聚类之 FCM 算法原理及应用

Arrow Luo

2016 年 9 月 21 日

1 原理部分

模糊 C 均值（Fuzzy C-means）算法简称 FCM 算法，是一种基于目标函数的模糊聚类算法，主要用于数据的聚类分析。

首先介绍一下模糊这个概念，所谓模糊就是不确定，确定性是指：东西是什么那就是什么，而不确定性是指：东西就说很像什么。比如说把 20 岁作为年轻不年轻的标准，那么一个人 21 岁按照确定性的划分就属于不年轻，而我们印象中的观念是 21 岁也很年轻，这个时候可以模糊一下，认为 21 岁有 0.9 分像年轻，有 0.1 分像不年轻，这里 0.9 与 0.1 不是概率，而是一种相似的程度，把这种一个样本属于结果的这种相似的程度称为样本的隶属度，一般用 u 表示，表示一个样本相似于不同结果的一个程度指标。

基于此，假定数据集为 X，如果把这些数据划分成 c 类的话，那么对应的就有 c 个类中心为 C，每个样本 j 属于某一类 i 的隶属度为 u_{ij}，那么定义一个 FCM 目标函数（1）及其约束条件（2）如下所示。

$$J = \sum_{i=1}^{c}\sum_{j=1}^{n} u_{ij}^{m}\|x_j - c_i\|^2 \tag{1}$$

$$\sum_{i=1}^{c} u_{ij} = 1, j = 1, 2 \cdots, n \tag{2}$$

看一下目标函数（1）可知，由相应样本的隶属度与该样本到各个类中心的距离相乘组成的，m 是一个隶属度的因子，个人理解为属于样本的轻缓程度，就像 x^2 与 x^3 这种一样。式（2）为约束条件，也就是一个样本属于所有类的隶属度之和要为 1。观察式（1）可以发现，其中的变量有 u_{ij}、c_i，并且还有约束条件，那么如何求这个目标函数的极值呢？

这里首先采用拉格朗日乘数法将约束条件拿到目标函数中去，前面加上系数，并把式（2）的所有 j 展开，那么式（1）变成下列所示：

$$J = \sum_{i=1}^{c}\sum_{j=1}^{n} u_{ij}^{m}\|x_j - c_i\|^2 + \lambda_1(\sum_{i=1}^{c} u_{i1} - 1) + \cdots + \lambda_j(\sum_{i=1}^{c} u_{ij} - 1) + \cdots + \lambda_n(\sum_{i=1}^{c} u_{in} - 1) \tag{3}$$

现在要求该式的目标函数极值，那么分别对其中的变量 u_{ij}、c_i 求导数，首先对 u_{ij} 求导。

分析式 (3)，先对第一部分的两级求和的 u_{ij} 求导，对求和形式下如果直接求导不熟悉，可以把求和

展开如下：

$$\begin{bmatrix} u_{11}^m\|x_1-c_1\|^2 & \cdots & u_{1j}^m\|x_j-c_1\|^2 & \cdots & u_{1n}^m\|x_n-c_1\|^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ u_{i1}^m\|x_1-c_i\|^2 & \cdots & u_{ij}^m\|x_j-c_i\|^2 & \cdots & u_{in}^m\|x_n-c_i\|^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ u_{c1}^m\|x_1-c_c\|^2 & \cdots & u_{cj}^m\|x_j-c_c\|^2 & \cdots & u_{cn}^m\|x_n-c_c\|^2 \end{bmatrix}$$

这个矩阵要对 u_{ij} 求导，可以看到只有 u_{ij} 对应的 $u_{ij}^m\|x_j-c_i\|^2$ 保留，其他的所有项中因为不含有 u_{ij}，所以求导都为 0。那么 $u_{ij}^m\|x_j-c_i\|^2$ 对 u_{ij} 求导后就为 $m\|x_j-c_i\|^2u_{ij}^{m-1}$

再来看后面那个对 u_{ij} 求导，同样把求和展开，再去除和 u_{ij} 不相关的（求导为 0），那么只剩下这一项：$\lambda_j(u_{ij}-1)$，它对 u_{ij} 求导就是 λ_j 了。

那么最终 J 对 u_{ij} 的求导结果并让其等于 0 就是：

$$\frac{\partial J}{\partial u_{ij}} = m\|x_j-c_i\|^2u_{ij}^{m-1}+\lambda_j=0$$

这个式子化简下，将 u_{ij} 解出来就是：

$$u_{ij}=\left(\frac{-\lambda_j}{m}\right)^{\frac{1}{m-1}}\left(\frac{1}{\|x_j-c_i\|^{(\frac{2}{m-1})}}\right) \tag{4}$$

要解出 u_{ij} 则需要把 λ_j 去掉才行。这里重新使用公式（2）的约束条件，并把算出来的 u_{ij} 代入式（2）中有：

$$1=\sum_{i=1}^{c}u_{ij}=\sum_{i=1}^{c}\Big(\frac{-\lambda_j}{m}\Big)^{\frac{1}{m-1}}\Big(\frac{1}{\|x_j-c_i\|^{(\frac{2}{m-1})}}\Big)=\Big(\frac{-\lambda_j}{m}\Big)^{\frac{1}{m-1}}\sum_{i=1}^{c}\Big(\frac{1}{\|x_j-c_i\|^{(\frac{2}{m-1})}}\Big)$$

这样就有（其中把符号 i 换成 k）：

$$\Big(\frac{-\lambda_j}{m}\Big)^{\frac{1}{m-1}}=\frac{1}{\sum\limits_{i=1}^{c}\Big(\frac{1}{\|x_j-c_i\|^{(\frac{2}{m-1})}}\Big)}=\frac{1}{\sum\limits_{k=1}^{c}\Big(\frac{1}{\|x_j-c_k\|^{(\frac{2}{m-1})}}\Big)}$$

把这个重新代入到式（4）中有：

$$\begin{aligned} u_{ij}&=\left(\frac{1}{\sum\limits_{k=1}^{c}\Big(\frac{1}{\|x_j-c_k\|^{(\frac{2}{m-1})}}\Big)}\right)\left(\frac{1}{\|x_j-c_i\|^{(\frac{2}{m-1})}}\right) \\ &=\frac{1}{\sum\limits_{k=1}^{c}\left(\frac{\|x_j-c_i\|}{\|x_j-c_k\|}\right)^{(\frac{2}{m-1})}} \end{aligned} \tag{5}$$

好了，式子（5）就是最终的 u_{ij} 迭代公式。

下面在来求 J 对 c_i 的导数。由公式（2）可以看到只有 $\sum\limits_{i=1}^{c}\sum\limits_{j=1}^{n}u_{ij}^m\|x_j-c_i\|^2$ 这一部分里面含有 c_i，对其二级求和展开如前面所示的，那么它对 c_i 的导数就是：

$$\frac{\partial J}{\partial c_i}=\sum_{j=1}^{n}(-u_{ij}^m*2*(x_j-c_i))=0$$

解出：

$$c_i = \frac{\sum\limits_{j=1}^{n}(x_j u_{ij}^m)}{\sum\limits_{j=1}^{n} u_{ij}^m} \tag{6}$$

公式（6）就是类中心的迭代公式。

我们发现 u_{ij} 与 c_i 是相互关联的，彼此包含对方，所以程序开始的时候我们随便赋值给 u_{ij} 或者 c_i 其中的一个，只要数值满足条件即可。然后就开始迭代，比如一般的都赋值给 u_{ij}，那么有了 u_{ij} 就可以计算 c_i，然后有了 c_i 又可以计算 u_{ij}，反反复复，在这个过程中还有一个目标函数 J 一直在变化，逐渐趋向稳定值。那么当 J 不在变化的时候就认为算法收敛到一个比较好的结了。可以看到 u_{ij} 和 c_i 在目标函数 J 下似乎构成了一个正反馈一样，这一点很像 EM 算法，先 E 在 M，有了 M 在 E，在 M 直至达到最优。

式（5），（6）是算法的关键。现在来重新从宏观的角度来整体看看这两个公式，先看（5）式

$$u_{ij} = \frac{1}{\sum\limits_{k=1}^{c}\left(\frac{\|x_j - c_i\|}{\|x_j - c_k\|}\right)^{\left(\frac{2}{m-1}\right)}}$$

现在假设样本集中有三个类即 $c = 3$(假设指数 $m = 3$)，如图 1 求样本 j 到三个类中心的隶属度

$$u_{1j} = \frac{1}{\frac{x_j - c_1}{x_j - c_1} + \frac{x_j - c_1}{x_j - c_2} + \frac{x_j - c_1}{x_j - c_3}} = \frac{(x_j - c_2)(x_j - c_3)}{(x_j - c_1)(x_j - c_2) + (x_j - c_1)(x_j - c_3) + (x_j - c_2)(x_j - c_3)}$$

$$u_{2j} = \frac{1}{\frac{x_j - c_2}{x_j - c_1} + \frac{x_j - c_2}{x_j - c_2} + \frac{x_j - c_2}{x_j - c_3}} = \frac{(x_j - c_1)(x_j - c_3)}{(x_j - c_1)(x_j - c_2) + (x_j - c_1)(x_j - c_3) + (x_j - c_2)(x_j - c_3)}$$

$$u_{3j} = \frac{1}{\frac{x_j - c_3}{x_j - c_1} + \frac{x_j - c_3}{x_j - c_2} + \frac{x_j - c_3}{x_j - c_3}} = \frac{(x_j - c_1)(x_j - c_2)}{(x_j - c_1)(x_j - c_2) + (x_j - c_1)(x_j - c_3) + (x_j - c_2)(x_j - c_3)}$$

可以看出 u_{ij} 分母相同，u_{ij} 越大，则分子越大，表示到 i 之外的类的距离乘积越大，即越远离其他类而靠近 i 类。

再来宏观看看公式（6），考虑当类 i 确定后，式（6）的分母求和其实是一个常数，那么式（6）可以写成：

$$c_i = \frac{\sum\limits_{j=1}^{n}(x_j u_{ij}^m)}{\sum\limits_{j=1}^{n} u_{ij}^m} = \sum_{j=1}^{n} \frac{u_{ij}^m}{\sum\limits_{k=1}^{n} u_{ik}^m} x_j$$

这是类中心的更新法则。这个公式本质上就是对样本点加权平均，类 i 确定后，首先将所有点到该类的隶属度 u 求和，然后对每个点，隶属度除以这个和就是所占的比重，乘以 x_j 就是这个点对于这个类 i 的贡献值。假设样本集中类 i 和 6 个样本点如图 2所示。

$$c_i = \frac{u_{i1}}{u_{i1} + u_{i2} + \cdots + u_{i6}} x_1 + \frac{u_{i2}}{u_{i1} + u_{i2} + \cdots + u_{i6}} x_2 + \cdots + \frac{u_{i6}}{u_{i1} + u_{i2} + \cdots + u_{i6}} x_6$$

由上述的宏观分析可知，这两个公式的迭代关系式是可以理解的。

2 简单程序实现

下面我们在 matlab 下用最基础的循环实现上述的式（5）与式（6）的 FCM 过程。首先，我们需要产生可用于 FCM 的数据，为了可视化方便，我们产生一个二维数据便于在坐标轴上显示，也就是每个样

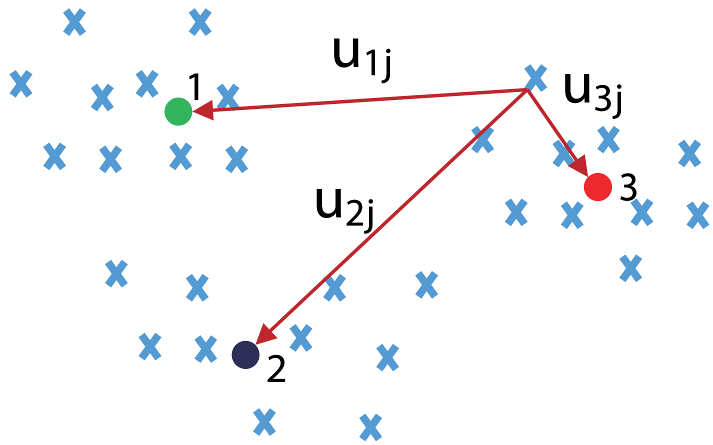

图 1. FCM 隶属度直观解释

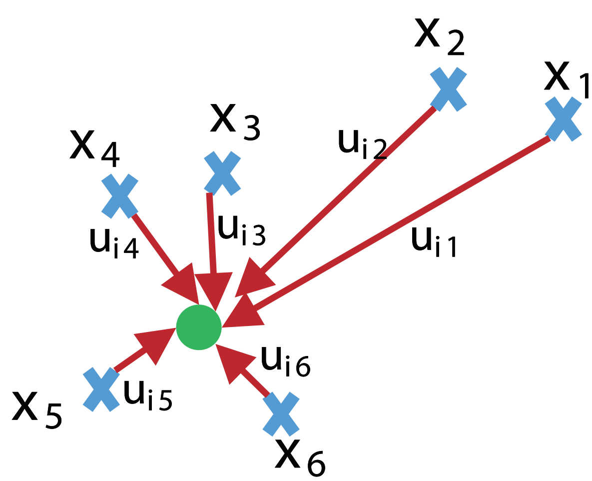

图 2. FCM 中心点直观解释

本由两个特征（或者 x 坐标与 y 坐标构成），生成 100 个这样的点，当然我们在人为改变一下，让这些点看起来至少属于不同的类。生成的点画出来如图 3。

那么我们说 FCM 算法的一般步骤为：

(1) 确定分类数，指数 m 的值，确定迭代次数（这是结束的条件，当然结束的条件可以有多种）。

(2) 初始化一个隶属度 U（注意条件—和为 1）；

(3) 根据 U 计算聚类中心 C；

(4) 计算目标函数 J

(5) 根据 C 返回去计算 U，回到步骤 (3)，一直循环直到结束。

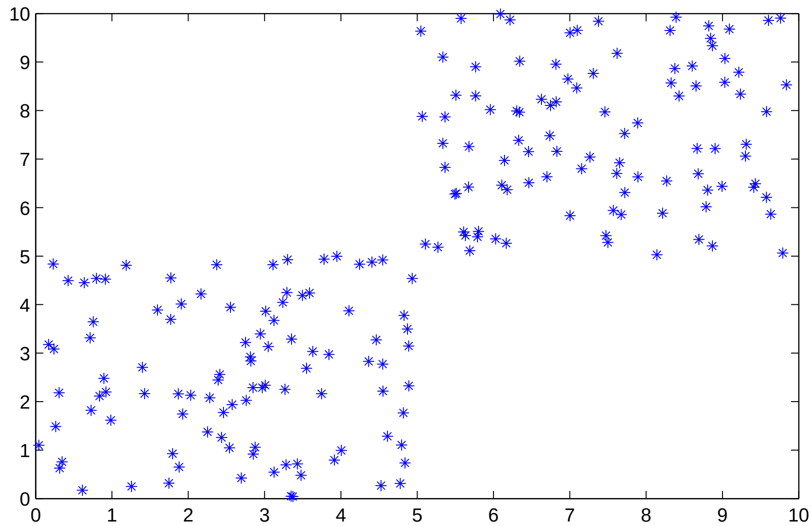

图 3. FCM 运行数据

得到结果如图 4。

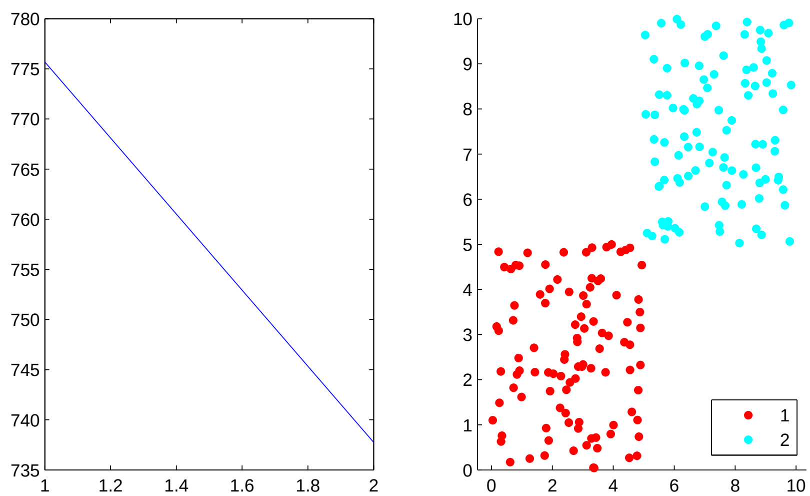

图 4. FCM 运行结果

还需要说一点的是，当程序结束后，怎么去判断哪个点属于哪个类呢？在结束后，肯定有最后一次计算的 U 吧，对于每一个点，它属于各个类都会有一个 u，那么找到其中的最大的 u 就认为这个点就属于这一类。基于此一个基础的程序如下：

```
clc
clear
close all
%for i=1:30
```

```
    x1(i) = rand()*5;
    y1(i) = rand()*5;
    x2(i) = rand()*5 + 5;
    y2(i) = rand()*5 + 5;
end
x = [x1,x2];
y = [y1,y2];
data = [x;y];
data = data'; %每一行代表一个样本
%plot(data(:,1),data(:,2),'*');


cluster_n = 2;              %类别数
iter = 50;                  %迭代次数
min_impro = 1e-5;            %收敛精度
m = 3;                      %指数

num_data = size(data,1);    % 样本个数
num_d = size(data,2);       % 样本维度

%-初始化隶属度 u，条件是每一列和为 1
U = rand(cluster_n, num_data);
col_sum = sum(U);
U = U./col_sum(ones(cluster_n,1),:);
%for i = 1:iter
    %更新 c
    for j = 1:cluster_n
        u_ij_m = U(j,:).^m;
        sum_u_ij = sum(u_ij_m);
        sum_1d = u_ij_m./sum_u_ij;
        c(j,:) = u_ij_m*data./sum_u_ij;
    end
    %更新 U
    for j = 1:cluster_n
        for k = 1:num_data
            sum1 = 0;
            for j1 = 1:cluster_n
                temp = (norm(data(k,:)-c(j,:))/norm(data(k,:)-c(j1,:))).^(2/(m-1));
                sum1 = sum1 + temp;
            end
            U(j,k) = 1./sum1;
        end
    end
    %-计算目标函数 J
    temp1 = zeros(cluster_n,num_data);
    for j = 1:cluster_n
        for k = 1:num_data
            temp1(j,k) = U(j,k)^m*(norm(data(k,:)-c(j,:)))^2;
        end
    end
    J(i) = sum(sum(temp1));
    if i>1
        if abs(J(i)-J(i-1) < min_impro)
```

```
            break;
        end;
    end;
end
figure;
subplot(1,2,1),plot(J);
[~,label] = max(U); %找到所属的类
subplot(1,2,2);
gscatter(data(:,1),data(:,2),label)
```

3 参考

1、on2way. 聚类之详解 FCM 算法原理及应用 [EB/OL].
http://blog.csdn.net/on2way/article/details/47087201
2、李政軒. Fuzzy C-Means 迭代公式推導 [EB/OL]. http://www.powercam.cc/slide/20878
注：有改动