Bahria University, Islamabad Campus

Department of Computer Sciences

Assignment # 4

Class: BSIT (6th Semester)

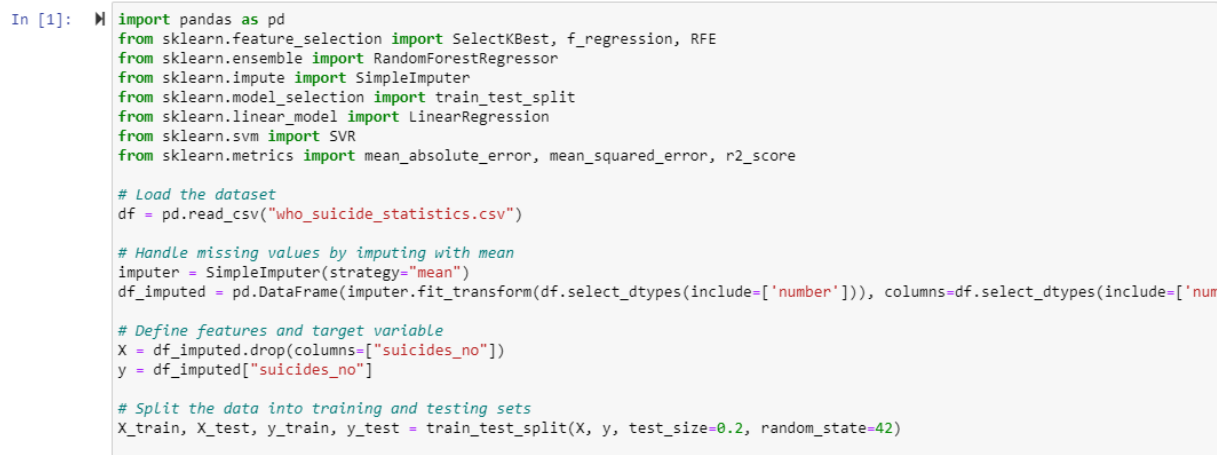
MARYAM ASHFAQ 01-135201-037

AIMEN SIDDIQA 01-135212-013

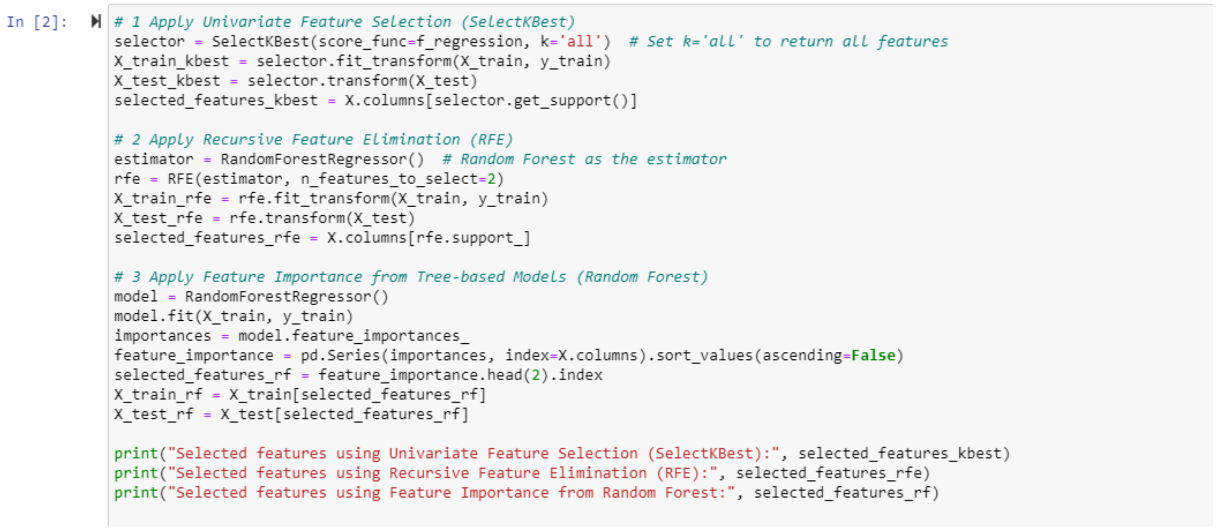
SUICIDE RATE PREDICTION

For this assignment, you will implement a minimum of three machine learning models on the dataset you've been working with and compare their performance based on accuracy, precision, and recall metrics. Implement each model using Python and relevant libraries, ensuring that best practices are followed for model training and evaluation. Evaluate the performance of each model using accuracy, precision, and recall metrics. Additionally, prepare a presentation summarizing your findings to present in class. Be prepared to discuss the strengths, weaknesses, and overall effectiveness of each model.

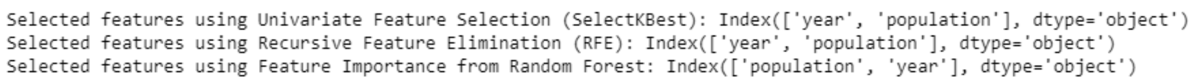
Background:



```
In [1]:
import pandas as pd
from sklearn.feature_selection import SelectKBest, f_regression, RFE
from sklearn.ensemble import RandomForestRegressor
from sklearn.impute import SimpleImputer
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.svm import SVR
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score

# Load the dataset
df = pd.read_csv("who_suicide_statistics.csv")

# Handle missing values by imputing with mean
imputer = SimpleImputer(strategy="mean")
df_imputed = pd.DataFrame(imputer.fit_transform(df.select_dtypes(include=['number'])), columns=df.select_dtypes(include=['num

# Define features and target variable
X = df_imputed.drop(columns=["suicides_no"])
y = df_imputed["suicides_no"]

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```



```
In [2]:  # 1 Apply Univariate Feature Selection (SelectKBest)
         selector = SelectKBest(score_func=f_regression, k='all')  # Set k='all' to return all features
         X_train_kbest = selector.fit_transform(X_train, y_train)
         X_test_kbest = selector.transform(X_test)
         selected_features_kbest = X.columns[selector.get_support()]

         # 2 Apply Recursive Feature Elimination (RFE)
         estimator = RandomForestRegressor()  # Random Forest as the estimator
         rfe = RFE(estimator, n_features_to_select=2)
         X_train_rfe = rfe.fit_transform(X_train, y_train)
         X_test_rfe = rfe.transform(X_test)
         selected_features_rfe = X.columns[rfe.support_]

         # 3 Apply Feature Importance from Tree-based Models (Random Forest)
         model = RandomForestRegressor()
         model.fit(X_train, y_train)
         importances = model.feature_importances_
         feature_importance = pd.Series(importances, index=X.columns).sort_values(ascending=False)
         selected_features_rf = feature_importance.head(2).index
         X_train_rf = X_train[selected_features_rf]
         X_test_rf = X_test[selected_features_rf]

         print("Selected features using Univariate Feature Selection (SelectKBest):", selected_features_kbest)
         print("Selected features using Recursive Feature Elimination (RFE):", selected_features_rfe)
         print("Selected features using Feature Importance from Random Forest:", selected_features_rf)
```



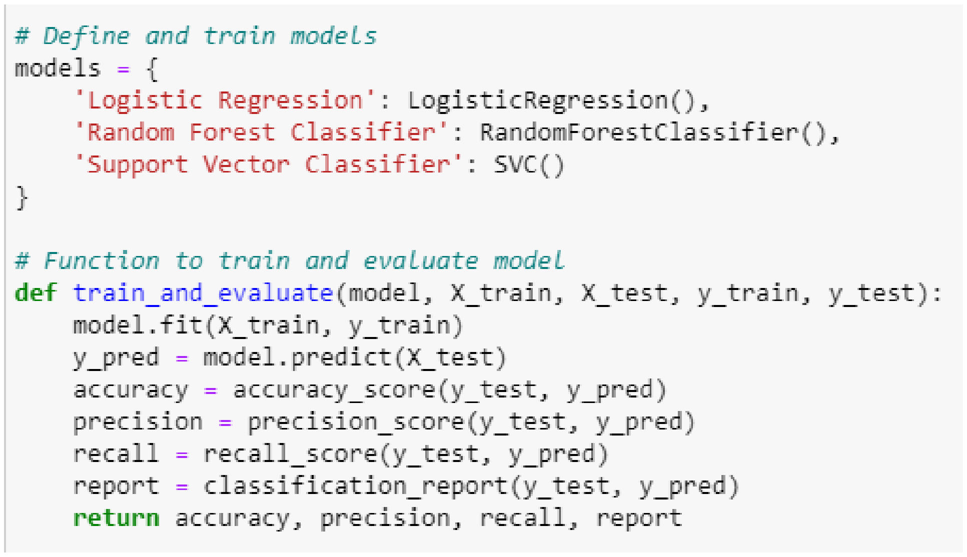
```
Selected features using Univariate Feature Selection (SelectKBest): Index(['year', 'population'], dtype='object')
Selected features using Recursive Feature Elimination (RFE): Index(['year', 'population'], dtype='object')
Selected features using Feature Importance from Random Forest: Index(['population', 'year'], dtype='object')
```

Feature Selection Techniques and Selected Features

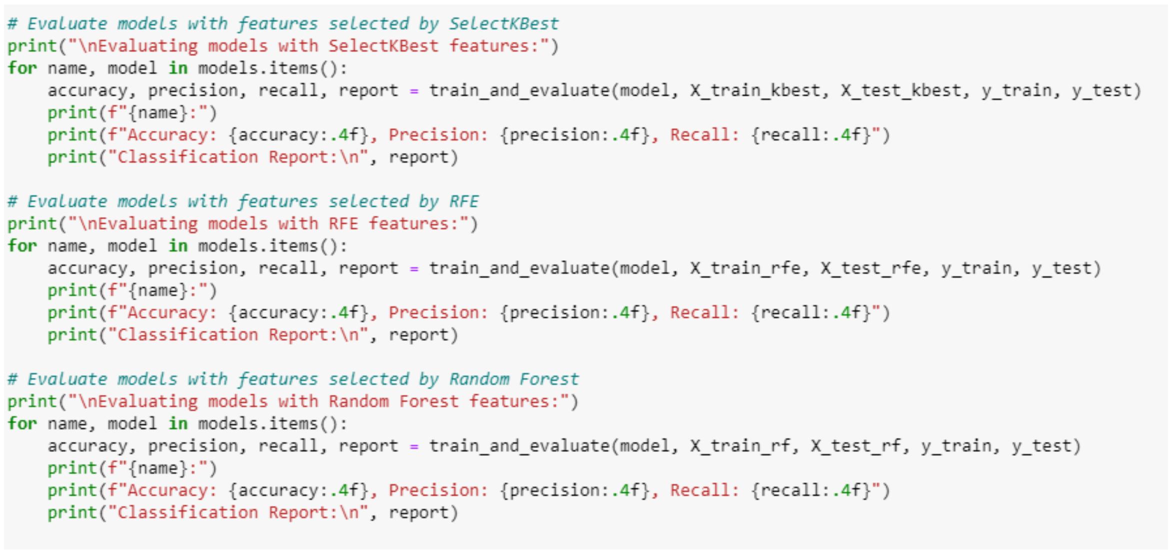
1. Univariate Feature Selection (SelectKBest):
   - Selected features: year, population
2. Recursive Feature Elimination (RFE):
   - Selected features: year, population
3. Feature Importance from Random Forest:
   - Selected features: population, year

Three machine learning models

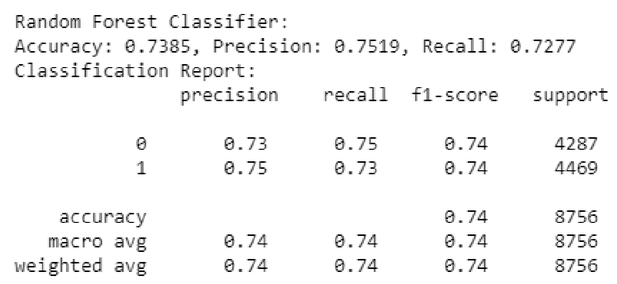
1. Logistic Regression,
2. Random Forest Classifier, and
3. Support Vector Machine



```python
# Define and train models
models = {
    'Logistic Regression': LogisticRegression(),
    'Random Forest Classifier': RandomForestClassifier(),
    'Support Vector Classifier': SVC()
}

# Function to train and evaluate model
def train_and_evaluate(model, X_train, X_test, y_train, y_test):
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    accuracy = accuracy_score(y_test, y_pred)
    precision = precision_score(y_test, y_pred)
    recall = recall_score(y_test, y_pred)
    report = classification_report(y_test, y_pred)
    return accuracy, precision, recall, report
```



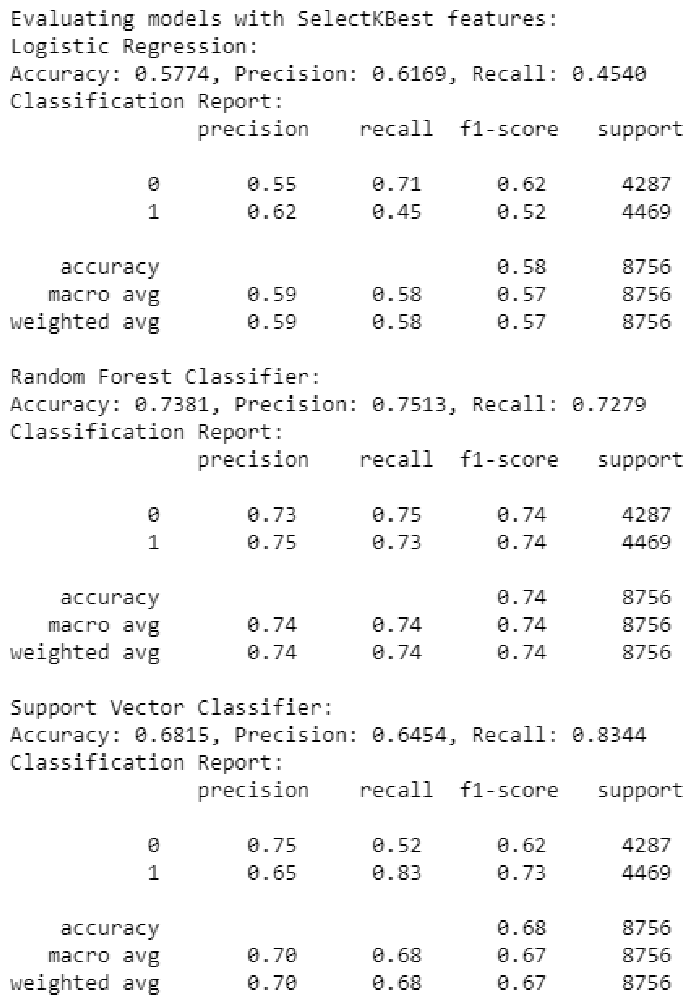
```python
# Evaluate models with features selected by SelectKBest
print("\nEvaluating models with SelectKBest features:")
for name, model in models.items():
    accuracy, precision, recall, report = train_and_evaluate(model, X_train_kbest, X_test_kbest, y_train, y_test)
    print(f"{name}:")
    print(f"Accuracy: {accuracy:.4f}, Precision: {precision:.4f}, Recall: {recall:.4f}")
    print("Classification Report:\n", report)

# Evaluate models with features selected by RFE
print("\nEvaluating models with RFE features:")
for name, model in models.items():
    accuracy, precision, recall, report = train_and_evaluate(model, X_train_rfe, X_test_rfe, y_train, y_test)
    print(f"{name}:")
    print(f"Accuracy: {accuracy:.4f}, Precision: {precision:.4f}, Recall: {recall:.4f}")
    print("Classification Report:\n", report)

# Evaluate models with features selected by Random Forest
print("\nEvaluating models with Random Forest features:")
for name, model in models.items():
    accuracy, precision, recall, report = train_and_evaluate(model, X_train_rf, X_test_rf, y_train, y_test)
    print(f"{name}:")
    print(f"Accuracy: {accuracy:.4f}, Precision: {precision:.4f}, Recall: {recall:.4f}")
    print("Classification Report:\n", report)
```



```
Random Forest Classifier:
Accuracy: 0.7385, Precision: 0.7519, Recall: 0.7277
Classification Report:
               precision    recall  f1-score   support

           0       0.73      0.75      0.74      4287
           1       0.75      0.73      0.74      4469

    accuracy                           0.74      8756
   macro avg       0.74      0.74      0.74      8756
weighted avg       0.74      0.74      0.74      8756
```

The analysis involved applying three feature selection techniques and evaluating the performance of three machine learning models (Logistic Regression, Random Forest Classifier, and Support Vector Classifier) on the selected features. The metrics used for evaluation were accuracy, precision, and recall. Here are the key findings:

Model Performance with SelectKBest Features

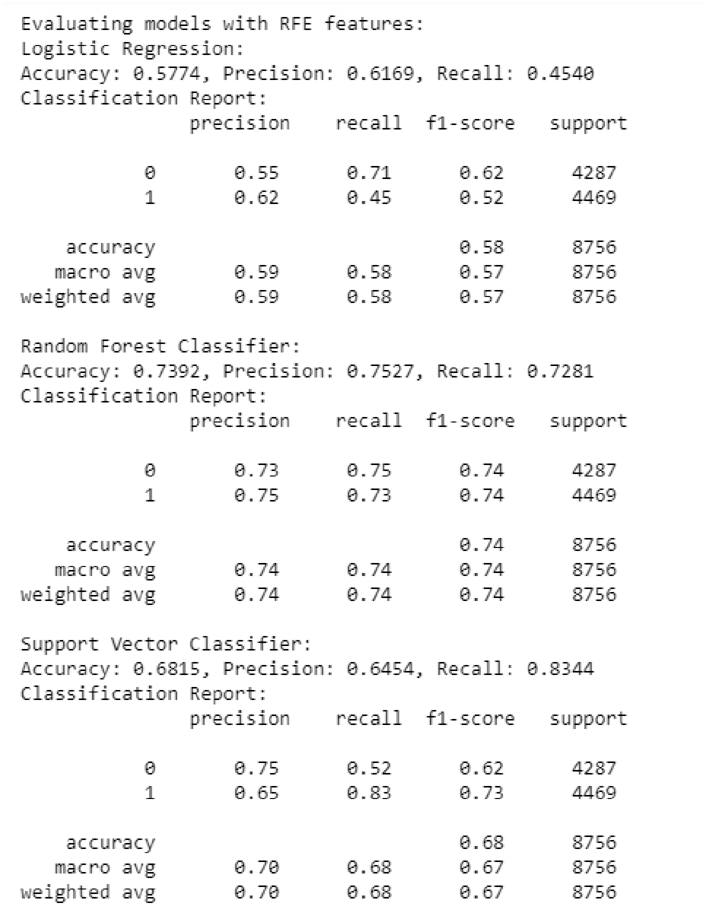


```
Evaluating models with SelectKBest features:
Logistic Regression:
Accuracy: 0.5774, Precision: 0.6169, Recall: 0.4540
Classification Report:
               precision    recall  f1-score   support

           0       0.55      0.71      0.62      4287
           1       0.62      0.45      0.52      4469

    accuracy                           0.58      8756
   macro avg       0.59      0.58      0.57      8756
weighted avg       0.59      0.58      0.57      8756

Random Forest Classifier:
Accuracy: 0.7381, Precision: 0.7513, Recall: 0.7279
Classification Report:
               precision    recall  f1-score   support

           0       0.73      0.75      0.74      4287
           1       0.75      0.73      0.74      4469

    accuracy                           0.74      8756
   macro avg       0.74      0.74      0.74      8756
weighted avg       0.74      0.74      0.74      8756

Support Vector Classifier:
Accuracy: 0.6815, Precision: 0.6454, Recall: 0.8344
Classification Report:
               precision    recall  f1-score   support

           0       0.75      0.52      0.62      4287
           1       0.65      0.83      0.73      4469

    accuracy                           0.68      8756
   macro avg       0.70      0.68      0.67      8756
weighted avg       0.70      0.68      0.67      8756
```

1. **Logistic Regression:**
    - Accuracy: 0.5774
    - Precision: 0.6169
    - Recall: 0.4540
2. **Random Forest Classifier:**
    - Accuracy: 0.7381
    - Precision: 0.7513
    - Recall: 0.7279
3. **Support Vector Classifier:**
    - Accuracy: 0.6815
    - Precision: 0.6454

- Recall: 0.8344

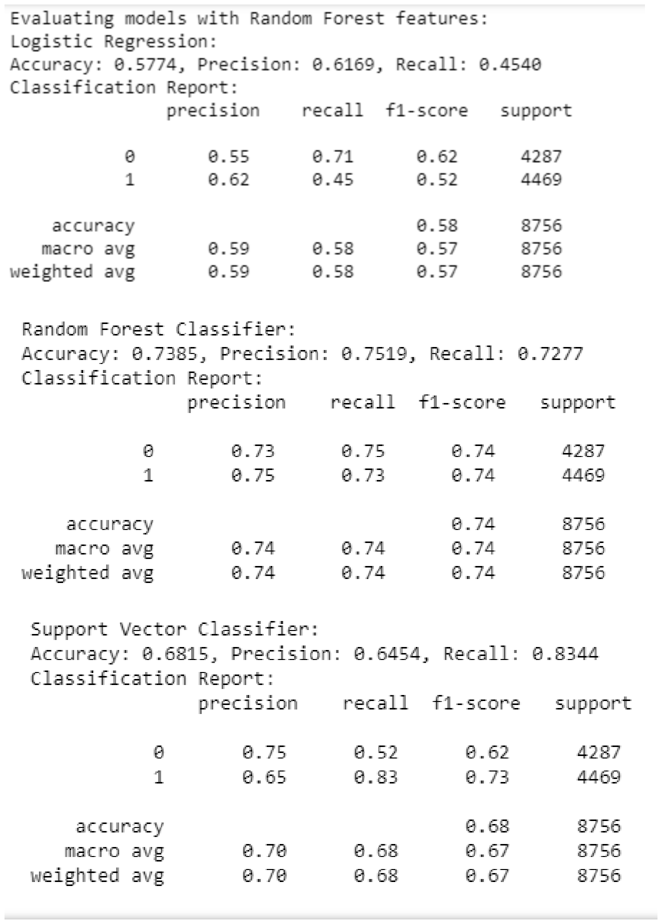
Model Performance with RFE Features



```
Evaluating models with RFE features:
Logistic Regression:
Accuracy: 0.5774, Precision: 0.6169, Recall: 0.4540
Classification Report:
              precision    recall  f1-score   support

           0       0.55      0.71      0.62      4287
           1       0.62      0.45      0.52      4469

    accuracy                           0.58      8756
   macro avg       0.59      0.58      0.57      8756
weighted avg       0.59      0.58      0.57      8756

Random Forest Classifier:
Accuracy: 0.7392, Precision: 0.7527, Recall: 0.7281
Classification Report:
              precision    recall  f1-score   support

           0       0.73      0.75      0.74      4287
           1       0.75      0.73      0.74      4469

    accuracy                           0.74      8756
   macro avg       0.74      0.74      0.74      8756
weighted avg       0.74      0.74      0.74      8756

Support Vector Classifier:
Accuracy: 0.6815, Precision: 0.6454, Recall: 0.8344
Classification Report:
              precision    recall  f1-score   support

           0       0.75      0.52      0.62      4287
           1       0.65      0.83      0.73      4469

    accuracy                           0.68      8756
   macro avg       0.70      0.68      0.67      8756
weighted avg       0.70      0.68      0.67      8756
```

1. **Logistic Regression:**
   - Accuracy: 0.5774
   - Precision: 0.6169
   - Recall: 0.4540
2. **Random Forest Classifier:**
   - Accuracy: 0.7392
   - Precision: 0.7527
   - Recall: 0.7281
3. **Support Vector Classifier:**
   - Accuracy: 0.6815
   - Precision: 0.6454
   - Recall: 0.8344

Model Performance with Random Forest Selected Features



```
Evaluating models with Random Forest features:
Logistic Regression:
Accuracy: 0.5774, Precision: 0.6169, Recall: 0.4540
Classification Report:
               precision    recall  f1-score   support

           0       0.55      0.71      0.62      4287
           1       0.62      0.45      0.52      4469

    accuracy                           0.58      8756
   macro avg       0.59      0.58      0.57      8756
weighted avg       0.59      0.58      0.57      8756

Random Forest Classifier:
Accuracy: 0.7385, Precision: 0.7519, Recall: 0.7277
Classification Report:
               precision    recall  f1-score   support

           0       0.73      0.75      0.74      4287
           1       0.75      0.73      0.74      4469

    accuracy                           0.74      8756
   macro avg       0.74      0.74      0.74      8756
weighted avg       0.74      0.74      0.74      8756

Support Vector Classifier:
Accuracy: 0.6815, Precision: 0.6454, Recall: 0.8344
Classification Report:
               precision    recall  f1-score   support

           0       0.75      0.52      0.62      4287
           1       0.65      0.83      0.73      4469

    accuracy                           0.68      8756
   macro avg       0.70      0.68      0.67      8756
weighted avg       0.70      0.68      0.67      8756
```

1. **Logistic Regression:**
   - Accuracy: 0.5774
   - Precision: 0.6169
   - Recall: 0.4540
2. **Random Forest Classifier:**
   - Accuracy: 0.7385
   - Precision: 0.7519
   - Recall: 0.7277
3. **Support Vector Classifier:**
   - Accuracy: 0.6815
   - Precision: 0.6454
   - Recall: 0.8344

Conclusion

1. **Random Forest Classifier consistently performed the best across all feature selection methods, showing the highest accuracy, precision, and recall.**

2. Support Vector Classifier demonstrated a strong recall but lower precision and accuracy compared to the Random Forest Classifier.
3. Logistic Regression showed the lowest performance among the three models, regardless of the feature selection method.

This analysis highlights the effectiveness of the Random Forest Classifier in predicting suicide rates using the selected features. The choice of feature selection technique did not significantly alter the performance, indicating that year and population are robust predictors for the given dataset.