**Practical 8**

**Implement the K-NN algorithm for classification or regression.**

**• Apply the K-NN algorithm to a given dataset and predict the class or value for test data.**

**• Evaluate the accuracy or error of the predictions and analyse the results**

**Theory:**

* K-Nearest Neighbors (KNN) is a simple and intuitive machine learning algorithm used for both classification and regression tasks.
* It falls under the category of supervised learning, where the algorithm learns patterns from labeled training data to make predictions or decisions on new, unlabeled data points.
* KNN is a non-parametric algorithm, which means it doesn't make any assumptions about the underlying data distribution.
* The KNN algorithm can be used for both classification and regression problems.
* The KNN algorithm uses ‘**feature similarity**’ to predict the values of any new data points.
* This means that the new point is assigned a value based on how closely it resembles the points in the training set.

The K-NN working can be explained on the basis of the below algorithm:

* **Step-1:** Select the number K of the neighbors
* **Step-2:** Calculate the Euclidean distance of **K number of neighbors**
* **Step-3:** Take the K nearest neighbors as per the calculated Euclidean distance.
* **Step-4:** Among these k neighbors, count the number of the data points in each category.
* **Step-5:** Assign the new data points to that category for which the number of the neighbor is maximum.
* **Step-6:** Our model is ready.

Advantages of KNN Algorithm:

* It is simple to implement.
* It is robust to the noisy training data
* It can be more effective if the training data is large.

Disadvantages of KNN Algorithm:

* Always needs to determine the value of K which may be complex some time.
* The computation cost is high because of calculating the distance between the data points for all the training samples.

# Import necessary modules

from sklearn.neighbors import KNeighborsClassifier

from sklearn.model\_selection import train\_test\_split

from sklearn.datasets import load\_iris

# Loading data

irisData = load\_iris()

# Create feature and target arrays

X = irisData.data

y = irisData.target

# Split into training and test set

X\_train, X\_test, y\_train, y\_test = train\_test\_split(

      X, y, test\_size = 0.2, random\_state=42)

knn = KNeighborsClassifier(n\_neighbors=7)

knn.fit(X\_train, y\_train)

# Predict on dataset which model has not seen before

print(knn.predict(X\_test))

Output:

[1 0 2 1 1 0 1 2 2 1 2 0 0 0 0 1 2 1 1 2 0 2 0 2 2 2 2 2 0 0]

In the example shown above following steps are performed:

1. The k-nearest neighbor algorithm is imported from the scikit-learn package.
2. Create feature and target variables.
3. Split data into training and test data.
4. Generate a k-NN model using neighbors value.
5. Train or fit the data into the model.
6. Predict the future.