**TOPSIS.m--------------------**

data = xlsread('data.xlsx');

[x,y] = size(data);

X = data(:,1:y);

[n,m] = size(X);

% It needs to be converted into very large ones

for i = 16:m

X(:,i) = Positivization(X(:,i),1,i);

end

disp('The forward-forward matrix is X=');

disp(X);

%standardizing

Z = X ./ repmat(sum(X.\*X) .^ 0.5, n, 1);

disp('Standardization matrix Z = ')

disp(Z)

Judge = 1; % Control weighting

if Judge == 1

if sum(sum(Z<0))>0

disp('The presence of negative numbers in the normalized matrix is being re-normalized')

for j=1:m

minn=min(Z(:,j));

maxx=max(Z(:,j));

for i=1:n

Z(i,j)=(Z(i,j)-minn)/(maxx-minn);

end

end

disp('Standardized completion matrix Z= ');

disp(Z);

end

W = Entropy\_Method(Z);

disp('The weight determined by the entropy weight method is：');

disp(W);

else

W = ones(1,m) ./ m ;

end

Tmp\_Max = (Z - repmat(max(Z),n,1)) .^ 2;

Tmp\_Min = (Z - repmat(min(Z),n,1)) .^ 2;

for i = 1 : n

R\_Max(i,1:y) = Tmp\_Max(i,1:y).\*W;

R\_Min(i,1:y) = Tmp\_Min(i,1:y).\*W;

end

D\_P = sum(R\_Max,2) .^ 0.5;%The optimal distance

D\_N = sum(R\_Min,2) .^ 0.5;%The worst distance

S = D\_N ./ (D\_P+D\_N);%Relative proximity (available as scores)

disp('The final score：')

stand\_S = S / sum(S);%Scores were normalized and the final scores of each scheme were summed to 1

**mylog.m---------------------**

function [lnp] = mylog(p)

n = length(p);

lnp = zeros(n,1);

for i = 1:n

if p(i) == 0

lnp(i) = 0;

else

lnp(i) = log(p(i));

end

end

end

**Positivization.m-----------------------**

function [posit\_x]=Positivization(x,type,i)

if type==1 %Very small

% posit\_x=max(x)-x;

posit\_x=1./x;

end

end

**Entropy\_Method.m------------------**

%Calculate the weight

function [W] = Entropy\_Method(Z)

[n,m]=size(Z);

d=zeros(1,m);

for i=1:m

x = Z(:,i);

p = x./sum(x);%Probability matrix

e = -sum(p .\* mylog(p)) / mylog(n);%entropy of information

d(i)=1-e;%Information utility value

end

W=d./sum(d);%Entropy right

end

**get\_W.m----------------**

function [W]=get\_W(X)

[n,m] = size(X);

% It needs to be converted into very large ones

for i = 16:m

X(:,i) = Positivization(X(:,i),1,i);

end

%standardizing

Z = X ./ repmat(sum(X.\*X) .^ 0.5, n, 1);

Judge = 1; % Control weighting

if Judge == 1

if sum(sum(Z<0))>0

for j=1:m

minn=min(Z(:,j));

maxx=max(Z(:,j));

for i=1:n

Z(i,j)=(Z(i,j)-minn)/(maxx-minn);

end

end

end

W = Entropy\_Method(Z);

end

End

**Problem3.m-------------------**

data = xlsread('data.xlsx');

[x,y] = size(data);

X = data(:,1:y);

[n,m] = size(X);

Z = X;

W = get\_W(X); %Obtain weights

Tmp = X;

Tmp(:,9) = Tmp(:,9) \* 0.8;

Tmp(:,15) = Tmp(:,15) \* 0.8;

Tmp(:,25) = Tmp(:,25) \* 1.2;

SS = Untitled2(Tmp, W);

for i = 16:m

X(:,i) = Positivization(X(:,i),1,i);

end

for i = 1 : n

Score(i) = sum(W .\* X(i,:));

end

for i = 16:m

Tmp(:,i) = Positivization(Tmp(:,i),1,i);

end

for i = 1 : n

Score1(i) = sum(W .\* Tmp(i,:));

end

ans = (Score - Score1) ./ Score;

**Problem4.m-----------------------**

data = xlsread('data.xlsx');

[x,y] = size(data);

X = data(:,1:y);

[n,m] = size(X);

W = get\_W(X);

% Shanghai

for i = 1 : m

Z = X;

if i > 15

Z(9,i) = Z(9,i) \* 1.3;

else

Z(9,i) = Z(9,i) \* 0.7;

end

score = Untitled2(Z, W);

shanghai\_score(i) = score(9);

end

for i = 1:28

if i == 1

tmp = Untitled2(X, W);

res1(i) = tmp(9);

else

res1(i) = shanghai\_score(i-1);

end

end

% Sichuan

for i = 1 : m

Z = X;

if i > 15

Z(23,i) = Z(23,i) \* 1.3;

else

Z(23,i) = Z(23,i) \* 0.7;

end

score = Untitled2(Z, W);

shanghai\_score(i) = score(23);

end

for i = 1:28

if i == 1

tmp = Untitled2(X, W);

res2(i) = tmp(23);

else

res2(i) = shanghai\_score(i-1);

end

end