STAT R note

Wenxiao Yang*

*Department of Mathematics, University of Illinois at Urbana-Champaign

2021

目录

1 Basic

1.1 q-value of χ^2_n

```r
qchisq(0.95, n)
```

1.2 读取数据 txt (galton)

```r
galton <- read.table("Galton.txt", header=TRUE)
```

1.3 读取数据 csv (bikeshares)

```r
bikeshares <-  read.csv("BikeShares.csv", header=TRUE)
#分隔符为";"
whitewines.data<-read.csv("whitewines.csv",sep=";",header = TRUE)
```

1.4 查看数据维度 (bikeshares)

```r
dim(bikeshares)
## [1] 17414    10
```

1.5 数据中删除列 (bikeshares)

```r
# We remove columns 1,  7, 8, 9, 10:
bikeshares.reg = bikeshares[,c(-1,-7,-8,-9,-10)] #-i 即删除i列
head(bikeshares.reg)
##    cnt  t1  t2   hum wind_speed
## 1  182 3.0 2.0  93.0        6.0
## 2  138 3.0 2.5  93.0        5.0
## 3  134 2.5 2.5  96.5        0.0
## 4   72 2.0 2.0 100.0        0.0
## 5   47 2.0 0.0  93.0        6.5
## 6   46 2.0 2.0  93.0        4.0
```

1.6 数据“列”处理：赋值，条件选中 (galton)

```r
# Define the Adjusted Heigth Variable (according to Galton)
galton$AH <- galton$Height
galton$AH[galton$Gender=="F"]<-galton$Height[galton$Gender=="F"]*1.08
galton$MP <- (galton$Father + 1.08*galton$Mother)/2
head(galton)
##    Family Father Mother Gender Height Kids     AH    MP
## 1       1   78.5   67.0      M   73.2    4 73.200 75.43
## 2       1   78.5   67.0      F   69.2    4 74.736 75.43
## 3       1   78.5   67.0      F   69.0    4 74.520 75.43
## 4       1   78.5   67.0      F   69.0    4 74.520 75.43
## 5       2   75.5   66.5      M   73.5    4 73.500 73.66
## 6       2   75.5   66.5      M   72.5    4 72.500 73.66
```

1.7 查看数据类型 (bikeshare)

```r
class(numeric(n.iter))
## [1] "numeric"
class(bikeshares.reg)
## [1] "data.frame"
```

1.8 data.frame

1.8.1 修改列名

```r
names(myCR) = c("t1","hum");
```

1.8.2 data.frame 列拼接 cbind() (bikeshares)

```r
bikeshare.mlr1$fitted[1:5]
## 1          2          3          4          5
## 158.12967 152.85747  42.50091 -77.95731 126.47427
bikeshare.mlr1$residuals[1:5]
## 1          2          3          4          5
## 23.87033 -14.85747  91.49909 149.95731 -79.47427
cbind(bikeshare.mlr1$fitted[1:5], bikeshare.mlr1$residuals[1:5])
##           [,1]      [,2]
## 1 158.12967  23.87033
```

```
## 2 152.85747 -14.85747
## 3  42.50091  91.49909
## 4 -77.95731 149.95731
## 5 126.47427 -79.47427
```

1.8.3 data.frame 行拼接 rbind() (bikeshares)

```
rbind(bikeshare.mlr1$fitted[1:5], bikeshare.mlr1$residuals[1:5])
## 1         2         3         4         5
## [1,] 158.12967 152.85747 42.50091 -77.95731 126.47427
## [2,]  23.87033 -14.85747 91.49909 149.95731 -79.47427
```

1.8.4 data.frame 抽样

```
head(bikeshares.reg)
##      cnt t1  t2  hum   wind_speed
## 1    182 3.0 2.0 93.0  6.0
## 2    138 3.0 2.5 93.0  5.0
## 3    134 2.5 2.5 96.5  0.0
## 4    72  2.0 2.0 100.0 0.0
## 5    47  2.0 0.0 93.0  6.5
## 6    46  2.0 2.0 93.0  4.0
bikeshares.reg[sample(5), c(3,4)] #前五行（第3，4列）中随机抽样
##      t2  hum
## 2    2.5 93.0
## 5    0.0 93.0
## 3    2.5 96.5
## 1    2.0 93.0
## 4    2.0 100.0
```

1.9 集体求均值

```
apply(bikeshares.reg,2,mean)
## cnt        t1        t2        hum  wind_speed
## 1143.10164  12.46809  11.52084  72.32495  15.91306
```

1.10 numeric

1.10.1 numeric(k): 生成 k 个 0 的 numeric

```
numeric(5)
## [1] 0 0 0 0 0
class(numeric(5))
## [1] "numeric"
```

1.10.2 numeric 数值修改

```
A=numeric(5)
A[1]=2
A
## [1] 2 0 0 0 0
```

1.11 matrix

1.11.1 data.frame 转成 matrix

```
M=data.matrix(X)
```

1.11.2 修改列名

```
colnames(x)=c("t1", "t2", "hum")
```

1.11.3 去掉矩阵列/行的名字

```
rownames(A)<-NULL
colnames(A)<-NULL
```

1.11.4 自己创建 matrix

```
A=matrix(1:12,nrow=3,ncol=4)
A
##      [,1] [,2] [,3] [,4]
## [1,]    1    4    7   10
```

```
## [2,]    2    5    8   11
## [3,]    3    6    9   12
```

1.11.5 Transpose of matrix 转置矩阵

```
t(A)
##      [,1] [,2] [,3]
## [1,]    1    2    3
## [2,]    4    5    6
## [3,]    7    8    9
## [4,]   10   11   12
```

1.11.6 Multiplication of matrix 矩阵乘法

```
A%*%t(A)
##     [,1] [,2] [,3]
## [1,]  166  188  210
## [2,]  188  214  240
## [3,]  210  240  270
```

1.11.7 解 $Ax = b$: solve(A,b)

Solve $ax = b$

```
A.data=data.frame(a=c(1,43,765,9),b=c(2,455,787,2),
                  c=c(213,434,67,24),d=c(672,332,7,123))
A=data.matrix(A.data)
b=matrix(c(1,10,8,9))
solve(A,b)
##         [,1]
## a -0.6723499
## b  0.7380811
## c -0.9034118
## d  0.2866412
```

1.11.8 矩阵行列式: det()

```
det(A)
```

1.11.9 生成对角阵：diag(1,2,3,4)

```
diag(c(1,2,3,4))
##      [,1] [,2] [,3] [,4]
## [1,]    1    0    0    0
## [2,]    0    2    0    0
## [3,]    0    0    3    0
## [4,]    0    0    0    4
```

1.11.10 提取对角线上的元素：diag()

```
diag(A)
## [1]   1 455  67 123
```

1.11.11 特征值和特征向量：eigen()

```
eigen(A)
## eigen() decomposition
## $values
## [1]  962.54862 -533.15335  195.96895   20.63578
##
## $vectors
##             [,1]        [,2]       [,3]       [,4]
## [1,] -0.18050353 -0.31476395  0.7098847  0.5218457
## [2,] -0.65689212 -0.36245740 -0.6561850 -0.5428550
## [3,] -0.73165231  0.87683413  0.2141936  0.6319310
## [4,] -0.02441547 -0.02664961  0.1400217 -0.1834356
```

1.11.12 逆矩阵 solve(A)

```
solve(A)
##             [,1]          [,2]          [,3]        [,4]
## [1,]  0.015470466 -0.0038533021  0.0023771584 -0.07425607
## [2,] -0.016656510  0.0038972675 -0.0011449021  0.08054712
```

```
## [3,]  0.019498924 -0.0018420827  0.0012737127 -0.10163107
## [4,] -0.004665816  0.0005780095 -0.0004038514  0.03208420
```

1.11.13 列或行的函数处理 $apply(A, 1/2, func)$

```
apply(A,1,mean) #1表示对行求均值
apply(A,2,mean) #2表示对列求均值
apply(x,2,sd)
apply(x,2,var)
```

2 Simple Linear Regression

2.1 拟合 slr (galton)

```
# Simple Linear Regression
slr.fit <- lm(AH ~ MP, data=galton)
summary(slr.fit)
##
## Call:
## lm(formula = AH ~ MP, data = galton)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.4947 -1.4779  0.0995  1.5175  9.1262
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 18.76698    2.84062   6.607 6.74e-11 ***
## MP           0.72906    0.04102  17.772  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 01 ' ' 1
##
## Residual standard error: 2.233 on 896 degrees of freedom
## Multiple R-squared:  0.2606, Adjusted R-squared:  02598
## F-statistic: 315.9 on 1 and 896 DF,  p-value: < 2.2e-16
```

2.2 Summary 中提取 R-square (galton)

```
summary(slr.fit)$r.square
```

2.3 Summary 中提取 coefficients (galton)

```
galton.coef = summary(slr.fit)$coef
galton.coef
##                Estimate  Std. Error   t value     Pr(>|t|)
## (Intercept) 18.7669821 2.84062068  6.606648 6.735528e-11
## MP           0.7290562 0.04102226 17.772211 9.224505e-61

galton.coef[2,1]
galton.coef[2,3]  ## 提取t-test
```

2.4 回归中提取 degrees of freedom (galton)

```
slr.fit$df
## [1] 896
```

2.5 Hypothesis test

2.5.1 p-value of t-test (galton)

```
# pt(t-statistics, df)
# $H_0:\beta_1=0$, 由于检验0对称, 我们需要乘2
2*pt(-galton.coef[2,1]/galton.coef[2,2], 896)
## [1] 9.224505e-61
```

2.5.2 Critical value of $\alpha = 0.05$ in t(n)

```
qt(.05, n)
## -1.644941
```

2.5.3 ANOVA(F-test) (HW1)

```r
grade.anova=anova(slr.fit)
grade.anova
## Analysis of Variance Table
##
## Response: final
##              Df SumSq MeanSq F value Pr(>F)
## QuizAverage   1 69812  69812  423.19 < 2.2e-16 ***
## Residuals   380 62687    165
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

grade.anova[1,4]   ## 提取F-value from ANOVA Table
```

2.5.4 p-value of F-test (HW1)

```r
pf(grade.anova[1,4],df1=1,df2=380,lower.tail = FALSE)
# lower.tail: if TRUE (default), probabilities are P[X $\leq$ x],
# otherwise, P[X > x].
```

2.5.5 Critical value of $\alpha = 0.05$ in F(p,n)

```r
qf(.05, p, n, lower.tail = FALSE)
```

2.6 Confidence interval 置信区间 (HW1)

```r
confint(slr.fit, 'QuizAverage', level=0.9)
##                5 %      95 %
##QuizAverage 0.7880018 0.9253306
```

2.7 Prediction

2.7.1 模型带入数据 (galton)

```r
predict(slr.fit, newdata=data.frame(MP=70))
##        1
## 69.80092
```

2.7.2 Confidence interval (HW1) $\hat{\beta}_0 + \hat{\beta}_1 x^\star \pm T_{n-2}(\alpha/2)\hat{\sigma}\sqrt{\frac{1}{n} + \frac{(x^\star - \bar{x})^2}{S_{xx}}}$

```
predict(slr.fit,newdata = data.frame(QuizAverage=85),
interval = 'confidence', level=0.9)
##        fit      lwr      upr
## 1 76.7638 75.44682 78.08077
```

2.7.3 Prediction interval (HW1) $\hat{\beta}_0 + \hat{\beta}_1 x^\star \pm T_{n-2}(\alpha/2)\hat{\sigma}\sqrt{1 + \frac{1}{n} + \frac{(x^\star - \bar{x})^2}{S_{xx}}}$

```
predict(slr.fit,newdata = data.frame(QuizAverage=85),
interval = 'prediction', level=0.9)
##        fit      lwr      upr
## 1 76.7638 55.54486 97.98273
```

3 Multiple Linear Regression

3.1 拟合 mlr (bikeshares)

```
bikeshare.mlr1 = lm(cnt ~ t1 + t2 + hum + wind_speed,
                                   data=bikeshares.reg )
summary(bikeshare.mlr1)
##
## Call:
## lm(formula = cnt ~ t1 + t2 + hum + wind_speed,
##                                   data = bikeshares.reg)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1970.1  -602.7  -252.7   332.6  6007.4
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 2582.5618    64.7237  39.901  < 2e-16 ***
## t1            66.1963     9.4206   7.027 2.19e-12 ***
## t2           -18.2313     7.7565  -2.350 0.018762 *
## hum          -27.5645     0.5865 -46.999  < 2e-16 ***
## wind_speed    -3.8435     0.9899  -3.883 0.000104 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 936 on 17409 degrees of freedom
## Multiple R-squared:  0.2562, Adjusted R-squared:  0.256
## F-statistic:  1499 on 4 and 17409 DF,  p-value: < 2.2e-16
```

3.2 Update regression, add or delete predictor

```
rat.lm_body = update(rat.lm, ~ liver+dose)
rat.lm_body = lm(Y~liver+dose, data = rat)
# 两者等价
```

3.3 回归中提取 residuals, fitted values (bikeshare)

```
bikeshare.mlr1$res
bikeshare.mlr$residuals
bikeshare.mlr$fitted.values
```

3.4 Summary 中提取 F-test statitic

```
summary(bikeshare.mlr1)$fstat
## value     numdf     dendf
## 1499.07   4.00      17409.00
summary(bikeshare.mlr1)$fstat[1]
## 1499.07
```

3.4.1 得到 RSS: $\sum_{i=1}^{n} r_i^2$

```
sum(bikeshare.mlr1$res^2) #方法1
deviance(bikeshare.mlr1)  #方法2
```

3.5 Correlation matrix (bikeshares) *cor*()

```
cor(bikeshares.reg[,-1]) #这里[,-1]是不想算第一列
##                          t1            t2          hum    wind_speed
```

```
## t1          1.0000000  0.98834422 -0.4477810  0.14547097
## t2          0.9883442  1.00000000 -0.4034951  0.08840854
## hum        -0.4477810 -0.40349514  1.0000000 -0.28778917
## wind_speed  0.1454710  0.08840854 -0.2877892  1.00000000
```

```
round(cor(seatpos), dig=2)
# 打印出来的数据保留两位小数
```

3.6 Plot all pairs of variables

```
pairs(rat)
```

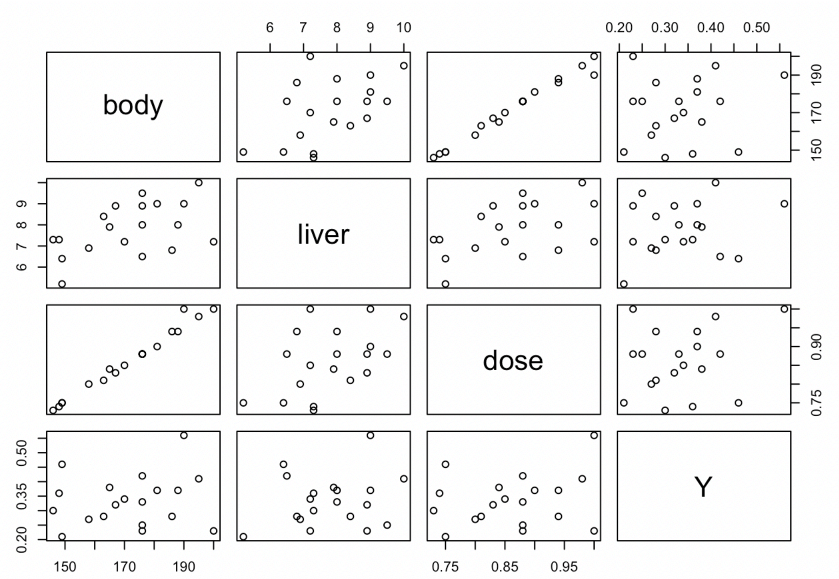

图 1:

3.7 Partial F–Tests (bikeshare)

```
bikeshare.mlr.full = lm(cnt ~ t1 + t2+ hum + wind_speed,
                        data=bikeshares.reg ) #先回归full model
bikeshare.mlr.reduced = lm(cnt ~ hum + wind_speed ,
                        data=bikeshares.reg ) #回归reduced model
anova(bikeshare.mlr.reduced, bikeshare.mlr.full)
                             #do the partial F-test by "anova(.)"
```

```
## Analysis of Variance Table
##
## Model 1: cnt ~ hum + wind_speed
## Model 2: cnt ~ t1 + t2 + hum + wind_speed
##   Res.Df        RSS Df Sum of Sq      F    Pr(>F)
## 1  17411 1.6103e+10
## 2  17409 1.5250e+10  2 853010396 486.88 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Sum of Square 853010396 是 $RSS_0 - RSS_\alpha = 1.6103e + 10 - 1.5250e + 10 = 853010396$
我们也可以按照公式算:

```
rss.full = sum(bikeshare.mlr.full$res^2)
# You can also compute it with
# deviance(bikeshare.mlr.full)
rss.reduced = sum(bikeshare.mlr.reduced$res^2)
# deviance(bikeshare.mlr.reduced)
Fstat = (rss.reduced - rss.full)/2/(rss.full/17409)
Fstat
## [1] 486.8763
1-pf(Fstat, 2, 17409)
## [1] 0
```

3.8 Permutation Tests (bikeshares)

$$\begin{cases} H_0 : bikeshares \sim humidity + windspeed \\ H_\alpha : bikeshares \sim RealTemp + FeelsLikeTemp + humidity + windspeed \end{cases}$$

If *RealTemp* and *FeelsLikeTemp* are insignificant (Under H_0), the F-statstic of regression model will not be affected by switching the orders of these two data. Then new F-statistic will be equal(or less) to the old. i.e. High new F-statistic is more extreme than H_0. So lower $p-$value will support H_α : *RealTemp* and *FeelsLikeTemp* are significant.

```
n.iter = 2000;
fstats = numeric(n.iter);
for(i in 1:n.iter){
  newbikes = bikeshares.reg;
  newbikes[, c(3,4)] = bikeshares.reg[sample(17414), c(3,4)];
```

```r
  ge = lm(cnt ~ t1 + t2+ hum + wind_speed, data=newbikes);
  fstats[i] = summary(ge)$fstat[1]
}

# Estimated p-value
length(fstats[fstats > summary(bikeshare.mlr.full)$fstat[1]])/n.iter
## [1] 0
```

3.9 Confidence/Prediction Interval

3.9.1 Estimators' Confidence Interval

```r
confint(bikeshare.mlr)
##                    2.5 %      97.5 %
## (Intercept) 2543.679114 2766.669772
## t1            41.516598   47.099480
## hum          -28.984942  -26.739780
## wind_speed    -4.941603   -1.262794
confint(bikeshare.mlr, 't1', level=0.99)
##        0.5 %   99.5 %
## t1  40.63932 47.97676
```

3.9.2 Estimators' Confidence regions

```r
library(ellipse)
library(ggplot2)
CR95 = ellipse(bikeshare.mlr, c(2,3))
CR99 = ellipse(bikeshare.mlr, c(2,3), level=0.99)
CR998 = ellipse(bikeshare.mlr, c(2,3), level=0.998)
# Plot Confidence Regions for column 2,3
dim(CR95)
## [1] 100   2
head(CR95)
##            t1       hum
## [1,] 47.25426 -26.67754
## [2,] 47.13012 -26.63239
## [3,] 46.99462 -26.59219
## [4,] 46.84830 -26.55710
```

```
## [5,] 46.69175 -26.52728
## [6,] 46.52561 -26.50282
```

```
myCR = rbind(CR95, CR99, CR998);
# 行连接
myCR = data.frame(myCR);
names(myCR) = c("t1","hum");
myCR[, 'level']=as.factor(c(rep(0.95, dim(CR95)[1]),
                            rep(0.99, dim(CR99)[1]),
                            rep(0.998, dim(CR998)[1])));
# 添加列 'level', 给各行根据精度赋值

ggplot(data=myCR, aes(x=t1, y=hum, colour=level)) +
  geom_path(aes(linetype=level), size=1.5) +
  geom_point(x=coef(bikeshare.mlr)[2], y=coef(bikeshare.mlr)[3]
  , shape=3, size=3, colour='red') +
  geom_point(x=0, y=0, shape=1, size=3, colour='red')
```

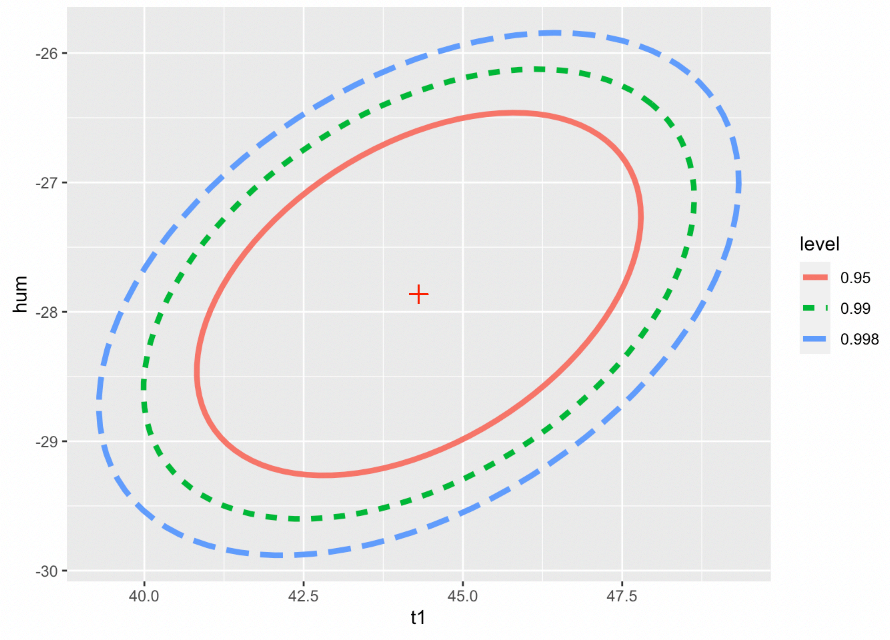

图 2:

3.9.3 Confidence Interval for new observation

```
x=data.frame(t(meanvalue))
predict.lm(bikeshare.mlr,x,interval="confidence",level=0.95)
##        fit      lwr      upr
## 1 1143.102 1129.198 1157.006
```

3.9.4 Prediction Interval for new observation

```
predict.lm(bikeshare.mlr,x,interval="prediction",level=0.95)
##        fit       lwr      upr
## 1 1143.102 -691.7461 2977.949
```

3.10 Unusual Observation

3.10.1 Leverage Points

```
lev=influence(bikeshare.mlr)$hat
# H matrix 的对角上的所有元素
newlev = lev[lev>2*p/n]
# 找出所有high leverage points
bikeshares.reg[lev > 2*p/n,]
# 筛选出bikeshares中high leverage points的项
```

3.10.2 Half-norm Plot

Designed to identify unusually large values and assess positive data.

Plot the data against the positive normal quantiles. Specifically,

1. Sort the data:

$$x_{[1]} \leq ... \leq x_{[n]}.$$

2. Compute the quantiles:

$$u_i = \Phi^{-1}(\frac{n+i}{2n+1})$$

3. Plot $x_{[i]}$ against u_i.

```
library(faraway)
halfnorm(newlev, 6, labs=as.character(1:length(newlev)),
 ylab="Leverages")
# 6是nlab, 即给几个点标注
```

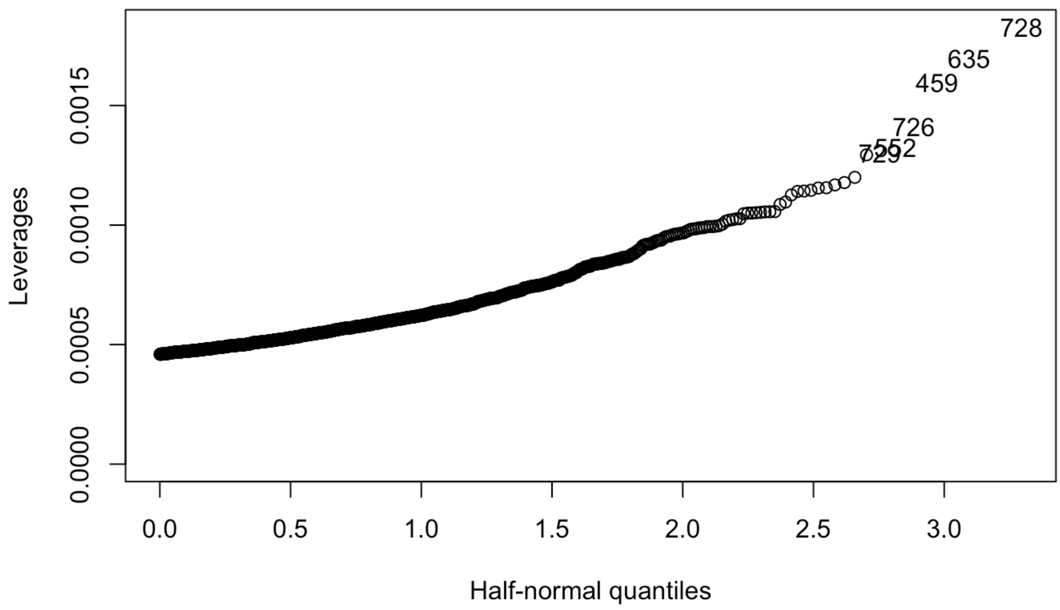

图 3:

3.10.3 Standardized Residuals, Studentized Residuals, *rstandard*(), *rstudent*()

```
rstandard(model)
rstudent(model)
```

3.10.4 Outliers

```
# Compute Studentized Residuals
jack=rstudent(bikeshare.mlr);
# The critical value WITH Bonferroni correction is
qt(.05/(2*n), n-p-1)
## [1] -4.681361
# The critical value WITHOUT Bonferroni correction is
qt(.05/2, n-p-1)
## [1] -1.9601
# Sort the residuals indescending order to find outliers (if any)
sort(abs(jack), decreasing=TRUE)[1:10]
##     4462     5130     5139     4471    15888     5140
## 6.408782 5.665958 5.499140 5.317999 4.807279 4.787554
##    15217    15385    16727    14905
```

```
## 4.746059 4.738005 4.661289 4.522918
```

As we can see here, we have 8 outliers, i.e. the values that are higher (in absolute value) of the critical T distribution value with Bonferroni correction ($|-4.681361|$). These are observations: #4462, #5130, #5139, #4471, #15888, #5140, #15217, #15385.

3.11 High influential points

```
# Compute Cook's Distance
cook = cooks.distance(bikeshare.mlr)
# Extract max Cook's Distance
max(cook)
## [1] 0.005641587
which.max(cook)
## 4471
# Prepare a Half Normal Plot of Cook's distances
halfnorm(cook, 6, labs=as.character(1:length(cook)),
 ylab="Cook's␣distances")
```

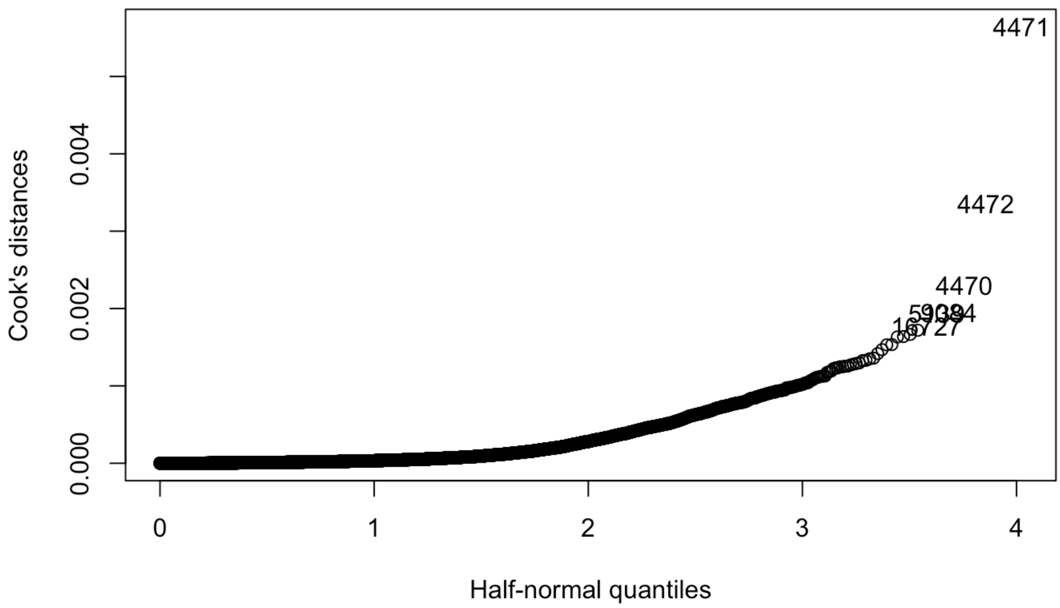

图 4:

3.12 Diagnostics

3.12.1 Checking Homoskedasticity Graph

```
plot(bikeshare.mlr, which=1)
```

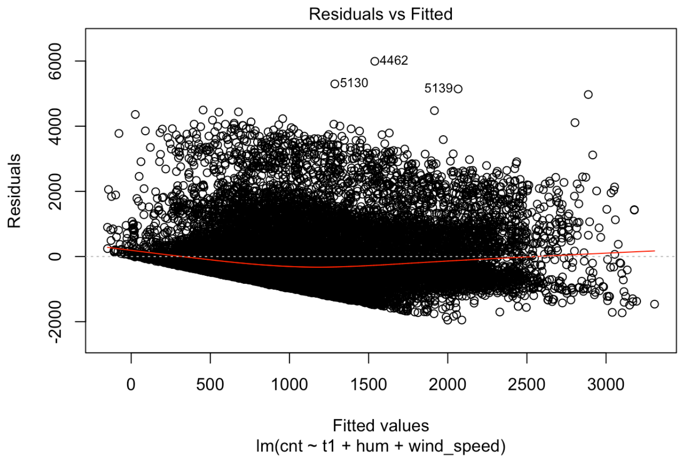

图 5:

Which is same as

```
plot(bikeshare.mlr$fitted.values, bikeshare.mlr$residuals)
```

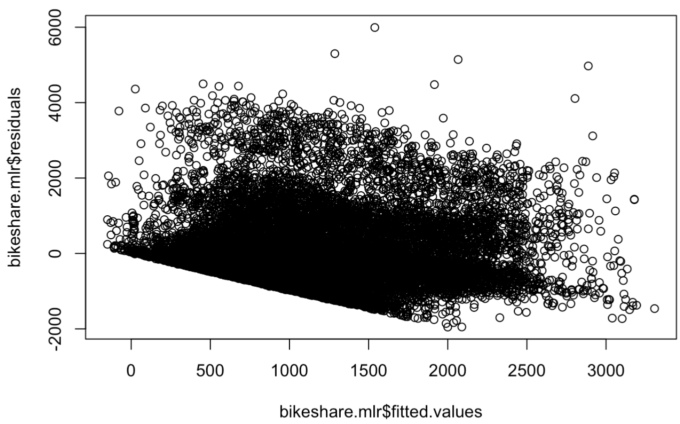

图 6:

3.12.2 Breusch-Pagan Test

```r
library(lmtest)
bptest(bikeshare.mlr)
##
## 	studentized Breusch-Pagan test
##
## data:  bikeshare.mlr
## BP = 133.29, df = 3, p-value < 2.2e-16
```

We can also perform the BP test by hand:

```r
tmp.fit = lm(bikeshare.mlr$res^2 ~ t1 +  hum + wind_speed,
  data=bikeshares.reg  )
summary(tmp.fit)$r.sq*dim(bikeshares.reg)[1]
```

3.12.3 Checking Normality Graph

QQ-Plot

```r
plot(bikeshare.mlr, which=2)
```

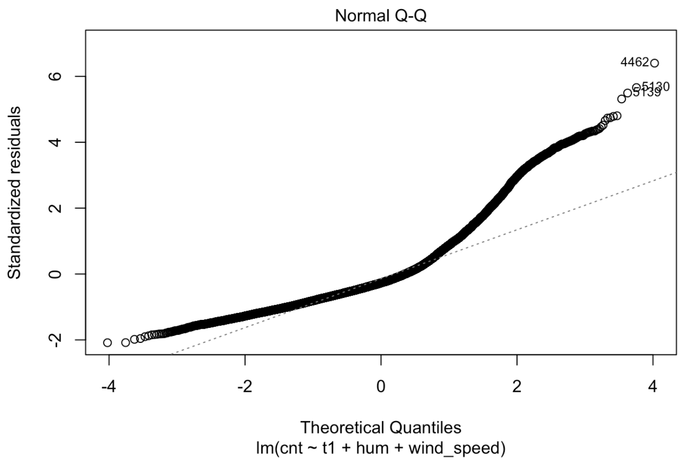

图 7:

Histogram

```r
hist(bikeshare.mlr$residuals)
```

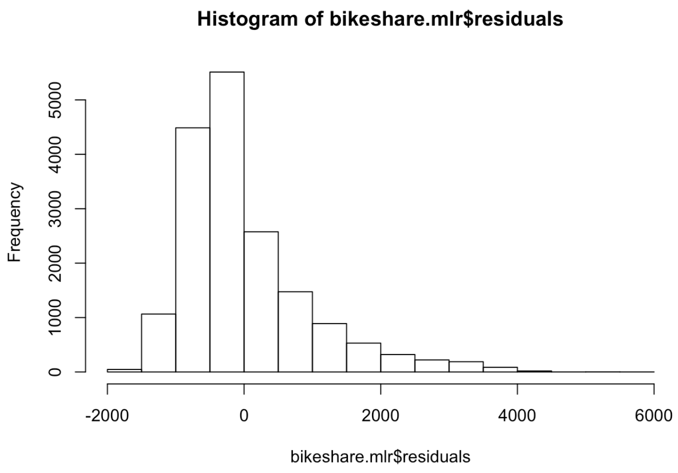

图 8:

3.12.4 Shapiro test

```r
shapiro.test(residuals(bikeshare.mlr))
```

3.12.5 Kolmogorov-Smirnov test

```r
ks.test(residuals(bikeshare.mlr), y=pnorm)
##
##  One-sample Kolmogorov-Smirnov test
##
## data:  residuals(bikeshare.mlr)
## D = 0.63627, p-value < 2.2e-16
## alternative hypothesis: two-sided
```

The p-value is low, which implies that the normality assumption is not satisfied either.

3.12.6 Checking Serial Dependence: Durbin Watson test

```r
library(lmtest)
dwtest(lm.sales)
```

```
##
##  Durbin-Watson test
##
## data:  lm.sales
## DW = 0.73473, p-value = 0.0001748
## alternative hypothesis: true autocorrelation is greater than 0
```

3.12.7 Checking the Linearity Assumption with Partial Regression Plots

Test $t1$

```
bikeshare.mlr = lm(cnt ~ hum + wind_speed, data=bikeshares.reg)
bikeshare.mlr.t1 = lm(t1 ~ hum + wind_speed, data=bikeshares.reg)
plot(bikeshare.mlr.t1$residuals, bikeshare.mlr$residuals)
```

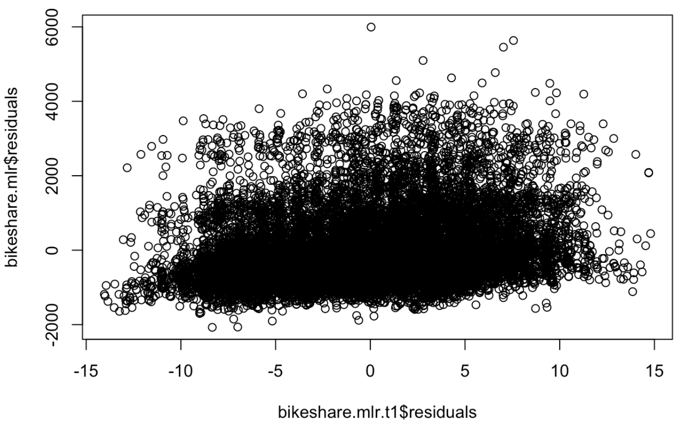

图 9:

3.12.8 Box Cox Transformations

First we need to make sure each $y_i > 0$:

```
min(bikeshares.reg$cnt) # this is the min value in the y's
## [1] 0
which(bikeshares.reg$cnt==0)
# this is the location of the min value
## [1] 2016
```

```
bikeshares.reg$cnt[2016]=0.01
# we replace the min with a small positive value
min(bikeshares.reg$cnt)
# we checke whether the 0 value was replaced
# by the small positive number
## [1] 0.01
```

Now, we are ready to apply the *boxcox* function:

```
bikes.transformation = boxcox(bikeshare.mlr,
lambda=seq(-2, 2, length=400))
```

which also same as

```
boxcox(bikeshare.mlr,plotit=T) # plotit=T is the default setting
```

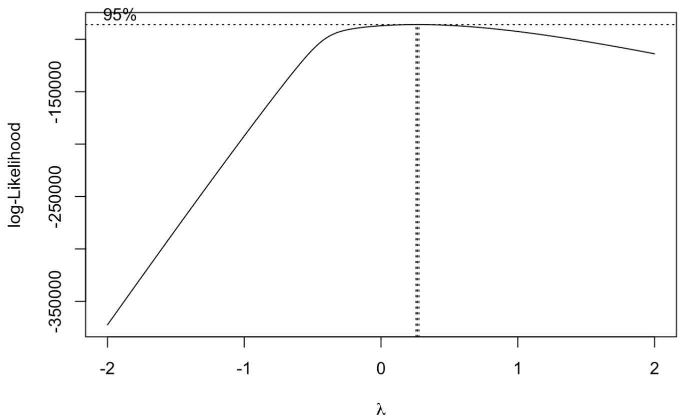

图 10:

改变范围

```
boxcox(bikeshare.mlr,plotit=T,lambda=seq(0.1,1,by=0.05)) # zoom-in
```

Find the λ that maximizes the $Log - likelihood$.

```
names(bikes.transformation)
```

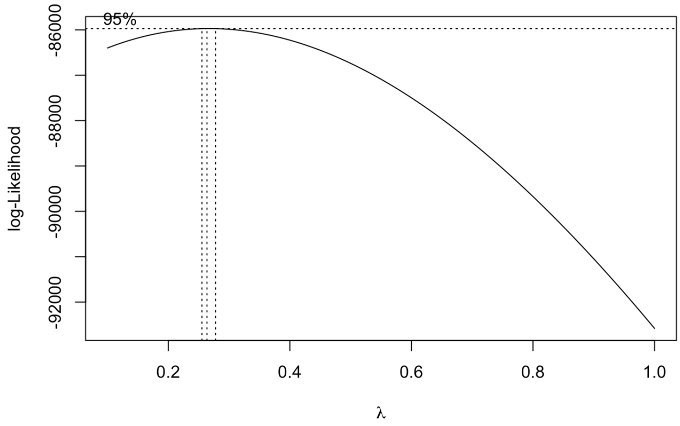

图 11:

```
## [1] "x" "y"
bikes.transformation$x[1:10]
##  [1] -2.000000 -1.989975 -1.979950 -1.969925 -1.959900 -1.949875 -1.939850
##  [8] -1.929825 -1.919799 -1.909774
bikes.transformation$y[1:10]
##  [1] -372422.1 -370582.3 -368742.9 -366904.0 -365065.6 -363227.6 -361390.0
##  [8] -359553.0 -357716.4 -355880.2
bikes.transformation$x[bikes.transformation$y ==
max(bikes.transformation$y)] # lambda.hat
## [1] 0.2656642
```

$$\hat{\lambda} = 0.2656642$$

Construct a Confidence Interval for λ as follows:

$$\{\lambda : L(\lambda) > L(\hat{\lambda}) - \frac{1}{2}\chi_1^2(1-\alpha)\}$$

```
tmp=bikes.transformation$x[bikes.transformation$y >
      max(bikes.transformation$y) - qchisq(0.95, 1)/2];
range(tmp) # 95% CI. Read Chapter 9 in the Faraway textbook for details.
## [1] 0.2556391 0.2756892
```

3.12.9 Summary of Diagnostic Plots

```
fit=lm(Y~., data=rat)
par(mfrow=c(2,2))
```

```
plot(fit)
```

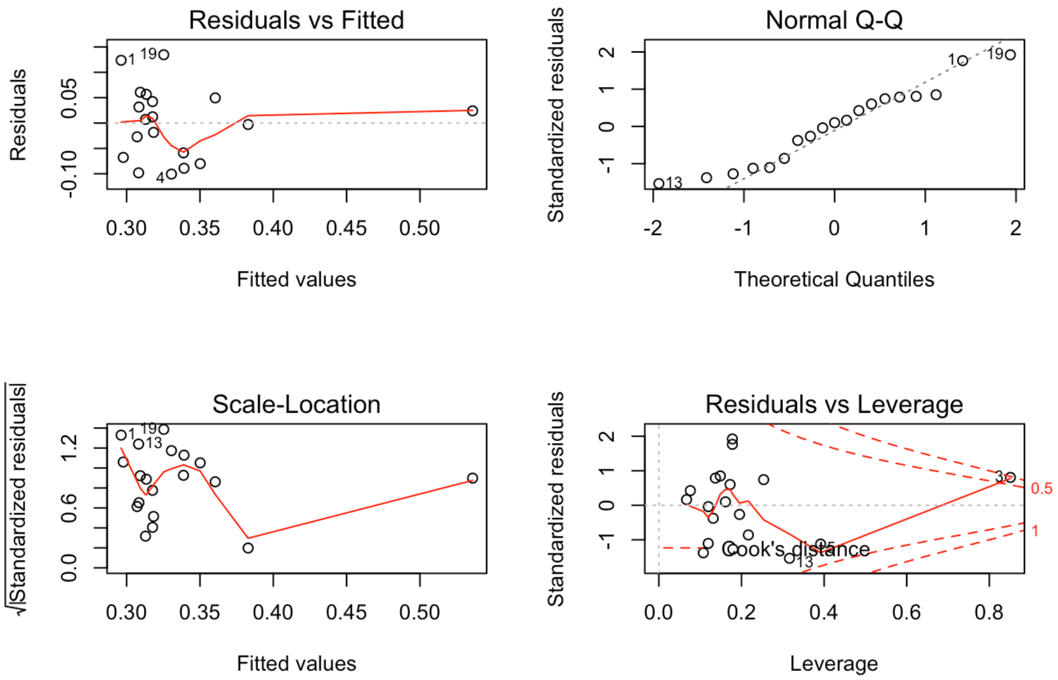

图 12:

3.13 Collinearity

```
library(faraway)
# 提取数据seatpos
data(seatpos)
attach(seatpos)

# Fit the FULL model
position.full=lm(hipcenter ~ ., seatpos)
x = model.matrix(position.full)[,-1]
# remove the column that corresponds to the intercept
```

3.13.1 Standardized each colum of X

```
x = model.matrix(position.full)[,-1] #去除第一列(即intercept)
x = x - matrix(apply(x,2, mean), 38,8, byrow=TRUE)
x = x / matrix(apply(x, 2, sd), 38,8, byrow=TRUE)
apply(x,2,mean)
##           Age        Weight       HtShoes            Ht        Seated
## -2.193512e-17  2.810252e-16  9.566280e-16  1.941574e-16 -1.073010e-15
##           Arm         Thigh           Leg
## -1.070022e-16  8.909895e-17 -9.114182e-17
apply(x,2,var)
##     Age  Weight HtShoes      Ht  Seated     Arm   Thigh     Leg
##       1       1       1       1       1       1       1       1
```

3.13.2 Condition number of the $X^T X$ matrix

```
e = eigen(t(x) %*% x) # compute the eigenvalues
sqrt(e$val[1]/e$val[8])
## [1] 59.7662
```

The condition number is 59.77, larger than 30, so we conclude that collinearity is present.

3.13.3 Variance Inflation Factor (VIF)

```
# Variance Inflation Factor (VIF)
round(vif(x), dig=2)
##     Age  Weight HtShoes      Ht  Seated     Arm   Thigh     Leg
##    2.00    3.65  307.43  333.14    8.95    4.50    2.76    6.69
```

```
sqrt(307.43)
## [1] 17.53368
```

Note that the se for the coef associated with *HtShoes* is 17.5 times larger than it would have been without collinearity.

3.13.4 Pairwise correlations and partial F-tests

```
cor(Seated+Thigh, Ht)
## [1] 0.9389819
cor(Seated+Leg, Ht)
```

```
## [1] 0.965607
cor(Seated+Arm, Ht)
## [1] 0.9465523
```

```
position.red1 = lm(hipcenter ~ Age + Weight + Ht + Seated, data=seatpos)
position.red2 = lm(hipcenter ~ Ht, data=seatpos)
anova(position.red2, position.red1)
## Analysis of Variance Table
##
## Model 1: hipcenter ~ Ht
## Model 2: hipcenter ~ Age + Weight + Ht + Seated
##   Res.Df   RSS Df Sum of Sq      F Pr(>F)
## 1     36 47616
## 2     33 44774  3    2841.6 0.6981 0.5599
```

Based on the $F-$test provided in the $ANOVA$ table, we conclude that the reduced model with Ht as the only variable is better than the model that includes $Age, Weight, Ht$ and $Seated$.

4 Time Series

4.1 First Order Autoregressive Model

```
library(nlme)
lm.sales.cor = gls(company_sales~industry_sales,
correlation = corAR1(form= ~ index), data=sales)

summary(lm.sales.cor)
## Generalized least squares fit by REML
##   Model: company_sales ~ industry_sales
##   Data: sales
##         AIC       BIC   logLik
##   -31.74311 -28.18162 19.87156
##
## Correlation Structure: AR(1)
##  Formula: ~index
##  Parameter estimate(s):
## Phi
##   1
##
```

```
## Coefficients:
##                        Value Std.Error   t-value p-value
## (Intercept)       -0.3189197 2041.6945 -0.00016  0.9999
## industry_sales     0.1684878    0.0051 33.06272  0.0000
##
##  Correlation:
##                (Intr)
## industry_sales 0
##
## Standardized residuals:
##           Min            Q1           Med            Q3
##           Max
## -9.036061e-05 -4.156415e-05 -3.013053e-06  8.080346e-05
##  1.091922e-04
##
## Residual standard error: 2041.694
## Degrees of freedom: 20 total; 18 residual
```

5 Polynomials Regression

5.1 Orthogonal Polynomials

```
poly(.)
```

5.2 B-Splines Basis

```
bs(x, df, knots, degree=3, intercept=FALSE)
# x是数据
# df是输出的design matrix的columns数, 和真正的df无关
# intercept=FALSE, df=真df-1
# intercept=TRUE, df=真df
# knots=k, 代表k是那个唯一的knot, 所以knot数是1, 无论k多大
new.knots= c(1/6, 3/6, 5/6)
bs(x, knots=new.knots, intercept=TRUE)
bs(x, knots=quantile(x, c(1/3,2/3)), intercept=TRUE)
```

5.3 Natural Cubic Splines

```
ns(x, df, knots, Boundary.knots, degree=3, intercept=FALSE)
# knots只表示interior knots，还有俩boundary knots。
# 所以 真df=#knots+2
# 其他一样
ns(x, knots=new.knots, Boundary.knots=c(0,1), intercept=TRUE)
```

6 Categorical ANOVA

6.1 Effect tests

When the levels of the categorical variable are in text (instead of number), R assigns 0 and 1 in alphabetical order: 0 first and 1 second.

```
quest.full=lm(rate~lot.size*color,quest.data)
anova(quest.full)
##Analysis of Variance Table
##Response: rate
##                Df Sum Sq  Mean Sq F value  Pr(>F)
##lot.size         1 43.226  43.226  7.1024 0.01765 *
##color            1 20.052  20.052  3.2947 0.08955 .
##lot.size:color   1  0.166   0.166  0.0272 0.87111
##Residuals       15 91.293   6.086
##---
##Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

第一行 intercept only vs. intercept+lot.size

第二行 intercept+lot.size vs. intercept+lot.size+color

第三行 intercept+lot.size+color vs. intercept+lot.size+color+lot.size*color

6.2 ANOVA Type III

This type tests for the presence of an effect given that both the other effects are in the model.

```
Anova(lm(1/time ~ treat*poison, data=newrats), type="III")
```

```
## Anova Table (Type III tests)
##
## Response: 1/time
##                Sum Sq Df F value    Pr(>F)
## (Intercept)   15.0605  1 66.5967 1.298e-09 ***
## treat          2.1340  3  3.1455   0.03723 *
## poison        11.7375  2 25.9514 1.225e-07 ***
## treat:poison   1.9800  6  1.4592   0.22073
## Residuals      7.9151 35
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

图 13:

7 Variation Selection

7.1 Leap and Bounds method

Use function *regsubsets* from library leaps to evaluate different scores for sub-sets of models up to size p (including the intercept).

```
library(leaps)
Hitters.leaps=regsubsets(Salary~.,data=data.reg,nvmax=16)
rs=summary(Hitters.leaps)
rs$adjr2
rs$which[which.max(rs$adjr2),]
rs$cp
rs$bic
n=dim(data.reg)[1]
m=2:17
Aic=n*log(rs$rss/n)+2*m
```

7.2 Searching algorithm based on AIC and BIC

Use function step from the *stats* library to apply searching algorithms based on the AIC (default) or BIC criteria (k = log(n)). The option direction=both uses the Stepwise searching algorithm. You can also use the options: $direction = forward$ and $direction = backward$.

```
step(full.model, direction="both")
step(full.model, direction="both", k=log(n))
We can also use direction=forward and direction=backward
```

8 Shrinkage Methods

8.1 PCR, PCA

Function prcomp can be used to calculate the PCs and extract the λ's squared-roots (sdev) and eigenvectors (rotation) of the variance-covariance matrix:

```
data(meatspec,package="faraway")
trainmeat<-meatspec[1:172,]
testmeat<-meatspec[173:215,]
mod1<-lm(fat~.,trainmeat)
meatpca<-prcomp(trainmeat[,-101])
round(meatpca$sdev,3)[1:50]
```

```
##  [1] 5.055 0.511 0.282 0.168 0.038 0.025 0.014 0.011 0.005 0.003 0.002 0.002
## [13] 0.001 0.001 0.001 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
## [25] 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
## [37] 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
## [49] 0.000 0.000
```

图 14:

The pcr function (principal component regression) from the pls package has useful features for prediction and cross-validation. We can easily calculate the RMSE for the training set and the testing set.

```
library(pls)
modpcr<-pcr(fat ~ ., data=trainmeat,ncomp=50)
#summary(modpcr)
#RMSE with 4 PCAs
rmse(predict(modpcr,ncom=4),trainmeat$fat)
## [1] 4.064745
rmse(predict(modpcr,testmeat,ncomp=4),testmeat$fat)
## [1] 4.533982
```

You can use the function RMSEP instead, to select the number of PC's that minimize the 10-fold Cross-Validation error. The resulting Cross-Validation error is < 2.5

```
set.seed(123)
# Minimize RMSE using function RMSEP
pcrmse<-RMSEP(modpcr,newdata=testmeat)
plot(pcrmse)
```

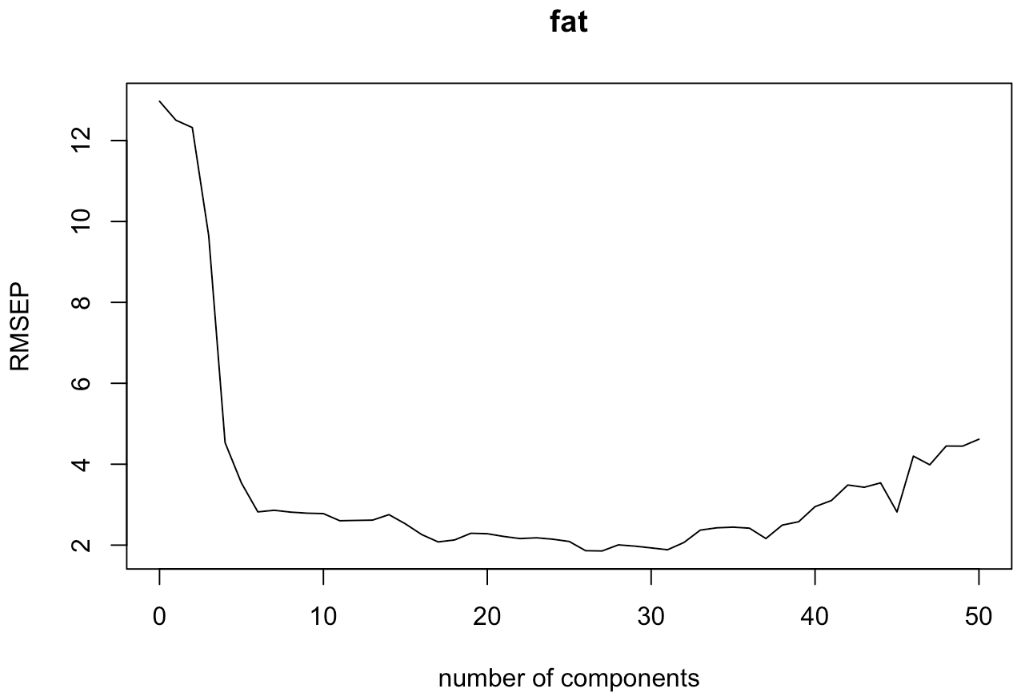

图 15:

9 One/Two Way ANOVA

9.1 Pairwise comparisons

Construct 90% family confidence intervals for all pairwise comparisons of classroom environments.

```
> TukeyHSD(aov(score ~ classroom, data=class.data), "classroom", conf.level=0.9)
  Tukey multiple comparisons of means
    90% family-wise confidence level

Fit: aov(formula = score ~ classroom, data = class.data)

$classroom
    diff       lwr        upr     p adj
B-A -3.0 -4.750606 -1.2493942 0.0029108
C-A -2.7 -4.450606 -0.9493942 0.0073313
C-B  0.3 -1.450606  2.0506058 0.9285599
```

图 16:

10 Experimental Designs

10.1 Paired t-test

```
t.test(shoes$A-shoes$B)
```

10.2

We use the drop1 function instead of anova, because of the lack of orthogonality due to the incompleteness of the design.

```
lmodbibd <- lm(gain~block+treat, rabbit)
drop1(lmodbibd, test="F")
## Single term deletions
##
## Model:
## gain ~ block + treat
##        Df Sum of Sq    RSS     AIC F value    Pr(>F)
## <none>              150.77  78.437
## block   9    595.74 746.51 108.426  6.5854 0.0007602 ***
## treat   5    158.73 309.50  90.013  3.1583 0.0381655 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

11 画图

11.1 2×2 的画布

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

11.2 plot 点图，接上节 (bikeshares)

```
par(mfrow=c(2,2))
# Plot of t1 vs. cnt
plot(bikeshares.reg$t1, bikeshares.reg$cnt, xlab="Real
Temperature in C", ylab="New Bike Shares")
# Plot of t2 vs. cnt
plot(bikeshares.reg$t2, bikeshares.reg$cnt, xlab=" Feels
```

```
Like␣Temperature␣in␣C", ylab="New␣Bike␣Shares")
# Plot of t1 vs. t2
plot(bikeshares.reg$t1, bikeshares.reg$t2, xlab="␣Feels
Like␣Temperature␣in␣C", ylab="Real␣Temperature␣in␣C")
# Plot of hum vs. t1
plot(bikeshares.reg$hum, bikeshares.reg$t1, xlab="Humidity",
ylab="Real␣Temperature␣in␣C")
```

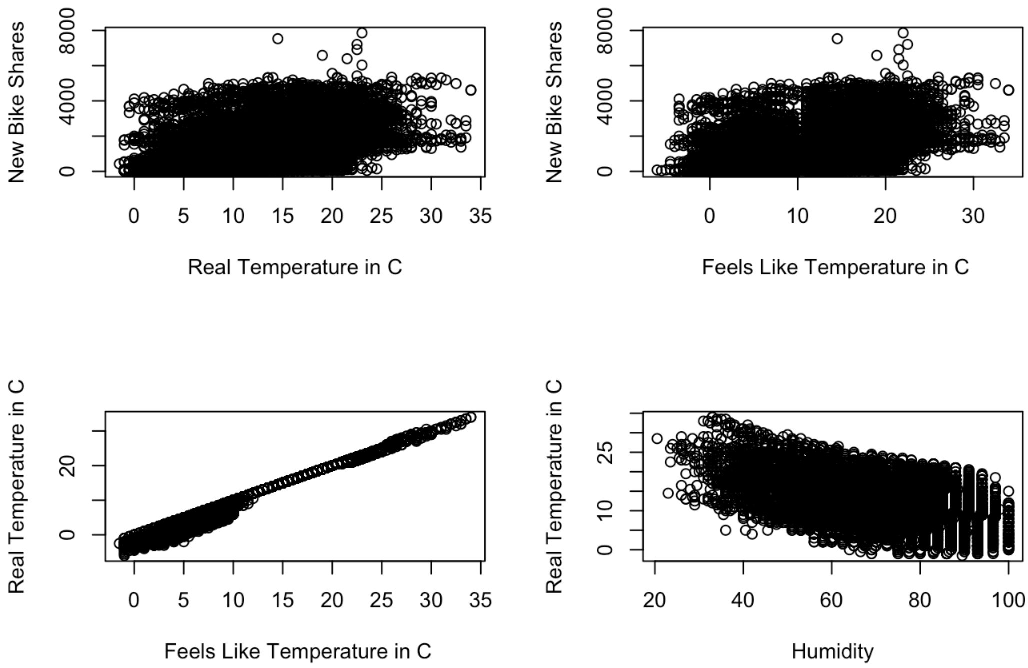

图 17:

11.3 ggplot

```
library(ggplot2)
```

11.3.1 Plot the regression line along with the connected "point-wise"confidence intervals (galton)

```
library(ggplot2)
```

```
ggplot(galton, aes(MP,AH)) + geom_point() + geom_smooth(method=lm)
```

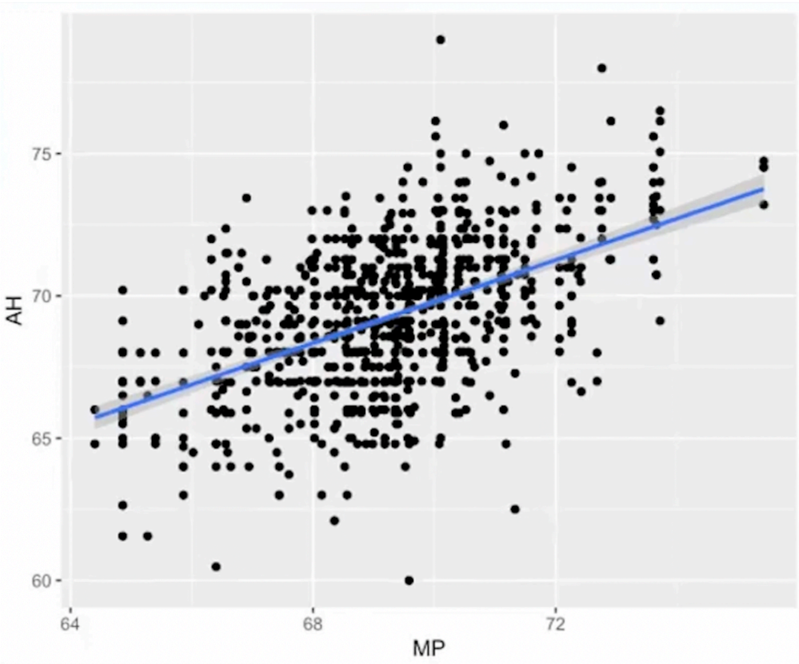

图 18:

11.3.2 给颜色取名，竖直的线，坐标 label

```
# Form the data frame for plotting
ggplot(data=NULL, aes(x=0:56)) +
  geom_line(aes(y=myCI[,1], colour="LSfit"), size=1) +
  geom_line(aes(y=myCI[,2], colour="90%␣CI"), size=1) +
  geom_line(aes(y=myCI[,3], colour="90%␣CI"), size=1) +
  geom_line(aes(y=myPI[,2], colour="90%␣PI"), size=1, linetype=2)+
  geom_line(aes(y=myPI[,3], colour="90%␣PI"), size=1, linetype=2)+
  scale_colour_manual("", values=c("LSfit" = "black",
                                   "90%␣CI" = "blue",
                                   "90%␣PI"="red"))+
  xlab("wind_speed") +ylab("bike␣shares")+
  geom_vline(xintercept = mean(bikeshares.reg$wind_speed),
  colour="purple", size=1, linetype=3)
```

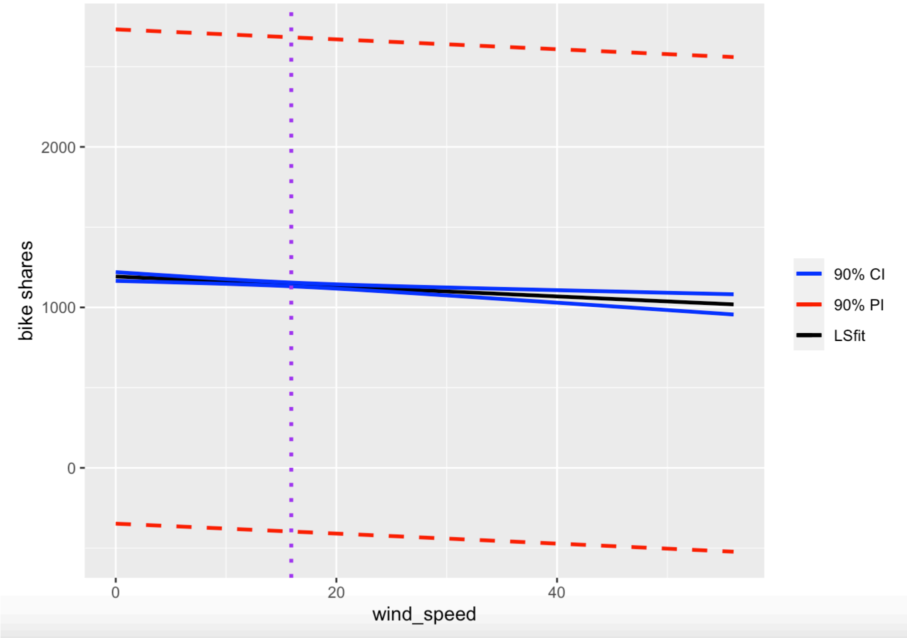

图 19: