Algorithmic Data Science

3rd Assignment

Nikos Stamatis
MSc Student
Graduate Programme in Data Science and Machine Learning
SN: 03400115
nikolaosstamatis@mail.ntua.gr

Exercise 7.2.5

We begin with the lower left cluster containing the points $x_1 = (3,4)$, $x_2 = (2,2)$ and $x_3 = (5,2)$. Their pairwise distances are

$$\begin{aligned} d(x_1,x_2) &= \sqrt{(3-2)^2+(4-2)^2} = \sqrt{1+4} = \sqrt{5}, \\ d(x_1,x_3) &= \sqrt{(3-5)^2+(4-2)^2} = \sqrt{4+4} = \sqrt{8}, \\ d(x_2,x_3) &= \sqrt{(5-2)^2+(2-2)^2} = \sqrt{9} = 3. \end{aligned}$$

For each point x_i, the sum of distances between it and the rest of the points $D(x_i) = \sum_k d(x_i,x_k)$ is

$$\begin{aligned} D(x_1) &= \sqrt{5}+\sqrt{8} = 5.06, \\ D(x_2) &= \sqrt{5}+3 = 5.23, \\ D(x_3) &= \sqrt{8}+3 = 5.83, \end{aligned}$$

and is minimized for $x_1 = (3,4)$. So the clustroid of this cluster is the point x_1.

We will compute the rest of the clustroids using the clustroid function we created in R:

```
clustroid <- function(C){
  # Given a list of points C, it computes its clustroid,
  # namely the point in C for which the sum of distances
  # between it and the rest of the points in C is minimized.

  # Number of vectors in cluster C.
  n = length(C)

  # Pairwise distances will be stored here.
  dist_mat = matrix(0, n, n)

  for (i in 1:n){
    for (j in 1:n) {
      # Computes the pairwise distance of the two points.
      dist_mat[i, j] = norm(C[[i]] - C[[j]], type='2')
    }
  }

  # Sum of pairwise distances.
  dist_sum = rowSums(dist_mat)

  ind = C[[which.min(dist_sum)]]   # Clustroid.
  return(ind)
}
```

We create the three clusters of the exercise:

```
y1 = c(2, 2)
y2 = c(5, 2)
y3 = c(3, 4)

CL1 = list(y1, y2, y3)

yy1 = c(4, 10)
yy2 = c(7, 10)
yy3 = c(4, 8)
yy4 = c(6, 8)

CL2 = list(yy1, yy2, yy3, yy4)

yyy1 = c(12, 6)
yyy2 = c(10, 5)
yyy3 = c(11, 4)
yyy4 = c(9, 3)
yyy5 = c(12, 3)

CL3 = list(yyy1, yyy2, yyy3, yyy4, yyy5)
```

and apply the clustroid function on them:

```
clustroid(CL1)
[1] 3 4

clustroid(CL2)
[1] 6 8

clustroid(CL3)
[1] 11  4
```

We conclude that the three clustroids are the points $(3,4)$, $(6,8)$ and $(11,4)$.

Exercise 7.3.2

Let (X,d) be a metric space and $C \subseteq X$. We define its diameter $\operatorname{diam}(C)$ as $\operatorname{diam}(C) = \sup\{d(x,y) : x,y \in C\}$. Additionally, given a partition $C = (C_i)_{i=1}^n$ of X we define the minimum intercluster distance MID of this partition as $\mathrm{MID}(C) = \min\{d(C_i, C_j) : i \neq j = 1, \ldots, n\}$, where $d(C_i, C_j) = \inf\{d(x, y) : x \in C_i, y \in C_j\}$.

The exercise requires to provide a theorem that involves the quantities $m = \max\{\mathrm{diam}(C_i)\}_i$ and $\mathrm{MID}(C)$ and asserts that no matter which initial point we choose, the k-means algorithm will initialize with n points which will necessarily belong to all n-different clusters $C_1, \ldots, C_n$.

Although it is easy to provide such a theorem, an issue arises for the particular example the exercise requires us to solve. Let us explain: We can easily compute the diameters of the three clusters. For example, based on the previous exercise, for the first cluster we have that

$$\mathrm{diam}(C_1) = \max\left\{\sqrt{3}, \sqrt{8}, \sqrt{9}\right\} = 3.$$

We computed the rest of the diameters in R:

```
diameter <- function(c){
  # Computes the diameter of a list of vectors c.

  # Number of vectors.
  n = length(c)

  # Constructs every possible pair of vectors.
  all_subsets = combn(1:n, 2, simplify = FALSE)

  min_d = -Inf
  for (x in all_subsets){
    x1 = x[1]
    x2 = x[2]
    x1 = c[[x1]]
    x2 = c[[x2]]
    d = norm(x1 - x2, type="2")
    if (d > min_d) {min_d = d}
  }
  return (min_d)
}
```

```
diameter(CL1)
[1] 3

diameter(CL2)
[1] 3.605551

diameter(CL3)
[1] 4.242641
```

We computed the intercluster distances using the min_among_cl function:

```
min_among_cl = function(c1, c2){
  # Computes the distance between the lists c1, c2.

  min_d = Inf
  for (x in c1){
    for (y in c2){
      d = norm(x-y, type="2")
      if (d < min_d) min_d = d
    }
  }
  return(min_d)
}

min_among_cl(CL3, CL1)
[1] 4.123106

min_among_cl(CL3, CL2)
[1] 5

min_among_cl(CL1, CL2)
[1] 4.123106
```

To sum up, the diameters of the clusters are $d(C_1) = 3$, $d(C_2) = 3.61$ and $d(C_3) = 4.24$, whereas the minimum intercluster distance is $\mathrm{MID}(C) = 4.12$.

An obvious theorem involving only these four quantities would be that the maximum diameter should be strictly smaller than MID. This would indeed suffice for the process to work, since at any given stage, any point that belongs to a cluster not used so far will have a larger distance to the points chosen than any point in an already used cluster. However, this property does not hold in the example of the exercise as $d(C_3) = 4.24 > \mathrm{MID} = 4.12$.

Since the main point of the exercise is to prove the claim for the example, we will state a slightly different theorem with a weaker hypothesis, so that it can be applied here. The assumption that all diameters must be less than the MID will be replaced by slightly weaker, but equally useful, property. Instead of demanding all points in all unused clusters to be sufficiently far, it suffices to demand that at least one such point in any unused cluster will always exist.

For this reason, for any cluster C_i we define the quantity

$$D(C_i) = \max_{x \in C_i}\{d(x, \cup_{k \neq i} C_k)\}. \tag{1}$$

The quantity $D(C_i)$ computes the maximum possible distance between a point in C_i and all the remaining clusters.

Theorem 1. *Using the same notation as before and given n clusters $(C_i)_{i=1}^n$, let $m = \max_i\{\mathrm{diam}(C_i)\}$ and $M = \min_i\{D(C_i)\}$. If $m < M$, then the procedure of the exercise will pick n elements that belong to n distinct clusters.*

Proof. We will use induction. During the first step,

x_1 will belong to some cluster C_{i_1} which has not picked before, so we have nothing to prove. Suppose that the claim holds up to $k < n$, namely we have picked $x_1 \in C_{i_1}, \dots, x_k \in C_{i_k}$ such that all $i_1, \dots, i_k$ are distinct.

Let $y \in \cup_{l=1}^{k} C_{i_l}$. Then y belongs to some C_{i_l} and

$$d(y, \{x_1, \dots, x_l\}) \le d(y, x_l) \le \operatorname{diam}(C_{i_l}) \le m. \tag{2}$$

Now pick a cluster which has not been used so far, say C_m. Let also y^* denote the element in C_m where $D(C_m)$ is attained. Then

$$\begin{aligned} d(y^*, \{x_1, \dots, x_k\}) &\ge d\left(y^*, \bigcup_{l=1}^{k} C_{i_l}\right) \\ &\ge d\left(y^*, \bigcup_{j \ne m} C_j\right) \\ &= D(C_m) \ge M > m \\ &\ge d(y, \{x_1, \dots, x_l\}). \end{aligned}$$

Since there exists some y^* with larger distance than any $y \in \cup_{l=1}^{k} C_{i_l}$, the next point picked by our algorithm will have to belong to a cluster that hasn't appeared so far. This concludes the inductive step and the proof. □

We computed the $D(C_i)$'s for all clusters using the point_to_cluster_min function.

```
distance <- function(x, c){
  # Computes the distance between a point x and a list C,
  # d(x, C) = min{ d(x,y): y in C }

  n = length(c)
  res = rep(0, n)
  for (i in 1:n){
    res[i] = norm(x-c[[i]], type='2')
  }
  return(min(res))
}

point_to_cluster_min <- function(original_cluster, c1, c2){
  # Computes the distances between all points in
  # the original_cluster, and the union of c1 and c2.
  # Returns the maximum distance achieved among them.

  new_c = c(c1, c2)
  curr_max = -Inf
  n = length(original_cluster)
  new = rep(0, n)
  for (i in 1:n){
    new[i] = distance(original_cluster[[i]], new_c)
  }
  return(max(new))
}
```

```
point_to_cluster_min(CL1, CL2, CL3)
[1] 6.324555

point_to_cluster_min(CL2, CL1, CL3)
[1] 6.082763

point_to_cluster_min(CL3, CL1, CL2)
[1] 7.071068
```

As we can see, $M = 6.08 > 4.24 = m$, so the theorem can be applied to our example.

Exercise 7.4.1

The exercise involves a ball and a ring in the plane, but the same reasoning can be applied to any normed space $(X, \|\cdot\|)$ instead of $\mathbb{R}^2$. The first cluster is a ball centered at zero, $C_1 = B(0, c) = \{x \in X : \|x\| \le c\}$ and the second one is the ring $C_2 = \{x \in X : i \le \|x\| \le o\}$. The centroids of both clusters are at $x = 0$. We will use C_1^r and C_2^r to denote their representative points respectively.

Given a point $x \in X$, the resulting point x' after moving it towards 0 is given by the convex combination $x' = (1-\lambda)x + \lambda 0 = (1-\lambda)x$. For $\lambda = 0.2$ we obtain that $x' = 0.8x$.

Let $x, y \in X$ be two points and let $x' = (1-\lambda)x$ and $y' = (1-\lambda)y$ denote the points resulting after moving them a $100 \cdot \lambda\%$ towards zero. Then

$$\|x' - y'\| = \|(1-\lambda)x - (1-\lambda)y\| = (1-\lambda)\|x - y\|,$$

so their new distance is just their previous one multiplied by a factor of $1-\lambda$. If we set $m = \min\{\|x-y\| : x \in C_1^r, y \in C_2^r\}$ to be the minimum distance between the representative points of the two clusters, and $m' = \min\{\|x'-y'\| : x \in C_1^r, y \in C_2^r\}$ to be the same distance after the points have been moved towards the center, then

$$\begin{aligned} m' &= \min\left\{\|x'-y'\| : x \in C_1^r, y \in C_2^r\right\} \\ &= \min\left\{(1-\lambda)\|x-y\| : x \in C_1^r, y \in C_2^r\right\} \\ &= (1-\lambda)\left\{\|x-y\| : x \in C_1^r, y \in C_2^r\right\} \\ &= (1-\lambda)m. \end{aligned}$$

The two clusters will be merged into one, if and only if $m' \le d$, namely if and only if $m \le \frac{d}{1-\lambda} = \frac{d}{0.8}$. What is left, is to express m as a function of c, i and o.

The boundary assumption for C_1 means that all the representative points will lie on the sphere $S_c = \{x \in X : \|x\| = c\}$. For the second cluster C_2 they will lie in the union $S_i \cup S_o$ of the two spheres that contain the ring C_2.

At first, the representative points of C_2 will mostly lie on the outer sphere S_o. For a really small number N of representative points, one may get some very bad worst case bound. For example, for $N = 1$ the worst case scenario will yield $m = c + o$, whereas for $N = 2$ the worst case results in $m = \sqrt{c^2 + o^2}$.

However, as the number of representative points N increases, points will start to appear in the inner circle, and furthermore, the minimum pairwise distances between the two clusters will start to tend to $i - c$. So, for N sufficiently large, m will be approximately $m \approx i - c$ and the two clusters will be merged if and only if $i - c \leq \frac{d}{1-\lambda} = \frac{d}{0.8}$.

Exercise 10.4.1

The adjacency matrix A contains 1 in the (i, j) cell if the nodes i and j are connected and 0 otherwise. Therefore,

$$A = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \end{bmatrix}.$$

The degree matrix D is a diagonal matrix the i-th element of which is the degree of node i:

$$D = \mathrm{diag}(2, 3, 3, 3, 3, 2, 3, 3, 2).$$

Lastly, the Laplacian matrix L is defined as $L = D - A$, so

$$L = \begin{bmatrix} 2 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 3 & -1 & 0 & 0 & 0 & 0 & -1 & 0 \\ -1 & -1 & 3 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 3 & -1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 3 & -1 & -1 & 0 & 0 \\ 0 & 0 & 0 & -1 & -1 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 3 & -1 & -1 \\ 0 & -1 & 0 & 0 & 0 & 0 & -1 & 3 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & 2 \end{bmatrix}.$$

Exercise 10.4.2

We first define the Laplacian matrix in R:

```
r1 = c(0, 1, 1, 0, 0, 0, 0, 0, 0)
r2 = c(1, 0, 1, 0, 0, 0, 0, 1, 0)
r3 = c(1, 1, 0, 1, 0, 0, 0, 0, 0)
r4 = c(0, 0, 1, 0, 1, 1, 0, 0, 0)
r5 = c(0, 0, 0, 1, 0, 1, 1, 0, 0)
r6 = c(0, 0, 0, 1, 1, 0, 0, 0, 0)
r7 = c(0, 0, 0, 0, 1, 0, 0, 1, 1)
r8 = c(0, 1, 0, 0, 0, 0, 1, 0, 1)
r9 = c(0, 0, 0, 0, 0, 0, 1, 1, 0)

# Adjacency matrix.
A = rbind(r1, r2, r3, r4, r5, r6, r7, r8, r9)

# Degree matrix.
D = diag(9)
diag(D) <- c(2, 3, 3, 3, 3, 2, 3, 3, 2)

#Laplacian matrix.
L = D - A
```

Then we calculate the eigenvalues and eigenvectors of the Laplacian L:

```
eig = eigen(L)

eig$values
# [1] 5.000000e+00 4.302776e+00 4.302776e+00 3.0000e+00
#     3.000000e+00 3.000000e+00 6.972244e-01  6.9722e-01
#     4.440892e-15
```

The smallest nonzero eigenvalue is equal to $\lambda_2 = 0.67$ and there are two distinct eigenvectors that correspond to it:

```
eig$values[c(7,8)]
# [1] 0.6972244 0.6972244

eig$vectors[, c(7,8)]

#              [,1]          [,2]
# [1,]  0.58772230 -0.008451891
# [2,]  0.38453627  0.112786193
# [3,]  0.38113402 -0.123797111
# [4,] -0.09459243 -0.389411274
# [5,] -0.29777845 -0.268173189
# [6,] -0.30118070 -0.504756494
# [7,] -0.28994384  0.276625080
# [8,] -0.08335557  0.391970300
# [9,] -0.28654160  0.513208385
```

The first one yields the partition $C_1 = \{A, B, C\}$, $C_2 = \{D, E, F, G, H, I\}$, whereas the second one the $C_1 = \{B, G, H, I\}$ and $C_2 = \{A, C, D, E, F\}$.

Exercise 5.2.2

The transition matrix for the graph of Figure 5.4 is:

Page	Degree	Destinations
A	3	B, C, D
B	2	A, D
C	1	E
D	2	B, C
E	0	∅

Similarly, the transition matrix for Figure 5.7 is:

Page	Degree	Destinations
a	3	a, b, c
b	2	a, c
c	2	b, c

Exercise 5.2.2

The block M_{11} is built based on the sub-matrix with starting pages A, B and destinations also A, B:

Page	Degree	Destinations
A	3	B
B	2	A

The block M_{12} is built based on the sub-matrix with starting pages C, D and destinations A, B:

Page	Degree	Destinations
D	2	B

The block M_{21} is built based on the sub-matrix with starting pages A, B and destinations C, D:

Page	Degree	Destinations
A	3	C, D
B	2	D

The block M_{22} is built based on the sub-matrix with starting pages C, D and destinations C, D:

Page	Degree	Destinations
D	2	C

Exercise 9.3.2

(a) We first replace the values of the matrix with zeros and ones according to the instructions:

	a	b	c	d	e	f	g	h
A	1	1	0	1	0	0	1	0
B	0	1	1	1	0	0	0	0
C	0	0	0	1	0	1	1	1

Then, we compute all the pairwise Jaccard distances between the columns of this matrix. For example, all the pairwise distances involving a are:

$$J(a,b) = 1 - \frac{1}{2} = 0.5,$$

$$J(a,c) = 1 - \frac{0}{2} = 1,$$
$$J(a,d) = 1 - \frac{1}{3} = \frac{2}{3},$$
$$J(a,e) = 1 - \frac{0}{1} = 1,$$
$$J(a,f) = 1 - \frac{0}{2} = 1,$$
$$J(a,g) = 1 - \frac{1}{2} = 0.5,$$
$$J(a,h) = 1 - \frac{0}{2} = 1.$$

We proceed in a similar manner to obtain the rest. We summarize the results below:

$J(x,y)$	a	b	c	d	e	f	g	h
a	0	1/2	1	2/3	1	1	1/2	1
b		0	1/2	1/3	1	1	2/3	1
c			0	2/3	1	1	1	1
d				0	1	2/3	1/3	2/3
e					0	1	1	1
f						0	1/2	0
g							0	1/2
h								0

Table I: Pairwise Jaccard distances.

Throughout the clustering procedure we will use the alphabetical order to break possible ties. During the first step we will have 8 clusters, each containing a single element $C_1 = \{a\}$, $C_2 = \{b\}$ and so on. In the second step we merge together the two closest clusters, namely $\{f\}$ and $\{h\}$ will now form a single cluster since their distance is zero. So after the second step we have the following clusters:

$$C_1 = \{a\}, \quad C_2 = \{b\}, \quad C_3 = \{c\},$$
$$C_4 = \{d\}, \quad C_5 = \{e\}, \quad C_6 = \{g\}, \quad C_7 = \{f,h\}.$$

In the next step we will merge two clusters the distance of which is $\frac{1}{3}$. Due to the alphabetical criterion, these clusters will be $C_2 = \{b\}$ and $C_4 = \{d\}$. So the new clusters are:

$$C_1 = \{a\}, \quad C_2 = \{c\}, \quad C_3 = \{b,d\},$$
$$C_4 = \{e\}, \quad C_5 = \{g\}, \quad C_6 = \{f,h\}.$$

Next, we will merge $\{a\}$ with $\{b,d\}$ to obtain

$$C_1 = \{a,b,d\},\ C_2 = \{c\},\ C_3 = \{e\},\ C_4 = \{g\},\ C_5 = \{f,h\}.$$

Then, C_1 and C_4 are merged together:

$$C_1 = \{a,b,d,g\}, \quad C_2 = \{c\}, \quad C_3 = \{e\}, \quad C_4 = \{f,h\}.$$

At this point we can stop the procedure as four clusters have been formed.

(b) When computing the average of each row, we will treat the missing values in the original matrix as zero. The resulting matrix is the following:

	C_1	C_2	C_3	C_4
A	17/4	0	1	2
B	7/4	4	1	2
C	10/4	1	0	7/2

(c) The norms of the vectors A, B, C are $\|A\| = 4.8$, $\|B\| = 4.9$ and $\|C\| = 4.4$. The cosine similarity scores are

$$\cos(A,B) = \frac{\frac{17}{4}\cdot\frac{7}{4} + 1 + 2\cdot 2}{\|A\|\cdot\|B\|} = 0.528,$$
$$\cos(A,C) = \frac{\frac{17}{4}\cdot\frac{10}{4} + 2\cdot\frac{7}{2}}{\|A\|\cdot\|C\|} = 0.831,$$
$$\cos(B,C) = \frac{\frac{7}{4}\cdot\frac{10}{4} + 1 + 4\cdot 1 + 2\cdot\frac{7}{2}}{\|B\|\cdot\|C\|} = 0.71,$$

with corresponding cosine distances

$$d(A,B) = \arccos(0.528) = 58.1,$$
$$d(A,C) = \arccos(0.831) = 33.8,$$
$$d(B,C) = \arccos(0.710) = 44.8.$$