一、PPC 投影寻踪评估模型介绍

1974 年 Friedman 等[1]提出了将高维样本数据投影到低维子空间上的投影寻踪聚类(简称 PPC)原理,通过研究不同投影方向(空间角度)上样本投影点的分布规律、结构特性,以确定最优(也称为"感兴趣")投影方向。由于求最优解非常困难,目前最常用的是一维 PPC 模型,其目标函数是"使样本投影值 $z(i)$ 的标准差 S_z 与局部密度值 D_z 的乘积最大化",建模基本思想是"使样本点在整体上尽可能分散,并形成若干类(团),类与类之间尽可能分开,而各类内的样本点则应尽可能密集"。PPC 建模基本思想与人类进行综合评价的思维方式基本一致,而且可同时求得评价指标的客观权重和研究对象(样本)的综合得分,是一种较理想的综合评价方法,在学术界获得了广泛的应用[2-6]。根据上述 PPC 建模基本思想,不同学者提出了多种不同目标函数型式,但仍以 Friedman 等[1]提出的 PPC 模型效果最好。

为了从样本数据中挖掘出更多的有效信息,Friedman 等[1]提出可逐次建立第 2、第 3 和第 4 维的 SPPC 模型。由于求解 SPPC 模型目标函数的最优解非常困难,本文提出建立低维(1—4 维)逐次 PPC(简称 LDSPPC)模型的基本原理(已申请国家发明专利);建立逐次 PPC 模型的目标函数与约束条件;提出判定最优化过程是否求得真正全局最优解的判定准则和方法,从而确保求得各 SPPC 模型真正的全局最优解;提出采用矢量合成法将多个 SPPC 模型构建成综合的 LDSPPC 模型;将 LDSPPC 模型应用于供应商选择与评估研究,进行实证建模。结果表明,LDSPPC 模型能比一维 PPC 模型挖掘出更多的样本数据信息,是一种理想的综合评价新方法。

1.2 投影寻踪评价模型的建立

投影寻踪的基本思想是利用计算机技术,把高维数据通过某种组合,投影到低维(1~3 维)子空间上,并通过最优化某个投影指标,寻找出能反映高维数据结构或特征的投影,在低维空间上对数据结构进行分析,以达到研究和分析高维数据的目的[4]。投影寻踪评价模型(projection pursuit evaluation model, PPE)的建模过程包括如下 4 步[10]:

(1) 评价指标值的归一化。设定样本集为 $\{x^*(i,j)\mid i=1,n,j=1,p\}$。其中 $x^*(i,j)$ 为第 i 个样本的第 j 个指标值,n, p 分别为样本的个数和指标的个数。为消除各指标值的量纲效应和统一各指标值的变化范围,可用下式对指标值进行极值归一化处理:

$$x(i,j)=[x^*(i,j)-x_{\min}(j)]/[x_{\max}(j)-x_{\min}(j)] \quad (1)$$

其中,$x_{\max}(j)$、$x_{\min}(j)$分别为第 j 个指标的上限值和下限值。

(2) 投影指标函数 $Q(\alpha)$的构造。将 p 维数据$\{x(i,j)\mid j=1,p\}$综合成以$\{\alpha(j)\mid j=1,p\}$为投影方向的一维投影值进行研究,其表达式为

$$z(i)=\sum_{j=1}^{p}\alpha(j)x(i,j)\ (i=1,n) \quad (2)$$

其中,α 为单位长度向量。

根据投影寻踪思想,投影值 $z(i)$的分布在整体上应该尽量散开,在局部上应该尽量密集,故投影指标函数 $Q(\alpha)$可构造如下:

$$Q(a)=S(a)D(a) \quad (3)$$

其中,$S(a)$、$D(a)$分别为投影值 $z(i)$的标准差(数据散布特征)和局部密度。

$$S(a)=\sqrt{\sum_{i=1}^{n}(z(i)-\bar{z})^2/(n-1)} \quad (4)$$

$$D(a)=\sum_{i=1}^{n}\sum_{j=1}^{n}(R-r_{ij})u(R-r_{ij}) \quad (5)$$

其中,$\bar{z}$ 为投影值 $z(i)$的均值;R 为局部密度的窗口半径,一般可取 $R=0.1S(a)$;r_{ij} 为样本之间的距离,$r_{ij}=|z(i)-z(j)|$;$u(t)$为单位阶跃函数,当 $t\geq 0$ 时,$u(t)=1$,否则取 0。

(3) 投影指标函数的优化。由于不同的投影方向反映不同的数据结构特征,最佳投影方向 α 就是能够最大可能暴露多维数据某类结构特征的投影方向。因而可以通过求解如下非线性问题来估计 α:

$$\max\quad Q(a)=S(a)D(a) \quad (6)$$

$$\text{s.t.}\ \sum_{j=1}^{p}\alpha^2(j)=1 \quad (7)$$

(4) 评价模型的建立。将由步骤 3 求得的 α 代入(2)式,求得各样本的投影值 $z(i)$。$z(i)$与 $z(j)$越接近,则样本 i 与 j 越倾向于分为同一类。还可以通过比较投影值 $z(i)$的大小,来判断样本的优劣。

归一化处理

由于所选取指标的单位不统一,为了消除各评价指标量纲的影响,我们对正负向性不同的指标分别进行归一化处理。

对于正向性指标,归一化公式如下:

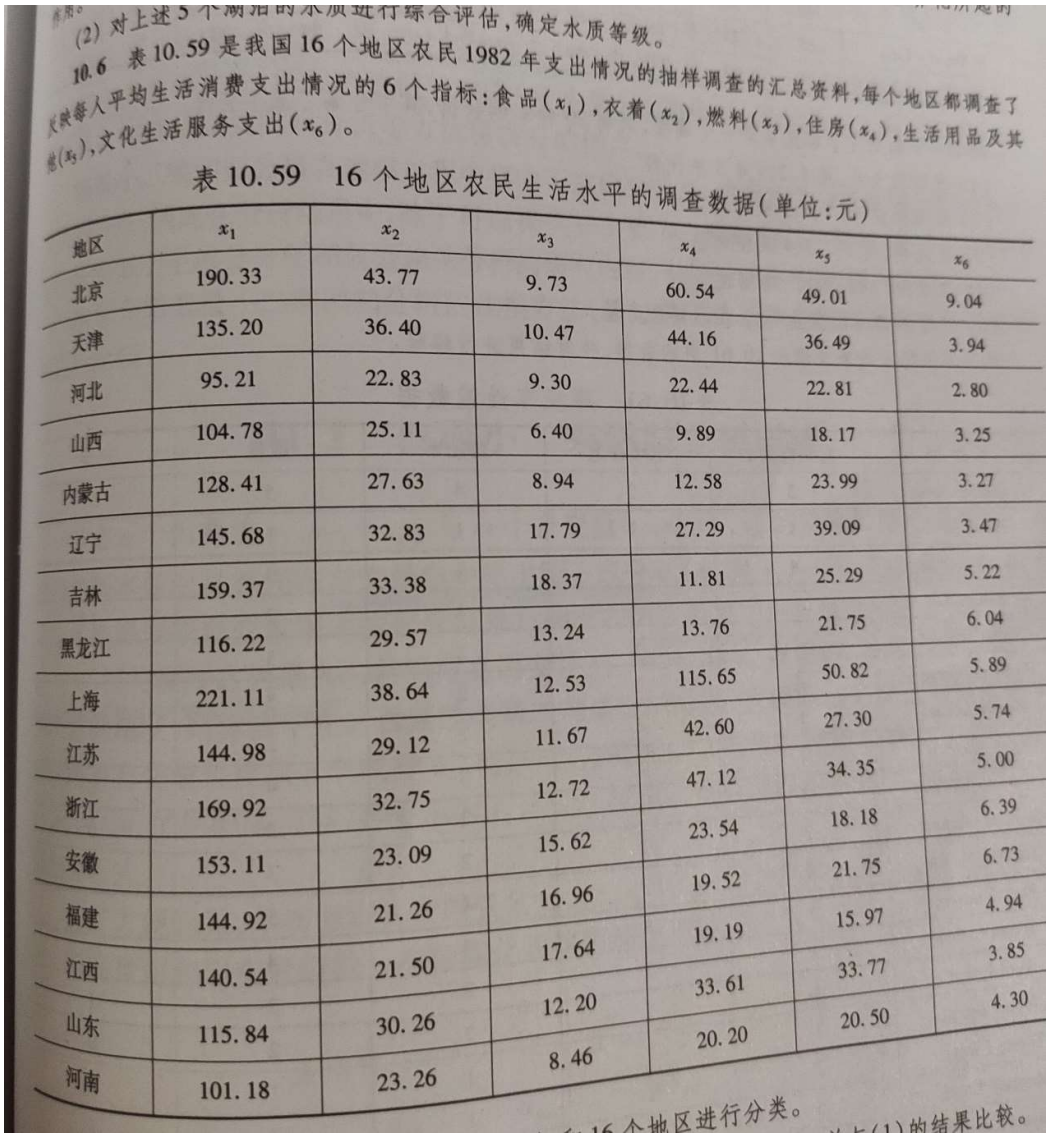
$$x_j=\frac{x_j-\min x_j}{\max x_j-\min x_j}$$

对于负向性指标,归一化公式如下:

$$x_j=\frac{\max x_j-x_j}{\max x_j-\min x_j}$$

其中 $\max x_j$ 为第 j 列指标的最大值, $\min x_j$ 为第 j 列指标的最小值。

二、例子：利用 PPC 评估 16 个地区消费水平



(2) 对上述 5 个湖泊的水质进行综合评估，确定水质等级。

10.6 表 10.59 是我国 16 个地区农民 1982 年支出情况的抽样调查的汇总资料，每个地区都调查了反映各人平均生活消费支出情况的 6 个指标：食品(x_1)，衣着(x_2)，燃料(x_3)，住房(x_4)，生活用品及其他(x_5)，文化生活服务支出(x_6)。

表 10.59　16 个地区农民生活水平的调查数据(单位:元)

地区	x_1	x_2	x_3	x_4	x_5	x_6
北京	190.33	43.77	9.73	60.54	49.01	9.04
天津	135.20	36.40	10.47	44.16	36.49	3.94
河北	95.21	22.83	9.30	22.44	22.81	2.80
山西	104.78	25.11	6.40	9.89	18.17	3.25
内蒙古	128.41	27.63	8.94	12.58	23.99	3.27
辽宁	145.68	32.83	17.79	27.29	39.09	3.47
吉林	159.37	33.38	18.37	11.81	25.29	5.22
黑龙江	116.22	29.57	13.24	13.76	21.75	6.04
上海	221.11	38.64	12.53	115.65	50.82	5.89
江苏	144.98	29.12	11.67	42.60	27.30	5.74
浙江	169.92	32.75	12.72	47.12	34.35	5.00
安徽	153.11	23.09	15.62	23.54	18.18	6.39
福建	144.92	21.26	16.96	19.52	21.75	6.73
江西	140.54	21.50	17.64	19.19	15.97	4.94
山东	115.84	30.26	12.20	33.61	33.77	3.85
河南	101.18	23.26	8.46	20.20	20.50	4.30

16 个地区进行分类。

(1)的结果比较。

2.1 Python Code

```
import numpy as np
from numpy import exp, array, std, sqrt
from numpy.random import rand


def open_txt(file:str, func=float):
    data = []
    f = open(file=file, mode='r', encoding='UTF-8')
    for line in f:
        tmp = line.split()
        for i, v in enumerate(tmp):
            tmp[i] = func(tmp[i])
```

```
        data.append(tmp)
    return array(data)

data = open_txt("consume.txt")
n, m = len(data), len(data[0])

# standardization
maxmin = [[data[0][j], data[0][j]] for j in range(m)]
for i in range(n):
    for j in range(m):
        maxmin[j][0] = max(maxmin[j][0], data[i][j])
        maxmin[j][1] = min(maxmin[j][1], data[i][j])
for i in range(n):
    for j in range(m):
        data[i][j] = (data[i][j] - maxmin[j][1]) / (maxmin[j][0] - maxmin[j][1])

def f(x):
    global data, n, m
    z = [0 for _ in range(n)]
    for i in range(n):
        tmp = 0
        for j in range(m):
            tmp += (x[j] * data[i][j])
        z[i] = tmp
    S = std(z)
    R = 0.1 * S
    D = 0
    for i in range(n):
        for j in range(m):
            r = abs(z[i] - z[j])
            t, u = R - r, 1
            if t <= 0:
                u = 0
            D += (t * u)
    return S * D

def get_new_x(x:np.array, step):
    m = len(x)
    new = rand(m)
    tmp = sum(new ** 2)
    new = sqrt(new ** 2 / tmp)
    return new
    new = x + step * rand(m)
    tmp = new ** 2
```

```
    new = sqrt(tmp / sum(tmp))
    return new

def SA():
    global n, m, data
    T, finalT, coef = 1000, 1, 0.97
    K, step, niter = 1, 1, 100
    tmp = rand(m)
    x0 = x = ansX = sqrt((tmp ** 2) / sum(tmp ** 2))
    y0, y, ansY = f(x0), f(x), f(ansX)
    while T > finalT:
        for _ in range(niter):
            newx = get_new_x(x, step)
            newy = f(newx)
            df1, df2 = newy - y, newy - ansY
            if df1 > 0:
                x, y = newx, newy
            elif exp(df1 / (K * T)) > rand():
                x, y = newx, newy
            if df2 > 0:
                ansX, ansY = newx, newy
        T *= coef
    #print(x0, y0)
    print(ansX, ansY)
    return ansX, ansY, x0, y0

def evaluate(x):
    global data, n, m
    z = []
    for i in range(n):
        tmp = 0
        for j in range(m):
            tmp += (data[i][j] * x[j])
        z.append(tmp)
    print(z)
    return z


#from datetime import datetime
#s = datetime.now()
x, y, x0, y0 = SA()
#print("-------------初始状态评估-------------")
#z0 = evaluate(x0)
print("-------------投影寻踪评估-------------")
```

```
z = evaluate(x)
#e = datetime.now()
#print("Running Time:", e - s)
province = ['北京', '天津', '河北', '山西', '内蒙古', '辽宁', '吉林', '黑龙江',
'上海', '江苏', '浙江', '安徽', '福建', '江西', '山东', '河南']
res = []
for i in range(n):
    res.append([province[i], z[i]])
res.sort(key=lambda x:x[1], reverse=True)
print('-------------消费水平由高到低-------------')
print(res)


'''
[0.50526319   0.32080724   0.1190597      0.33010081   0.44054848   0.56970791]
0.31253574570640813
-------------投影寻踪评估-------------
[1.8811314385096543,          0.8871869429853091,          0.17685785455327394,
0.16217123398791233, 0.4019762874941192, 0.8884765746708561, 0.8940325372108173,
0.6517394289551314, 1.8666952338699188, 0.8779154738696757, 1.0758501078324756,
0.7484590084640912, 0.7664622116587704, 0.5215459911643342, 0.6636626651327545,
0.299346035643722]
-------------消费水平由高到低-------------
[['北京', 1.8811314385096543], ['上海', 1.8666952338699188], ['浙江',
1.0758501078324756], ['吉林', 0.8940325372108173], ['辽宁', 0.8884765746708561],
['天津', 0.8871869429853091], ['江苏', 0.8779154738696757], ['福建',
0.7664622116587704], ['安徽', 0.7484590084640912], ['山东', 0.6636626651327545],
['黑龙江', 0.6517394289551314], ['江西', 0.5215459911643342], ['内蒙古',
0.4019762874941192], ['河南', 0.299346035643722], ['河北', 0.17685785455327394],
['山西', 0.16217123398791233]]

'''
```