PRACTICAL-10

Subject : Machine Learning Lab

Class:MCA6A

Sub\_Code: CAP-858

**Q.10)Implementation of K NN Clustering on Jupyter Notebook using Python.**

**Ans:**

import csv

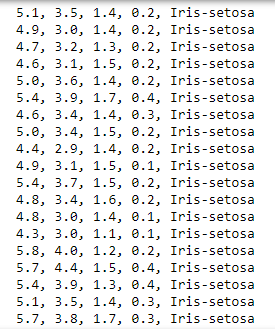
with open(r'iris.txt') as csvfile:

lines = csv.reader(csvfile)

for row in lines:

print (', '.join(row))

output:



import csv

import random

def handleDataset(filename, split, trainingSet=[] , testSet=[]):

with open(filename, 'r') as csvfile:

lines = csv.reader(csvfile)

dataset = list(lines)

for x in range(len(dataset)-1):

for y in range(4):

dataset[x][y] = float(dataset[x][y])

if random.random() < split:

trainingSet.append(dataset[x])

else:

testSet.append(dataset[x])

trainingSet=[]

testSet=[]

handleDataset(r'iris.txt.', 0.66, trainingSet, testSet)

print ('Train: ' + repr(len(trainingSet)))

print ('Test: ' + repr(len(testSet)))

OutPut:



import math

def euclideanDistance(instance1, instance2, length):

distance = 0

for x in range(length):

distance += pow((instance1[x] - instance2[x]), 2)

return math.sqrt(distance)

data1 = [2, 2, 2, 'a']

data2 = [4, 4, 4, 'b']

distance = euclideanDistance(data1, data2, 3)

print ('Distance: ' + repr(distance))

OUTPUT:



import operator

def getKNeighbors(trainingSet, testInstance, k):

distances = []

length = len(testInstance)-1

for x in range(len(trainingSet)):

dist = euclideanDistance(testInstance, trainingSet[x], length)

distances.append((trainingSet[x], dist))

distances.sort(key=operator.itemgetter(1))

neighbors = []

for x in range(k):

neighbors.append(distances[x][0])

return neighbors

trainSet = [[2, 2, 2, 'a'], [4, 4, 4, 'b']]

testInstance = [5, 5, 5]

k = 1

neighbors = getKNeighbors(trainSet, testInstance, 1)

print(neighbors)

OUTPUT:



import operator

def getResponse(neighbors):

classVotes = {}

for x in range(len(neighbors)):

response = neighbors[x][-1]

if response in classVotes:

classVotes[response] += 1

else:

classVotes[response] = 1

sortedVotes = sorted(classVotes.items(), key=operator.itemgetter(1), reverse=True)

return sortedVotes[0][0]

neighbors = [[1,1,1,'a'], [2,2,2,'a'], [3,3,3,'b']]

print(getResponse(neighbors))

OUTPUT:

a

def getAccuracy(testSet, predictions):

correct = 0

for x in range(len(testSet)):

if testSet[x][-1] is predictions[x]:

correct += 1

return (correct/float(len(testSet))) \* 100.0

testSet = [[1,1,1,'a'], [2,2,2,'a'], [3,3,3,'b']]

predictions = ['a', 'a', 'a']

accuracy = getAccuracy(testSet, predictions)

print(accuracy)

OUTPUT:

