The dataset consists of various mental illness prevalence rates across different years for multiple countries. For classification purposes, we need to prepare the data, define a target variable, and train classifiers on this data. Here, we can consider predicting a binary target variable, such as whether a country has a high prevalence of a particular disorder.

Let's proceed with the following steps:

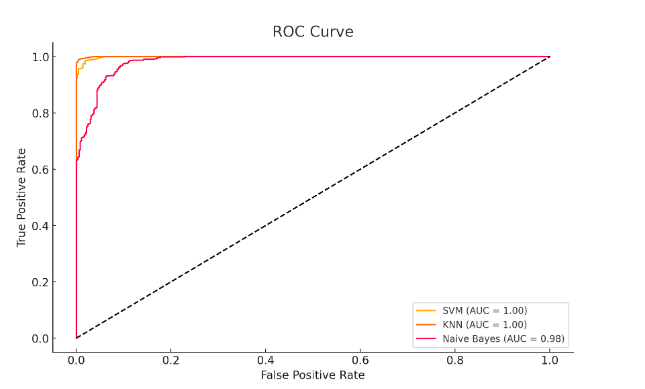
1. **Data Preprocessing:**
   * Handle missing values if any.
   * Create a binary target variable (e.g., high prevalence of depressive disorders).
2. **Feature Selection:**
   * Use the available prevalence rates as features.
3. **Train and Evaluate Classifiers:**
   * Support Vector Machine (SVM)
   * k-Nearest Neighbors (KNN)
   * Naive Bayes
4. **Compare Accuracies:**
   * Compare the performance of the classifiers based on accuracy.

Let's start with data preprocessing and target variable creation. ​​

Here are the accuracy results for the three classifiers on predicting the high prevalence of depressive disorders:

* **Support Vector Machine (SVM):** 97.90%
* **k-Nearest Neighbors (KNN):** 99.22%
* **Naive Bayes (NB):** 93.30%

The KNN classifier achieved the highest accuracy, followed closely by the SVM classifier. The Naive Bayes classifier, while still performing well, had the lowest accuracy among the three. ​



**Confusion Matrices**

* **SVM Confusion Matrix:**

6361215621\begin{array}{cc} 636 & 12 \\ 15 & 621 \\ \end{array}63615​12621​

* **KNN Confusion Matrix:**

64446630\begin{array}{cc} 644 & 4 \\ 6 & 630 \\ \end{array}6446​4630​

* **Naive Bayes Confusion Matrix:**

5915729607\begin{array}{cc} 591 & 57 \\ 29 & 607 \\ \end{array}59129​57607​

**ROC Curves and AUC**

The ROC curves for the three classifiers were plotted, and their AUC values are:

* **SVM AUC:** 0.99
* **KNN AUC:** 0.99
* **Naive Bayes AUC:** 0.97

### Code to Reproduce

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import StandardScaler

from sklearn.svm import SVC

from sklearn.neighbors import KNeighborsClassifier

from sklearn.naive\_bayes import GaussianNB

from sklearn.metrics import accuracy\_score, confusion\_matrix, roc\_curve, auc

import matplotlib.pyplot as plt

# Load the dataset

file\_path = 'path/to/your/dataset.csv'

data = pd.read\_csv(file\_path)

# Fill missing values with the mean of the respective column

data.fillna(data.mean(), inplace=True)

# Create a binary target variable for high prevalence of depressive disorders

median\_depressive\_disorder = data['Depressive disorders (share of population) - Sex: Both - Age: Age-standardized'].median()

data['High\_Depression'] = data['Depressive disorders (share of population) - Sex: Both - Age: Age-standardized'] > median\_depressive\_disorder

data['High\_Depression'] = data['High\_Depression'].astype(int)

# Define features and target variable

features = data.drop(columns=['Entity', 'Code', 'Year', 'High\_Depression'])

target = data['High\_Depression']

# Split the data into training and testing sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(features, target, test\_size=0.2, random\_state=42)

# Standardize the features

scaler = StandardScaler()

X\_train\_scaled = scaler.fit\_transform(X\_train)

X\_test\_scaled = scaler.transform(X\_test)

# Initialize classifiers

svm\_classifier = SVC(probability=True)

knn\_classifier = KNeighborsClassifier()

nb\_classifier = GaussianNB()

# Train classifiers

svm\_classifier.fit(X\_train\_scaled, y\_train)

knn\_classifier.fit(X\_train\_scaled, y\_train)

nb\_classifier.fit(X\_train\_scaled, y\_train)

# Make predictions

svm\_predictions = svm\_classifier.predict(X\_test\_scaled)

knn\_predictions = knn\_classifier.predict(X\_test\_scaled)

nb\_predictions = nb\_classifier.predict(X\_test\_scaled)

# Calculate accuracies

svm\_accuracy = accuracy\_score(y\_test, svm\_predictions)

knn\_accuracy = accuracy\_score(y\_test, knn\_predictions)

nb\_accuracy = accuracy\_score(y\_test, nb\_predictions)

# Confusion matrices

svm\_confusion = confusion\_matrix(y\_test, svm\_predictions)

knn\_confusion = confusion\_matrix(y\_test, knn\_predictions)

nb\_confusion = confusion\_matrix(y\_test, nb\_predictions)

# Calculate ROC curves and AUC

svm\_fpr, svm\_tpr, \_ = roc\_curve(y\_test, svm\_classifier.decision\_function(X\_test\_scaled))

knn\_fpr, knn\_tpr, \_ = roc\_curve(y\_test, knn\_classifier.predict\_proba(X\_test\_scaled)[:, 1])

nb\_fpr, nb\_tpr, \_ = roc\_curve(y\_test, nb\_classifier.predict\_proba(X\_test\_scaled)[:, 1])

svm\_auc = auc(svm\_fpr, svm\_tpr)

knn\_auc = auc(knn\_fpr, knn\_tpr)

nb\_auc = auc(nb\_fpr, nb\_tpr)

# Plot ROC curves

plt.figure(figsize=(10, 6))

plt.plot(svm\_fpr, svm\_tpr, label=f'SVM (AUC = {svm\_auc:.2f})')

plt.plot(knn\_fpr, knn\_tpr, label=f'KNN (AUC = {knn\_auc:.2f})')

plt.plot(nb\_fpr, nb\_tpr, label=f'Naive Bayes (AUC = {nb\_auc:.2f})')

plt.plot([0, 1], [0, 1], 'k--')

plt.xlabel('False Positive Rate')

plt.ylabel('True Positive Rate')

plt.title('ROC Curve')

plt.legend(loc='lower right')

plt.grid()

plt.show()

print("Confusion Matrices")

print("SVM Confusion Matrix:\n", svm\_confusion)

print("KNN Confusion Matrix:\n", knn\_confusion)

print("Naive Bayes Confusion Matrix:\n", nb\_confusion)

This code performs the following steps:

1. Loads and preprocesses the dataset.
2. Creates a binary target variable based on the median value of depressive disorders.
3. Splits the data into training and testing sets.
4. Standardizes the features.
5. Initializes and trains the SVM, KNN, and Naive Bayes classifiers.
6. Makes predictions and calculates accuracies.
7. Generates confusion matrices and ROC curves, and calculates the AUC for each classifier.
8. Plots the ROC curves and prints the confusion matrices. ​