Topsis法算法精讲

史上最全数学建模综合教程（数学建模写作、算法、编程从入门、速成到进阶）

模型原理+Matlab/Python双语言代码演示

主讲人：江北

目录 Contents

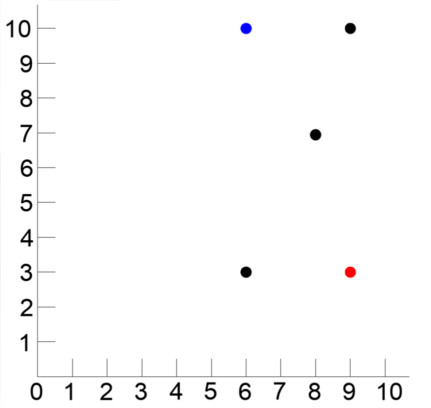
Topsis | 模型引出

问题的提出

- 生活中我们常常要进行评价，上节课我们讲到了层次分析法，通过构造判断矩阵，确定各指标的权重，然后对指标数值进行加权来进行打分，那还有别的方法吗?

明星Kun想找个对象，但喜欢他的人太多，不知道怎么选，经过层层考察，留下三个候选人。

候选人	颜值	脾气（争吵次数）
A	9	10
B	8	7
C	6	3



- 理想情况下:

  最好的对象应该是颜值9，脾气3

  最差的对象应该是颜值6，脾气10

- 那怎么衡量A、B、C和最好、最差的距离呢?

  把（9，3），（6，10）作为二维平面的一个点

- 距离**最好点最近**或者距离**最差点最远**的的就是综合条件最好的

Topsis | 模型引出

➢ 基本概念

C. L. Hwang和 K. Yoon于1981年首次提出 **TOPSIS** (Technique for Order Preference by Similarity to an Ideal Solution)，可翻译为逼近理想解排序法，国内常简称为**优劣解距离法**。

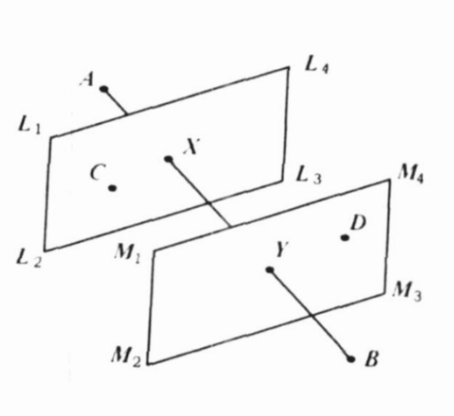
TOPSIS法是一种常用的综合评价方法，能充分利用原始数据的信息，其结果能精确地反映各评价方案之间的差距。

TOPSIS法引入了两个基本概念：

理想解：设想的最优的解(方案)，它的各个属性值都达到各备选方案中的最好的值；

负理想解：设想的最劣的解(方案)，它的各个属性值都达到各备选方案中的最坏的值。

方案排序的规则是把各备选方案与理想解和负理想解做比较，若其中有一个方案最接近理想解，而同时又远离负理想解，则该方案是备选方案中最好的方案。TOPSIS通过最接近理想解且最远离负理想解来**确定最优选择**。



Topsis | 基本原理

➢ 模型原理

TOPSIS法是一种理想目标相似性的顺序选优技术，在**多目标决策分析**中是一种非常有效的方法。它通过归一化后(去量纲化)的数据规范化矩阵，找出多个目标中最优目标和最劣目标(分别用理归想一解化和反理想解表示)，分别计算各评价目标与理想解和反理想解的距离，获得各目标与理想解的贴近度，按理想解贴近度的大小排序，以此作为评价目标优劣的依据。贴近度取值在0～1之间，该值愈接近1，表示相应的评价目标越接近最优水平；反之，该值愈接近0，表示评价目标越接近最劣水平。

➢ 基本步骤

- 将原始矩阵正向化

  将原始矩阵正向化，就是要将所有的指标类型统一转化为**极大型指标**。

- 正向矩阵标准化

  标准化的方法有很多种，其主要目的就是去除量纲的影响，保证不同评价指标在同一数量级，且数据**大小排序不变**。

- 计算得分并归一化

  $S_i=\frac{D_i^-}{D_i^++D_i^-}$ 其中S_i为得分，D_i^+为评价对象与最大值的距离，D_i^-为评价对象与最小值的距离

Topsis | 典型例题

➢ 我们继续帮明星Kun选对象

明星Kun考虑了一下觉得光靠颜值和脾气可能考虑的还不够全面，就又加上了身高和体重两个指标，而且他认为身高165是最好，体重在90-100斤是最好。

候选人	颜值	脾气（争吵次数）	身高	体重
A	9	10	175	120
B	8	7	164	80
C	6	3	157	90

➢ **原始矩阵正向化**

指标名称	指标特点	例子
极大型（效益型）指标	越大（多）越好	颜值、成绩、GDP增速
极小型（成本型）指标	越小（少）越好	脾气、费用、坏品率、污染程度
中间型指标	越接近某个值越好	身高、水质量评估时的PH值
区间型指标	落在某个区间最好	体重、体温

➢ 原始矩阵正向化

- 将原始矩阵正向化，就是要将所有的指标类型统一转化为**极大型指标**。

指标名称	公式
极大型（效益型）指标	/
极小型（成本型）指标	$\tilde{x}=max-x$，$\tilde{x}$为转化后指标，max为指标最大值，x为指标值
中间型指标	$\{x_i\}$是一组区间型序列，最优值是x_{best} $M=max\{\lvert x_i-x_{best}\rvert\}$，$\tilde{x}_i=1-\frac{\lvert x_i-x_{best}\rvert}{M}$
区间型指标	$\{x_i\}$是一组区间型序列，最佳区间为$[a,b]$，正向化公式如下 $M=max\{a-min\{x_i\},\ max\{x_i\}-b\}$，$\tilde{x}_i=\begin{cases}1-\frac{a-x_i}{M}, & x_i<a\\ 1, & a\le x_i\le b\\ 1-\frac{x_i-b}{M}, & x_i>b\end{cases}$

Topsis | 典型例题

原始矩阵正向化

- 将原始矩阵正向化，就是要将所有的指标类型统一转化为**极大型指标**。

候选人	颜值	脾气（争吵次数）	身高165	体重90-100
A	9	10	175	120
B	8	7	164	80
C	6	3	157	90

⬇

候选人	颜值	脾气（争吵次数）	身高	体重
A	9	0	0	0
B	8	3	0.9	0.5
C	6	7	0.2	1

Topsis | 典型例题

➢ 正向化矩阵标准化

- 标准化的目的是**消除不同指标量纲**的影响。

  假设有n个要评价的对象，m个评价指标（已正向化）构成的正向化矩阵如下：

$$X=\begin{bmatrix} x_{11} & \cdots & x_{1m} \\ \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{nm} \end{bmatrix}$$

  那么，对其标准化的矩阵记为Z，Z中的每一个元素：

$$z_{ij}=\frac{x_{ij}}{\sqrt{\sum_{i=1}^{n} x_{ij}^2}} \text{（每一个元素}/\sqrt{\text{其所在列的元素的平方和}}\text{）}$$

- 标准化后，还需给不同指标加上权重，采用的权重确定方法有层次分析法、熵权法、Delphi法、对数最小二乘法等。在这里认为**各个指标的权重相同**。

候选人	颜值	脾气（争吵次数）	身高	体重
A	9	0	0	0
B	8	3	0.9	0.5
C	6	7	0.2	1

→

候选人	颜值	脾气（争吵次数）	身高	体重
A	0.669	0	0	0
B	0.595	0.394	0.976	0.447
C	0.446	0.919	0.217	0.894

数模加油站

计算得分并归一化

- 上一步得到标准化矩阵Z

$$Z=\begin{bmatrix} z_{11} & \cdots & z_{1m} \\ \vdots & \ddots & \vdots \\ z_{n1} & \cdots & z_{nm} \end{bmatrix}$$

- 定义最大值$Z^+=(Z_1^+,Z_2^+,\dots,Z_m^+)=(max\{z_{11},z_{21},\dots,z_{n1}\},\ max\{z_{12},z_{22},\dots,z_{n2}\},\dots,max\{z_{1m},z_{2m},\dots,z_{nm}\})$
- 定义最小值$Z^-=(Z_1^-,Z_2^-,\dots,Z_m^-)=(min\{z_{11},z_{21},\dots,z_{n1}\},\ min\{z_{12},z_{22},\dots,z_{n2}\},\dots,min\{z_{1m},z_{2m},\dots,z_{nm}\})$
- 定义第$i(\mathrm{i}=1,2,\dots,n)$个评价对象与最大值的距离$D_i^+=\sqrt{\sum_{j=1}^m (Z_j^+-z_{ij})^2}$
- 定义第$i(\mathrm{i}=1,2,\dots,n)$个评价对象与最小值的距离$D_i^-=\sqrt{\sum_{j=1}^m (Z_j^--z_{ij})^2}$
- 那么，我们可以计算得出第 $i(\mathrm{i}=1,2,\dots,n)$ 个评价对象未归一化的得分：$S_i=\frac{D_i^-}{D_i^++D_i^-}$
- 很明显$0\le S_i\le 1$，**且S_i越大D_i^+越小，即越接近最大值**
- 我们可以将得分归一化并换成百分制：$\widetilde{S}_\mathrm{i}=\frac{S_\mathrm{i}}{\sum_{i=1}^n S_\mathrm{i}}\times 100$

Topsis | 典型例题

MATH 数模加油站

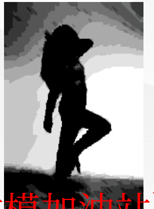
➢ 计算得分并归一化

候选人	颜值	脾气(争吵次数)	身高	体重
A	0.669	0	0	0
B	0.595	0.394	0.976	0.447
C	0.446	0.919	0.217	0.894

候选人	得分
A	0.122
B	0.624
C	0.622

候选人	得分
A	8.9
B	45.7
C	45.5

- 明星K选择了B！！！



Topsis | 相关Matlab代码

主代码

```
clear;clc
% 1.判断是否需要正向化
X=input('指标矩阵A='); %输入判断矩阵
[n,m] = size(X);
disp(['共有' num2str(n) '个评价对象, ' num2str(m) '个评价指标'])
Judge = input(['这' num2str(m) '个指标是否需要经过正向化处理，需要请输入1 ，不需要输入0：  ']);
if Judge == 1
Position = input('请输入需要正向化处理的指标所在的列，例如第2、3、6三列需要处理，那么你需要输入[2,3,6]：  '); %[2,3,4]
disp('请输入需要处理的这些列的指标类型（1：极小型， 2：中间型， 3：区间型） ')
Type = input('例如：第2列是极小型，第3列是区间型，第6列是中间型，就输入[1,3,2]：  '); %[2,1,3]
% 注意，Position和Type是两个同维度的行向量
for i = 1 : size(Position,2) %这里需要对这些列分别处理，因此我们需要知道一共要处理的次数，即循环的次数
X(:,Position(i)) = Positivization(X(:,Position(i)),Type(i),Position(i));
% Positivization是我们自己定义的函数，其作用是进行正向化，其一共接收三个参数
% 第一个参数是要正向化处理的那一列向量 X(:,Position(i)) 回顾上一讲的知识，X(:,n)表示取第n列的全部元素
% 第二个参数是对应的这一列的指标类型（1：极小型， 2：中间型， 3：区间型）
% 第三个参数是告诉函数我们正在处理的是原始矩阵中的哪一列
% 该函数有一个返回值，它返回正向化之后的指标，我们可以将其直接赋值给我们原始要处理的那一列向量
end
```

主代码

```
disp('正向化后的矩阵 X = ')
disp(X)
end
%% 2.对正向化后的矩阵进行标准化
Z = X ./ repmat(sum(X.*X) .^ 0.5, n, 1);
disp('标准化矩阵 Z = ')
disp(Z)
%% 3.计算与最大值的距离和最小值的距离，并算出得分
D_P = sum([(Z - repmat(max(Z),n,1)) .^ 2 ],2) .^ 0.5; % D+ 与最大值的距离向量
D_N = sum([(Z - repmat(min(Z),n,1)) .^ 2 ],2) .^ 0.5; % D- 与最小值的距离向量
S = D_N ./ (D_P+D_N); % 未归一化的得分
disp('最后的得分为：')
stand_S = 100*S / sum(S)
[sorted_S,index] = sort(stand_S ,'descend')
```

数模加油站

➢ Positivization函数

```
function [posit_x] = Positivization(x,type,i)
% 输入变量有三个:
% x: 需要正向化处理的指标对应的原始列向量
% type:  指标的类型 (1: 极小型,  2: 中间型,  3: 区间型)
% i: 正在处理的是原始矩阵中的哪一列
% 输出变量posit_x表示: 正向化后的列向量
if type == 1 %极小型
disp(['第' num2str(i) '列是极小型, 正在正向化'] )
posit_x = Min2Max(x); %调用Min2Max函数来正向化
disp(['第' num2str(i) '列极小型正向化处理完成'] )
disp('——————————————————分界线——————————————————' )
elseif type == 2 %中间型
disp(['第' num2str(i) '列是中间型'] )
best = input('请输入最佳的那一个值:  ');
posit_x = Mid2Max(x,best);
disp(['第' num2str(i) '列中间型正向化处理完成'] )
disp('——————————————————分界线——————————————————')
elseif type == 3 %区间型
disp(['第' num2str(i) '列是区间型'] )
a = input('请输入区间的下界:  ');
b = input('请输入区间的上界:  ');
posit_x = Inter2Max(x,a,b);
disp(['第' num2str(i) '列区间型正向化处理完成'] )
disp('——————————————————分界线——————————————————')
else
disp('没有这种类型的指标, 请检查Type向量中是否有除了1、2、3之外的其他值')
end
end
```

Topsis | 相关Matlab代码

➢ 其他函数

```
function [posit_x] = Min2Max(x)
posit_x = max(x) - x;
%posit_x = 1 ./ x; %如果x全部都大于0，也可以这样正向化
End

function [posit_x] = Mid2Max(x,best)
    M = max(abs(x-best));
    posit_x = 1 - abs(x-best) / M;
End
```

```
function [posit_x] = Inter2Max(x,a,b)
r_x = size(x,1); % row of x
M = max([a-min(x),max(x)-b]);
posit_x = zeros(r_x,1); %zeros函数用法: zeros(3)
zeros(3,1) ones(3)
% 初始化posit_x全为0 初始化的目的是节省处理时间
for i = 1: r_x
if x(i) < a
posit_x(i) = 1-(a-x(i))/M;
elseif x(i) > b
posit_x(i) = 1-(x(i)-b)/M;
else
posit_x(i) = 1;
end
end
end
```

Topsis | 相关Python代码

- **Python代码**

详见Topsis.py

欢迎关注数模加油站

THANKS