《系统工程导论》第四次作业
黑箱建模2

2015080072 自53 韩载贤

1.

病态线性回归问题中，显著性检验是需要的。某些特征根值特别小，所以它的倒数特别大。因此，显著性检验在自变量降维去线性之后是更可靠。

2.

一般线性回归分析：

```
ans =

    "< Linear Regression > "

ans =

    "A linear relationship exists.
     Regression equation ： y^=-9.987-0.082x1+0.586x2-0.008x3+0.333x4
     Confidence interval = （y0-0.979，y0+0.979） "
```

病态线性回归分析：

```
ans =

    "<ill-conditoned Linear regression>."

ans =

    "ill-conditoned regression equation:y=-9.151+0.073x1+0.599x2+0.002x3+0.105x4"
```

通过 F 显著性检验：

$F = 125.4320 > F_\alpha = 4.5337$，

```
ans =

    "A linear relationship exists."

ans =

    "Confidence interval ：（y0-1.248，y0+1.248） "
```

[MATLAB CODE]

-main.m

```
data = [149.3   4.2 80.3    108.1   15.9
161.2   4.1 72.9    114.8   16.4
171.5   3.1 45.6    123.2   19.0
175.5   3.1 50.2    126.9   19.1
180.8   1.1 68.8    132.0   18.88
190.7   2.2 88.5    137.7   20.4
202.1   2.1 87.0    146.0   22.7
212.4   5.6 96.9    154.1   26.5
226.1   5.0 84.9    162.3   28.1
231.9   5.1 60.7    164.3   27.6
239.0   0.7 70.4    167.6   26.3]
alpha = 0.05;

% Enter data into function
LinearRegression(data, alpha);
```

-LinearRegression.m

```
% set function linear_regression
function LinearRegression(data, alpha)
% function if it is regular linear regression, non ill-conditoned.
   sprintf("< Linear Regression >\n")
% extract data
Y = data( : , 5);
X = data( : , 1 : 4);
[N, n] = size(X);
X1 = X( : , 1);
X2 = X( : , 2);
X3 = X( : , 3);
X4 = X( : , 4);


X1a = mean(X1);
X2a = mean(X2);
X3a = mean(X3);
X4a = mean(X4);
Ya = mean(Y);

% sum of data
X1s = X1a * N;
X2s = X2a * N;
X3s = X3a * N;
X4s = X4a * N;
Ys = Ya * N;

% square
X11 = X1' * X1;
X22 = X2' * X2;
X33 = X3' * X3;
X44 = X4' * X4;

X12 = X1' * X2;
X13 = X1' * X3;
X14 = X1' * X4;
```

```
X23 = X2' * X3;
X24 = X2' * X4;
X34 = X3' * X4;

X1Y = X1' * Y;
X2Y = X2' * Y;
X3Y = X3' * Y;
X4Y = X4' * Y;

% lxx, lxy
L11 = X11 - X1s * X1s / N;
L22 = X22 - X2s * X2s / N;
L33 = X33 - X3s * X3s / N;
L44 = X44 - X4s * X4s / N;

L12 = X12 - X1s * X2s / N;
L13 = X13 - X1s * X3s / N;
L14 = X14 - X1s * X4s / N;
L23 = X23 - X2s * X3s / N;
L24 = X24 - X2s * X4s / N;
L34 = X34 - X3s * X4s / N;

L1 = X1Y - X1s * Ys / N;
L2 = X2Y - X2s * Ys / N;
L3 = X3Y - X3s * Ys / N;
L4 = X4Y - X4s * Ys / N;

% matrix A, B
A = [ N 0 0 0 0
   0 L11 L12 L13 L14
   0 L12 L22 L23 L24
   0 L13 L23 L33 L34
   0 L14 L24 L34 L44 ];
B = [ Ys ; L1 ; L2 ; L3 ; L4 ];

% regression coeffiConfidence intervalent matrix => A^-1*B
Beta = A \ B;
% regression equation
Y_ = Beta(1) + Beta(2) * (X1 - X1a) + Beta(3) * (X2 - X2a) + Beta(4) * (X3
- X3a) + Beta(5) * (X4 - X4a);
b0 = Beta(1) - Beta(2) * X1a - Beta(3) * X2a - Beta(4) * X3a - Beta(5) *
X4a;

% TSS = ESS + RSS
ESS = (Y_ - Ya)' * (Y_ - Ya);
RSS = (Y - Y_)' * (Y - Y_);

% F-test, df=(n, N-n-1)
F = (N - n - 1) * ESS / (n * RSS);
Fa = finv(1 - alpha, n, N - n - 1);

% if F > Fa
% null hypothesis is not true, there be linear relationship
if F > Fa
   Sdelta = sqrt(RSS / (N - n - 1));
   Zalpha = norminv(1 - alpha / 2);
   SZ = Sdelta*Zalpha;

   sprintf("A linear relationship exists.\nRegression equation :
y^=%.3f%.3fx1+%.3fx2%.3fx3+%.3fx4\nConfidence interval
```

```
=£¨y0-%.3f£¬y0+%.3f£©", ...
       b0, Beta(2), Beta(3), Beta(4), Beta(5), SZ, SZ)
else
   sprintf("Non linear relationship.")
end
%ill-conditoned Linear regression
sprintf("<ill-conditoned Linear regression>.\n")

% Standarizing variables
Yb = (Y - mean(Y)) / std(Y);
Xb1 = (X1 - mean(X1)) / std(X1);
Xb2 = (X2 - mean(X2)) / std(X2);
Xb3 = (X3 - mean(X3)) / std(X3);
Xb4 = (X4 - mean(X4)) / std(X4);
Xb = [Xb1 Xb2 Xb3 Xb4];

Xb=Xb';
Yb=Yb';
% get matrix Q and lamda : X*X'=Q*lamda*Q'
[Q , lamda] = eig(Xb * Xb');
lamda = abs(sum(lamda));
% descending order
[lamda , i] = sort(lamda, 'descend');
Q = Q( : , i);

% find the minimum m
lamdas = 0;
for j = 1 : n
   lamdas = lamdas + lamda(j);
end

m = n - 1;
r = lamdas - lamda(m + 1);

% ratio>=0.9, min r = ratio * lamdas, get min m = 3
while r > 0.9 * lamdas
   m = m - 1;
   r = r - lamda(m + 1);
end


m = m + 1;
Qm = Q( : , 1 : m);
Z = Qm' * Xb;
d = (Z * Z') \ Z * Yb';
c = Qm * d;

c0 = mean(Y);
for j = 1 : n
   c0 = c0 - c(j) * mean(X( : , j)) * std(Y) / std(X( : , j));
end
c1 = c(1) * std(Y) / std(X( : , 1));
c2 = c(2) * std(Y) / std(X( : , 2));
c3 = c(3) * std(Y) / std(X( : , 3));
c4 = c(4) * std(Y) / std(X( : , 4));

% ill-conditoned regression equation
Y_ = c0 + c1 * X1 + c2 * X2 + c3 * X3 + c4 * X4;

sprintf("ill-conditoned regression
equation:y=%.3f+%.3fx1+%.3fx2+%.3fx3+%.3fx4", ...
    c0, c1, c2, c3, c4)

% TSS = ESS + RSS
ESS = (Y_ - mean(Y))' * (Y_ - mean(Y));
RSS = (Y - Y_)' * (Y - Y_);

% F-test, df=(n, N-n-1)
F = (N - n - 1) * ESS / (n * RSS);
Fa = finv(1 - alpha, n, N - n - 1);

% if F > Fa
% null hypothesis is not true, there be linear relationship
if F > Fa
    Sdelta = sqrt(RSS / (N - n - 1));
    Zalpha = norminv(1 - alpha / 2);
    SZ = Sdelta*Zalpha;
    sprintf("A linear relationship exists.\n")
    sprintf("Confidence interval :£¨y0-%.3f£¬y0+%.3f£©", SZ, SZ)
else
    sprintf("Non linear relationship.")
end

end
```