OLS

AUTHOR
ZhongKai Han, WenTing Huang

前置知识

行列式计算

对角线法

代数余子式法

等价转化法

逆序数法

矩阵转置

$$x = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}, x' = \begin{pmatrix} 1 & 4 & 7 \\ 2 & 5 & 8 \\ 3 & 6 & 9 \end{pmatrix}$$

```r
x <- matrix(c(1,2,3,4,5,6,7,8,9),nrow=3,byrow=T)
print(x)
```

```
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9
```

```r
Ax <- t(x)
print(Ax)
```

```
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
```

```r
MatrixTranspose <-function(Matrix){
  n_row = nrow(Matrix)
  n_col = ncol(Matrix)
  newMatrix = matrix(nrow=n_col,ncol=n_row) #copy
  for(i in 1:n_row){
    for(j in 1:n_col){
      newMatrix[j,i] = Matrix[i,j]
    }
  }
  return(newMatrix)
}
Ax2 <- MatrixTranspose(Ax)
print(Ax2)
```

```
     [,1] [,2] [,3]
[1,]    1    2    3
```

```
[2,]    4    5    6
[3,]    7    8    9
```

矩转求逆

初等变换法

对A求逆:A|E,将A通过初等变化为单位矩阵,对E做相同操作,当A变为单位矩阵时,变换后的E就是逆矩阵 ##### 伴随矩阵法

将矩阵A的每个元素A_{ij}替换为其余子式$A_{ij}(-1)^(i+j)$，并将行列号互换得到的矩阵，记为$A^*A^{-1}=\frac{1}{|A|}*A^*$

LU分解 略

```
#x <- matrix(c(1,2,3,4,5,6,7,8,9),nrow=3,byrow=T)
x <- matrix(c(1,2,3,4),nrow=2,byrow=T)
cat("行列式:",det(x),"\n")
```

```
行列式: -2
```

```
if(abs(det(x))<=0.1e-6){
  #stop("矩阵行列式为0,不可求行列式")
}
x <- solve(x)
print(x)
```

```
     [,1] [,2]
[1,] -2.0  1.0
[2,]  1.5 -0.5
```

多元回归分析

最小二乘法多元回归推导

$y=\beta_0+\beta_1x_1+\beta_2x_2+...+\beta_nx_n$ 由于需要求β是最接近线性方程的解,所以就是得到最小残差平方和,既为0. 残差就是实现值减估计值(通过估算的β构成的方程得出的结果)
y_i(真实采集到的数值) $-\hat{y}_i$(通过公式推导出的数值) x_1,x_2,x_n是n个自变量,多组数据可得:
$\hat{y}_i=\beta_0+\beta_1x_{1i}+\beta_2x_{2i}+...+\beta_nx_{ni}$ 但是由于β是我们最终基于自已计算出来的所以y也只是估算值,因此需要用$\hat{y}$表示. 现有我们有多个y和x(在现实中通过测量或采样得到的数据),我们就会得到n个这样的方程:
$\hat{y}_1=\beta_0+\beta_1x_{11}+\beta_2x_{21}+...+\beta_nx_{n1}$ $\hat{y}_2=\beta_0+\beta_1x_{12}+\beta_2x_{22}+...+\beta_nx_{n2}$ ...
$\hat{y}_n=\beta_0+\beta_1x_{1n}+\beta_2x_{2n}+...+\beta_nx_{nn}$ 将其向量化: 需要在β_0后面补一个x_0否则无法匹配,但是值为1,因此得到新公式: $\hat{y}_i=\beta_0x_{0i}+\beta_1x_{1i}+\beta_2x_{2i}+...+\beta_nx_{ni}$ 向量化:

$$\beta=\begin{pmatrix}\beta_0 & \beta_1 & \dots & \beta_n\end{pmatrix},\ X=\begin{pmatrix}1 & x_{11} & x_{21} & \dots & x_{n1}\\ 1 & x_{12} & x_{22} & \dots & x_{n2}\\ \dots \\ 1 & x_{1n} & x_{2n} & \dots & x_{nn}\end{pmatrix}$$

$\hat{Y} = \beta X^T$ $\hat{Y} = X\beta^T$ or

$$X = \begin{pmatrix} 1 & x_{11} & x_{21} & \dots & x_{n1} \\ 1 & x_{12} & x_{22} & \dots & x_{n2} \\ \dots \\ 1 & x_{1n} & x_{2n} & \dots & x_{nn} \end{pmatrix},\ \beta = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \dots \\ \beta_n \end{pmatrix}$$

$\hat{Y} = X\hat{\beta}$ 由于我们想要$\hat{Y}$(估计值,既方程代入x算出的值)与Y(真实样本数据)最接近,甚至一样,因此有: $Y - \hat{Y} = 0$ 我们想要的是每个$\hat{Y}$与Y的偏离最小,所以总体函数需要求方差最小,既: $\underset{\beta}{min}(\sum_{i=1}^{n}(Y-\hat{Y})^2) = min(\sum_{i=1}^{n}(y_i - \hat{y}_i)^2)$ 由此可以推倒出: $\underset{\beta_0,\beta_1,\beta_2,\dots\beta_n}{min} \sum_{i=1}^{n}(y_i - \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + \dots + \beta_n x_{ni})^2$ 求偏导为0:既: *求一阶导函数推导需要看看 $\frac{\partial Q}{\partial \hat{\beta}_i} = 0$

$$\begin{cases} \frac{\partial Q}{\partial \hat{\beta}_0} = \sum 2(y_i - \beta_0 - \beta_1 x_{1i} - \beta_2 x_{2i} - \dots - \beta_n x_{ni})(-1(\text{因为减}\beta_0)) = -2\sum e_i = 0 \\ \frac{\partial Q}{\partial \hat{\beta}_1} = \sum 2(y_i - \beta_0 - \beta_1 x_{1i} - \beta_2 x_{2i} + \dots + \beta_n x_{ni})(-x_{1i}) = -2\sum e_i x_{1i} = 0 \\ \dots \\ \frac{\partial Q}{\partial \hat{\beta}_n} = \sum 2(y_i - \beta_0 - \beta_1 x_{ni} - \beta_2 x_{ni} + \dots + \beta_n x_{ni})(-x_{ni}) = -2\sum e_i x_{ni} = 0 \end{cases}$$

上述n+1个方程称为正规方程,转为矩阵表示:

$$\begin{pmatrix} \sum e_i \\ \sum e_i x_{1i} \\ \dots \\ \sum e_i x_{1i} \end{pmatrix} = \begin{pmatrix} 1 & 1 & \dots & 1 \\ x_{11} & x_{12} & \dots & x_{1n} \\ \dots \\ x_{n1} & x_{n2} & \dots & x_{nn} \end{pmatrix} \begin{pmatrix} e_1 \\ e_2 \\ \dots \\ e_n \end{pmatrix} = X'e = 0$$

根据前面向量化得到的公式:$Y = X\hat{\beta} + e$ 两边同时左乘X': $X'Y = X'X\hat{\beta} + X'e$ $X'Y = X'X\hat{\beta}$ 两边同时左乘$(X'X)^{-1}$ $(X'X)^{-1}X'Y = (X'X)^{-1}X'X\hat{\beta} \to (X'X)^{-1}X'Y = \hat{\beta} \to$

$$X'X = \begin{pmatrix} 1 & 1 & \dots & 1 \\ x_{11} & x_{12} & \dots & x_{1n} \\ \dots \\ x_{n1} & x_{n2} & \dots & x_{nn} \end{pmatrix} \begin{pmatrix} 1 & x_{11} & x_{21} & \dots & x_{n1} \\ 1 & x_{12} & x_{22} & \dots & x_{n2} \\ \dots \\ 1 & x_{1n} & x_{2n} & \dots & x_{nn} \end{pmatrix} =$$

$$\begin{pmatrix} n & \sum x_{1i} & \sum x_{2i} & \dots & \sum x_{ni} \\ \sum x_{1i} & \sum(x_{1i}x_{1i}) & \sum(x_{1i}x_{2i}) & \dots & \sum(x_{1i}x_{ni}) \\ \dots \\ \sum x_{ni} & \sum(x_{ni}x_{1i}) & \sum(x_{ni}x_{2i}) & \dots & \sum(x_{ni}x_{ni}) \end{pmatrix}$$

再根据矩阵求逆方法求逆,

$$X'Y = \begin{pmatrix} 1 & 1 & \dots & 1 \\ x_{11} & x_{12} & \dots & x_{1n} \\ \dots \\ x_{n1} & x_{n2} & \dots & x_{nn} \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ \dots \\ y_n \end{pmatrix} = \begin{pmatrix} \sum y_i \\ \sum(x_{1_i} y_i) \\ \dots \\ \sum(x_{n_i} y_i) \end{pmatrix}$$

$$\hat{\beta} = \begin{pmatrix} \hat{\beta}_0 \\ \hat{\beta}_1 \\ \dots \\ \hat{\beta}_n \end{pmatrix} = \begin{pmatrix} n & \sum x_{1i} & \sum x_{2i} & \dots & \sum x_{ni} \\ \sum x_{1i} & \sum(x_{1i}x_{1i}) & \sum(x_{1i}x_{2i}) & \dots & \sum(x_{1i}x_{ni}) \\ \dots \\ \sum x_{ni} & \sum(x_{ni}x_{1i}) & \sum(x_{ni}x_{2i}) & \dots & \sum(x_{ni}x_{ni}) \end{pmatrix}^{-1} \begin{pmatrix} \sum y_i \\ \sum(x_{1_i}y_i) \\ \dots \\ \sum(x_{n_i}y_i) \end{pmatrix}$$

- $$\begin{pmatrix} y_1 \\ y_2 \\ .. \\ y_n \end{pmatrix} - \begin{pmatrix} 1 & x_{11} & x_{21} & \dots & x_{n1} \\ 1 & x_{12} & x_{22} & \dots & x_{n2} \\ \dots \\ 1 & x_{1n} & x_{2n} & \dots & x_{nn} \end{pmatrix} * \begin{pmatrix} \beta_0 \\ \beta_1 \\ \dots \\ \beta_n \end{pmatrix}$$

3.51 $Var(\hat{\beta}_j) = \frac{\sigma^2}{SST_j(1-R_j^2)}$ $SST_j = \sum_{i=1}^n (x_{ij} - \bar{x}_j)^2$

方差膨胀因子(variance inflation factor, VIF) $VIF_j = \frac{1}{1-R_j^2}$ $Var(\hat{\beta}_j) = \frac{\sigma^2}{SST_j} * VIF_j$

*习题 14 和习题 15 3.56 $\sigma^2 = (\sum_{i=1}^n \hat{\mu}_i^2)/(n-k-1) = SSR/(n-k-1)$

自由度 degrees of freedom, df $df = n - (k+1) =$ 观测次数 − 估计参数的个数

回归标准误 standard error the regression, SER 误差项之标准差的估计量 也被称为估计值的标准误或均方根误 $SER = \sqrt{\sigma^2}$

标准差 standard deviation $\hat{\beta}_j$的标准差 $sd(\hat{\beta}_j) = \sigma/[SST_j(1-R_j^2)]^{\frac{1}{2}}$ 标准误 standard error of $\hat{\beta}_j$ because σ is unknown, so we use $\hat{\sigma}$ $sd(\hat{\beta}_j) = \hat{\sigma}/[SST_j(1-R_j^2)]^{\frac{1}{2}}$ $sd(\hat{\beta}_j) = \frac{\hat{\sigma}}{\sqrt{SST_j(1-R_j^2)}}$

最小二乘法多元实现 ordinary least squares

```
library(memisc)
```

```
Loading required package: lattice

Loading required package: MASS


Attaching package: 'memisc'

The following objects are masked from 'package:stats':

    contr.sum, contr.treatment, contrasts

The following object is masked from 'package:base':

    as.array
```

```
data0 = as.data.set(spss.system.file("254359000_36_36.sav"))
```

```
File character set is 'UTF-8'.

Converting character set to UTF-8.
```

```
data = as.data.frame(data0)
options(digits=10)
```

```
options(scipen=999)
data$index <- as.numeric(as.character(data$index))
data$gender <- as.numeric(data$gender)-1
data$age <- as.numeric(as.character(data$age))
data$totalseconds <- as.numeric(as.character(data$totalseconds))
#function
OLS_getBeta=function(y,...){
#y 为一个列向量
#x 为多个自变量
# x11 x12 x13 x14...
# xn1 xn2 xn3 xn4
  ym = matrix(y,ncol=1)
  xdata = c(rep(1,nrow(ym)))
  for (arg in list(...)) {
    xdata = c(xdata,arg)
  }
  #print(xdata)
  cat("y:",nrow(ym),length(list(...)),"\n")
  xm_t = matrix(xdata,nrow=length(list(...))+1,ncol = nrow(ym),byrow=T)
  xm = t(xm_t)
  #print(ym)
  #print(xm)
  #\hat{\beta} =(X'X)^{-1}X'Y
  #求(x'x)^-1
  xmsum <- xm_t %*% xm
  #print(xmsum)
  if(det(xmsum)<=0.1e-6){
    stop("矩阵行列式为0,不可求行列式")
  }
  xmsum_solve = solve(xmsum)
  #print(xmsum_solve)
  #求x'y
  x_tysum <- xm_t %*% ym
  #print(x_tysum)
  beta = xmsum_solve %*% x_tysum
  for(i in 1:nrow(beta)){
    cat(paste("beta",i-1,sep=""),beta[i,1]," ")
  }
  cat("\n")
  return(beta)
  #print(beta)
}
OLS_getYhat<-function(beta,xdata){
  y_hat_list = c()#array(NA, dim =c(1,length(y)))
  for(i in 1:ncol(xdata)){
    y_hat = beta[1,1]
    #cat(i,beta[1,1]," ")
    for(j in 1:nrow(xdata)){
      y_hat = y_hat+beta[j+1,1]*xdata[j,i]
      #cat(beta[j+1,1],xdata[j,i],beta[j+1,1]*xdata[j,i]," ")
    }
    y_hat_list = c(y_hat_list,y_hat)
    #cat(y_hat,y[i],"\n")
  }
  return(y_hat_list)
}
```

```
OLS <- function(y,...){
  k = length(list(...))
  N = length(y)
  df = N-k-1
  xdata = array( NA , dim=c(k,N) )
  i=1
  for (arg in list(...)) {
    xdata[i,] <- arg
    i=i+1
  }
  print(xdata)
  beta <- OLS_getBeta(y,...)
  y_hat = OLS_getYhat(beta,xdata)
  var_y = 0
  var_y_sum = 0
  for(i in 1:length(y_hat)){
    var_y = y[i]-y_hat[i]
    var_y_sum = var_y_sum+var_y^2
    #cat(i,var_y,var_y^2," \n")
  }
  cat("Residual standard error",sqrt(var_y_sum/df),"\n")
  #print(y_hat)
  #print(y)
}


OLS(data$pressure,data$depression,data$anxiety)
```

```
            [,1]        [,2]        [,3]        [,4] [,5]        [,6]
[1,] 1.857142857 1.285714286 2.857142857 1.000000000    2 3.428571429
[2,] 1.714285714 1.000000000 3.142857143 2.714285714    2 3.857142857
            [,7]        [,8] [,9] [,10]       [,11]       [,12]       [,13]
[1,] 3.142857143 2.285714286    1     3 2.428571429 3.000000000 1.285714286
[2,] 3.714285714 2.571428571    1     3 3.000000000 2.714285714 1.000000000
           [,14]       [,15]       [,16]       [,17] [,18]       [,19]
[1,] 1.857142857 2.142857143 1.714285714 2.142857143     1 2.285714286
[2,] 2.714285714 2.142857143 2.857142857 3.000000000     1 2.857142857
           [,20]       [,21]       [,22]       [,23]       [,24]       [,25]
[1,] 3.142857143 3.857142857 3.714285714 3.428571429 2.714285714 1.000000000
[2,] 3.428571429 3.000000000 3.285714286 2.714285714 2.285714286 1.714285714
           [,26]       [,27] [,28]       [,29]       [,30]       [,31] [,32]
[1,] 3.285714286 2.714285714     3 2.714285714 3.142857143 2.142857143     1
[2,] 2.857142857 3.285714286     3 2.428571429 3.142857143 2.857142857     1
           [,33] [,34]       [,35]       [,36]
[1,] 2.571428571     1 1.285714286 2.285714286
[2,] 3.285714286     2 1.142857143 1.714285714
y: 36 2
beta0 0.001536354617  beta1 0.3884424009  beta2 0.6661934111
Residual standard error 0.4330252557
```

```
##--
model<-lm(pressure~depression+anxiety,data=data)
summary(model)
```

```
Call:
```

```
lm(formula = pressure ~ depression + anxiety, data = data)

Residuals:
        Min          1Q      Median          3Q         Max 
-0.67337265 -0.20885061 -0.07938469  0.17758980  1.39712088 

Coefficients:
               Estimate  Std. Error t value    Pr(>|t|)    
(Intercept) 0.001536355 0.231883966 0.00663   0.9947535    
depression  0.388442401 0.133856235 2.90194   0.0065594 ** 
anxiety     0.666193411 0.139702300 4.76866 0.000036414 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.4330253 on 33 degrees of freedom
Multiple R-squared:  0.802297,  Adjusted R-squared:  0.790315 
F-statistic:  66.9585 on 2 and 33 DF,  p-value: 0.000000000002422258
```

```
#print( resid(model) )
print( sum(resid(model)^2)/33)
```

```
[1] 0.1875108721
```

```
#anova(model)
#summary(model)$r.squared
```

异方差性:

随机误差项(无法解释的误差部分)具有不同的方差(每个数据点与整体均值的偏差的平方的平均值) 将自变量分组分段后,随机误差项方差不同,则存在异方差性

OLS的无偏性 SLR.1 线性于参数 SLR.2 随机抽样 SLR.3 解释变量的样本有波动 样本标准差为零，则不成立 SLR.4 零条件均值 给定解释变量(自变量)的任何值,误差的期望值为零 $E(\mu|x) = 0$ SLR.5 同方差性 $Var(\mu|x) = \sigma^2(方差)$

高斯-马儿科夫假定

MLR.1 线性于参数 MLR.2 随机抽样 MLR.3 不存在完全共线性 MLR.4 零条件均值 给定解释变量(自变量)的任何值,误差的期望值为零 $E(u|x_1, x_2, \ldots, x_k) = 0$ MLR.5 同方差性 给定解释变量的任何值,误差都具有相同的方差

经典线性模型 Classical linear model

CLM 假定 classical linear model (CLM) assumptions 高斯-马儿科夫假定 MLR.1~5 MLR.6 正态性 总体中不可观测的误差是正态分布的 正态性假定

有序分类回归

因变量Y是分类变量

$$\begin{cases} Excellent \\ Good \\ Average \\ Fair \\ Poor \end{cases} \Leftarrow Y^* = \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_k x_k + \varepsilon$$

潜变量 Latent variable 无法直接观测到的一个变量,(与因变量Y之间存在联系的变量) Y*
$Y^* = \beta_1 x_1 + \beta_2 x_2 + ... + \beta_k x_k + \varepsilon$ $Y^* <= c_1$ Y=1(第一组) $Y^* <= c_2$ Y=2(第二组) ... $c_{n-1} < Y^*$ Y=n(第n组) c称为临界值

有序逻辑回归

当因变量Y是有充分类变量时,可以使用 因为有排序,所以会一级一级的比较 没有常数项, 有临界值(也可以理解为就是常数项) 其它有序分类回归: Ordered Logit Regression, Ordered Probit Regression 普通分类回归 multinomial logit regression 公式推导:https://www.bilibili.com/video/BV1e14y147cq/?p=17&spm_id_from=pageDriver

比例优势假设 Proportional odds assumption

因变量取到不同分类或者不同选项时，自变量x对应的斜率都是相同的，自变量与因变量的关系不受组别的影响。自变量x每增加一个单位，对因变量成为下一个临近类别的影响程度是相同的。

平行性检验

最大似然估计法

卡方检验

似然比检验

比较两个或多个统计模型的统计检验方法。它基于似然函数的最大化原理，通过比较模型拟合数据的好坏来判断是否存在显著的差异，从而确定哪个模型更适合描述数据。其中一个模型通常是另一个模型的简化版本。
分类变量 Categorical data 男,女,评价级别,

有序分类变量

可以按一定次序排列, 好,非常好