In [9]:

```
from sklearn.datasets import load_iris
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report,confusion_matrix
```

In [2]:

```
# Load Iris dataset
iris = load_iris()
```

In [11]:

```
iris.feature_names
```

Out[11]:

```
['sepal length (cm)',
 'sepal width (cm)',
 'petal length (cm)',
 'petal width (cm)']
```

In [12]:

```
iris.target
```

Out[12]:

```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2])
```

In [13]:

```
iris.data
```

Out[13]:

```
array([[5.1, 3.5, 1.4, 0.2],
       [4.9, 3. , 1.4, 0.2],
       [4.7, 3.2, 1.3, 0.2],
       [4.6, 3.1, 1.5, 0.2],
       [5. , 3.6, 1.4, 0.2],
       [5.4, 3.9, 1.7, 0.4],
       [4.6, 3.4, 1.4, 0.3],
       [5. , 3.4, 1.5, 0.2],
       [4.4, 2.9, 1.4, 0.2],
       [4.9, 3.1, 1.5, 0.1],
       [5.4, 3.7, 1.5, 0.2],
       [4.8, 3.4, 1.6, 0.2],
       [4.8, 3. , 1.4, 0.1],
       [4.3, 3. , 1.1, 0.1],
       [5.8, 4. , 1.2, 0.2],
       [5.7, 4.4, 1.5, 0.4],
       [5.4, 3.9, 1.3, 0.4],
       [5.1, 3.5, 1.4, 0.3],
```

In [3]:

```
# Split dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(iris.data, iris.target, test_size=0.
```

In [4]:

```
# Train k-NN classifier
knn = KNeighborsClassifier(n_neighbors=3)
knn.fit(X_train, y_train)
```

Out[4]:

```
KNeighborsClassifier
KNeighborsClassifier(n_neighbors=3)
```

In [5]:

```
# Test k-NN classifier
y_pred = knn.predict(X_test)
```

```
C:\Users\khana\miniconda3\envs\ML_Experiments\lib\site-packages\sklearn\ne
ighbors\_classification.py:237: FutureWarning: Unlike other reduction func
tions (e.g. `skew`, `kurtosis`), the default behavior of `mode` typically
preserves the axis it acts along. In SciPy 1.11.0, this behavior will chan
ge: the default value of `keepdims` will become False, the `axis` over whi
ch the statistic is taken will be eliminated, and the value None will no l
onger be accepted. Set `keepdims` to True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
```

In [6]:

```
# Print classification report
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        19
           1       1.00      1.00      1.00        13
           2       1.00      1.00      1.00        13

    accuracy                           1.00        45
   macro avg       1.00      1.00      1.00        45
weighted avg       1.00      1.00      1.00        45
```

In [7]:

```
for i in range(len(y_test)):
    if y_test[i] == y_pred[i]:
        print("Correct prediction: ", iris.target_names[y_pred[i]])
    else:
        print("Wrong prediction: ", iris.target_names[y_pred[i]])
```

```
Correct prediction:  versicolor
Correct prediction:  setosa
Correct prediction:  virginica
Correct prediction:  versicolor
Correct prediction:  versicolor
Correct prediction:  setosa
Correct prediction:  versicolor
Correct prediction:  virginica
Correct prediction:  versicolor
Correct prediction:  versicolor
Correct prediction:  virginica
Correct prediction:  setosa
Correct prediction:  setosa
Correct prediction:  setosa
Correct prediction:  setosa
Correct prediction:  versicolor
Correct prediction:  virginica
Correct prediction:  versicolor
Correct prediction:  versicolor
Correct prediction:  virginica
Correct prediction:  setosa
Correct prediction:  virginica
Correct prediction:  setosa
Correct prediction:  virginica
Correct prediction:  virginica
Correct prediction:  virginica
Correct prediction:  virginica
Correct prediction:  virginica
Correct prediction:  setosa
Correct prediction:  setosa
Correct prediction:  setosa
Correct prediction:  setosa
Correct prediction:  versicolor
Correct prediction:  setosa
Correct prediction:  setosa
Correct prediction:  virginica
Correct prediction:  versicolor
Correct prediction:  setosa
Correct prediction:  setosa
Correct prediction:  setosa
Correct prediction:  virginica
Correct prediction:  versicolor
Correct prediction:  versicolor
Correct prediction:  setosa
Correct prediction:  setosa
```

In [10]:

```
# Print confusion matrix
print("Confusion matrix:\n", confusion_matrix(y_test, y_pred))
```

```
Confusion matrix:
 [[19  0  0]
 [ 0 13  0]
 [ 0  0 13]]
```

In [ ]:

```

```