**9. To implement KNN Algorithm**

# Step 1: Import necessary libraries

import pandas as pd

import numpy as np

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

from sklearn.neighbors import KNeighborsClassifier

from sklearn.metrics import accuracy\_score, classification\_report, confusion\_matrix

import seaborn as sns

import matplotlib.pyplot as plt

# Step 2: Load the dataset

df = pd.read\_csv('/content/IRIS.csv')

print("First 5 rows of the dataset:")

print(df.head())

# Step 3: Check for missing values

print("\nMissing values in dataset:")

print(df.isnull().sum())

# Step 4: Split dataset into features and target

X = df.iloc[:, :-1] # all rows, all columns except the last

y = df.iloc[:, -1] # all rows, only the last column (target class)

# Step 5: Split into training and testing sets (80% train, 20% test)

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Step 6: Create and train the KNN classifier

k = 3 # You can experiment with different k values

knn = KNeighborsClassifier(n\_neighbors=k)

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

# Step 7: Make predictions

y\_pred = knn.predict(X\_test)

# Step 8: Evaluate the model

print("\nAccuracy:", accuracy\_score(y\_test, y\_pred))

print("\nClassification Report:\n", classification\_report(y\_test, y\_pred))

print("\nConfusion Matrix:")

conf\_matrix = confusion\_matrix(y\_test, y\_pred)

sns.heatmap(conf\_matrix, annot=True, cmap='Blues', fmt='d', xticklabels=knn.classes\_, yticklabels=knn.classes\_)

plt.xlabel('Predicted')

plt.ylabel('Actual')

plt.title('Confusion Matrix')

plt.show()