Online Fuzzy-C-Means clustering

Authors: Dezső Kancsár, Ágnes B. Simon
Author’s Addresses: H-1157 Budapest, Nyírpalota u. 79/C 2/8; College of Nyíregyháza, Rákóczi út 69.
Contact: berszoft@ax.hu, simona@nyf.hu

Abstract
In this paper you will find short information about the program of Online Fuzzy-Clustering. To understand the background processes, it gives a summary of the architecture and basics of the communications, to positioning it shows the software and hardware requirements of both: the server and the client side. Because the practice is the best way to usage, it guides on the whole process that a new ‘customer’ has to do while presents some screenshots of the working program surface. And finally it also provides some working interest and inside tricks. This documentation will not touch upon the mathematical issue of the applied Fuzzy C-means Clustering Algorithm although it based on it [FCM]. It concentrates on the technical implementation.

1. Technical summary

1.1. Architecture and Communication basics
If we want to categorise this application than we can say it can deliver its service across the Web; it is based on 3-tier (layer) architecture logic: the Client-tier, the Application Logic-tier, and the Data-tier. (Figure 1.) These tiers are separated basically on the understanding its functionality instead of the location of them. In the present working configuration the Application Logic and the Data-tiers are installed on the same machine, and of cause a Client-tier also can start from this machine.

![Figure 1. The 3-tier Architecture](image-url)
The communication between the Client and the Application Logic-tier based on industry standards HTTP protocol without any encryption, but the HTTPS protocol can supplement without changing any of the core application code.

In the Client-tier a Web Browser downloads the dynamically generated web pages, and sends them back to the Application-tier with the filled input fields in it for next pages to get [Apache]+[Tomcat]. The Application-tier collects, checks and interprets the information from the Client-tier and the Data-tier, as well as it executes commands and runs the main function: the fuzzy clustering algorithm with the specified parameters and data [Tomcat]+[FCM]. Finally it also generates dynamic web pages for the actual user with the actual information (data and/or input fields) on it [Tomcat].

The communication between the Application Logic and the Data-tier is based on the standard ‘jdbc’ driver.

1.2. Software and hardware requirements

Because the applied component, there are some software and hardware requirements before we can start work with the Fuzzy on-Line Application. At first, we have to install the Data and the Application Logic-tier (back-end) than we have to start these services before we can get contact them from an installed Client.

1.2.1. Back-end requirements

The server implements the Application Logic-tier needs to be installed with a Web-server [Apache], a Java Software Development Kit [Java], and a Java Server Pages Engine [Tomcat]. For the database server connection the Java environment has to be extended with the jdbc-driver of the applied database server, too.

The server implements the Data-tier needs to be installed with same database server [MySQL], which is configured at the Application Logic-tier and needs to be configured for it. (Database, tables, accounts, grants…)

All of the listed software components also have software and hardware requirements, the Fuzzy on-Line Application has no more extension to them; its recommendations can be followed. All of them installable on several OS including Linux, Windows 98/NT/2000…

1.2.2. Front-end requirements

In the client-side there are much easier things to do, only the Web Browser has to be installed; unfortunately the present systems is optimised only for the Internet Explorer 4.0 or above with JavaScript and cookie acceptances.
1.3. The properties of the present working system

The present working system is located in the College of Nyíregyháza based on the college’s hardware and infrastructure. The server machine has an Intel Pentium 233MHz CPU, 128MB RAM, 10/100 MB Network card installed in the public domain network; DN: pingvin.nyf.hu, and the application’s service works on port: 8080. The Application Logic and the Data-tiers are implemented on the same machine (this server). All of the installed software components are under the GNU/Open Source/Freeware licence/policy:

- OS: Debian-Linux;
- Web-server: Apache v1.3.39 [Apache];
- JSDK: Sun J2SE v1.3.1 [Java];
- JSP Engine: Tomcat 3.3.1 [Tomcat];
- Database Server: MySQL 3.23.37 [MySQL];
- Application Source: Java [FCM].

2. Using the Fuzzy on-Line Application

There are two different levels of the Fuzzy on-Line Application’s usage. The first level is called ‘demo’ where some parameters and the size of the input data matrix are limited. In the second level is called ‘full’ where are no limitations for any parameters or data. The ‘demo’ level is open for everyone who has already registered (the registration very easy and it is free), while the ‘full’ level is open for only members. The both levels having run the same application, the only difference is the limitations.

2.1. Registration

Every new user has to be registered before she/he can start the Fuzzy on-Line Application. (Figure 2.) In the registration form there are some mandatory and some optional fields, and every new user has to be asked for a unique username before her/his data will be acceptable by the system. Only successfully registered users can come in to the next pages. The username and the password set before have to repeat all the times she/he comes back to the login page. The user’s information will be saved in the user table for future contact.
2.2. The User’s own Calculations

There is one predefined (demo) data available for everyone, based on artificial data set was originally published by J. C. Bezdek, and every new calculation/recalculation also will be listed in the user’s calculations table. This nice feature of the system is available right after the login screen when has to be decided if a recalculation from the list or new calculation will be started. (Figure 3.) There is a possibility for every user to name her/his calculations in the input parameter section and reuse a complete data table more than once, and/or recalculate them with another parameter setting and/or change the input data table. To have a look at once more to the results of a finished calculation is also available for all of the listed calculations.
2.3. Preparing Parameters and Data

The next step is to set the parameters and the input data want to be grouped. While changing some parameters like number of features and/or objects, the input page can be reloaded to redraw the input data table that’s way it is recommended to set the parameters first than set the input data. (Figure 4.) If any of the parameters or the input data will not be filled up the Browser (the Client-tier) warns the user to correct before the page to be sent to the server (the Application-tier)
2.4. Confirming

Before the actual calculations starts the user is asked to confirm the input parameters and data the server received. There is one more chance to correct any of them if it does not fit to the user needs. Pushing the ‘Start’ button, the server will pass all of this input information to the database server (the Data-tier) and invokes the calculating algorithm.

2.5. Calculating

During the calculations the Client automatically refresh its screen to check the status of the running algorithm and to show the actual process rate of the calculations.

2.6. Summary of the Results

If the calculation process finishes, the user’s refresh request automatically pass to the Result of the calculations page (Figure 5.), including the main parameters of the FCM algorithm, the Membership Matrix, the Cluster Centres’ Matrix and once more the Input parameters and data table. While all of these information automatically saved for future analysis in the database server than the validation is one of the next user’s tasks before the final conclusions.
3. Secret of the trade

3.1. Loading and Performance

The implemented software (the Application-tier) has no load balancing and/or fail over component. In the other hand the other applied software components (like OS, DB, JSP…) provides several opportunities to have minimal or higher fail over and/or load balancing feature: like raiserfs file system, db tuning, JSP concurrent user account limitation, session time… Why java can devour the memory of the system, more memory always can be a good choice ;-) 

3.1.1. Sessions

Every client who opens any of the pages from the server (downloads a Web-page) in her/his Browser’s window invokes a new session in the server-side. One user has one session for one open Browser’s window, but the same session belongs to her/him while she/he goes through the pages of the Fuzzy on-Line Application on the same Browser’s window. The sessions have
unique identifications and every session automatically ends if there is no activity from the client-side for more than half an hour.

3.1.2. Threads

Every session what starts the FCM calculations also starts a new thread for it in the JVM (Java Virtual Machine). In this case the session has its new thread if the user has confirmed the input parameters and table. There is only one ‘calculating-thread’ for a session in its lifecycle even if the user starts a new calculations/recalculations with the same session.

3.2. Database tricks

While a new user registers herself/himself the application pass her/his data to the database server and creates a new role for her/him, as well. The roles are the other technique of the applied security techniques (the first are the username/password). With roles the user can be grouped and one user can be a member of more than one group, too. One of the groups has rights to run the ‘full’ version of the application. The basic role what every successfully registered user will get the ‘demouser’ (a right to run the enter the demo application zone).

The precision of the number drawing under the calculations is different than the precision of it in the database. During the calculation the java double type is used, while the demo database was configured to store the data as decimal (20,10) and decimal (15,5).

4. Future plans

In the near future we plan to integrate an SVG and/or Chart based graphical summeries into the Results of Calculations, and to implement an XLS type input data entry and output copy as well. These developments are also plan to base on GNU/Open source projects using the POI, JFreeChart and SVG. Every comment is welcome! Have fun and see you at our demo pages!

5. References


[Tomcat] - Tomcat Documentation (V3.3.1): The Apache Software Foundation