Aim:

To implement a K-Nearest Neighbors (KNN) classifier to classify the Iris dataset into its respective species based on flower measurements.

Procedure:

1. Import libraries — Import numpy, pandas, sklearn.model_selection, and sklearn.neighbors.
2. Load dataset — Read Iris.csv using pd.read_csv().
3. Explore data — Use df.info() and df.variety.value_counts() to understand dataset structure.
4. Define features and labels —
   Features → first four columns (sepal & petal measurements)
   Labels → variety
5. Split dataset — Use train_test_split() to split data into training and test sets.
6. Train model — Create a KNeighborsClassifier(n_neighbors=5) and fit it with training data.
7. Evaluate model — Print training and test accuracy using .score().
8. Generate metrics — Display confusion matrix and classification report.
9. Interpret results — Observe accuracy and class performance for Setosa, Versicolor, and Virginica.

In [9]:
```
#Experiment no.: 08
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import confusion_matrix, classification_report

df = pd.read_csv('Iris (1).csv')

df.info()
print(df.variety.value_counts())
print(df.head())

features = df.iloc[:, :-1].values
label = df.iloc[:, 4].values

xtrain, xtest, ytrain, ytest = train_test_split(features, label, test_size=0.2,

model_KNN = KNeighborsClassifier(n_neighbors=5)
model_KNN.fit(xtrain, ytrain)

print(model_KNN.score(xtrain, ytrain))
print(model_KNN.score(xtest, ytest))

print(confusion_matrix(label, model_KNN.predict(features)))
print(classification_report(label, model_KNN.predict(features)))
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   sepal.length  150 non-null    float64
 1   sepal.width   150 non-null    float64
 2   petal.length  150 non-null    float64
 3   petal.width   150 non-null    float64
 4   variety       150 non-null    object 
dtypes: float64(4), object(1)
memory usage: 6.0+ KB
variety
Setosa        50
Versicolor    50
Virginica     50
Name: count, dtype: int64
   sepal.length  sepal.width  petal.length  petal.width variety
0           5.1          3.5           1.4          0.2  Setosa
1           4.9          3.0           1.4          0.2  Setosa
2           4.7          3.2           1.3          0.2  Setosa
3           4.6          3.1           1.5          0.2  Setosa
4           5.0          3.6           1.4          0.2  Setosa
0.95
0.9666666666666667
[[50  0  0]
 [ 0 45  5]
 [ 0  2 48]]
              precision    recall  f1-score   support

      Setosa       1.00      1.00      1.00        50
  Versicolor       0.96      0.90      0.93        50
   Virginica       0.91      0.96      0.93        50

    accuracy                           0.95       150
   macro avg       0.95      0.95      0.95       150
weighted avg       0.95      0.95      0.95       150
```

In [ ]:

Result:

Thus, the model efficiently classified the Iris dataset into three categories — Setosa, Versicolor, and Virginica.