**Department of Computing**

**CS370: Artificial Intelligence**

**Class: BSCS 8C**

**Time: 9.00Am to 12.00Pm**

**Instructor: Dr. Hashir Moheed**

# Lab Tasks

**Task #1**

from sklearn.neighbors import KNeighborsClassifier  
import time  
import numpy as np  
import pandas as pd  
from scipy.stats import mode  
  
dataset = pd.read\_csv('TrainingSet.csv')  
y\_train = dataset['plant'].values  
x\_train = dataset[['leaf.length', 'leaf.width', 'flower.length', 'flower.width']].values  
  
  
def eucledian(p1, p2):  
 dist = np.sqrt(np.sum((p1 - p2) \*\* 2))  
 return dist  
  
  
def predict(x\_train, y, x\_input, k):  
 op\_labels = []  
 for item in x\_input:  
 point\_dist = []  
 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)  
 dist = np.argsort(point\_dist)[:k]  
 labels = y[dist]  
 lab = mode(labels).mode[0]  
 op\_labels.append(lab)  
 return op\_labels  
  
  
test\_data = pd.read\_csv('TestSet1.csv')  
x\_test = test\_data[['leaf.length', 'leaf.width', 'flower.length', 'flower.width']].values  
  
k\_values = [3, 5, 7]  
prediction = []  
times = {}  
  
for i in k\_values:  
 # predictions from self made formula  
 start\_time = time.time()  
 y\_test = predict(x\_train, y\_train, x\_test, i)  
 times['Time for k = ' + str(i) + ' (Self)'] = (time.time() - start\_time)  
  
 prediction.append(y\_test)  
  
 # predictions from scikit learn  
 classifier = KNeighborsClassifier(n\_neighbors=i)  
 start\_time = time.time()  
 classifier.fit(x\_train, y\_train)  
 y\_pred = classifier.predict(x\_test)  
 times['Time for k = ' + str(i) + ' (Scikit)'] = (time.time() - start\_time)  
  
 prediction.append(list(y\_pred))  
  
result = pd.concat(  
 [pd.DataFrame(x\_test), pd.DataFrame(prediction[0]), pd.DataFrame(prediction[1]), pd.DataFrame(prediction[2]),  
 pd.DataFrame(prediction[3]), pd.DataFrame(prediction[4]), pd.DataFrame(prediction[5])], axis=1)  
result.columns = ['leaf.length', 'leaf.width', 'flower.length', 'flower.width', 'k3', 'k3\_scikit', 'k5', 'k5\_scikit',  
 'k7', 'k7\_scikit']  
print(result)  
  
print("\nTime for prediction by both methods:")  
print("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_")  
for i, j in times.items():  
 print(i + " " + str(j))  
  
print("\nCheck Predictions")  
print("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_")  
precentage = (sum(((result['k3'] == result['k3\_scikit']) == (result['k5'] == result['k5\_scikit'])) == (  
 result['k7'] == result['k7\_scikit'])) / 30) \* 100  
print(str(precentage) + '% of entries are equal')  
  
result.to\_csv('result.csv', index=False)

Graphical user interface, text, application

Description automatically generated