1.init.m 初始化函数，得到1-频繁项集及支持度，输入数据集和最小支持度

function [L A]=init(D,min\_sup)  %D表示数据集  min\_sup 最小支持度

[m n]=size(D);  %注释 size矩阵的行数和列数

A=eye(m,n);   %产生扩展的单位矩阵

B=(sum(D))';   %列求和，再转秩

i=1;

while(i<=m)

    if B(i)<min\_sup

        B(i)=[];

        A(i,:)=[];

        m=m-1;

    else

        i=i+1;

    end

end

L=[A B];

   2、apriori\_gen.m

function [C]=apriori\_gen(A,k) %产生Ck（实现组内连接及剪枝 ）

%A表示第k-1次的频繁项集 k表示第k-频繁项集

[m n]=size(A);

C=zeros(0,n);  %组内连接

for i=1:1:m

    for j=i+1:1:m

        flag=1;

        for t=1:1:k-1

            if ~(A(i,t)==A(j,t))

                flag=0;

                break;

            end

        end

        if flag==0 break;

        end

        c=A(i,:)|A(j,:);

        flag=isExit(c,A);   %剪枝

        if(flag==1)C=[C;c];

        end

    end

end

      3、 isExit.m

function flag=isExit(c,A)%判断c串的子串在A中是否存在

[m n]=size(A);

b=c;

for i=1:1:n

    c=b;

    if c(i)==0 continue

    end

    c(i)=0;

    flag=0;

    for j=1:1:m

        A(j,:);

        a=sum(xor(c,A(j,:)));  % xor 函数判断是否相同

        if a==0

            flag=1;

            break;

        end

    end

    if flag==0 return

    end

end

 4、get\_k\_itemset.m

function [L C]=get\_k\_itemset(D,C,min\_sup)%D为数据集  C为第K次剪枝后的候选

%集 获得第k次的频繁项集

m=size(C,1);

M=zeros(m,1);

t=size(D,1);

i=1;

while i<=m

    C(i,:);

    H=ones(t,1);

    ind=find(C(i,:)==1);  %[find函数](https://www.baidu.com/s?wd=find%E5%87%BD%E6%95%B0&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1YzrjbdPhP9P1f4rjbknWKB0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EnHT3nW6vrH0v)用于返回所需要元素的所在位置

    n=size(ind,2);

    for j=1:1:n

        D(:,ind(j));

        H=H&D(:,ind(j));

    end

        x=sum(H');

        if x<min\_sup

            C(i,:)=[];

            M(i)=[];

            m=m-1;

        else

            M(i)=x;

            i=i+1;

        end

end

L=[C M];

5、主函数   apriori.m

function [L]=apriori(D,min\_sup)

[L A]=init(D,min\_sup)%A为1-频繁项集  L中为包含1-频繁项集以及对应的支持度

k=1;

C=apriori\_gen(A,k) %产生2项的集合

while ~(size(C,1)==0)

[M C]=get\_k\_itemset(D,C,min\_sup)%产生k-频繁项集 M是带支持度  C不带

if ~(size(M,1)==0)L=[L;M]

end

k=k+1;

C=apriori\_gen(C,k)%产生组合及剪枝后的候选集

end