聚类——WKFCM的matlab程序

作者：凯鲁嘎吉 - 博客园 http://www.cnblogs.com/kailugaji/

在聚类——WKFCM文章中已介绍了WKFCM算法的理论知识，现在用matlab进行实现,下面这个例子是用FCM初始化聚类中心，也可以随机初始化聚类中心。

1.matlab程序

WKFCM_main.m

```
%function [ave_acc_WKFCM,max_acc_WKFCM,min_acc_WKFCM,ave_iter_WKFCM,ave_run_time]=WKFCM_main(X,real_label,K)
function [ave_acc_WKFCM,max_acc_WKFCM,min_acc_WKFCM,ave_iter_FCM,ave_iter_WKFCM,ave_run_time]=WKFCM_main(X,real_label,K)
%输入K:聚的类，max_iter是最大迭代次数,T:遗传算法最大迭代次数，n:种群个数 X：未归一化
%输出ave_acc_KFCM：迭代max_iter次之后的平均准确度,iter:实际KFCM迭代次数
t0=cputime;
max_iter=20;
s=0;
s_1=0;
s_2=0;
accuracy=zeros(max_iter,1);
iter_WKFCM_t=zeros(max_iter,1);
iter_FCM_t=zeros(max_iter,1);
%对data做最大-最小归一化处理
% [data_num,~]=size(data);
% X=(data-ones(data_num,1)*min(data))./(ones(data_num,1)*(max(data)-min(data)));
for i=1:max_iter
    %[label,~,iter_WKFCM]=My_WKFCM(X,K);
    [label,~,iter_WKFCM,iter_FCM]=My_WKFCM(X,K);
    iter_WKFCM_t(i)=iter_WKFCM;
    iter_FCM_t(i)=iter_FCM;
    accuracy(i)=succeed(real_label,K,label);
    s=s+accuracy(i);
    s_1=s_1+ iter_WKFCM_t(i);
    s_2=s_2+ iter_FCM_t(i);
    %fprintf('第 %2d 次，WKFCM的迭代次数为：%2d，准确度为：%.8f\n', i, iter_WKFCM_t(i), accuracy(i));
    fprintf('第 %2d 次，FCM的迭代次数为：%2d，WKFCM的迭代次数为：%2d，准确度为：%.8f\n', i, iter_FCM_t(i), iter_WKFCM_t(i), accuracy(i));
end
ave_iter_FCM=s_2/max_iter;
ave_iter_WKFCM=s_1/max_iter;
ave_acc_WKFCM=s/max_iter;
```

```
max_acc_WKFCM=max(accuracy);
min_acc_WKFCM=min(accuracy);
run_time=cputime-t0;
ave_run_time=run_time/max_iter;
```

My_WKFCM.m

```
%function [label_1,para_miu,iter]=My_WKFCM(X,K)
function [label_1,para_miu,iter,iter_FCM]=My_WKFCM(X,K)
%输入K：聚类数
%输出：label_1:聚的类, para_miu_new:模糊聚类中心μ，responsivity:模糊隶属度
format long
eps=1e-4;  %定义迭代终止条件的eps
%sigma_2=2^(-4);  %高斯核函数的参数sigma^2
sigma_2=150;  %高斯核函数的参数sigma^2
beta=2;
alpha=2;  %模糊加权指数，[1,+无穷)
T=100;  %最大迭代次数
fitness=zeros(T,1);
[X_num,X_dim]=size(X);
distant=zeros(X_num,K,X_dim);
kernel_fun=zeros(X_num,K,X_dim);
R_temp=zeros(X_num,K,X_dim);
miu_up=zeros(X_num,K,X_dim);
miu_down=zeros(X_num,K,X_dim);
W_temp=zeros(X_num,K,X_dim);
J_temp=zeros(X_num,K,X_dim);
count=zeros(X_num,1);  %统计distant中每一行为0的个数
responsivity=zeros(X_num,K);
R_up=zeros(X_num,K);
W_up=zeros(K,X_dim);
%-----------------------------------------------------------------------------------------------------
%随机初始化属性权重K*X_dim
para_weight=ones(K,X_dim)./X_dim;
%随机初始化K个聚类中心
% rand_array=randperm(X_num);  %产生1~X_num之间整数的随机排列
% para_miu=X(rand_array(1:K),:);  %随机排列取前K个数，在X矩阵中取这K行作为初始聚类中心
%用FCM初始聚类中心
[~,para_miu,iter_FCM]=My_FCM(X,K);
% -----------------------------------------------------------------------------------------------------
% WKFCM算法
for t=1:T
    %计算隶属函数K*X_num
    for j=1:X_dim
        for i=1:X_num
            for k=1:K
```

```
            distant(i,k,j)=(X(i,j)-para_miu(k,j))^2;
            kernel_fun(i,k,j)=exp((-distant(i,k,j))/sigma_2);
            R_temp(i,k,j)=(para_weight(k,j)^beta)*(1-kernel_fun(i,k,j));
        end
    end
end
R_down=sum(R_temp,3);
for i=1:X_num
    count(i)=sum(R_down(i,:)==0);
    if count(i)>0
        for k=1:K
            if R_down(i,k)==0
                responsivity(i,k)=1./count(i);
            else
                responsivity(i,k)=0;
            end
        end
    else
        R_up(i,:)=R_down(i,:).^(-1/(alpha-1));  %隶属度矩阵的分子部分N*K
        responsivity(i,:)= R_up(i,:)./sum( R_up(i,:),2);
    end
end
 %更新聚类中心K*X_dim
for j=1:X_dim
    for i=1:X_num
        for k=1:K
            miu_up(i,k,j)=responsivity(i,k)*kernel_fun(i,k,j)*X(i,j);
            miu_down(i,k,j)=responsivity(i,k)*kernel_fun(i,k,j);
        end
    end
end
miu_up_sum=sum(miu_up,1);
miu_down_sum=sum(miu_down,1);
for k=1:K
    for j=1:X_dim
        if para_weight(k,j)==0
            para_miu(k,j)=0;
        else
            para_miu(k,j)=miu_up_sum(1,k,j)/miu_down_sum(1,k,j);
        end
    end
end
%更新属性权重K*X_dim
 for j=1:X_dim
    for i=1:X_num
        for k=1:K
            distant(i,k,j)=(X(i,j)-para_miu(k,j))^2;
```

```
                kernel_fun(i,k,j)=exp((-distant(i,k,j))./sigma_2);
                W_temp(i,k,j)=(responsivity(i,k)^alpha)*(1-kernel_fun(i,k,j));
            end
        end
     end
    W_down=sum(W_temp,1);
    for k=1:K
        for j=1:X_dim
                if W_down(1,k,j)==0
                    para_weight(k,j)=1./X_dim;
                else
                    W_up(k,:)=W_down(1,k,:).^(-1/(beta-1));  %属性权重矩阵的分子部分K*X_dim
                    para_weight(k,:)= W_up(k,:)./sum( W_up(k,:),2);
                end
        end
    end
    %计算目标函数值
    for j=1:X_dim
        for i=1:X_num
            for k=1:K
                distant(i,k,j)=(X(i,j)-para_miu(k,j))^2;
                kernel_fun(i,k,j)=exp((-distant(i,k,j))./sigma_2);
                J_temp(i,k,j)=(responsivity(i,k)^alpha)*(para_weight(k,j)^beta)*(1-kernel_fun(i,k,j));
            end
        end
    end
    fitness(t)=2*sum(sum(sum( J_temp)));
    if t>1
        if abs(fitness(t)-fitness(t-1))<eps
            break;
        end
    end
end
iter=t;  %实际迭代次数
[~,label_1]=max(responsivity,[],2);
```

2.在UCI数据库的iris上的运行结果

```
>> [ave_acc_WKFCM,max_acc_WKFCM,min_acc_WKFCM,ave_iter_FCM,ave_iter_WKFCM,ave_run_time]=WKFCM_main(data,real_label,3)
第  1 次，FCM的迭代次数为：14，WKFCM的迭代次数为： 4，准确度为：0.92666667
第  2 次，FCM的迭代次数为：17，WKFCM的迭代次数为： 4，准确度为：0.92666667
第  3 次，FCM的迭代次数为：28，WKFCM的迭代次数为： 4，准确度为：0.92666667
第  4 次，FCM的迭代次数为：14，WKFCM的迭代次数为： 4，准确度为：0.92666667
第  5 次，FCM的迭代次数为：20，WKFCM的迭代次数为： 4，准确度为：0.92666667
第  6 次，FCM的迭代次数为：11，WKFCM的迭代次数为： 4，准确度为：0.92666667
第  7 次，FCM的迭代次数为：19，WKFCM的迭代次数为： 4，准确度为：0.92666667
```

```
第  8 次，FCM的迭代次数为：15，WKFCM的迭代次数为： 4，准确度为：0.92666667
第  9 次，FCM的迭代次数为：14，WKFCM的迭代次数为： 4，准确度为：0.92666667
第 10 次，FCM的迭代次数为：11，WKFCM的迭代次数为： 4，准确度为：0.92666667
第 11 次，FCM的迭代次数为：21，WKFCM的迭代次数为： 4，准确度为：0.92666667
第 12 次，FCM的迭代次数为：20，WKFCM的迭代次数为： 4，准确度为：0.92666667
第 13 次，FCM的迭代次数为：10，WKFCM的迭代次数为： 4，准确度为：0.92666667
第 14 次，FCM的迭代次数为：28，WKFCM的迭代次数为： 4，准确度为：0.92666667
第 15 次，FCM的迭代次数为：18，WKFCM的迭代次数为： 4，准确度为：0.92666667
第 16 次，FCM的迭代次数为：16，WKFCM的迭代次数为： 4，准确度为：0.92666667
第 17 次，FCM的迭代次数为：12，WKFCM的迭代次数为： 4，准确度为：0.92666667
第 18 次，FCM的迭代次数为：20，WKFCM的迭代次数为： 4，准确度为：0.92666667
第 19 次，FCM的迭代次数为：12，WKFCM的迭代次数为： 4，准确度为：0.92666667
第 20 次，FCM的迭代次数为：13，WKFCM的迭代次数为： 4，准确度为：0.92666667

ave_acc_WKFCM =
   0.926666666666666

max_acc_WKFCM =
   0.926666666666667

min_acc_WKFCM =
   0.926666666666667

ave_iter_FCM =
  16.649999999999999

ave_iter_WKFCM =
     4

ave_run_time =
   0.232812500000000
```