



Data Collection and Preprocessing Phase

Date	6 th July 2024
Team ID	SWTID1720151909
Project Title	PANIC DISORDER DETECTION
Maximum Marks	6 Marks

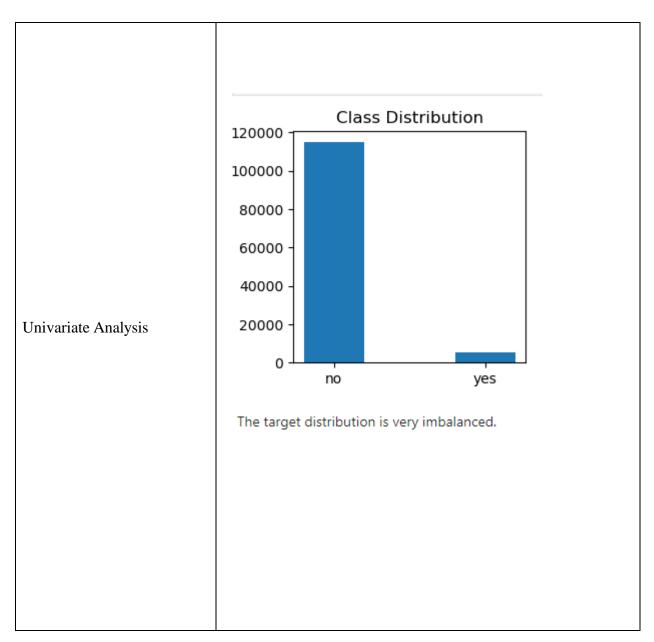
Data Exploration and Preprocessing Template

Identifies data sources, assesses quality issues like missing values and duplicates, and implements resolution plans to ensure accurate and reliable analysis.

Section	Descripti	on							
	Rows - 50	Dimensions: Rows - 5656458, columns - 6 Basic structure of the data.							
		Year	Value						
	count	5.656458e+06	5.656458e+06						
Data Overview	mean	1.994464e+03	1.070501e+12						
	std	1.387895e+01	4.842469e+13						
	min	1.960000e+03	-9.824821e+15						
	25%	1.984000e+03	5.566242e+00						
	50%	1.997000e+03	6.357450e+01						
	75%	2.006000e+03	1.346722e+07						
	max	2.015000e+03	1.103367e+16						

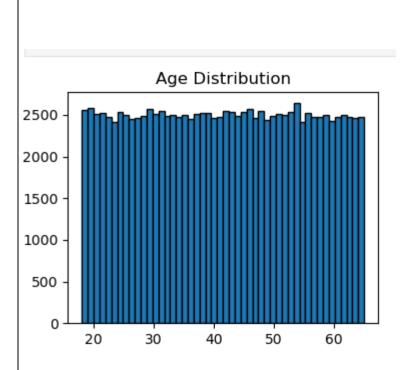








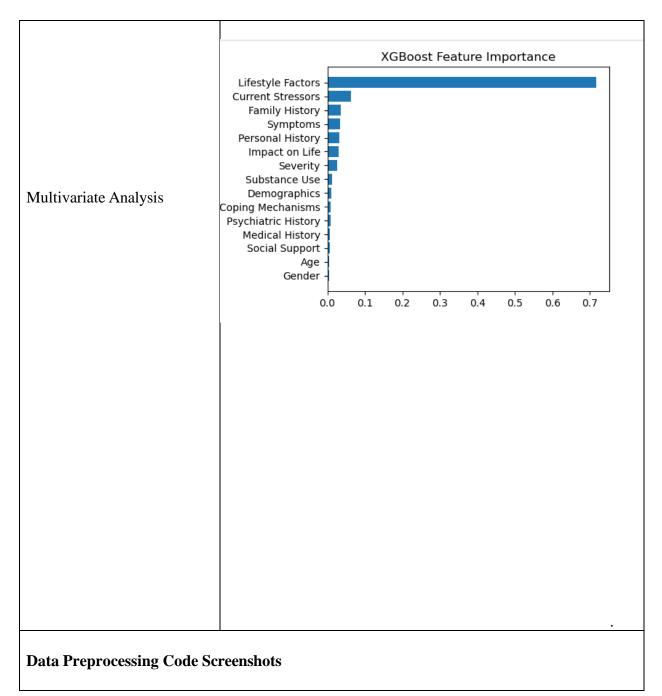




Bivariate Analysis











Loading Data		Participant ID	Age	Gender	Family History	Personal History	Current Stressors	Symptoms	Severity	Impact on Life	Demographics	Medical History	Psychiatri Histor
	7812	7813	46	Male	No	Yes	Low	Shortness of breath	Severe	Significant	Rural	None	Anxiet disorde
	49038	49039	58	Female	Yes	No	Moderate	Chest pain	Severe	Moderate	Urban	Diabetes	Anxiet disorde
	60517	60518	33	Male	No	Yes	Low	Shortness of breath	Mild	Significant	Urban	Diabetes	Anxiet disorde
	29625	29626	22	Male	No	Yes	High	Panic attacks	Severe	Mild	Rural	Diabetes	Depressiv disorde
	61384	61385	62	Female	Yes	No	Low	Panic attacks	Moderate	Significant	Urban	Asthma	Depressiv disorde





```
def columns_summary():
                                                                       col_list = df.columns.to_list()
                                                                      dtype_list = []
                                                                      null_list = []
                                                                      unique_list = []
                                                                      for col in df.columns:
                                                                            dtype_list.append(df[col].dtype)
                                                                            \verb|null_list.append(df[col].isnull().sum())|\\
                                                                            unique_list.append(df[col].nunique())
Handling Missing Data
                                                                      return df_sum.style.hide(axis='index')
                                                                 def update classification():
                                                                      from sklearn.utils.class_weight import compute_sample_weight
                                                                      from sklearn.svm import SVC
                                                                      from sklearn.linear model import SGDClassifier
                                                                     from sklearn.metrics import accuracy_score, f1_score, precision_score, recall_score
from sklearn.metrics import roc_curve, auc, confusion_matrix, matthews_corrcoef
                                                                     def extract_class_weights():
    classes = df['Panic Disorder Diagnosis'].unique()
    weights = compute_sample_weight(class_weight='balanced', y=df['Panic Disorder Diagnosis'])
Data Transformation
                                                                           return dict(zip(classes, np.unique(weights)))
                                                                     class_weights = extract_class_weights()
                                                               Attached codes in the final folder
Feature Engineering
                                                                 def ann_classification():
    from tensorflow.keras.models import Sequential
                                                                      from tensorflow.keras.layers import Dense
                                                                       from tensorflow.keras.metrics import BinaryAccuracy
                                                                       from tensorflow.keras.optimizers import Adam
                                                                      classifier = Sequential(
                                                                               Dense(units=9, kernel_initializer='he_uniform', activation='relu', input_dim=np.shape(X_train)[1]), Dense(units=9, kernel_initializer='he_uniform', activation='relu'), Dense(units=1, kernel_initializer='glorot_uniform', activation='sigmoid')
Save Processed Data
                                                                      adam_opt = Adam(learning_rate=0.001)
classifier.compile(optimizer=adam_opt, loss='binary_crossentropy', metrics=[BinaryAccuracy(name='accuracy')])
                                                                       classifiers.append(code)
                                                                      \label{eq:history} \verb| history = classifier.fit(X_train, y_train, batch_size=100, epochs=100, verbose=0)| \\
                                                                      predictions = classifier.predict(X_test)
                                                                      y_pred = predictions > 0.5
                                                                      from sklearn.metrics import accuracy_score, f1_score, precision_score, recall_score
from sklearn.metrics import roc_curve, auc, confusion_matrix, matthews_corrcoef
                                                                      accuracies[code] = accuracy_score(y_test, y_pred)
f1_scores[code] = f1_score(y_test, y_pred, zero_division=0)
precision_scores[code] = precision_score(y_test, y_pred, zero_division=0)
```