Simple Linear Regression
Week 4 – Tuesday

Applied Regression Analysis (STAT 757)

Paul J. Hurtado

Tuesday, 9 Feb, 2016

Announcements

Recall the SLR Model:

Estimates $\hat{\beta}_0$, $\hat{\beta}_1$, and $\hat{\sigma^2}$ yield the *best fit model*

$$Y_i | X = x_i \sim \text{Normal}(\text{mean} = \hat{\beta}_0 + \hat{\beta}_1 \, x_i, \; \text{var} = \hat{\sigma}^2)$$

or, alternatively stated

$$Y_i = \hat{\beta}_0 + \hat{\beta}_1 \, x_i + \epsilon_i$$

where ϵ_i is Normal error with mean 0, variance $\hat{\sigma^2}$.

Parameter Estimation via Optimization

Recall $\widehat{\beta}_0$ and $\widehat{\beta}_1$ were obtained by finding the slope (b_1) and intercept (b_0) that **minimize** the *RSS*

$$RSS = \sum_{i=1}^{n} (y_i - \overbrace{(b_0 + b_1\, x_i)}^{\widehat{y}_i})^2$$

This is an example of an **optimization** problem: Finding function arguments (i.e., input values) that *minimize* or *maximize* an **objective function**.

Optimization in Practice

Suppose we aim to minimize the objective function

$$G(\theta) = G(a, b, c).$$

Typically, two approaches are used:

1. **Analytical:** Solve $\nabla G(\theta) = 0$, i.e.,

$$\frac{\partial G}{\partial a} = 0, \qquad \frac{\partial G}{\partial b} = 0, \qquad \frac{\partial G}{\partial c} = 0.$$

2. **Computational:** Use minimization algorithms (see `optimize()`, `optimx()` in R)

Analytical Example

$$\frac{\partial RSS}{\partial b_0} = -2\sum_{i=1}^{n}(y_i - b_0 - b_1 x_i) = 0$$

$$\frac{\partial RSS}{\partial b_1} = -2\sum_{i=1}^{n} x_i\,(y_i - b_0 - b_1 x_i) = 0$$

which rearranges to two linear equations in b_0 and b_1:

$$\sum_{i=1}^{n} y_i = b_0\, n + b_1 \sum_{i=1}^{n} x_i; \qquad \sum_{i=1}^{n} x_i y_i = b_0 \sum_{i=1}^{n} x_i + b_1 \sum_{i=1}^{n} x_i^2$$

Solving yields the estimates $\widehat{\beta}_1 = S_{XY}/S_{XX}$ and $\widehat{\beta}_0 = \bar{y} - \widehat{\beta}_1 \bar{x}$

Computational

```
library(optimx)
# Minimize f(a,b)=a^2+b^2
f <- function(ps){ return(ps[1]^2+ps[2]^2) }
opt=optimx(c(a=1,b=1),f)
opt

##                          a             b        value fevals gevals niter
## Nelder-Mead  3.754010e-05  5.179101e-05 4.091568e-09     63     NA    NA
## BFGS        -4.263536e-16 -4.263536e-16 9.087931e-30      8      3    NA
##             convcode kkt1 kkt2 xtimes
## Nelder-Mead        0 TRUE TRUE      0
## BFGS               0 TRUE TRUE      0
```

How close to the analytical optimum (0,0) are these estimates?

$$a = -4.2635361 \times 10^{-16} \qquad b = -4.2635361 \times 10^{-16}$$

Exercise

Edit the following code to compare estimates of the slope and intercept obtained from `optimx()` versus `lm()`.

```
library(optimx)
# Simulated data set
set.seed(757)
x=1:20
y=rnorm(length(x),11+1.2*x,sd=pi)

# Minimize obj()=RSS
obj <- function(ps){
   return( sum( (???)^2 ) )
}
p.initial=c(b0=0,b1=0)
opt=optimx(p.initial,obj)
opt

# lm() gives...
summary(lm(y~x))
```

Prediction & Confidence Intervals

Recall that our estimators $\widehat{\beta}_0$ and $\widehat{\beta}_1$ are Normal r.v.s

Properties of Normal Distributions:
Suppose $Y \sim Normal(\mu, \sigma)$ and $a \in \mathbb{R}$.

1. If $Z = a + Y$ then $Z \sim Normal(\mu + a, \sigma)$.
2. If $Z = Y/a$ then $Z \sim Normal(\mu/a, \sigma/a)$.
3. (Standard Normal) If $Z = (X - \mu)/\sigma$ then $Z \sim Normal(0, 1)$.

Note $\bar{X}$ is $Normal(\mu, \sigma/\sqrt{n}$

Prediction & Confidence Intervals

To characterize uncertainty in estimates of β_0 and β_1 (or predictions of $Y|X = x$), use the distributions of $\widehat{\beta_0}$ and $\widehat{\beta_1}$ (or $\widehat{y_i} = \widehat{\beta_0} + \widehat{\beta_1}\, x_i$) to compute **confidence intervals** (or **prediction intervals**). Compare to **credible interval**.

1. A $100(1-\alpha)\%$ **confidence interval** is the interval (a, b) that will contain the expected value of the given distribution approximately $100(1-\alpha)\%$ of the time.
2. If *the complete process of data collection and analysis* were repeated, a $100(1-\alpha)\%$ **prediction interval** will contain the next observation of $Y|X = x$ approximately $100(1-\alpha)\%$ of the time.

Confidence Interval

```
x=1:20
B0=11
B1=1.2
Nreps=1000
CIdat=data.frame(L0=rep(NA,Nreps),U0=NA,B0.in.CI=NA,L1=NA,U1=NA,B1.in.CI=NA)
for(i in 1:Nreps) {
  y=rnorm(length(x),B0+B1*x,sd=pi)
  M=confint(lm(y~x),level = 0.95)
  CIdat$L0[i] = M[1,1];   CIdat$U0[i] = M[1,2]
  CIdat$L1[i] = M[2,1];   CIdat$U1[i] = M[2,2]
  CIdat$B0.in.CI[i] = ( M[1,1]<B0 & B0<M[1,2] )
  CIdat$B1.in.CI[i] = ( M[2,1]<B1 & B1<M[2,2] )
}
sum(CIdat$B0.in.CI)/Nreps

## [1] 0.946

sum(CIdat$B1.in.CI)/Nreps

## [1] 0.954
```

Prediction Interval (SLR)

```r
x=1:20
B0=11
B1=1.2
Nreps=1000
PIdat=data.frame(L=rep(NA,Nreps),U=NA,Y.in.PI=NA)
for(i in 1:Nreps) {
  y=rnorm(length(x),B0+B1*x,sd=pi)
  PI=predict(lm(y~x),data.frame(x=12),interval="prediction",level=0.95)
  PIdat$L[i] = PI[2]
  PIdat$U[i] = PI[3]
  Y=rnorm(1,B0+B1*12,sd=pi)
  PIdat$Y.in.PI[i] = ( PI[2]<Y & Y<PI[3] )
}
sum(PIdat$Y.in.PI)/Nreps

## [1] 0.95
```

Credible Intervals

The difference between *credible intervals* and *confidence intervals* is mostly philosophical: the former arising in Bayesian frameworks, the latter in Frequentist frameworks. *The two can differ substantially* in more complex models, but it's reassuring that for many models (e.g., linear with Normal errors) *they are often indistinguishable*.

For now, it suffices to know (1) that they're different concepts, and (2) what *exactly* defines a confidence interval.

1. The **$100(1-\alpha)$% confidence interval** is an interval calculated from a single data set that for $100(1-\alpha)$% of such data sets includes the true value in question.
2. The $100(1-\alpha)$**% credible interval** is the interval that with $100(1-\alpha)$% probability contains the true value.

For more information, see comparisons in applications, e.g.,

Lu, Ye, and Hill. 2012. **Analysis of regression confidence intervals and Bayesian credible intervals for uncertainty quantification.** *URL: people.sc.fsu.edu/ mye/pdf/paper31.pdf*

Exercise

1. See `?qt`. Modify the **Confidence Interval** code above so that instead of using `confint()` you calculate upper and lower limits using `qt()` and the formulas in Ch. 2.
2. Modify the code resulting from the exercise above to instead (erroneously!) use the Normal distribution, i.e., assume we can use the mean for the expected value and the sample standard deviation for the population standard deviation. *Does the t distribution or the Normal distribution give the broader Confidence Interval?*