Bahria University, Islamabad Campus

Department of Computer Sciences

Assignment # 2

Class: BSIT (6th Semester)

Roll No: 01-135201-037 **Name: MARYAM ASHFAQ**

Extend your previous work in assignment 1 by applying feature selection techniques to the same dataset you selected. Choose three feature selection algorithms to implement, describing their workings and graphs. Apply these algorithms to the dataset, assess their performance in selecting relevant features, and compare their effectiveness. Submit a detailed report, including code snippets, visualizations, and analysis, following the provided submission guidelines and evaluation criteria.

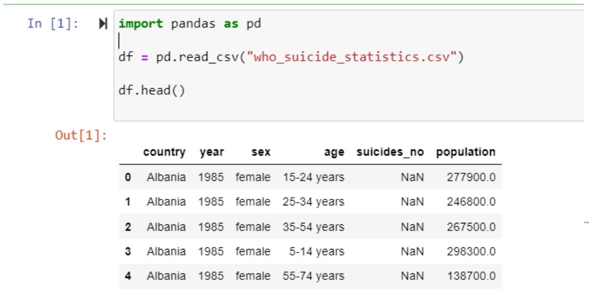
Feature Selection Algorithms for Suicide Rate Prediction

Introduction:

Feature selection is a crucial step in machine learning model development, as it helps identify the most relevant features for predicting the target variable. In this report, we explore three feature selection algorithms applied to a dataset containing information about suicide statistics across different countries. The goal is to assess the effectiveness of these algorithms in selecting relevant features for predicting suicide rates.

Dataset Description:

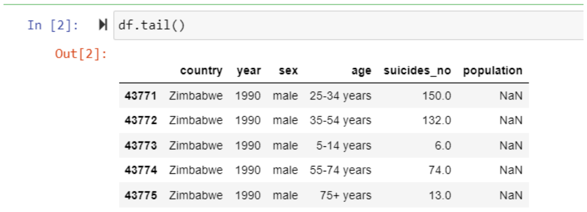
The dataset consists of several columns, including 'country', 'year', 'sex', 'age', 'suicides_no', and 'population'. The target variable we aim to predict is 'suicides_no', representing the number of suicides recorded. The other columns serve as potential features for prediction.



In [1]:
```
import pandas as pd

df = pd.read_csv("who_suicide_statistics.csv")

df.head()
```

Out[1]:

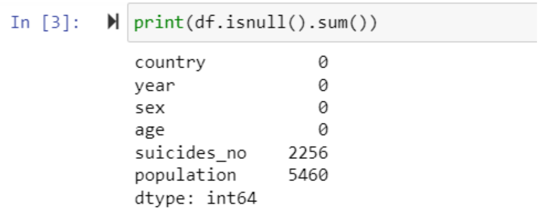
	country	year	sex	age	suicides_no	population
0	Albania	1985	female	15-24 years	NaN	277900.0
1	Albania	1985	female	25-34 years	NaN	246800.0
2	Albania	1985	female	35-54 years	NaN	267500.0
3	Albania	1985	female	5-14 years	NaN	298300.0
4	Albania	1985	female	55-74 years	NaN	138700.0



In [2]:
```
df.tail()
```

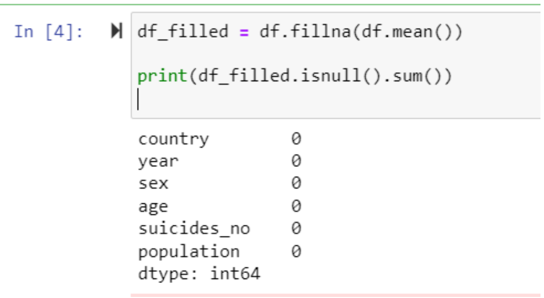
Out[2]:

	country	year	sex	age	suicides_no	population
43771	Zimbabwe	1990	male	25-34 years	150.0	NaN
43772	Zimbabwe	1990	male	35-54 years	132.0	NaN
43773	Zimbabwe	1990	male	5-14 years	6.0	NaN
43774	Zimbabwe	1990	male	55-74 years	74.0	NaN
43775	Zimbabwe	1990	male	75+ years	13.0	NaN



In [3]:
```
print(df.isnull().sum())
```

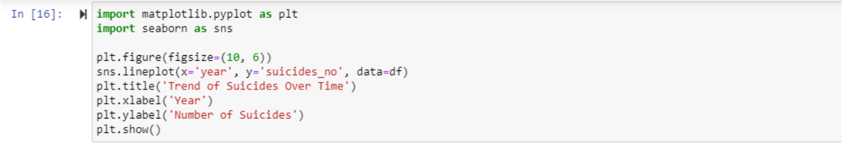
```
country            0
year               0
sex                0
age                0
suicides_no     2256
population      5460
dtype: int64
```



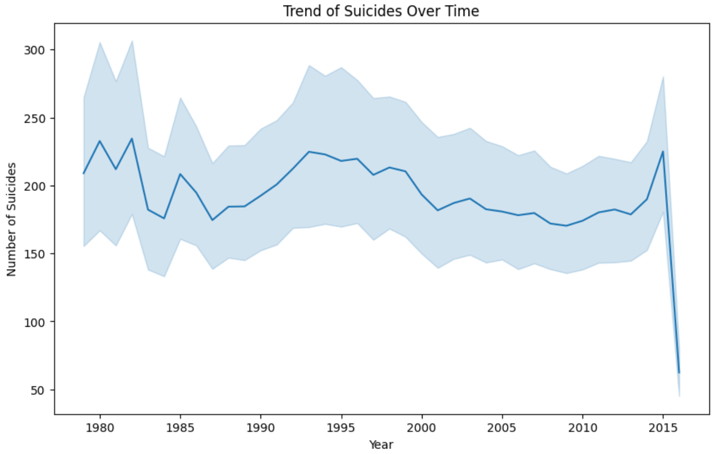
In [4]:
```
df_filled = df.fillna(df.mean())

print(df_filled.isnull().sum())
```

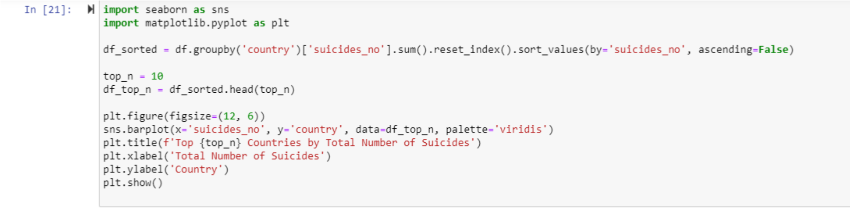
```
country        0
year           0
sex            0
age            0
suicides_no    0
population     0
dtype: int64
```

Dataset Visualization:

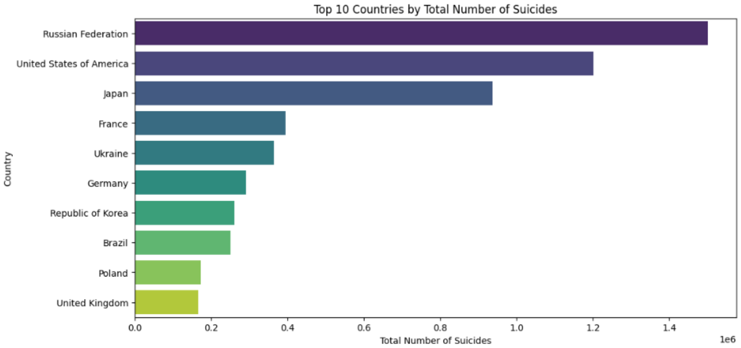


```
In [16]:  import matplotlib.pyplot as plt
          import seaborn as sns

          plt.figure(figsize=(10, 6))
          sns.lineplot(x='year', y='suicides_no', data=df)
          plt.title('Trend of Suicides Over Time')
          plt.xlabel('Year')
          plt.ylabel('Number of Suicides')
          plt.show()
```



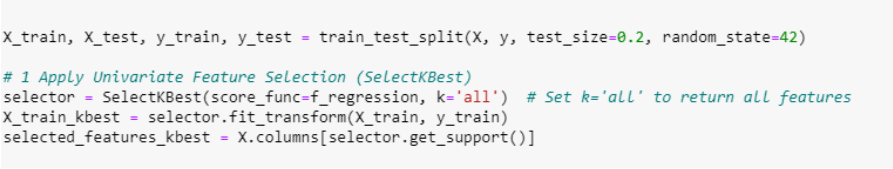


```
In [21]:  import seaborn as sns
          import matplotlib.pyplot as plt

          df_sorted = df.groupby('country')['suicides_no'].sum().reset_index().sort_values(by='suicides_no', ascending=False)

          top_n = 10
          df_top_n = df_sorted.head(top_n)

          plt.figure(figsize=(12, 6))
          sns.barplot(x='suicides_no', y='country', data=df_top_n, palette='viridis')
          plt.title(f'Top {top_n} Countries by Total Number of Suicides')
          plt.xlabel('Total Number of Suicides')
          plt.ylabel('Country')
          plt.show()
```

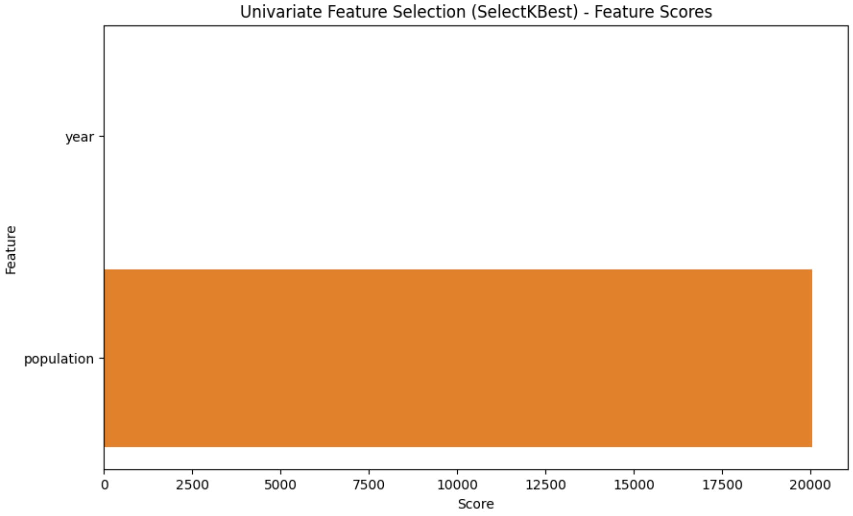


Three feature selection algorithms:

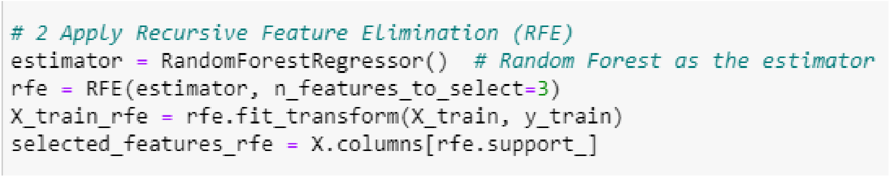
1. **The Univariate Feature Selection (SelectKBest)** method selects the top features based on their individual relationship with the target variable which is number of suicides. Here, it selects all features as 'k' is set to 'all'.



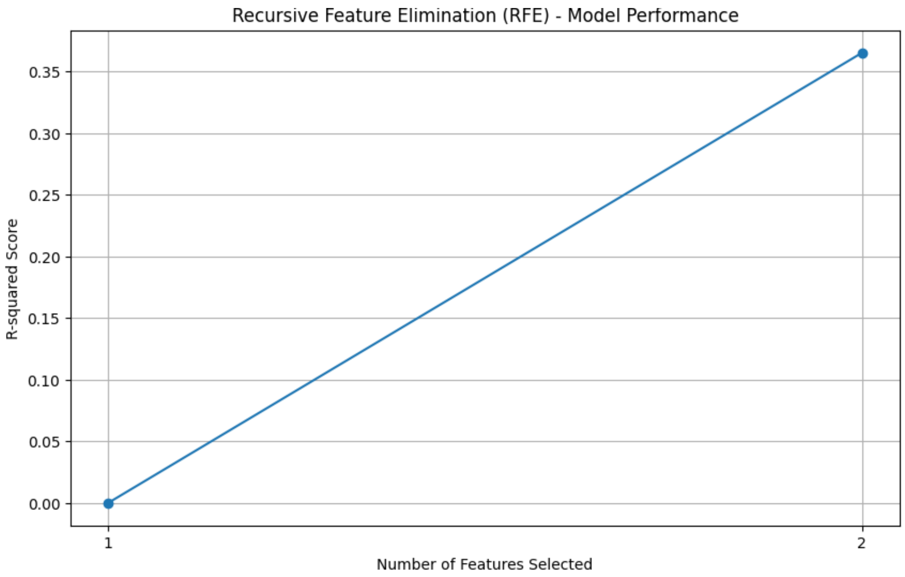
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# 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)
selected_features_kbest = X.columns[selector.get_support()]
```



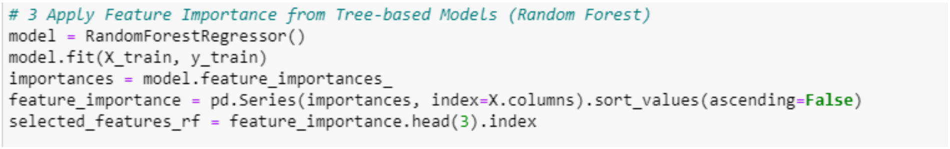
2. **Recursive Feature Elimination (RFE) method** selects features by recursively considering smaller subsets of features. It selects the top features based on the importance of features in the RandomForestRegressor.



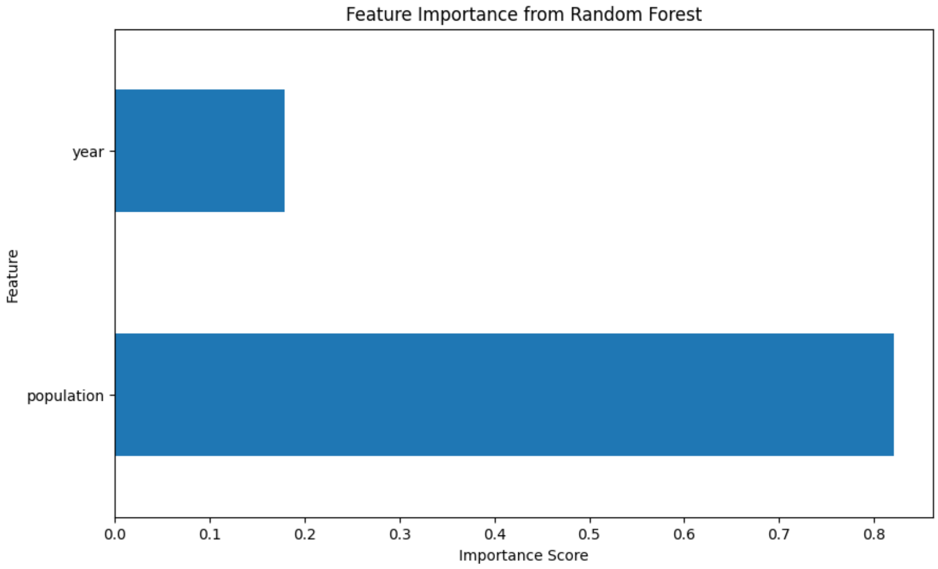
```
# 2 Apply Recursive Feature Elimination (RFE)
estimator = RandomForestRegressor()  # Random Forest as the estimator
rfe = RFE(estimator, n_features_to_select=3)
X_train_rfe = rfe.fit_transform(X_train, y_train)
selected_features_rfe = X.columns[rfe.support_]
```

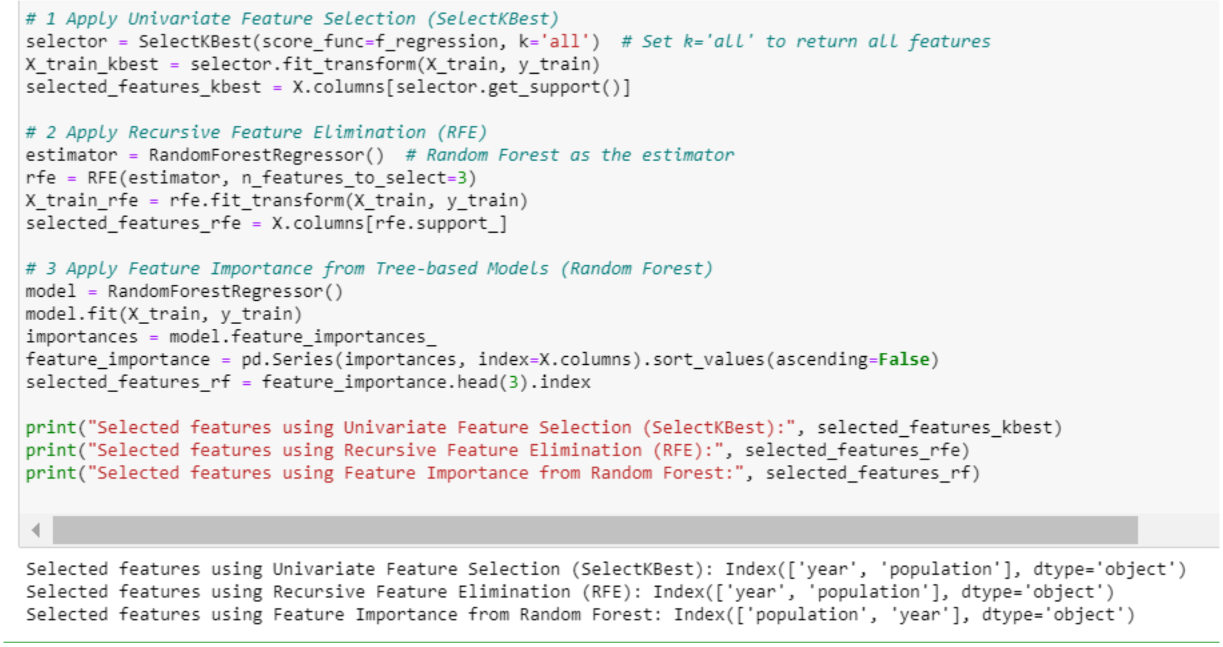


3. **Feature Importance from Random Forest method** ranks the features based on their importance scores from the RandomForestRegressor model. It selects the top features with the highest importance scores.



```
# 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(3).index
```





```
# 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)
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=3)
X_train_rfe = rfe.fit_transform(X_train, y_train)
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(3).index

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')
```

Overall, it appears that 'year' and 'population' are consistently identified as important features across all three techniques. This suggests that these features may have a significant impact on predicting suicide rates. The consistency in feature selection across different methods lends support to the importance of these features in our analysis.