Fuzzy Pattern Recofnition

DECEMBER 8

Saeed77t@gmail.com / student number: 40160957
Authored by: Reza Tahmasebi

Table of Contents

Clustering method based on fuzzy equivalence relations and pattern recognition

The goal of this homework is to cluster some data based on how much they are similar to each other.

How codes work

First, the data has been read with the *loadmat* method from the *mat4py library. After* that, the data was divided into two sections the *digits* and the *labels.*

Use PCA

PCA method from *sklearn* library used to extract features from data.

Rfunc function :

This function gets X1, X2 and q as arguments, and it will return the distance of two data And it implements the formula bellow

$$R(\mathbf{x}_i, \mathbf{x}_k) = 1 - \delta(\sum_{j=1}^{p} |x_{ij} - x_{kj}|^q)^{\frac{1}{q}}$$

This function returns the distance between two data.

Use Rfunc:

In two nested for loops with the length of our data (150), the Rfunc is called. And it created a fuzzy compatibility relation in our data.

```
matrix of fuzzy compatibility relation
[[1.    0.285 0.357 ... 0.394 0.387 0.398]
 [0.285 1.    0.39  ... 0.414 0.355 0.361]
 [0.357 0.39  1.    ... 0.369 0.355 0.322]
 ...
 [0.394 0.414 0.369 ... 1.    0.341 0.357]
 [0.387 0.355 0.355 ... 0.341 1.    0.368]
 [0.398 0.361 0.322 ... 0.357 0.368 1.   ]]
```

The figure above is a model of compatibility relation.

RoR function :

This function gets the table that was created with Rfunc as an argument and, with the algorithm below, returns the matrix of transitive closure.

1. $R' = R \cup (R \circ R)$.
2. If $R' \neq R$, make $R = R'$ and go to Step 1.
3. Stop: $R' = R_T$.

This function uses 3 for loops and a while loop to create the matrix of transitive closure, so it has enormous computational complexity.

```
table of fuzzy transitive closer
[[1.    0.285 0.357 ... 0.394 0.387 0.398]
 [0.285 1.    0.39  ... 0.414 0.355 0.361]
 [0.357 0.39  1.    ... 0.369 0.355 0.322]
 ...
 [0.394 0.414 0.369 ... 1.    0.341 0.357]
 [0.387 0.355 0.355 ... 0.341 1.    0.368]
 [0.398 0.361 0.322 ... 0.357 0.368 1.   ]]
```

The table above is the matrix of transitive closure that the RoR function created.

calcAlphacut function:

This function gets the value of the alpha cut and the data as arguments and prints the clustering values.
The function uses the alpha cut value as a threshold and splits the data into parts.

```
 the alpha for 0.35 is =
 [1, 3, 5, 12, 14, 21, 26, 30, 32, 36, 37, 38, 39, 41, 43, 50, 51,
 53, 56, 59, 62, 65, 66, 81, 82, 92, 93, 95, 97, 98, 99, 100, 103,
 104, 108, 109, 112, 117, 119, 124, 130, 137, 139]
 the alpha for none is =
 [0, 2, 4, 6, 7, 8, 9, 10, 11, 13, 15, 16, 17, 18, 19, 20, 22, 23,
 24, 25, 27, 28, 29, 31, 33, 34, 35, 40, 42, 44, 45, 46, 47, 48, 4
 9, 52, 54, 55, 57, 58, 60, 61, 63, 64, 67, 68, 69, 70, 71, 72, 73
 , 74, 75, 76, 77, 78, 79, 80, 83, 84, 85, 86, 87, 88, 89, 90, 91,
 94, 96, 101, 102, 105, 106, 107, 110, 111, 113, 114, 115, 116, 11
 8, 120, 121, 122, 123, 125, 126, 127, 128, 129, 131, 132, 133, 13
 4, 135, 136, 138, 140, 141, 142, 143, 144, 145, 146, 147, 148, 14
 9]
the alpha for 0.32 is =
[1, 26, 30, 36, 39, 62, 65, 66, 98, 99, 100, 103, 109, 124]
the alpha for none is =
[0, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 1
9, 20, 21, 22, 23, 24, 25, 27, 28, 29, 31, 32, 33, 34, 35, 37, 38
, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55,
56, 57, 58, 59, 60, 61, 63, 64, 67, 68, 69, 70, 71, 72, 73, 74, 7
5, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91
, 92, 93, 94, 95, 96, 97, 101, 102, 104, 105, 106, 107, 108, 110,
111, 112, 113, 114, 115, 116, 117, 118, 119, 120, 121, 122, 123,
125, 126, 127, 128, 129, 130, 131, 132, 133, 134, 135, 136, 137,
138, 139, 140, 141, 142, 143, 144, 145, 146, 147, 148, 149]
```

```
the alpha for 0.4 is =
[1, 2, 3, 4, 5, 6, 7, 10, 11, 12, 13, 14, 15, 16, 17, 18, 20, 21,
22, 23, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 3
9, 40, 41, 42, 43, 44, 45, 47, 49, 50, 51, 52, 53, 55, 56, 57, 58
, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 70, 71, 72, 73, 75, 76,
77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 90, 92, 93, 94, 9
5, 96, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106, 108, 109, 1
10, 111, 112, 114, 115, 116, 117, 118, 119, 120, 121, 122, 123, 1
24, 125, 126, 127, 128, 129, 130, 131, 132, 133, 134, 137, 138, 1
39, 140, 141, 142, 143, 145, 146, 147, 148, 149]
the alpha for none is =
[0, 8, 9, 19, 24, 46, 48, 54, 69, 74, 89, 91, 107, 113, 135, 136,
144]
the alpha for 0.5 is =
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 1
9, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35
, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51,
52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 6
8, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84
, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100
, 101, 102, 103, 104, 105, 106, 107, 108, 109, 110, 111, 112, 113
, 114, 115, 116, 117, 118, 119, 120, 121, 122, 123, 124, 125, 126
, 127, 128, 129, 130, 131, 132, 133, 134, 135, 136, 137, 138, 139
, 140, 141, 142, 143, 144, 145, 146, 147, 148, 149]
the alpha for none is =
[0]
```

The cuts above are example of what the calcAlphacut function has created.