Write a program to learn how to select features for machine learning

In [4]:
```
import numpy as np
import pandas as pd
from sklearn.datasets import load_iris
from sklearn.feature_selection import SelectKBest, f_classif, RFE
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
```

In [5]:
```
# Load dataset
iris = load_iris()
```

In [6]:
```
X = pd.DataFrame(iris.data, columns=iris.feature_names)
y = iris.target
```

In [7]:
```
# Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
```

1. Univariate Feature Selection

What is it?

Univariate Feature Selection means selecting the best features based on individual statistical tests between each feature and the target variable.

```
“Univariate” = each feature is evaluated independently.

This is useful for removing irrelevant or less important features.
```

What is SelectKBest?

SelectKBest is a feature selection method in Scikit-learn that selects the top k features based on a scoring function.

Key Components:

Parameter	Description

Parameter	Description
`score_func`	The scoring function to evaluate each feature (e.g., `f_classif`, `chi2`)
`k`	Number of top features to keep

Common Scoring Functions:

Function	Use Case	Description
`f_classif`	Classification	ANOVA F-value between label/feature
`chi2`	Classification (non-negative features)	Chi-squared stats between each feature and target
`mutual_info_classif`	Classification	Information gain between each feature and target
`f_regression`	Regression	F-value for regression tasks
`mutual_info_regression`	Regression	Mutual information for regression tasks

In [8]:
```
# 1. Univariate Feature Selection
print("Univariate Feature Selection:")
# Apply SelectKBest
selector = SelectKBest(score_func=f_classif, k=2)
X_new = selector.fit_transform(X_train, y_train)
X_new
```

```
Univariate Feature Selection:
```

```
Out[8]: array([[3.7, 1. ],
               [5.1, 1.5],
               [5.5, 1.8],
               [4.4, 1.4],
               [6.1, 2.5],
               [4.2, 1.3],
               [6.6, 2.1],
               [4.5, 1.5],
               [1.4, 0.2],
               [6.7, 2. ],
               [4.1, 1. ],
               [1.4, 0.2],
               [1.3, 0.3],
               [1.9, 0.4],
               [3.5, 1. ],
               [4.9, 1.8],
               [1.9, 0.2],
               [1.6, 0.2],
               [1.7, 0.5],
               [4.2, 1.3],
               [1.5, 0.2],
               [4.2, 1.2],
               [6.7, 2.2],
               [1.4, 0.2],
               [4.3, 1.3],
               [5. , 2. ],
               [1.4, 0.2],
               [4.8, 1.8],
               [5.1, 1.9],
               [4. , 1. ],
               [4.5, 1.5],
               [5.4, 2.3],
               [4. , 1.3],
               [1.7, 0.4],
               [3.3, 1. ],
               [5.3, 1.9],
               [1.4, 0.2],
               [1.2, 0.2],
               [3.8, 1.1],
               [5. , 1.7],
               [1.5, 0.2],
               [5.1, 2.4],
```

```
       [1.5, 0.2],
       [1.6, 0.6],
       [4.8, 1.8],
       [3. , 1.1],
       [5.7, 2.3],
       [5.1, 1.6],
       [5.6, 1.4],
       [6.1, 2.3],
       [4. , 1.3],
       [1.4, 0.2],
       [1.1, 0.1],
       [5. , 1.5],
       [6. , 1.8],
       [1.5, 0.2],
       [1.4, 0.3],
       [1.3, 0.2],
       [4.9, 1.5],
       [5.6, 2.4],
       [1.4, 0.3],
       [5.5, 2.1],
       [6. , 2.5],
       [1.3, 0.2],
       [4.7, 1.4],
       [4.6, 1.5],
       [4.8, 1.8],
       [4.7, 1.4],
       [5.3, 2.3],
       [1.6, 0.2],
       [5.4, 2.1],
       [4.2, 1.5],
       [5.2, 2. ],
       [3.5, 1. ],
       [3.9, 1.4],
       [4.6, 1.4],
       [1.3, 0.3],
       [4.6, 1.3],
       [4.4, 1.2],
       [1.5, 0.2],
       [4.1, 1.3],
       [6.3, 1.8],
       [5.7, 2.1],
       [1.5, 0.4],
```

```
       [3.3, 1. ],
       [5.7, 2.5],
       [5.8, 1.6],
       [1.4, 0.1],
       [5.6, 2.4],
       [1.4, 0.2],
       [4.9, 1.5],
       [6.1, 1.9],
       [5.6, 1.8],
       [4.1, 1.3],
       [5.5, 1.8],
       [4.4, 1.3],
       [4.3, 1.3],
       [4.9, 2. ],
       [5.1, 1.8],
       [1.7, 0.2],
       [4. , 1.3],
       [4.5, 1.7],
       [1.2, 0.2],
       [4. , 1.2],
       [5.9, 2.1]])
```

In [9]:

```
# Display scores and selected features
print("Feature scores:", selector.scores_)
print("Selected features:", X_train.columns[selector.get_support()])
```

```
Feature scores: [ 74.7572012   33.41979913 713.45534904 526.54162416]
Selected features: Index(['petal length (cm)', 'petal width (cm)'], dtype='object')
```

2. Recursive Feature Elimination (RFE)

To select the best subset of features for your model by recursively removing the least important ones.

How it works (Step-by-Step):

```
Train a model (e.g., logistic regression, decision tree) on all features.

Rank features based on importance (e.g., model coefficients or feature_importances).

Remove the least important feature(s).
```

Repeat the process on the remaining features until:

You reach the desired number of features (n_features_to_select).

In [10]:
```
from sklearn.ensemble import RandomForestClassifier
model = RandomForestClassifier()
model.fit(X_train, y_train)
```

Out[10]:
```
▾ RandomForestClassifier

RandomForestClassifier()
```

In [11]:
```
# 2. Recursive Feature Elimination (RFE)
print("\nRecursive Feature Elimination (RFE):")
rfe = RFE(estimator=model, n_features_to_select=2)
rfe.fit(X_train, y_train)
# Display selected features
print("Selected features:", X_train.columns[rfe.support_])
```

```
Recursive Feature Elimination (RFE):
Selected features: Index(['petal length (cm)', 'petal width (cm)'], dtype='object')
```

3. Feature Importance from Random Forest

In [12]:
```
print("\nFeature Importance from Random Forest:")
# Display feature importance
importances = model.feature_importances_
indices = np.argsort(importances)[::-1]
print("Feature ranking:")
for f in range(X_train.shape[1]):
    print(f"{X_train.columns[indices[f]]}: {importances[indices[f]]}")
```

```
Feature Importance from Random Forest:
Feature ranking:
petal width (cm): 0.4330064240493466
petal length (cm): 0.3955833154085587
sepal length (cm): 0.12929497833439274
sepal width (cm): 0.04211528220770196
```