宝洁洗衣粉成分与效果关系的统计分析

Statistical Analysis of The Relationship Between P&G Washing Powder Elements and Effect

沈致远

00601089

北京大学数学科学学院

2010年6月18日

摘 要

本文是宝洁洗衣粉成分与功效关系的统计分析。主要用了最小二乘估计及其扩展模型、岭回归、lasso回归对问题进行了分析。由于原始的数据有缺失，所以先用了k-最邻近法对缺失数据进行了填补，然后分别用三种方法进行回归，最后对三种模型分别计算残差平方和，取残差平方和最小的模型。

关 键 字

k-最近邻　　最小二乘　　岭回归　　lasso回归

目　录

一 问题背景

洗衣粉是通过其中的化学成分溶于水后改变水溶液的物理化学性质来实现去污的作用的，因此通过测量洗衣产品溶于水后的溶液的一些属性就可以了解产品去污的功效。如果能建立溶液属性和产品功效之间的模型，就可以找出能够最大化产品功效的溶液的属性，根据这些属性和化工技术知识我们就可以找出最优的配方。

为了研究洗衣粉溶液的物理属性对去污功效的影响，我们分别测量了96个不同产品溶液的物理属性和它们的去污效果的数据

已有的数据：

1. 现有96个产品的物理属性及功效数据，从中随机选取了10个产品作为验证模型预测精度的数据，请用剩下的86组数据来建立模型

2. 每一个产品的21个属性作为输入变量(PP1-PP21)

3. 产品在18种污渍上的功效作为输出变量(O1-O18)

二 完整数据

采用k-最临近法，首先用前50组完整的数据算出PP1∼PP21,O1∼O18的均值和标准差，然后标准化86组数据。由于PP1∼PP21中，PP2∼PP5有缺失数据，所以我们打算用剩下的17个变量定义距离（可以将这86组数据看成17维空间的一个点）。然后计算分别第51∼86组数据与1∼50组数据的距离，取距离最近的3组数据，把他们PP2∼PP5的数据的均值填入。

如此，我们就完成了数据的完整。为了方便以后我们的处理，我们在这里顺便把数据进行了扩充，加入了交叉项和平方项。R code 可见附录。

三 最小二乘回归

1 最小二乘回归

我们先对O1进行最小二乘回归

所得结果如下所示：

Call:

lm(formula = O1 PP1 + PP2 + PP3 + PP4 + PP5 + PP6 + PP7 + PP8 +

PP9 + PP10 + PP11 + PP12 + PP13 + PP14 + PP15 + PP16 + PP17 + PP18 + PP19 + PP20 + PP21, data = x)

Residuals:

Min 1Q Median 3Q Max

-31.455 -9.517 1.971 8.589 34.389

Coefficients:

	$Estimate$	$Std.Error$	$tvalue$	$Pr(>\|t\|)$
$(Intercept)$	-18.973740	108.883051	-0.174	0.86221
$PP1$	0.001489	0.002214	0.673	0.50364
$PP2$	0.849526	2.283580	0.372	0.71111
$PP3$	2.927286	1.584391	1.848	0.06929.
$PP4$	-0.331505	0.534265	-0.620	0.53714
$PP5$	0.169217	0.321850	0.526	0.60087
$PP6$	-1.015953	0.655879	-1.549	0.12631
$PP7$	-0.773854	0.696729	-1.111	0.27085
$PP8$	-0.356656	0.353244	-1.010	0.31646
$PP9$	0.894093	0.319989	2.794	$0.00686**$
$PP10$	-4.546350	1.932127	-2.353	$0.02171*$
$PP11$	5.889037	2.698516	2.182	$0.03276*$
$PP12$	-4.182594	2.339202	-1.788	0.07850.
$PP13$	4.427934	1.544324	2.867	$0.00560**$
$PP14$	0.998679	2.884026	0.346	0.73027
$PP15$	-0.418696	2.575636	-0.163	0.87138
$PP16$	0.811263	2.265484	0.358	0.72145
$PP17$	-2.500560	3.399915	-0.735	0.46474
$PP18$	-2.320157	3.644807	-0.637	0.52668
$PP19$	0.038398	2.518068	0.015	0.98788
$PP20$	4.079415	3.689877	1.106	0.27305
$PP21$	0.544304	3.522722	0.155	0.87769

Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘’ 1

Residual standard error: 15.62 on 64 degrees of freedom

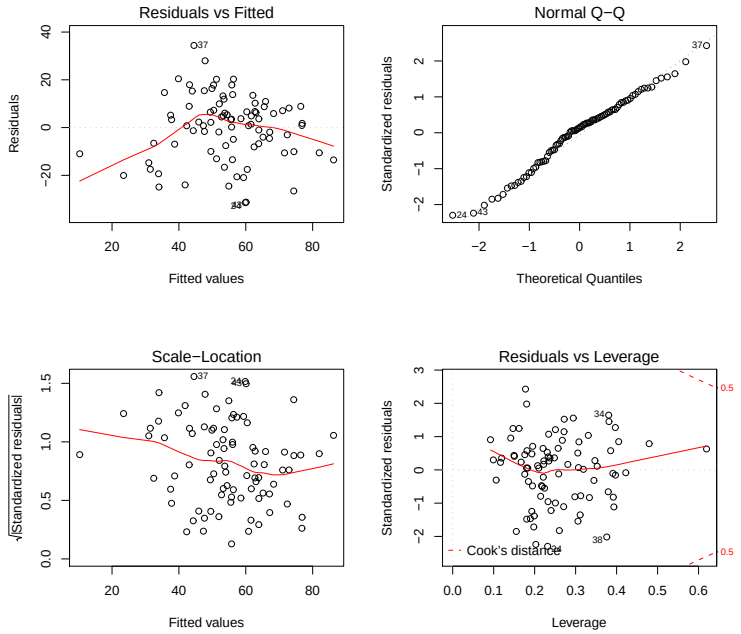
Multiple R-squared: 0.4914, Adjusted R-squared: 0.3246

F-statistic: 2.945 on 21 and 64 DF, p-value: 0.0004752

由上结果可知，只有PP9，PP10,PP11,PP13显著，所以必定存在很强的共线性。这里我们采用VIF检测共线性。

2　正态性及异方差检验

我们用QQ图来检测误差的正态性，用学生化残差与拟合值的散点图来判断异方差。有下图可知：误差基本服从正态分布，且不存在异方差问题。



3　共线性检验

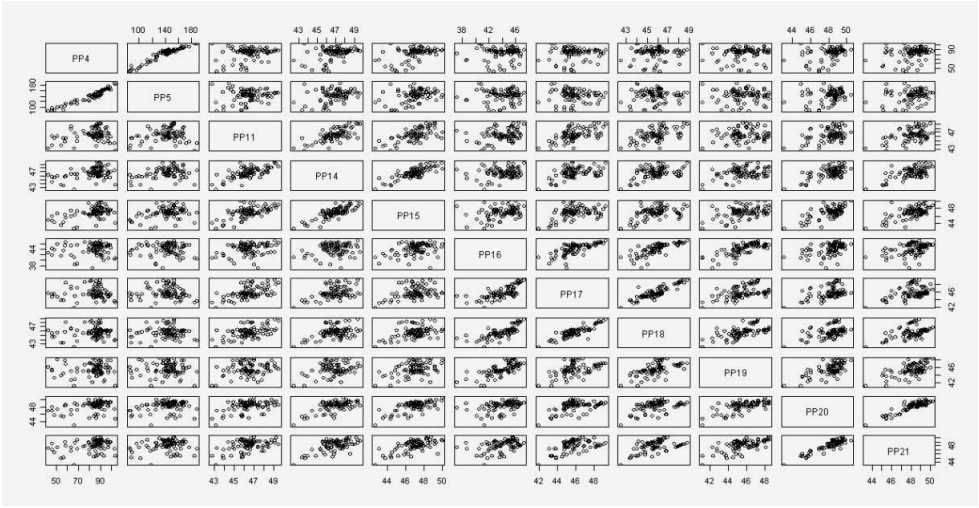
有上面的分析可知，自变量之间可能存在共线性，所以我们用VIF来检验共线性。这里我们之所以采用VIF为标准检验共线性，是因为我们可以证明

$$Var(\beta_j) = \sigma^2(\frac{1}{1-R_j^2})(\frac{1}{SXX_j}) \tag{3.1}$$

我们称$\frac{1}{1-R_j^2}$为方差扩大影子，即所谓的VIF。由此我们可以看出若VIF越大，则$\frac{1}{1-R_j^2}$越大, 而R_j^2为β_j与其他自变量的相关系数，所以R_j^2越大,共线性就越强。在这里，我们认为若VIF大于5就认为该变量与其他变量存在共线性。从VIF可以看

出PP4,PP5,PP11,PP14,PP15,PP16,PP17,PP18,PP19,PP20,PP21的VIF较大，则可以认为他们之间存在着共线性问题，下面我们画出他们之间的两两散点图，看一看是否存在共线性。

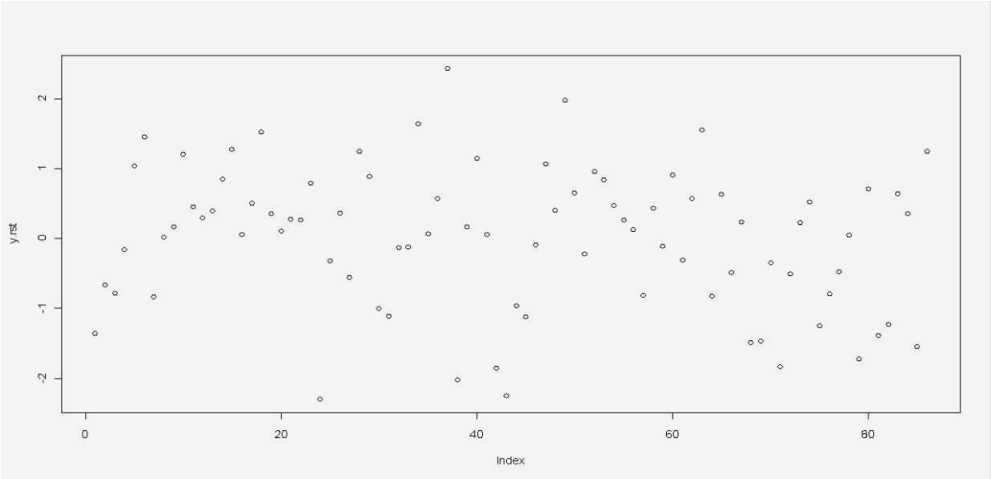
```
> vif(lm.o1)
    PP1     PP2     PP3     PP4     PP5     PP6     PP7     PP8     PP9
 3.1443  4.0223  3.2745 16.5960 15.8210  1.6987  1.6273  2.5579  3.2307
   PP10    PP11    PP12    PP13    PP14    PP15    PP16    PP17    PP18
 3.4522  4.9790  4.7060  3.0035  5.3995  4.7971  7.2765  8.9852  7.6964
   PP19    PP20    PP21
 5.9996  8.6997  7.5300
```



从上图可知自变量之间确实存在共线性。

4　异常值检验

这里，我们采用画出学生化内残差的散点图和cook距离来查看是否存在散点图。一般认为只要某个点的学生化残差与其他点有很大区别就认为这个点事异常值。这里，我们认为cook距离大于0.1即可认为可能是异常值，当然还得看看该店在残差图上的位置。



cooks.distance(lm.o1)

```
> cooks.distance(lm.o1)
           1            2            3            4            5            6            7            8            9           10           11           12
3.754511e-02 1.204561e-02 1.253139e-02 7.357998e-04 2.417560e-02 5.905419e-02 1.576083e-02 5.752495e-06 3.607481e-04 2.223884e-02 2.024289e-03 4.496393e-04
          13           14           15           16           17           18           19           20           21           22           23           24
2.124247e-03 2.224439e-02 4.848250e-02 3.140211e-05 5.125608e-03 3.938269e-02 1.819963e-03 2.968609e-04 1.879810e-03 9.497984e-04 2.596290e-02 7.233447e-02
          25           26           27           28           29           30           31           32           33           34           35           36
2.394083e-03 1.960595e-03 3.959659e-03 1.364543e-02 1.312375e-02 1.504943e-02 2.064224e-02 1.648747e-04 2.325302e-04 7.562977e-02 9.436711e-05 3.356561e-03
          37           38           39           40           41           42           43           44           45           46           47           48
5.788928e-02 1.114215e-01 4.828723e-04 2.231349e-02 3.143863e-05 2.857976e-02 5.802001e-02 1.256683e-02 3.654977e-02 2.448487e-04 1.595346e-02 1.332891e-03
          49           50           51           52           53           54           55           56           57           58           59           60
3.915635e-02 6.626731e-03 6.241526e-04 6.835713e-03 1.433748e-02 2.493870e-03 9.892729e-04 1.899251e-04 1.261012e-02 1.515920e-03 3.278449e-04 3.794859e-03
          61           62           63           64           65           66           67           68           69           70           71           72
5.098216e-04 9.477465e-03 4.590648e-02 1.970766e-02 2.923700e-02 2.554759e-03 4.988781e-04 2.199177e-02 2.286782e-02 1.260316e-03 5.324665e-02 3.207746e-03
          73           74           75           76           77           78           79           80           81           82           83           84
3.065075e-04 3.805526e-03 1.686343e-02 7.821033e-03 2.834304e-03 3.475284e-05 3.301341e-02 5.528315e-03 2.165170e-02 2.139126e-02 5.361545e-03 7.759387e-04
          85           86
4.754757e-02 1.212546e-02
```

从残差图和cook距离来看，并不存在异常值。

5 用AIC为标准删减变量

此处我们选择的是逐步向后选择的方法来进行子集选择。逐步的方法是通过寻找一条通过可能子集的好的路径，而不是搜索所有可能的子集（当变量个数较多时，搜索所有的子集不可行）。

逐步向后的方法，是从完全模型开始，在每一步中消去一个变量。如在方程的所有变量中，要消去的变量具有最小的t值或F值。当然，在这里我们选择的是AIC准则，而对于高斯模型，AIC统计量又等价于C_p统计量，C_p统计量由下式定义：

$$C_p = \frac{RSS_p}{\sigma^2} + 2p - n \tag{3.2}$$

$$= \frac{RSS_p - RSS_{k'}}{\sigma^2} + p - (k' - p) \tag{3.3}$$

$$= (k' - p)(F_p - 1) + p \tag{3.4}$$

其中，k'是完全模型的变量个数，σ^2来自完全模型，F_p是假设”所有在子集模型外，但在完全模型内的自变量系数都为0“的通常的F统计量。

这里当任何一个变量被删除时的AIC，都大于不删去变量时的AIC时，变量选择就停止。

$step(lm.o1)$

Step: AIC=473.72

O1 ~ PP3 + PP6 + PP8 + PP9 + PP10 + PP11 + PP12 + PP13 + PP17 + PP20

	Df	$SumofSq$	RSS	AIC
$none$			16430	473.72
$PP6$	1	665.2	17095	475.13
$PP8$	1	685.7	17116	475.23
$PP20$	1	1023.2	17453	476.91
$PP17$	1	1225.2	17655	477.90
$PP12$	1	1309.1	17739	478.31
$PP10$	1	1358.6	17789	478.55
$PP3$	1	1455.6	17886	479.02
$PP11$	1	2455.9	18886	483.70
$PP13$	1	2758.0	19188	485.06
$PP9$	1	3684.1	20114	489.12

Call:

lm(formula = O1 ~ PP3 + PP6 + PP8 + PP9 + PP10 + PP11 + PP12 + PP13 + PP17 + PP20, data = x)

Coefficients:

$(Intercept)$	$PP3$	$PP6$	$PP8$	$PP9$	$PP10$	$PP11$	$PP12$
−28.3262	2.3471	−0.9587	−0.5252	0.9645	−4.0852	6.3908	−4.5489

$PP13$	$PP17$	$PP20$
4.7145	−3.6334	3.6688

选出变量PP3，PP6，PP9，PP10，PP11，PP12,PP13,PP17,PP20继续做回归

$lm.o1.1 < -lm(O1\ PP3 + PP6 + PP8 + PP9 + PP10 + PP11 + PP12 + PP13 + PP17 + PP20, data = x)$

$summary(lm.o1.1)$

Call:

lm(formula = O1 PP3 + PP6 + PP8 + PP9 + PP10 + PP11 + PP12 + PP13 + PP17 + PP20, data = x)

Residuals:

Min 1Q Median 3Q Max

-30.562 -8.663 1.005 9.352 33.474

Coefficients:

	$Estimate$	$Std.Error$	$tvalue$	$Pr(>\|t\|)$
$(Intercept)$	-28.3262	80.4486	-0.352	0.725748
$PP3$	2.3471	0.9106	2.578	0.011908*
$PP6$	-0.9587	0.5502	-1.743	0.085504.
$PP8$	-0.5252	0.2969	-1.769	0.080928.
$PP9$	0.9645	0.2352	4.101	0.000103***
$PP10$	-4.0852	1.6404	-2.490	0.014976*
$PP11$	6.3908	1.9087	3.348	0.001274**
$PP12$	-4.5489	1.8608	-2.445	0.016853*
$PP13$	4.7145	1.3287	3.548	0.000673***
$PP17$	-3.6334	1.5364	-2.365	0.020624*
$PP20$	3.6688	1.6976	2.161	0.033870*

Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘’ 1

Residual standard error: 14.8 on 75 degrees of freedom

Multiple R-squared: 0.4648, Adjusted R-squared: 0.3934

F-statistic: 6.514 on 10 and 75 DF, p-value: 3.707e-07

选择变量PP3，PP9，PP10，PP11，PP12，PP13，PP17，PP20继续做回归

$lm.o1.2 < -lm(O1\ PP3 + PP9 + PP10 + PP11 + PP12 + PP13 + PP17 + PP20, data = x)$

$summary(lm.o1.2)$

Call:

lm(formula = O1 PP3 + PP9 + PP10 + PP11 + PP12 + PP13 + PP17 + PP20, data = x)

Residuals:

Min 1Q Median 3Q Max

-37.424 -10.180 2.960 9.662 32.646

Coefficients:

	$Estimate$	$Std.Error$	$tvalue$	$Pr(>\|t\|)$
$(Intercept)$	-35.2342	81.6758	-0.431	0.66739
$PP3$	2.2295	0.9421	2.367	0.02046*
$PP9$	0.6711	0.2121	3.164	0.00223 **
$PP10$	-3.8981	1.6967	-2.297	0.02431*
$PP11$	6.6522	1.9665	3.383	0.00113 **
$PP12$	-5.5914	1.8837	-2.968	0.00399 **
$PP13$	4.6073	1.3741	3.353	0.00124 **
$PP17$	-2.9179	1.5606	-1.870	0.06532.
$PP20$	2.3848	1.6907	1.410	0.16242

—

Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘’ 1

Residual standard error: 15.33 on 77 degrees of freedom

Multiple R-squared: 0.4105, Adjusted R-squared: 0.3493

F-statistic: 6.704 on 8 and 77 DF, p-value: 1.243e-06

选择PP3，PP9，PP10，PP11，PP12，PP13继续做回归

$lm.o1.3 < -lm(O1\ PP3 + PP9 + PP10 + PP11 + PP12 + PP13, data = x)$

$summary(lm.o1.3)$

$lm.o1.3 < -lm(O1\ PP3 + PP9 + PP10 + PP11 + PP12 + PP13, data = x)$

$summary(lm.o1.3)$

Call:

lm(formula = O1 PP3 + PP9 + PP10 + PP11 + PP12 + PP13, data = x)

Residuals:

Min 1Q Median 3Q Max

-42.178 -10.075 2.558 10.595 31.355

Coefficients:

$$\begin{pmatrix} & Estimate & Std.Error & tvalue & Pr(>|t|) \\ (Intercept) & -4.4586 & 68.5760 & -0.065 & 0.948325 \\ PP3 & 2.0500 & 0.9472 & 2.164 & 0.033464* \\ PP9 & 0.8464 & 0.1888 & 4.482 & 2.47e-05*** \\ PP10 & -4.7131 & 1.6685 & -2.825 & 0.005987** \\ PP11 & 6.4976 & 1.9751 & 3.290 & 0.001499** \\ PP12 & -6.4036 & 1.7771 & -3.603 & 0.000547*** \\ PP13 & 5.2134 & 1.3633 & 3.824 & 0.000261*** \end{pmatrix}$$

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 15.56 on 79 degrees of freedom

Multiple R-squared: 0.3767, Adjusted R-squared: 0.3294

F-statistic: 7.958 on 6 and 79 DF, p-value: 1.004e-06

此时所有变量都已经显著

6 回归扩展模型

将模型扩展，即加入交叉项和平方项后，再进行回归，再做一次AIC变量选择就得出最后的模型。

Call:

lm(formula = O1 PP3 + PP9 + PP10 + PP11 + PP12 + PP13 + PP3:PP12 + PP9:PP10 + PP9:PP13 + PP11:PP12, data = x)

Coefficients:

$$\begin{pmatrix} (Intercept) & PP3 & PP9 & PP10 & PP11 & PP12 & PP13 & PP3:PP12 \\ -4670.1751 & 98.0450 & 3.7382 & -11.5130 & 95.9509 & 92.1530 & 13.8849 & -2.0530 \end{pmatrix}$$

$$\begin{pmatrix} PP9:PP10 & PP9:PP13 & PP11:PP12 \\ 0.3034 & -0.3694 & -1.9241 \end{pmatrix}$$

然后可得残差平方和

[1] 14872.68

对于O2∼O18,方法类似，限于篇幅，本文不再赘述。

四 岭回归

1 岭回归的理论介绍

岭回归通过其容量加罚来收缩回归系数。岭回归的极小化罚残差平方和为:

$$\beta^{ridge} = argmin_{\beta} \sum_{i=1}^{N} (y_i - \beta_0 - \sum_{j=1}^{p})^2 + \lambda \sum_{j=1}^{p} \beta_j^2\} \tag{4.1}$$

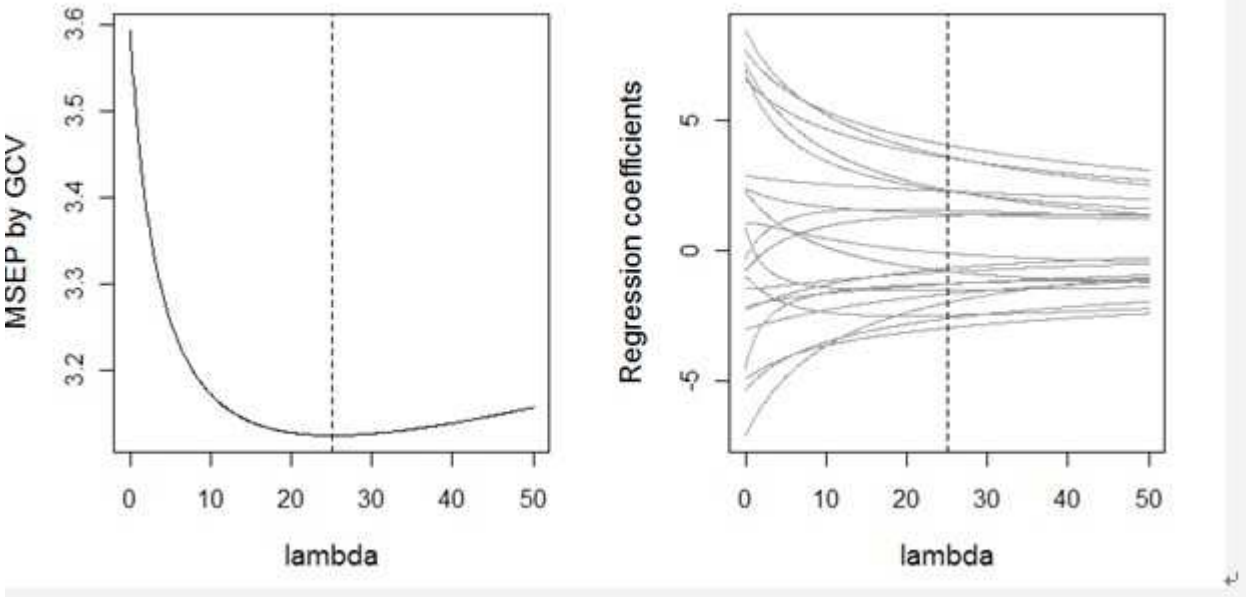
这里,λ是控制收缩量的复杂度参数:λ值越大，收缩量越大。当线性回归模型中存在多个相关变量时（前面已经通过考察共线性得知），它们的系数确定性变差，并呈现高方差。在一个变量上的很大的正系数可能被在其相关变量上类似大小的负系数抵消。通过在系数上施加一个量约束，可以避免这种现象的发生。

2 岭回归

对于第一个相应变量O1，先用chemometrics包里面的plotRidge函数考查岭回归的预测均方误差MSEP

$res_ridge1 < -plotRidge(O1\ PP1 + PP2 + PP3 + PP4 + PP5 + PP6 + PP7 + PP8 + PP9 + PP10 + PP11 + PP12 + PP13 + PP14 + PP15 + PP16 + PP17 + PP18 + PP19 + PP20 + PP21, data = yy, lambda = seq(0, 50, 0.1))$

得到如下结果：



可以看到最佳的岭回归参数lambda在25左右。可以用MASS包中的select函数精确计算：

$select(lm.ridge(O1\ PP1 + PP2 + PP3 + PP4 + PP5 + PP6 + PP7 + PP8 + PP9 + PP10 + PP11 + PP12 + PP13 + PP14 + PP15 + PP16 + PP17 + PP18 + PP19 + PP20 + PP21, lambda = seq(0, 50, 0.5)))$

输出结果为：

modified HKB estimator is 10.38313

modified L-W estimator is 24.97606

smallest value of GCV at 25

进一步精确确定lambda值：

$select(lm.ridge(O1\ PP1 + PP2 + PP3 + PP4 + PP5 + PP6 + PP7 + PP8 + PP9 + PP10 + PP11 + PP12 + PP13 + PP14 + PP15 + PP16 + PP17 + PP18 + PP19 + PP20 + PP21, lambda = seq(24, 26, 0.05)))$

输出结果为：

modified HKB estimator is 10.38313

modified L-W estimator is 24.97606

smallest value of GCV at 25.1

故岭回归参数取为25.1：

$fit1 <- lm.ridge(O1\ PP1+PP2+PP3+PP4+PP5+PP6+PP7+PP8+PP9+PP10+PP11+PP12+PP13+PP14+PP15+PP16+PP17+PP18+PP19+PP20+PP21, lambda = 25.1)$

岭回归结果为：

```
> fit1
                  PP1          PP2          PP3          PP4          PP5          PP6
35.754277650  0.001683697 -0.537104109  1.814563735 -0.089528438 -0.060775588 -0.493082555
         PP7          PP8          PP9         PP10         PP11         PP12         PP13
-0.411777037 -0.088413876  0.426086626 -1.219415739  1.660741801 -1.653780406  1.900840387
        PP14         PP15         PP16         PP17         PP18         PP19         PP20
 1.041925931  0.943837802 -0.051486011 -1.981326232 -1.922671321 -0.489088783  1.688881838
        PP21
 1.173600272
```

考察拟合的残差平方和：

输出为：[1] 17262.72

对于O2∼O18，方法类似，限于篇幅，不再赘述。

五　Lasso回归

1　Lasso回归理论介绍

Lasso是一种收缩方法，与岭回归类似。Lasso估计由下式定义:

$$\beta = argmin_\beta \sum_{i=1}^{N}(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{p} |\beta_j| \tag{5.1}$$

从上式，我们可以发现其实lasso回归就是把岭回归的惩罚函数$\sum_{j=1}^{p} \beta_j^2$换成了$\sum_{j=1}^{p} |\beta_j|$。而后一个约束使得解在y_i上非线性，由于该约束的特性，使得t充分小后，将导致某些系数恰好为0。这样，lasso相当于做了某种连续的子集选择。

对于t的选择，若$t = t_0 = \sum_{j=1}^{p} |\beta_j|$（其中$\beta_j$为最小二乘估计），则此时lasso估计为最小二乘估计。t应当自适应的选取，从而最小化期望预测误差估计。这里，期望预测误差估计我们成为cv误差。

对于如何来估计期望预测误差，我们这里采用10折交叉验证的方法。我们将数据分成容量大致相等的K部分（K=10），对于第k部分，我们用模型拟合数据的其余K-1部分，并用第k部分计算拟合模型的预测误差。我们对于k=1，2，……，

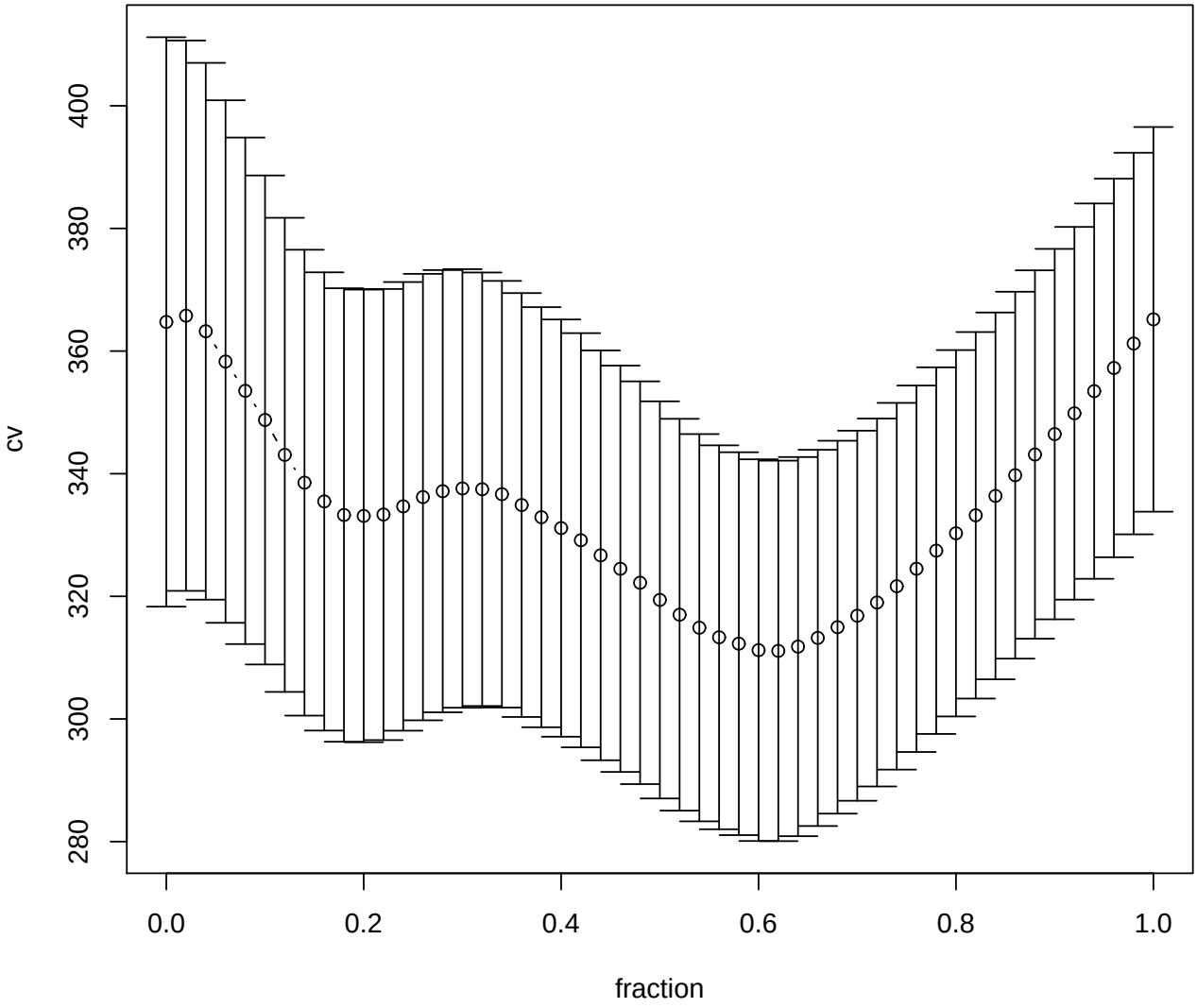
K这样做，并加总预测误差的K个估计。若用数学式子来表示，令$\kappa : 1, 2, ,,, N \mapsto 1, 2, ,,K$是一个指示函数，他指出观测i被随机指派到其上的划分。用$f^{-\kappa(i)}(x)$表示拟合函数，用删除第k部分后的数据来计算。于是预测误差的交叉验证估计为:

$$CV = \frac{1}{N}\sum_{i=1}^{N} L(y_i, f^{-\kappa(i)}(x_i)) \tag{5.2}$$

2 lasso回归

用lasso方法回归，我们先用10折交叉验证画出cv误差曲线图，取出cv误差值最小的收缩系数。然后我们算出最小二乘系数的绝对值之和，其与收缩系数的乘积为λ。接着，我们用前面算出的λ值做lasso回归得出回归系数。对于收缩系数的选取我们采用了lars 的软件包，软件包可以进行K折交叉验证，我们由此得出cv误差图，并选出使得cv误差最小的收缩系数。对于lasso回归我们编写了一个函数，其中用到了nlm函数命令，nlm可以用牛顿方法计算使得函数f最小的系数。

用10折交叉验证画cv误差图



如上图，我们取收缩系数为0.6。我们先提取出最小二乘方法系数的绝对值之和，再编写一个lasso回归的函数my.lasso，对于小于1-e03的系数，我们取为0。

$coef.o1 < -my.lasso(y, x, 0.6 * coefsum[1])$

$coef.o1$

coef.o1 [,1]

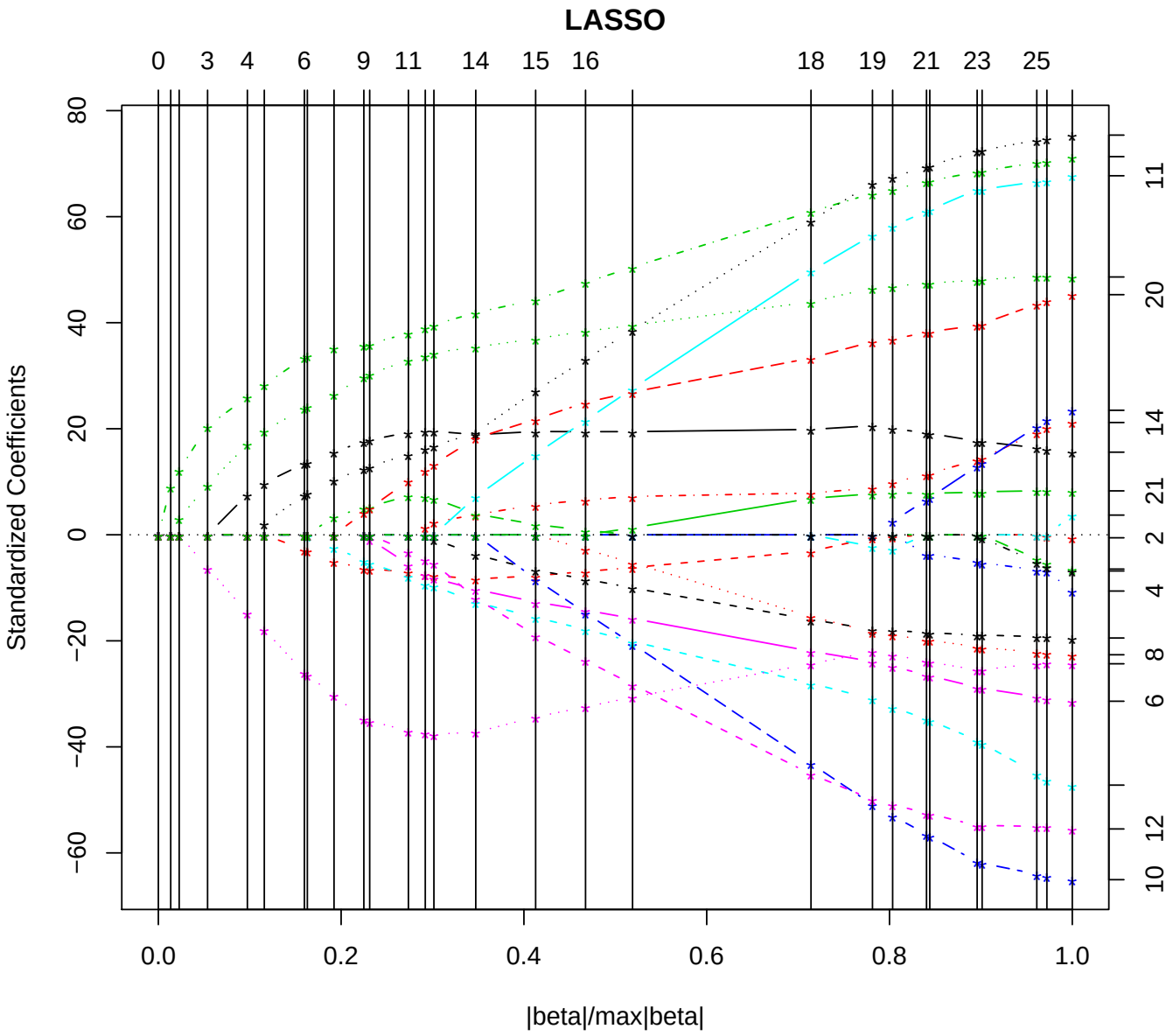
[1,] 54.664186047

[2,] 0.001552216
[3,] 0.000000000
[4,] 2.570786898
[5,] 0.088943095
[6,] 0.000000000
[7,] 0.000000000
[8,] 0.000000000
[9,] 0.000000000
[10,] 0.702464841
[11,] 0.000000000
[12,] 2.276442694
[13,] 0.000000000
[14,] 3.161132572
[15,] 1.288524126
[16,] 1.264564015
[17,] 0.000000000
[18,] 0.000000000
[19,] 0.000000000
[20,] 0.000000000
[21,] 2.384268725
[22,] 2.032665650

再算残差平方和

sse.o1
[,1]
[1,] 33048863

我们还可以画出系数收缩图



对于O2∼O18,方法类似，限于篇幅，不再赘述。

六　模型的选择

我们分别用三种方法构建出了三种模型，现在，我们根据残差平方和得出最优模型。我们将三种模型的18个分量的残差平方和列表如下：

Y	LS	$Ridge$	$Lasso$	$BestModel$
$O1$	14872.68	17262.72	33048863	LS
$O2$	1011.083	1150.052	283293.8	LS
$O3$	524.6595	614.7352	379786.6	LS
$O4$	2709.626	8550.442	15434515	LS
$O5$	8746.34	16246.11	31054440	LS
$O6$	16106.94	15559.99	36357751	$Ridge$
$O7$	1547.747	1419.099	1734393	$Ridge$
$O8$	297.5669	546.8619	449311.3	LS
$O9$	12516.21	12721.59	17183263	LS
$O10$	15799.50	14724.6	20831709	$Ridge$
$O11$	17843.23	20021.6	27259479	LS
$O12$	12301.66	18170.44	27721326	LS
$O13$	6906.18	10240.50	14931353	LS
$O14$	4515.069	8856.308	9702493	LS
$O15$	527.472	596.7319	1777604	LS
$O16$	1092.495	1673.691	658779.3	LS
$O17$	1358.440	1631.813	2021032	LS
$O18$	1611.024	1980.614	2809344	LS

七 总结与感谢

我之所以选择把宝洁洗衣粉的问题作为毕业论文，是因为我打算把大学四年期间所学的统计知识做一个梳理，并把其应用到实际问题的分析中去。在实际的操作过程中，我才发现原来有好多的问题，虽然理论上知道了怎么解决，但要通过实际编程解决，还是会碰到不少问题。只有通过不断的摸索试验，才能找到令人满意的解答。

比如，在做lasso回归时，我曾想用leave-one-out的交叉验证的方法计算lasso回归的λ值，一个原因是因为leave-one-out的编程容易实现，而且我想熟悉一下R的编程，但是结果并不理想。我得出的结果是，大部分的收缩系数都为1，这就意味着系数都得取最小二乘的系数，所以leave-one-out的方法无法得出最佳的值，我想主要是因为，leave-one-out只选择了一个观测值作为误差的估计，而我们需要更多的样本来进行误差估计，所以最后我选用了10折交叉验证的方

法，最后的结果表明，10折交叉验证确实比leave-one-out要好。

这次宝洁统计大赛，我和唐明宇、占翔一个小组参赛，在比赛期间，我得到了他们俩的不少帮助，我们共同探讨问题，解决疑惑，所以我要感谢他俩的帮助。当然，我还得感谢姚远老师，是他在讨论班上给我们指点迷津，才使得我们能攻克不少难关。最后，我还得感谢父母，正是他们对我的不懈支持才让我坚持走到了现在，才有了现在的成绩。

八 参考文献

[1] Trevor Hastie，Robert Tibshirani，Jerome Friedman *The elements of Statistical Learning*，世界图书出版公司，2009.

[2] S • Weisberg 应用线性回归, 中国统计出版社,2006.

[3] 汤银才，*R*语言与统计分析，高等教育出版社，2007.

九　附录：程序代码

```
#最小二乘回归
setwd("D:/宝洁")
x<-read.csv("数据一.csv")
y<-x[1:50, 2:40]
mean<-apply(y,MARGIN=2,FUN=mean)            #对1~86组数据标准化，方便定义距离
sd<-apply(y,MARGIN=2,FUN=sd)
yy<-x[,2:40]
for(i in 1:39){
yy[,i]<-(yy[,i]-mean[i])/sd[i]
}

xx<-x[,2:40]                                #对有缺失值的数据，选择距离最近的
三组数据，用此三组数据的均填填补，距离定义为PP1，PP6~PP21这17个分量距离差的绝对
值的和

for(r in 51:86){
summ<-rep(0,50)                             #计算yy[r,]与前50组数据中的第i组
数据的距离，记为summ[i]

for(i in 1:50){
summ[i]<-0
for(j in 6:21){
summ[i]<-abs(yy[i,j]-yy[r,j])+summ[i]
}
summ[i]<-summ[i]+abs(yy[i,1]-yy[r,1])
}

minorder<-order(summ)[1:3]                  #minorder存放与r行距离最近的三
行的行号，如果第r行2，3栏是缺失值，则用与第r行最近的三行的2，3的平均数填补，如果
第r行4，5栏是缺失值，则用与第r行最近的三行的4，5的平均数填补。

if(is.na(xx[r,2])) {                        #由于观察到PP2，PP3与PP4，PP5数
据是成对缺失，所以只需判断PP2或PP4是否为NA值
xx[r,2]<-(xx[minorder[1],2]+xx[minorder[2],2]+xx[minorder[3],2])/3
```

```
xx[r,3]<-(xx[minorder[1],3]+xx[minorder[2],3]+xx[minorder[3],3])/3
}
if(is.na(xx[r,4])) {
xx[r,4]<-(xx[minorder[1],4]+xx[minorder[2],4]+xx[minorder[3],4])/3
xx[r,5]<-(xx[minorder[1],5]+xx[minorder[2],5]+xx[minorder[3],5])/3
}

}
                                                    #为了以后便于处理，在这里我们就进
行扩展数据，即加入平方项和交叉项
xxx<-xx
for(i in 1:21){                                      #补充交叉项和二次项，V（39+(i-1）
*21+j)是PPi和PPj的乘积的交叉项

for(j in 1:21){
xxx[,39+(i-1)*21+j]<-xx[,i]*xx[,j]
}
}
zz<-xx
zzz<-xxx
xxx.mean<-apply(xxx,MARGIN=2,FUN=mean)          #对xxx进行标准化
xxx.sd<-apply(xxx,MARGIN=2,FUN=sd)
xxx<-sweep(xxx,MARGIN=2,STATS=xxx.mean,FUN="-")
xxx<-sweep(xxx,MARGIN=2,STATS=xxx.sd,FUN="/")
xx<-xxx[,1:39]                                       #提取xxx的前39项（不包括平方
项和交叉项）给xx

library(MASS)
library(rpart)
library(randomForest)
library(DAAG)
x<-zz[,1:39]
attach(x)
lm.o1<-lm(O1~PP1+PP2+PP3+PP4+PP5+PP6+PP7+PP8+PP9+PP10+PP11+PP12+PP13+PP14+PP15+PP16+PP17+PP18
summary(lm.o1)

par(mfrow=c(2,2))
```

```
plot(lm.o1)          #正态性检验
vif(lm.o1)           #共线性检验

pairs(x[,c(4,5,11,14,15,16,17,18,19,20,21)]) #两两散点图

y.rst<-rstandard(lm.o1) #异常值检验
y.fit<-predict(lm.o1)
plot(y.rst)

step(lm.o1) #用backward stepwise选择变量

lm.o1.1<-lm(O1~PP3+PP6+PP8+PP9+PP10+PP11+PP12+PP13+PP17+PP20,data=x) #删 去 不
显著变量 直到所有变量都显著
summary(lm.o1.1)

lm.o1.2<-lm(O1~PP3+PP9+PP10+PP11+PP12+PP13+PP17+PP20,data=x)
summary(lm.o1.2)

lm.o1.3<-lm(O1~PP3+PP9+PP10+PP11+PP12+PP13,data=x)
summary(lm.o1.3)


a<-c(3,9,10,11,12,13) #将模型扩展（即加入平方项和交叉项）再做回归
b<-a%*%t(a)
c<-c(c(3,9,10,11,12,13),(c(b[,1],b[2:6,2],b[3:6,3],b[4:6,4],b[5:6,5],b[6,6])+39),22)

x<-zzz[,c]

colnames(x)<-c("PP3","PP9","PP10","PP11","PP12","PP13","PP3*PP3","PP3*PP9","PP3*PP10","PP3*PP
"PP9*PP11","PP9*PP12","PP9*PP13","PP10*PP10","PP10*PP11","PP10*PP12","PP10*PP13","PP11*PP11"
"PP13*PP13","O1")

lm.o1.big<-lm
(O1~PP3+PP9+PP10+PP11+PP12+PP13+PP3*PP3+PP3*PP9+PP3*PP10+PP3*PP11+PP3*PP12+PP3*PP13+PP9*PP9+P
+PP10*PP13+PP11*PP11+PP11*PP12+PP11*PP13+PP12*PP12+PP12*PP13+PP13*PP13,data=x)
summary(lm.o1.big)
```

```
step(lm.o1.big)


#岭回归
Library（MASS）
Library（chemometrics）
res_ridge1<-plotRidge(O1~PP1+PP2+PP3+PP4+PP5+PP6+PP7+PP8+PP9+PP10+PP11+PP12+PP13+PP14+PP15+PP
select(lm.ridge(O1~PP1+PP2+PP3+PP4+PP5+PP6+PP7+PP8+PP9+PP10+PP11+PP12+PP13+PP14+PP15+PP16+PP
select(lm.ridge(O1~PP1+PP2+PP3+PP4+PP5+PP6+PP7+PP8+PP9+PP10+PP11+PP12+PP13+PP14+PP15+PP16+PP
fit1<-lm.ridge(O1~PP1+PP2+PP3+PP4+PP5+PP6+PP7+PP8+PP9+PP10+PP11+PP12+PP13+PP14+PP15+PP16+PP1

#Lasso回归
x<-cbind(PP1,PP2,PP3,PP4,PP5,PP6,PP7,PP8,PP9,PP10,PP11,PP12,PP13,PP14,PP15,PP16,PP17,PP18,PP
y<-x[,22]
x<-x[,-22]
library(lars)
cv.lars(x,y,K=10,fraction=seq(0,1,0.02),type="lasso") #用10折交叉验证画cv误差
图
reg<-function(i,t) {                #提取最小二乘系数
y<-x[,c(1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,i+21)]
lm.reg<-lm(t~PP1+PP2+PP3+PP4+PP5+PP6+PP7+PP8+PP9+PP10+PP11+PP12+PP13+PP14+PP15+PP16+PP17+PP18
summary(lm.reg)
coef<-lm.reg$coefficients
sum(abs(coef))
}
coefsum[1]<-reg(1,O1) #coefsum[i]中存放的是Oi的最小二乘回归系数的绝对值之和
coefsum[2]<-reg(2,O2)
coefsum[3]<-reg(3,O3)
coefsum[4]<-reg(4,O4)
coefsum[5]<-reg(5,O5)
coefsum[6]<-reg(6,O6)
coefsum[7]<-reg(7,O7)
coefsum[8]<-reg(8,O8)
coefsum[9]<-reg(9,O9)
coefsum[10]<-reg(10,O10)
coefsum[11]<-reg(11,O11)
coefsum[12]<-reg(12,O12)
```

```
coefsum[13]<-reg(13,O13)
coefsum[14]<-reg(14,O14)
coefsum[15]<-reg(15,O15)
coefsum[16]<-reg(16,O16)
coefsum[17]<-reg(17,O17)
coefsum[18]<-reg(18,O18)

my.lasso<-function(y,x,k)      #lasso回归函数
{       xm <- apply(x,2,mean)
        ym <- mean(y)
        y <- y - ym
        x <- t( t(x) - xm ) #中心化
        ss <- function (b)
        {
               t( y - x %*% b ) %*% ( y - x %*% b ) + k * sum(abs(b))
        }
        b <- nlm(ss, rep(0,dim(x)[2]))$estimate
        coef<-c(ym,b)
        coef<-matrix(coef,22,1)
        for (i in 1:22)  {
          if (coef[i,]<=1e-03) coef[i]<-0 #若系数小于1e-3，则取为0
        }
        coef
}

x<-cbind(PP1,PP2,PP3,PP4,PP5,PP6,PP7,PP8,PP9,PP10,PP11,PP12,PP13,PP14,PP15,PP16,PP17,PP18,PP
y<-x[,22]
x<-x[,-22]
coef.o1<-my.lasso(y,x,0.6*coefsum[1])
coef.o1
x<-cbind(PP1,PP2,PP3,PP4,PP5,PP6,PP7,PP8,PP9,PP10,PP11,PP12,PP13,PP14,PP15,PP16,PP17,PP18,PP
sse.o1<-0
for (i in 1:86 ) {    #算残差平方和
red<-x[i,22]-t(matrix(c(1,x[i,-22]),22,1))%*%coef.o1[,1]
sse.o1<-sse.o1+red^2
}
sse.o1
```

```
o1<-lars(x,y)
plot(o1)      #回归系数收缩图
```