Bell5820

Spencer Bellerose

160795820

bell5820@mylaurier.ca

In this assignment we have implemented code to calculate the d_min, d_max, d_mean, and d_avg between two classes. I have used the python math library.

Below is my code.

In [1]:
```
#imports
import math
```

In [36]:
```
#defined it as a function
def distances(class_a, class_b):


    #calculate Euclidean Distances between all variables:
    euclidean = []
    for i in class_a:
        for j in class_b:
            x_val = (i[0]-j[0])**2
            y_val = (i[1]-j[1])**2
            dist = math.sqrt(x_val+y_val)
            euclidean.append(dist)

    #calculate d_min, should be smallest euclidean distance in the chart
    d_min = 999 #to be replaced
    for i in euclidean:
        if i < d_min:
            d_min = i



    #calculate d_max, should be largest euclidean distance in the chart
    d_max = 0 #to be replaced
    for i in euclidean:
        if i > d_max:
            d_max = i

    #calculate d_avg
        #calculate the euclidean average
    euc_avg = 0
    for i in euclidean:
        euc_avg += i

    #calculate number of dimensions
    dim = len(class_a) * len(class_b)

    #finally, calcalate d_avg
    d_avg = euc_avg/dim


    #calculate d_mean
        #calculate centerpoint for class a
    centerpoint_a = 0
    len1 = 0
    centerpoint_b = 0
    len2 = 0

    for l in class_a:
        centerpoint_a += l[0]
        len1 += 1

    for p in class_a:
        centerpoint_b += p[1]
        len2 += 1
```

```
    norm_1 = [(centerpoint_a/len1), (centerpoint_b/len2)]


        #calculate centerpoint for class b
    centerpoint_c = 0
    len3 = 0
    centerpoint_d = 0
    len4 = 0

    for l in class_b:
        centerpoint_c += l[0]
        len3 += 1

    for p in class_b:
        centerpoint_d += p[1]
        len4 += 1

    norm_2 = [(centerpoint_c/len3), (centerpoint_d/len4)]

    #prepare values for calculation of euclidean distance
    x_mean = (norm_1[0]-norm_2[0])**2
    y_mean = (norm_1[1]-norm_2[1])**2
    d_mean = math.sqrt(x_mean+y_mean)

    #return a list with the returning values
    return[d_min, d_max, d_avg, d_mean]

#-------------------------------------------------------------------------------
---------------------------------

#Test program
    #case 1
print('Testing Case 1: Example from assignment Description')
class_a = [[1,1],[1,2]]
class_b = [[2,1],[3,1]]
test1 = distances(class_a, class_b)
print('using classes:',class_a, 'and,', class_b)
print(test1)
print()
print('-----------------------------------------------------')

    #case 2
print('Testing Case 2: Values given on My Learning Space')
class_c = [[1,2], [13,4], [4,4], [3,2]]
class_d = [[4,14], [4,16], [5,10]]
test2 = distances(class_c, class_d)
print('using classes:',class_c, 'and,', class_d)
print(test2)
print()
print('-----------------------------------------------------')
```

```
Testing Case 1: Example from assignment Description
using classes: [[1, 1], [1, 2]] and, [[2, 1], [3, 1]]
[1.0, 2.23606797749979, 1.6625703849682212, 1.5811388300841898]

--------------------------------------------------
Testing Case 2: Values given on My Learning Space
using classes: [[1, 2], [13, 4], [4, 4], [3, 2]] and, [[4, 14], [4, 16], [5,
10]]
[6.082762530298219, 15.0, 11.37427259209552, 10.373912258909632]

--------------------------------------------------
```

From these results we can see that upon using the above program, we can calculate the smallest euclidean distance (d_min), the largest euclidean distance (d_max), the average euclidean distance (d_avg), and the mean euclidean distance (d_mean) between any two classes.

as we can see from the results,

test 1: d_min = 1.0, d_max = 2.23, d_avg = 1.66, d_mean = 1.58

test2: d_min = 6.08, d_max = 15, d_avg = 11.37, d_mean = 10.37

I have shown this in two parts, one on the classes: [[1, 1], [1, 2]], [[2, 1], [3, 1]]

And the other classes: [[1, 2], [13, 4], [4, 4], [3, 2]], [[4, 14], [4, 16], [5, 10]]