In [1]:
```
!pip install pandas scikit-learn
```

```
Requirement already satisfied: pandas in /home/sargam/.conda/envs/myenv/li
b/python3.11/site-packages (2.2.3)
Requirement already satisfied: scikit-learn in /home/sargam/.conda/envs/my
env/lib/python3.11/site-packages (1.6.1)
Requirement already satisfied: numpy>=1.23.2 in /home/sargam/.conda/envs/m
yenv/lib/python3.11/site-packages (from pandas) (2.0.1)
Requirement already satisfied: python-dateutil>=2.8.2 in /home/sargam/.con
da/envs/myenv/lib/python3.11/site-packages (from pandas) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in /home/sargam/.conda/envs/my
env/lib/python3.11/site-packages (from pandas) (2024.1)
Requirement already satisfied: tzdata>=2022.7 in /home/sargam/.conda/envs/
myenv/lib/python3.11/site-packages (from pandas) (2025.2)
Requirement already satisfied: scipy>=1.6.0 in /home/sargam/.conda/envs/my
env/lib/python3.11/site-packages (from scikit-learn) (1.15.2)
Requirement already satisfied: joblib>=1.2.0 in /home/sargam/.conda/envs/m
yenv/lib/python3.11/site-packages (from scikit-learn) (1.5.0)
Requirement already satisfied: threadpoolctl>=3.1.0 in /home/sargam/.cond
a/envs/myenv/lib/python3.11/site-packages (from scikit-learn) (3.6.0)
Requirement already satisfied: six>=1.5 in /home/sargam/.conda/envs/myenv/
lib/python3.11/site-packages (from python-dateutil>=2.8.2->pandas) (1.17.
0)
```

In [2]:
```
# -------------------------------------
# 2. Import Required Libraries
# -------------------------------------
import pandas as pd
from sklearn import preprocessing
from sklearn.datasets import load_iris
```

In [3]:
```
# -------------------------------------
# 3. Dataset Description and Source
# -------------------------------------
"""
Dataset: Iris Flower Dataset
Description: This dataset consists of 150 samples from three species of I
(setosa, versicolor, virginica). Each sample contains four features:
sepal length, sepal width, petal length, and petal width.

Source URL: https://archive.ics.uci.edu/ml/datasets/iris
"""
```

Out[3]:
```
'\nDataset: Iris Flower Dataset\nDescription: This dataset consists of 1
50 samples from three species of Iris flowers \n(setosa, versicolor, vir
ginica). Each sample contains four features: \nsepal length, sepal widt
h, petal length, and petal width.\n\nSource URL: https://archive.ics.uc
i.edu/ml/datasets/iris\n'
```

In [4]:
```
# -------------------------------------
# 4. Load Dataset into Pandas DataFrame
# -------------------------------------
iris = load_iris()
df = pd.DataFrame(iris.data, columns=iris.feature_names)
df['Species'] = pd.Categorical.from_codes(iris.target, iris.target_names)

print(" 🔹 First 5 rows of the Iris Dataset:")
print(df.head())
```

```
🔹 First 5 rows of the Iris Dataset:
   sepal length (cm)  sepal width (cm)  petal length (cm)  petal width (c
m)  \
0                5.1               3.5                1.4               0.
2
1                4.9               3.0                1.4               0.
2
2                4.7               3.2                1.3               0.
2
3                4.6               3.1                1.5               0.
2
4                5.0               3.6                1.4               0.
2

  Species
0  setosa
1  setosa
2  setosa
3  setosa
4  setosa
```

In [5]:

```python
# ------------------------------------
# 5. Initial Exploration and Preprocessing
# ------------------------------------
print("\n🔍 Dataset Info:")
print(df.info())
```

```
🔍 Dataset Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   sepal length (cm)  150 non-null    float64
 1   sepal width (cm)   150 non-null    float64
 2   petal length (cm)  150 non-null    float64
 3   petal width (cm)   150 non-null    float64
 4   Species            150 non-null    category
dtypes: category(1), float64(4)
memory usage: 5.1 KB
None
```

In [6]:

```python
print("\n📊 Dataset Statistics:")
print(df.describe())
```

```
📊 Dataset Statistics:
       sepal length (cm)  sepal width (cm)  petal length (cm)  \
count         150.000000        150.000000         150.000000
mean            5.843333          3.057333           3.758000
std             0.828066          0.435866           1.765298
min             4.300000          2.000000           1.000000
25%             5.100000          2.800000           1.600000
50%             5.800000          3.000000           4.350000
75%             6.400000          3.300000           5.100000
max             7.900000          4.400000           6.900000

       petal width (cm)
count        150.000000
mean           1.199333
std            0.762238
min            0.100000
25%            0.300000
50%            1.300000
75%            1.800000
max            2.500000
```

In [7]:
```
print("\n❓ Missing Values:")
print(df.isnull().sum())
```

```
❓ Missing Values:
sepal length (cm)    0
sepal width (cm)     0
petal length (cm)    0
petal width (cm)     0
Species              0
dtype: int64
```

In [8]:
```
print(df['Species'].unique())
```

```
['setosa', 'versicolor', 'virginica']
Categories (3, object): ['setosa', 'versicolor', 'virginica']
```

In [9]:
```
# -------------------------------------
# 6. Check and Convert Data Types if Needed
# -------------------------------------
print("\nData Types:")
print(df.dtypes)
```

```
Data Types:
sepal length (cm)     float64
sepal width (cm)      float64
petal length (cm)     float64
petal width (cm)      float64
Species              category
dtype: object
```

In [10]:
```
# -------------------------------------
# 7. Normalize Numeric Features (Min-Max Scaling)
# -------------------------------------
min_max_scaler = preprocessing.MinMaxScaler()
x = df.iloc[:, :-1]  # Only numeric columns
x_scaled = min_max_scaler.fit_transform(x)
```

In [11]:
```
df_normalized = pd.DataFrame(x_scaled, columns=iris.feature_names)
print("\n✅ Normalized Dataset Preview:")
print(df_normalized.head())
```

```
✅ Normalized Dataset Preview:
   sepal length (cm)  sepal width (cm)  petal length (cm)  petal width (c
m)
0           0.222222          0.625000           0.067797          0.04166
7
1           0.166667          0.416667           0.067797          0.04166
7
2           0.111111          0.500000           0.050847          0.04166
7
3           0.083333          0.458333           0.084746          0.04166
7
4           0.194444          0.666667           0.067797          0.04166
7
```

In [12]:
```python
print("\nUnique values in Species column before Label Encoding:")
print(df['Species'].unique())
```

```
Unique values in Species column before Label Encoding:
['setosa', 'versicolor', 'virginica']
Categories (3, object): ['setosa', 'versicolor', 'virginica']
```

In [15]:
```python
# -------------------------------------
# 8. Label Encoding (Convert Species to Numeric)
# -------------------------------------
label_encoder = preprocessing.LabelEncoder()
df['Species_Label'] = label_encoder.fit_transform(df['Species'])

print("\nSpecies After Label Encoding:")
print(df['Species_Label'].unique())
```

```
Species After Label Encoding:
[0 1 2]
```

In [17]:
```python
# -------------------------------------
# 9. One-Hot Encoding
# -------------------------------------
print("\n🔍 One-Hot Encoding Preview:")
enc = preprocessing.OneHotEncoder(sparse_output=False)
enc_df = pd.DataFrame(enc.fit_transform(df[['Species_Label']]))
enc_df.columns = ['Iris-Setosa', 'Iris-Versicolor', 'Iris-Virginica']
df_one_hot = df.drop(columns=['Species', 'Species_Label']).join(enc_df)
print(df_one_hot.head())
```

```
🔍 One-Hot Encoding Preview:
   sepal length (cm)  sepal width (cm)  petal length (cm)  petal width (c
m)  \
0                5.1               3.5                1.4               0.
2
1                4.9               3.0                1.4               0.
2
2                4.7               3.2                1.3               0.
2
3                4.6               3.1                1.5               0.
2
4                5.0               3.6                1.4               0.
2

   Iris-Setosa  Iris-Versicolor  Iris-Virginica
0          1.0              0.0             0.0
1          1.0              0.0             0.0
2          1.0              0.0             0.0
3          1.0              0.0             0.0
4          1.0              0.0             0.0
```

In [18]:
```
df['Species'] = label_encoder.fit_transform(df['Species'])
print("\nUnique values in Species column after Label Encoding:")
print(df['Species'].unique())
```

```
Unique values in Species column after Label Encoding:
[0 1 2]
```

In [19]:
```
df_dummy_encoded = pd.get_dummies(df, drop_first=True)
print("\nDummy Encoded Iris Dataset:")
print(df_dummy_encoded)
```

```
Dummy Encoded Iris Dataset:
     sepal length (cm)  sepal width (cm)  petal length (cm)  petal width
(cm)  \
0                  5.1               3.5                1.4
0.2
1                  4.9               3.0                1.4
0.2
2                  4.7               3.2                1.3
0.2
3                  4.6               3.1                1.5
0.2
4                  5.0               3.6                1.4
0.2
..                 ...               ...                ...
...
145                6.7               3.0                5.2
2.3
146                6.3               2.5                5.0
1.9
147                6.5               3.0                5.2
2.0
148                6.2               3.4                5.4
2.3
149                5.9               3.0                5.1
1.8

     Species  Species_Label
0          0              0
1          0              0
2          0              0
3          0              0
4          0              0
..       ...            ...
145        2              2
146        2              2
147        2              2
148        2              2
149        2              2

[150 rows x 6 columns]
```

In [ ]: