```
#Import
import numpy as np
import pandas as pd
import scipy.spatial
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.preprocessing import StandardScaler
from scipy.stats import mode

#Getting the data
col_names = ['sepal_length', 'sepal_width', 'petal_length', 'petal_width', 'species']
data = pd.read_csv("iris.csv", skiprows=1, header=None, names=col_names)
data.head(10)

#Accuracy for Decision Tree as per Previous Assignment
accuracy_decision = 0.9333333333333333
print('Accuracy using Decision Tree:', accuracy_decision)

#Euclidean Distance
def eucledian(p1,p2):
  dist = np.sqrt(np.sum((p1-p2)**2))
  return dist

#Prediction using KNN
def predict(x_train, y, x_input, k):
  op_labels = []

  #Loop through the Datapoints to be classified
  for item in x_input:

    #Array to store distances
    point_dist = []

    #Loop through each training Data
    for j in range(len(x_train)):
      distances = eucledian(np.array(x_train[j,:]), item)
      #Calculating the distance
      point_dist.append(distances)
    point_dist = np.array(point_dist)

    #Sorting the array while preserving the index
    #Keeping the first K datapoints
    dist = np.argsort(point_dist)[:k]

    #Labels of the K datapoints from above
    labels = y[dist]

    #Majority voting
    lab = mode(labels)
    lab = lab.mode[0]
    op_labels.append(lab)

  return op_labels
  return op_labels

X = data.iloc[:, :-1].values
y = data.iloc[:, -1].values.reshape(-1,1)

#Random State: 41
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=41)
#print('Train Set:', X_train.shape, y_train.shape)
#print('Test Set:', X_test.shape, y_test.shape)

#Data Scaling
scaler = StandardScaler()
scaler.fit(X_train)
X_train = scaler.transform(X_train)
X_test = scaler.transform(X_test)

#Accuracy of 3NN
y_pred = predict(X_train, y_train, X_test, 3)

#Checking the accuracy of 3NN
nn3 = accuracy_score(y_test, y_pred)
print('Accuracy using KNN Classifier(K=3):', nn3)

#Accuracy of 5NN
y_pred = predict(X_train, y_train, X_test, 5)

#Checking the accuracy of 5NN
nn5 = accuracy_score(y_test, y_pred)
print('Accuracy using KNN Classifier(K=5):', nn5)

#Comparing the Accuracy of Decision Tree, 3NN and 5NN
if(nn3 < nn5):
  if(nn3 > accuracy_decision):
    print('Accuracy: Decision Tree < 3NN < 5NN')
  elif(nn3 == accuracy_decision):
    print('Accuracy: Decision Tree = 3NN < 5NN')
  elif(accuracy_decision > nn5):
    print('Accuracy: 3NN < 5NN < Decision Tree')
  elif(nn5 == accuracy_decision):
    print('Accuracy: 3NN < Decision Tree = 5NN')
  else:
    print('Accuracy: 3NN < Decision Tree < 5NN')
elif(nn3 > nn5):
  if(nn5 > accuracy_decision):
    print('Accuracy: Decision Tree < 5NN < 3NN')
  elif(nn5 == accuracy_decision):
    print('Accuracy: Decision Tree = 5NN < 3NN')
  elif(accuracy_decision == nn3):
    print('Accuracy: 5NN < 3NN = Decision Tree')
  elif(nn3 < accuracy_decision):
    print('Accuracy: 5NN < 3NN < Decision Tree')
  else:
    print('Accuracy: 5NN < Decision Tree < 3NN')
elif(nn3 == nn5):
  if(accuracy_decision < nn3):
    print('Accuracy: Decision Tree < 3NN = 5NN')
```

```
      print('Accuracy: Decision Tree < 3NN = 5NN')
    else:
      print('Accuracy: 3NN = 5NN < Decision Tree')
  else:
    print('Accuracy: Decision Tree = 3NN = 5NN')
```

```
Accuracy using Decision Tree: 0.9333333333333333
Accuracy using KNN Classifier(K=3): 0.9
Accuracy using KNN Classifier(K=5): 0.9
Accuracy: 3NN = 5NN < Decision Tree
```

✓ 0s completed at 23:06