Integrating Fuzzy *c*-Means Clustering with PostgreSQL[1]

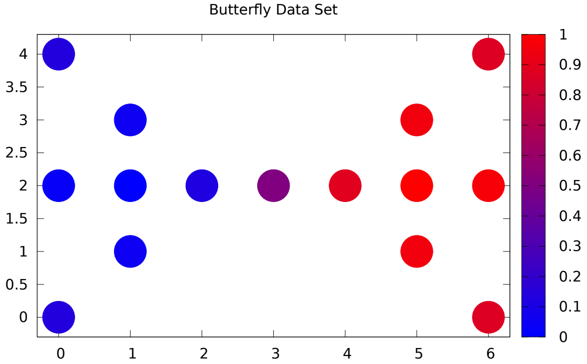
Ruslan Miniakhmetov

South Ural State University

June 2, 2011

[1] This paper is supported by the Russian Foundation for Basic Research (grant No. 09-07-00241-a).

Fuzzy Clustering



In fuzzy clustering we can observe that membership function can take non-integer values which represents probabilities. At the Figure above, blue points mostly belong to one cluster and red to another, their probabilities showed with shades of colors.

Fuzzy c-Means

The *Fuzzy c-Means (FCM)* algorithm provides *fuzzy* clusterization of data sets. Fuzzy partitioning is carried out through an iterative minimization of the *objective function* J_{FCM}:

$$J_{FCM}(X, k, m) = \sum_{i=1}^{N} \sum_{j=1}^{k} u_{ij}^{m} \rho^{2}(x_i, c_j), \quad 1 < m < \infty$$

Where $x_i \in X \subset \mathbb{R}^d$ — i-th data vector from input set of data $X, |X| = N$; $c_j \in C$ — center of cluster j, d-dimensional vector (*centroid*), $C, |C| = k$ — set of all centroids; u_{ij} — is a membership degree between vector x_i and cluster j; m — fuzzyfication degree of objective function; $\rho(x_i, c_j)$ — distance function, defines a membership degree between vector x_i and cluster j.

Fuzzy *c*-Means: Algorithm

```
Input: $X, k, m, \varepsilon$
Output: $U$
1: {initialization}
   $s := 0, U^{(0)} := (u_{ij})$
2: repeat
3:     {computation of new centroids' coordinates}
       Compute $C^{(s)} := (c_j)$ using formula $c_{jl} = \frac{\sum_{i=1}^{n} u_{ij}^m \cdot x_{il}}{\sum_{i=1}^{n} u_{ij}^m}$
       where $u_{ij} \in U^{(s)}$
4:     {update matrixes values}
       Compute $U^{(s)}$ and $U^{(s+1)}$ using
       formula $u_{ij} = \sum_{t=1}^{k} \left( \frac{\rho(x_i, c_j)}{\rho(x_i, c_t)} \right)^{\frac{2}{1-m}}$
5:     $s := s + 1$
6: until $\max_{ij}\{|u_{ij}^{(s)} - u_{ij}^{(s-1)}|\} \leqslant \varepsilon$
```

pgFCM: Relation Structure

Input data set

Matrix X

$$\begin{array}{c|ccc} & x_1 & \cdots & x_d \\ \hline 1 & 1.0 & \cdots & 2.1 \\ \vdots & \vdots & \ddots & \vdots \\ n & 3.4 & \cdots & 2.9 \end{array} \Longrightarrow$$

Table SH (horizontal)

i	x_1	$\cdots$	x_d
1	1.0	$\cdots$	2.1
$\vdots$	$\vdots$	$\ddots$	$\vdots$
n	3.4	$\cdots$	2.9

Table SV (vertical)

i	l	val
1	1	1.0
$\vdots$	$\vdots$	$\vdots$
n	d	2.9

Centroids' coordinates

Matrix C

$$\begin{array}{c|ccc} & x_1 & \cdots & x_d \\ \hline 1 & 2.2 & \cdots & 8.1 \\ \vdots & \vdots & \ddots & \vdots \\ k & 3.4 & \cdots & 6.9 \end{array} \Longrightarrow$$

Table C

j	l	val
1	1	2.2
$\vdots$	$\vdots$	$\vdots$
k	d	6.9

Membership degrees

Matrix U

$$\begin{array}{c|ccc} & 1 & \cdots & k \\ \hline 1 & 0.2 & \cdots & 0.1 \\ \vdots & \vdots & \ddots & \vdots \\ n & 0.8 & \cdots & 0.1 \end{array} \Longrightarrow$$

Table U (for step s)

i	j	val
1	1	0.2
$\vdots$	$\vdots$	$\vdots$
n	k	0.1

Table UT (for step s+1)

i	j	val
1	1	0.2
$\vdots$	$\vdots$	$\vdots$
n	k	0.1

pgFCM: Relation Tables

No.	Table	Semantics	Columns	No. of rows
1	SH	training set for data vectors (horizontal form)	$\underline{i}, x1, x2, \ldots, xd$	n
2	SV	training set for data vectors (vertical form)	$\underline{i, l}, val$	$n \cdot d$
3	C	centroids' coordinates	$\underline{j, l}, val$	$k \cdot d$
4	SD	distances between x_i and c_j	$\underline{i, j}, dist$	$n \cdot k$
5	U	degree of membership vector x_i to a cluster c_j on step s	$\underline{i, j}, val$	$n \cdot k$
6	UT	degree of membership vector x_i to a cluster c_j on step $s+1$	$\underline{i, j}, val$	$n \cdot k$
7	P	result of computation function $\delta = \max_{ij}\{\lvert u_{ij}^{(s+1)} - u_{ij}^{(s)}\rvert\}$ on step s	$\underline{d, k, n}, s, delta$	s

pgFCM: Subscripts

Subscript	Range	Semantics
i	$\overline{1, n}$	vector subscript
j	$\overline{1, k}$	cluster subscript
l	$\overline{1, d}$	vector's coordinate subscript

pgFCM: Algorithm

```
Input: $SH, k, m, eps$
Output: $U$
1: {initialization}
   Create and initialize temporary tables ($U, P, SV$, etc.)
2: repeat
3:    {computations}
4:    Compute centroids coordinates. Update table $C$.
5:    Compute distances $\rho(x_i, c_j)$. Update table $SD$.
6:    Compute membership degrees $UT = (ut_{ij})$.
      Update table $UT$.
7:    {update}
      Update tables $P$ and $U$.
8:    {check for termination}
9: until $P.delta \leqslant eps$
```

pgFCM: Initialization Phase

```
I1: INSERT INTO SV
      SELECT SH.i, 1, x1 FROM SH;
    ...
    INSERT INTO SV
      SELECT SH.i, d, xd FROM SH;

I2: INSERT INTO U (i, j, val)
      VALUES (1, 1, random());
    ...
    INSERT INTO U (i, j, val)
      VALUES (n, k, random());
```

pgFCM: Initialization Phase

```
I3: UPDATE U SET val = val / U1.tmp
    FROM (SELECT i, sum(val) AS tmp
         FROM U
         GROUP BY i) AS U1
    WHERE U1.i = U.i;

I4: INSERT INTO P(d, k, n, s, delta)
      VALUES (d, k, n, 0, 0.0);
```

pgFCM: Computation Phase

```
C1: INSERT INTO C
      SELECT R1.j, R1.l, R1.s1 / R2.s2 AS val
      FROM    (SELECT j, l, sum(U.val^m * SV.val) AS s1
               FROM U, SV
               WHERE U.i = SV.i
               GROUP BY j, l) AS R1,
              (SELECT j, sum(U.val^m) AS s2
               FROM U
               GROUP BY j) AS R2
      WHERE R1.j = R2.j;

C2: INSERT INTO SD
      SELECT i, j, sqrt(sum((SV.val - C.val)^2))) AS dist
      FROM SV, C
      WHERE SV.l = C.l
      GROUP BY i, j;
```

pgFCM: Computation Phase

```
C3:  INSERT INTO UT
       SELECT SD.i, j,
              SD.dist^(2.0/(1.0-m)) * SD1.den AS val
       FROM (SELECT i,
                    1.0 / sum(dist^(2.0/(1.0-m))) AS den
            FROM SD
            GROUP BY i) AS SD1, SD
       WHERE SD.i = SD1.i;
```

pgFCM: Update Phase

```
U1: INSERT INTO P
      SELECT L.d, L.k, L.n, L.s + 1 AS s,
             E.delta
      FROM (SELECT i, j,
                      max(abs(UT.val - U.val)) AS delta
            FROM U, UT
            GROUP BY i, j) AS E,
           (SELECT d, k, n, max(s)
            FROM P
            GROUP BY d, k, n) AS L) AS R

U2:  TRUNCATE U;
     INSERT INTO U
       SELECT * FROM UT;
```

pgFCM: Check Phase

```
CH1: SELECT delta INTO tmp
     FROM P, (SELECT d, k, n, max(s) AS max_s
              FROM P
              GROUP BY d, k, n) AS L
     WHERE P.s = L.max_s AND P.d = L.d
     AND P.k = L.k AND P.n = L.n;

CH2: IF (tmp < eps) THEN
       RETURN;
     END IF;
```

pgFCM: Test on ‘Butterfly’ Data Set

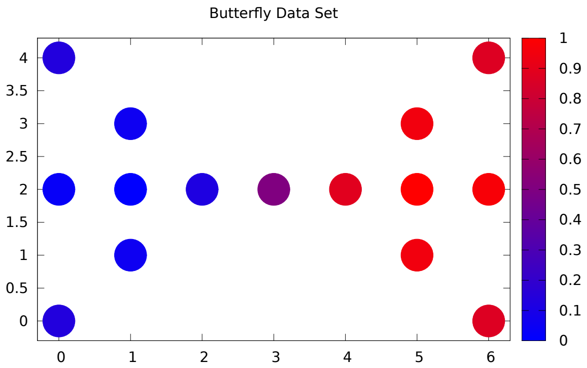


Table U

i	j	val
1	1	0.86
1	2	0.13
2	1	0.97
2	2	0.02
...	...	...
8	1	0.49
8	2	0.50
...	...	...
15	1	0.13
15	2	0.86

Conclusion

- We propose pgFCM algorithm which implements Fuzzy *c*-Means clustering algorithm and processes data stored in relational tables using PostgreSQL open-source DBMS.
- Future directions is developing a parallel version of pgFCM for distributed memory multiprocessors.

Thank You

Thanks for attention.

Contacts with author: `tavein@gmail.com`

Related Work

- Sarawagi, S. and Thomas, S. and Agrawal, R. Integrating association rule mining with relational database systems: alternatives and implications.
- Clear, J. and Dunn, D. and Harvey, B. and Heytens, M. and Lohman, P. and Mehta, A. and Melton, M. and Rohrberg, L. and Savasere, A. and Wehrmeister, R. and Xu, M. NonStop SQL/MX primitives for knowledge discovery.
- Graefe, G. and Fayyad, U. M. and Chaudhuri, S. On the Efficient Gathering of Sufficient Statistics for Classification from Large SQL Databases.
- Ordonez, C. Integrating K-Means Clustering with a Relational DBMS Using SQL.
- Ordonez, C. Programming the K-means clustering algorithm in SQL.

Contribution

The main contribution of the paper is an extension of results presented in Ordonez's works for the case where data vectors may belong to several clusters. Such a case is very important in problems connected with medicine data analysis.

- Shihab, A. I. Fuzzy Clustering Algorithms and their Applications to Medical Image Analysis
- Zhang, D. and Chen, S. A Novel Kernelized Fuzzy c-Means Algorithm with Application in Medical Image Segmentation