# 实验报告三

学号：20191060239

姓名：李品鑫

实验名称：聚类算法实现及应用

【实验目标】：

实现K-均值聚类算法、二分K-均值聚类算法，并在鸢尾花数据集上进行应用。

【实验任务】：

1. 阐述K-均值聚类算法的原理及流程，编程实现K-均值聚类算法。

2. 阐述二分K-均值聚类算法的原理及流程，编程实现二分K-均值聚类算法。

3. 针对数据集1，使用K-均值聚类算法按以下初始设置进行聚类，记录每次迭代的质心以及聚类结果。

（1）聚类数K=2，初始质心m1=(0,0,0), m2=(1,1,-1).

4. 分别使用K-均值聚类算法和二分K-均值聚类算法对鸢尾花数据集（数据集2）中的数据进行聚类，分析两种算法的聚类结果，至少使用一种簇评估方法对聚类结果进行评价。

【实验要求】：

（1）独立完成实验，拒绝抄袭。撰写实验报告，需包含实验目的，实验原理及过程，实验结果展示等。

（2）对实验任务中的1和2，可根据个人能力选做其中一个或两个都做，任务1的难度低于任务2。（评分会考虑任务难度，鼓励有余力的同学两个算法都实现）。

（3）可根据自己能力借助合适的工具（如sklearn库）完成实验，但需清楚算法原理。鼓励自己动手编程实现整个算法，评分时，自己动手编程实现>借助工具。

【实验过程】：

1. （对应任务1）阐述K-均值聚类算法的原理及流程，编程实现K-均值聚类算法。（对应任务3）针对数据集1，使用K-均值聚类算法进行聚类，记录每次迭代的质心以及聚类结果。

K-均值聚类算法假定所有数据对象可以分为K簇，对象间的相似性用距离度量，根据最小距离准则将每个数据对象分配到某一簇中，然后不断迭代计算各个簇的聚类中心并调整聚类情况，直到收敛（误差平方和不再变化或者是变化小于一个阈值）

针对数据集1编程进行的K-均值聚类算法聚类代码如下：

close all;

clear all;

clc;

data=csvread('数据集1、数据集.csv',1,1);

X=data(:,1);

Y=data(:,2);

Z=data(:,3);

u1 = [0,0,0];

u2 = [1,1,-1];

newu1 =[];newu2 =[];

newSSE=inf;

X1=[];X2=[];Y1=[];Y2=[];Z1=[];Z2=[];

k=1;

for i=(1:length(X))

if sqrt((X(i)-u1(1))^2+(Y(i)-u1(2))^2+(Z(i)-u1(3))^2)<sqrt((X(i)-u2(1))^2+(Y(i)-u2(2))^2+(Z(i)-u2(3))^2)

X1=[X1 X(i)];Y1=[Y1 Y(i)];Z1=[Z1 Z(i)];

else

X2=[X2 X(i)];Y2=[Y2 Y(i)];Z2=[Z2 Z(i)];

end

end

输出的聚类结果如下图，类1用红色 ’.’ 表示，类2用蓝色 ’.’ 表示，聚类中心用相应颜色的 ’\*’ 表示

figure;

scatter3(X1,Y1,Z1,'.','red');

hold on

scatter3(X2,Y2,Z2,'.','blue');

scatter3(u1(1),u1(2),u1(3),'\*','red');

scatter3(u2(1),u2(2),u2(3),'\*','blue');

title(sprintf("第%d次迭代",k));

hold off

SSE=sum((X-u1(1)).^2+(Y-u1(2)).^2+(Z-u1(3)).^2)+ ...

sum((X-u2(1)).^2+(Y-u2(2)).^2+(Z-u2(3)).^2);

u1list{k}=u1;u2list{k}=u2;

while abs(SSE-newSSE)>1e+04

SSE = newSSE;

k=k+1;

newu1 =[sum(X1)/length(X1) sum(Y1)/length(Y1) sum(Z1)/length(Z1)];

newu2 =[sum(X2)/length(X2) sum(Y2)/length(Y2) sum(Z2)/length(Z2)];

for i=(1:length(X))

if sqrt((X(i)-newu1(1))^2+(Y(i)-newu1(2))^2+(Z(i)- newu1(3))^2) ...

<sqrt((X(i)-newu2(1))^2+(Y(i)-newu2(2))^2+(Z(i)-newu2(3))^2)

X1=[X1 X(i)];Y1=[Y1 Y(i)];Z1=[Z1 Z(i)];

else

X2=[X2 X(i)];Y2=[Y2 Y(i)];Z2=[Z2 Z(i)];

end

end

figure;

scatter3(X1,Y1,Z1,'.','red');

hold on

scatter3(X2,Y2,Z2,'.','blue');

scatter3(newu1(1),newu1(2),newu1(3),'\*','red');

scatter3(newu2(1),newu2(2),newu2(3),'\*','blue');

title(sprintf("第%d次迭代",k));

hold off

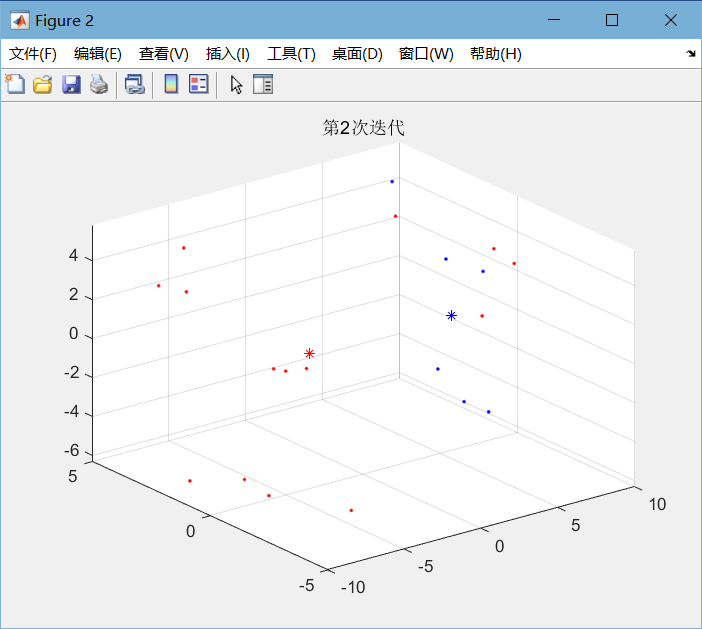
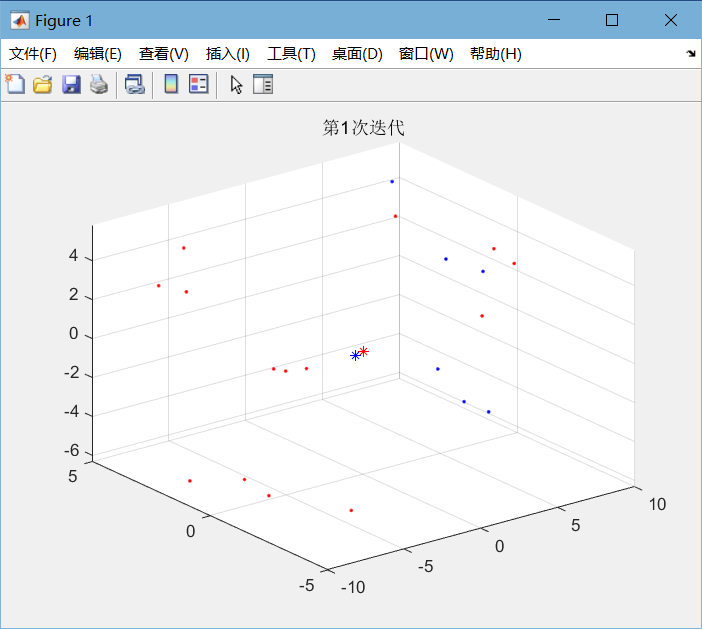
newSSE=sum((X-newu1(1)).^2+(Y-newu1(2)).^2+(Z-newu1(3)).^2)...

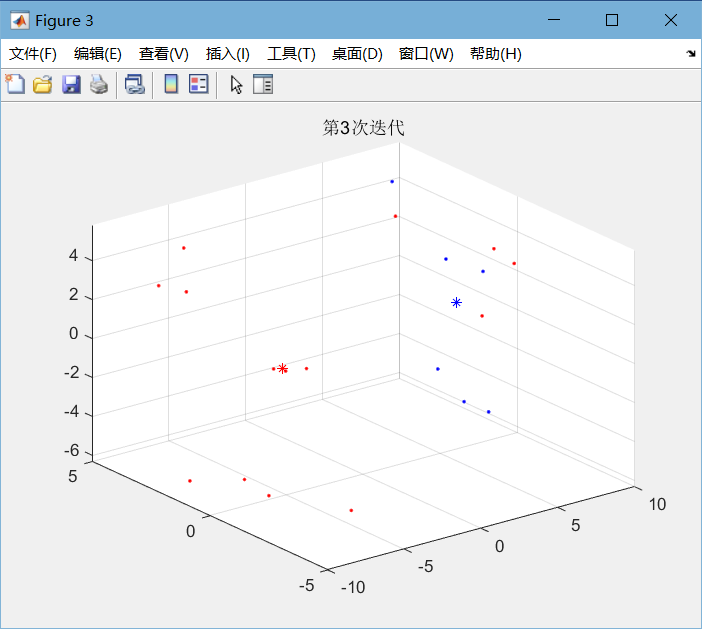
+sum((X-newu2(1)).^2+(Y-newu2(2)).^2+(Z-newu2(3)).^2);

u1list{k}=newu1;u2list{k}=newu2;

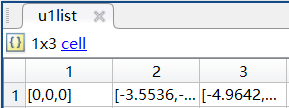
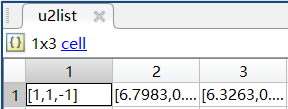
end

共进行了三次迭代，如下图所示：





数次迭代的质心存储在u1list与u2list两个元胞数组中，如下所示：

2. （对应任务2）阐述二分K-均值聚类算法的原理及流程，编程实现二分K-均值聚类算法。

（对应任务4）分别使用K-均值聚类算法和二分K-均值聚类算法对鸢尾花数据集（数据集2）中的数据进行聚类，分析两种算法的聚类结果，至少使用一种簇评估方法对聚类结果进行评价。

二分K-均值聚类算法先将所有数据集看作一个簇，多次使用随机质心利用基本K-均值聚类算法将其二分，从中选择具有最小总误差平方和SSE的两个簇进行第二次聚类。在上一次聚类中分出的簇中选择误差平方和最大的簇或是总数最大的簇重复进行以上操作，直至分出所需要的K个簇。

编写的代码分为两个部分，一部分为主函数，另一部分为聚类函数。

针对数据集2编程进行的二分K-均值聚类算法聚类代码如下

主函数 main.m ：

close all;

clear all;

clc;

csv=csvread('数据集2\iris.csv',1,0);

id=csv(:,1);

data=csv(:,2:5);

num=3;

ulist={};

newdata={};

ulist=BisectingKMeansForIris(data,num,ulist);

X1=[];X2=[];X3=[]; Y1=[];Y2=[];Y3=[];

Z1=[];Z2=[];Z3=[]; W1=[];W2=[];W3=[];

cls1=[];cls2=[];cls3=[];

X=data(:,1);Y=data(:,2);

Z=data(:,3);W=data(:,4);

for i=(1:length(data))

if min([dist([X(i) Y(i) Z(i) W(i)],ulist{1,1}'),dist([X(i) Y(i) Z(i) W(i)],ulist{1,2}')...

,dist([X(i) Y(i) Z(i) W(i)],ulist{1,3}')])==dist([X(i) Y(i) Z(i) W(i)],ulist{1,1}')

X1=[X1 X(i)];Y1=[Y1 Y(i)];Z1=[Z1 Z(i)];W1=[W1 W(i)];cls1=[cls1 id(i)];

elseif min([dist([X(i) Y(i) Z(i) W(i)],ulist{1,1}'),dist([X(i) Y(i) Z(i) W(i)],ulist{1,2}')...

,dist([X(i) Y(i) Z(i) W(i)],ulist{1,3}')])==dist([X(i) Y(i) Z(i) W(i)],ulist{1,2}')

X2=[X2 X(i)];Y2=[Y2 Y(i)];Z2=[Z2 Z(i)];W2=[W2 W(i)];cls2=[cls2 id(i)];

elseif min([dist([X(i) Y(i) Z(i) W(i)],ulist{1,1}'),dist([X(i) Y(i) Z(i) W(i)],ulist{1,2}')...

,dist([X(i) Y(i) Z(i) W(i)],ulist{1,3}')])==dist([X(i) Y(i) Z(i) W(i)],ulist{1,3}')

X3=[X3 X(i)];Y3=[Y3 Y(i)];Z3=[Z3 Z(i)];W3=[W3 W(i)];cls3=[cls3 id(i)];

end

end

for i=1:length(cls1)

if cls1(i)<=50

sum1=sum1+1;

elseif cls1(i)<=100 && cls1(i)>50

sum2=sum2+1;

elseif cls1(i)<=150 && cls1(i)>100

sum3=sum3+1;

end

end

sumT1=max([sum1 sum2 sum3]);

sum1=0;sum2=0;sum3=0;

主函数实现的功能有读取数据集，生成分类函数所需的参数并传递给分类函数BisectingKMeansForIris ，接收函数返回的聚类中心列表ulist并以此聚类并评价其纯度。

for i=1:length(cls2)

if cls2(i)<=50

sum1=sum1+1;

elseif cls2(i)<=100 && cls2(i)>50

sum2=sum2+1;

elseif cls2(i)<=150 && cls2(i)>100

sum3=sum3+1;

end

end

sumT2=max([sum1 sum2 sum3]);

sum1=0;sum2=0;sum3=0;

for i=1:length(cls3)

if cls3(i)<=50

sum1=sum1+1;

elseif cls3(i)<=100 && cls3(i)>50

sum2=sum2+1;

elseif cls3(i)<=150 && cls3(i)>100

sum3=sum3+1;

end

end

sumT3=max([sum1 sum2 sum3]);

sum1=0;sum2=0;sum3=0;

purity=(sumT1+sumT2+sumT3)/length(data);

聚类函数BisectingKMeansForIris.m

function [ulist]=BisectingKMeansForIris(data,num,ulist)

X=data(:,1);

Y=data(:,2);

Z=data(:,3);

W=data(:,4);

newu1 = [];newu2 = [];

curcls1 = [];curcls2 = [];

cls1 = [];cls2 = [];

newSSE=inf;

minSSE=inf;

X1=[];X2=[];Y1=[];Y2=[];Z1=[];Z2=[];W1=[];W2=[];

curX1=[];curX2=[];curY1=[];curY2=[];curZ1=[];curZ2=[];curW1=[];curW2=[];

for time=1:50

u1 = [rand(1,1)\*max(X),rand(1,1)\*max(Y),rand(1,1)\*max(Z),rand(1,1)\*max(W)];

u2 = [rand(1,1)\*max(X),rand(1,1)\*max(Y),rand(1,1)\*max(Z),rand(1,1)\*max(W)];

for i=(1:length(data))

if min([dist([X(i) Y(i) Z(i) W(i)],u1'),dist([X(i) Y(i) Z(i) W(i)],u2')])...

==dist([X(i) Y(i) Z(i) W(i)],u1')

curX1=[curX1 X(i)];curY1=[curY1 Y(i)];curZ1=[curZ1 Z(i)];

else

curX2=[curX2 X(i)];curY2=[curY2 Y(i)];curZ2=[curZ2 Z(i)];

end

end

SSE=sum((X-u1(1)).^2+(Y-u1(2)).^2+(Z-u1(3)).^2+(W-u1(4)).^2)+...

sum((X-u2(1)).^2+(Y-u2(2)).^2+(Z-u2(3)).^2+(W-u2(4)).^2);

while abs(SSE-newSSE)>0

SSE=newSSE;

newu1 =[sum(curX1)/length(curX1) sum(curY1)/length(curY1) sum(curZ1)/length(curZ1) sum(curW1)/length(curW1)];

newu2 =[sum(curX2)/length(curX2) sum(curY2)/length(curY2) sum(curZ2)/length(curZ2) sum(curW2)/length(curW2)];

curX1=[];curX2=[];curY1=[];curY2=[];curZ1=[];curZ2=[];curW1=[];curW2=[];curcls1=[];curcls2=[];

for i=(1:length(X))

if sqrt((X(i)-newu1(1))^2+(Y(i)-newu1(2))^2+(Z(i)-newu1(3))^2)<sqrt((X(i)-newu2(1))^2 ...

+(Y(i)-newu2(2))^2+(Z(i)-newu2(3))^2)

curX1=[curX1 X(i)];curY1=[curY1 Y(i)];curZ1=[curZ1 Z(i)];curW1=[curW1 W(i)];

else

curX2=[curX2 X(i)];curY2=[curY2 Y(i)];curZ2=[curZ2 Z(i)];curW2=[curW2 W(i)];

end

end

newSSE=sum((X-newu1(1)).^2+(Y-newu1(2)).^2+(Z-newu1(3)).^2+(W-u1(4)).^2)...

+sum((X-newu2(1)).^2+(Y-newu2(2)).^2+(Z-newu2(3)).^2+(W-u2(4)).^2);

end

end

该函数根据输入的数据集data，所需聚类簇数num和初始聚类中心空列表ulist，进行二分K-均值聚类。

if newSSE<minSSE

X1=curX1;Y1=curY1;Z1=curZ1;W1=curW1;

X2=curX2;Y2=curY2;Z2=curZ2;W2=curW2;

cls1=curcls1;cls2=curcls2;

minSSE=newSSE;

end

while length(ulist)<(num-2)

if sum((X-newu1(1)).^2+(Y-newu1(2)).^2+(Z-newu1(3)).^2+(W-newu1(4),^2) <...

sum((X-newu2(1)).^2+(Y-newu2(2)).^2+(Z-newu2(3)).^2+(W-newu2(4)).^2)

ulist=[ulist newu1];

newdata=cat(2,cls2',X2',Y2',Z2',W2');

else

ulist=[ulist newu2];

newdata=cat(2,cls1',X1',Y1',Z1',W1');

end

BisectingKMeansForIris(newdata,num,ulist);

end

ulist=[ulist newu1 newu2];

end

根据此聚类方法，重复多次产生的聚类纯度皆为：



而针对数据集2编程进行的一般K-均值聚类算法代码与前文类似，仅维度不同，故不多做赘述，仅列举重复多次产生的聚类纯度：

1. 

2. 

3. 

4. 

5. 

从聚类效果方面观察，发现一般K-均值聚类算法的聚类纯度浮动巨大，聚类最优时达到了0.9067的纯度，而最差时仅0.34。这是因为随机产生聚类中心导致的，而二分K-均值聚类算法采用了多次使用随机质心利用基本K-均值聚类算法将其二分，从中选择具有最小总误差平方和SSE的两个簇进行第二次聚类的方式大幅减小了随机性，保证了聚类的稳定。

从时间花费方面，一般K-均值聚类算法的聚类十分迅速，而二分K-均值聚类算法则需要花费一定的时间，这是由于二分K-均值聚类算法使用了许多循环与嵌套来保证算法效果的稳定，避免了随机性。

同时还观察到一个有趣的现象，在针对iris数据集的二分K-均值聚类算法中，虽然数据有三类，但总是有一个簇是空的，也就是说只被聚类成了两簇，说明在代码中把该数据集分成二簇才是SSE最小的分法，也体现了二分K-均值聚类算法的局限性。

【实验心得】：

该实验从简单的对三维数据的K-均值聚类算法起步，逐渐深入至多维数据的二分K-均值聚类算法以及聚类效果的评估方法。比起单纯的的理论授课，通过实际的编程操作更好地理解了K-均值聚类算法、二分K-均值聚类算法的原理、具体的实现以及各自的特点。代码的编写、纠错、调试也是一个很好的练习，极大地提升了基础编程能力，使将抽象的算法付诸于具体的计算机实践成为可能，而不是局限于试卷上的纸上谈兵。