In [1]:
```
# 🔹 PRACTICAL 4 – Feature Selection & Data Partitioning
```

In [ ]:
```
##Name: Ritika R. Junekar
#Sub: PD
#Roll_No:29
```

In [ ]:
```
#Part 1: SelectKBest example
```

In [3]:
```
from sklearn.datasets import load_iris
from sklearn.feature_selection import SelectKBest, f_classif
```

In [5]:
```
iris = load_iris()
X = iris.data
y = iris.target
```

In [7]:
```
sel = SelectKBest(score_func=f_classif, k=2)
X_new = sel.fit_transform(X, y)
selected_indices = sel.get_support(indices=True)
```

In [9]:
```
print("Selected feature indices:", selected_indices)
print("Selected feature names:", [iris.feature_names[i] for i in selected_indi
```

```
Selected feature indices: [2 3]
Selected feature names: ['petal length (cm)', 'petal width (cm)']
```

In [11]:
```
#4.2 Correlation + train-test split + LinearRegression
```

In [13]:
```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
```

In [15]:
```
np.random.seed(42)
X1 = np.random.rand(100) * 10
X2 = 2 * X1 + np.random.randn(100) * 0.5  # highly correlated with X1
X3 = np.random.rand(100) * 5
y = 5 + 1.5 * X1 - 2 * X3 + np.random.randn(100) * 2
```

In [17]:
```
df = pd.DataFrame({
    'Feature_X1': X1,
    'Feature_X2': X2,
    'Feature_X3': X3,
    'Target_Y': y
})
```

In [19]:
```
print("--- Original Data Shape ---")
print(df.shape)
```

```
--- Original Data Shape ---
(100, 4)
```

In [21]:
```
print("\n--- Correlation Matrix (for identifying highly correlated features) -
correlation_matrix = df.corr()
print(correlation_matrix)
```

```
--- Correlation Matrix (for identifying highly correlated features) ---
            Feature_X1  Feature_X2  Feature_X3  Target_Y
Feature_X1    1.000000    0.997069   -0.028591  0.811109
Feature_X2    0.997069    1.000000   -0.021861  0.801174
Feature_X3   -0.028591   -0.021861    1.000000 -0.511236
Target_Y      0.811109    0.801174   -0.511236  1.000000
```

In [23]:
```
# Choose non-redundant features
features = ['Feature_X1', 'Feature_X3']
X = df[features]
y = df['Target_Y']
```

In [25]:
```
# Train-test split (70/30)
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.3, random_state=42
)
```

In [27]:
```
print("\n--- Data Partitioning Results ---")
print(f"Total Samples: {len(df)}")
print(f"X_train shape: {X_train.shape}")
print(f"X_test shape: {X_test.shape}")
print(f"y_train shape: {y_train.shape}")
print(f"y_test shape: {y_test.shape}")
```

```
--- Data Partitioning Results ---
Total Samples: 100
X_train shape: (70, 2)
X_test shape: (30, 2)
y_train shape: (70,)
y_test shape: (30,)
```

In [29]:
```
# Quick model check
model = LinearRegression()
model.fit(X_train, y_train)
print(f"\nModel R-squared on Test set: {model.score(X_test, y_test):.4f}")
```

```
Model R-squared on Test set: 0.9302
```

In [ ]: