 0.03) [KNeighbors]
Accuracy: 0.33 (+/- 0.03) [RandomForest]
Accuracy: 0.32 (+/- 0.04) [ExtraTrees]
Accuracy: 0.32 (+/- 0.07) [HistGradientBoosting]
Accuracy: 0.30 (+/- 0.03) [Ensemble]
```

Modeling

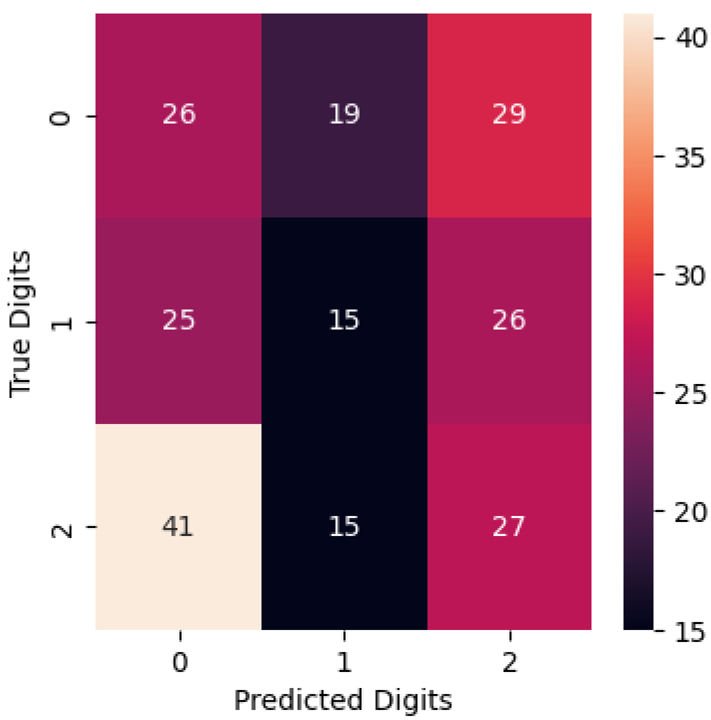
In [28]:

```
clf1 = clf1.fit(X_train, y_train)
clf2 = clf2.fit(X_train, y_train)
```

```
clf3 = clf3.fit(X_train, y_train)

clf4 = clf4.fit(X_train, y_train)

clf5 = clf5.fit(X_train, y_train)

clf6 = clf6.fit(X_train, y_train)

clf7 = clf7.fit(X_train, y_train)

clf8 = clf8.fit(X_train, y_train)

clf9 = clf9.fit(X_train, y_train)



Voting_model = eclf.fit(X_train, y_train)



y_pred_Voting = Voting_model.predict(X_eval) # predict our file
test data

Voting_acc = accuracy_score(y_eval, y_pred_Voting)

print("Voting accuracy is: {0:.3f}%".format(Voting_acc * 100))

cm = confusion_matrix(y_eval, y_pred_Voting)

plt.figure(figsize=(4, 4))

sns.heatmap(cm, annot=True, fmt='.0f')

plt.xlabel("Predicted Digits")
```

```
plt.ylabel("True Digits")

plt.show()
```

```
Voting accuracy is: 30.493%
```



[Reference link]