Assi5

Xiaolong Yang

1 Conceptual Questions (5 points)

1.a

K-Means is sensitive to noisy data and outliers, because it calculated the center by average of the dataset, and the outliers or noise can affect the mean, thus the K centers would be meaningless.

1.b

No.

1.c

Hierarchical clustering can be applied in more distance function while k-means get a good result merely in Euclidean distance.
Hierarchical clustering did not need the specific K – the number of clusters.

2 Advanced Classification: Perceptron (5 points)

x_1	x_2	y
0	0	+
0	1	+
1	0	+
1	1	-

Table 1: Data points with class labels

If y != sign($w^T x$), update $w = w + \eta * x * y$

	iter1	sign	$\eta*x*y$-4	iter2	sign	$\eta*x*y$-1	iter3	sign
W0	0.25		-0.5	-0.25		0.5	0.25	
W1	0.25		-0.5	-0.25		0	-0.25	
W2	0.25		-0.5	-0.25		0	-0.25	
Y1	0.25	1		-0.25	-1		0.25	1
Y2	0.5	1		-0.5	-1		0	1
Y3	0.5	1		-0.5	-1		0	1
Y4	0.75	1		-0.75	-1		-0.25	-1

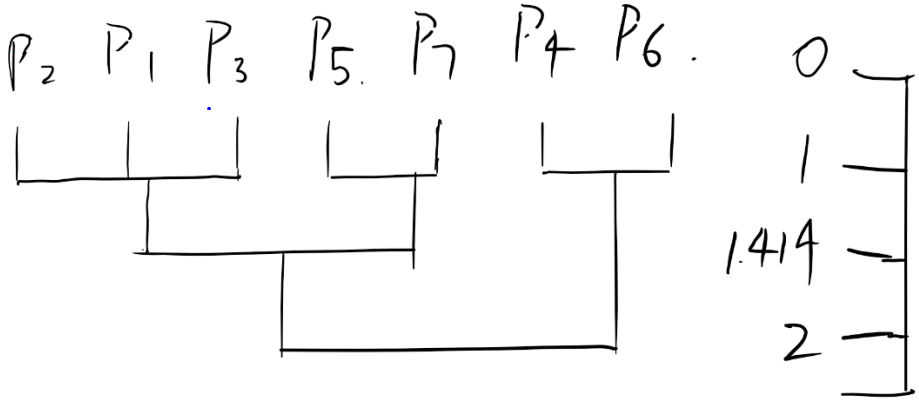
3 Hierarchical Agglomerative Clustering and B-Cubed Evaluation (8 points)

Point	x	y	Ground Truth
P1	1	1	C1
P2	1	2	C1
P3	2	1	C1
P4	5	1	C2
P5	3	2	C1
P6	5	2	C2
P7	3	3	C1

Table 2: Data Points

3.a.

L1 (value = 1): C1(P1, P2, P3), C2(P4, P6), C3(P5, P7);
L2 (value = 1.414): C4(P1, P2, P3, P5, P7), C5(P4, P6);
L3 (value = 2): C6(P1,P2, P3, P4, P5, P6, P7).



3.b

There would be C1(P2, P1, P3), C2(P4, P6), C3(P5, P7)

3.c

Precision 1= 3/3 =1
Precision 2= 2/2 = 1
Precision 3= 2/2 = 1

Final Precision = 1

Recall1 = 3/5 = 0.6
Recall2 = 2/2 = 1
Recall3 = 2/5 = 0.4

Final Recall = (Recall 1*3+ Recall 2*2+ Recall 3*2) * (3+2+2) = 0.657

Question 4

In [1]:

```
def eucl_dist(a,b):
    distance = 0
    for i in range(len(a)):
        distance += (a[i]-b[i])**2
    return round(float(distance)**0.5,3)
```

In [30]:

```
def find_center(cluster):
    C_x = 0
    C_y = 0
    for i in cluster:
        C_x += i[0]
        C_y += i[1]
    center = [float(C_x)/len(cluster), float(C_y)/len(cluster)]
    return center
```

In [35]:

```
def cluster(data,center1,center2):
    C1 = []
    C2 = []
    for i in data:
        if eucl_dist(i,center1) < eucl_dist(i,center2):
            C1.append(i)
        else:
            C2.append(i)
    print C1
    print C2
    print find_center(C1)
    print find_center(C2)
    return find_center(C1),find_center(C2)
```

Q.a

In [32]:

```
data = [[1,1],[1,2],[2,1],[5,1],[3,2],[5,2],[3,3]]
```

In [36]:

```
center1, center2 = cluster(data,data[0],data[2])
```

```
[[1, 1], [1, 2]]
[[2, 1], [5, 1], [3, 2], [5, 2], [3, 3]]
[1.0, 1.5]
[3.6, 1.8]
```

In [39]:

```
center1, center2 = cluster(data,center1, center2)
```

```
[[1, 1], [1, 2], [2, 1]]
[[5, 1], [3, 2], [5, 2], [3, 3]]
[1.3333333333333333, 1.3333333333333333]
[4.0, 2.0]
```

In [41]:

```
center1, center2 = cluster(data,center1, center2)
```

```
[[1, 1], [1, 2], [2, 1]]
[[5, 1], [3, 2], [5, 2], [3, 3]]
[1.3333333333333333, 1.3333333333333333]
[4.0, 2.0]
```

We can find there is no change in the **3rd** interation. The k-means finished.

Q.b

In [42]:

```
center1, center2 = cluster(data,data[1],data[5])
```

```
[[1, 1], [1, 2], [2, 1]]
[[5, 1], [3, 2], [5, 2], [3, 3]]
[1.3333333333333333, 1.3333333333333333]
[4.0, 2.0]
```

In [43]:

```
center1, center2 = cluster(data,center1, center2)
```

```
[[1, 1], [1, 2], [2, 1]]
[[5, 1], [3, 2], [5, 2], [3, 3]]
[1.3333333333333333, 1.3333333333333333]
[4.0, 2.0]
```

We can find there is no change in the **2nd** interation. The k-means finished.

Q.c

Randomly set one point as 1st center, find the longest point from the center as the 2nd center

In [51]:

```
import random as rd
```

In [64]:

```
a = data[rd.randint(0,6)]
```

In [65]:

```
for i in data:
    print eucl_dist(i,a)
```

```
2.828
2.236
2.236
2.828
1.0
2.236
0.0
```

We can find P7 is the randam select data, P3 is the farest from P7

In [66]:

```
b= data[2]
```

In [67]:

```
center1, center2 = cluster(data,a, b)
```

```
[[5, 1], [3, 2], [5, 2], [3, 3]]
[[1, 1], [1, 2], [2, 1]]
[4.0, 2.0]
[1.3333333333333333, 1.3333333333333333]
```

In [68]:

```
center1, center2 = cluster(data,center1, center2)
```

```
[[5, 1], [3, 2], [5, 2], [3, 3]]
[[1, 1], [1, 2], [2, 1]]
[4.0, 2.0]
[1.3333333333333333, 1.3333333333333333]
```

It get to the center in the **1st** step.