Intro to Stat Computing HW 1

Sarah Lotspeich

September 1, 2016

Create a Data Set

```
gender <- c('M','M','F','M','F','F','M','F','M')
age <- c(34, 64, 38, 63, 40, 73, 27, 51, 47)
smoker <- c('no','yes','no','no','yes','no','no','no','yes')
exercise <- factor(c('moderate','frequent','some','some','moderate','none','none','moderate','moderate'
                     levels=c('none','some','moderate','frequent'), ordered=TRUE)
los <- c(4,8,1,10,6,3,9,4,8)
x <- data.frame(gender, age, smoker, exercise, los)
x
```

```
##   gender age smoker exercise los
## 1      M  34     no moderate   4
## 2      M  64    yes frequent   8
## 3      F  38     no     some   1
## 4      M  63     no     some  10
## 5      F  40    yes moderate   6
## 6      F  73     no     none   3
## 7      M  27     no     none   9
## 8      F  51     no moderate   4
## 9      M  47    yes moderate   8
```

Create a Model

```
lm(los ~ gender + age + smoker + exercise, dat=x)
```

```
##
## Call:
## lm(formula = los ~ gender + age + smoker + exercise, data = x)
##
## Coefficients:
## (Intercept)      genderM          age    smokeryes   exercise.L
##    0.588144     4.508675     0.033377     2.966623    -2.749852
##  exercise.Q   exercise.C
##   -0.710942     0.002393
```

1. Looking at this output, genderM seems to have the undisputed largest effect on los. However, this output displays only coefficients without their respective standard errors, t test statistics, and P values.The implication of this output is that males spend 4.509 days longer.

2. Chiseling down from all of the original variables, I focused on a simple model relating variable los(y) to gender(x). Naming this new model mod, I then used the summary() function to explore not only coefficient estimates for hte model but also the standard errors, t test statistics, and P values.

```r
mod <- lm(los~gender, dat=x)
summary(mod)
```

```
##
## Call:
## lm(formula = los ~ gender, data = x)
##
## Residuals:
##    Min     1Q Median     3Q    Max
##   -3.8   -0.5    0.2    1.2    2.5
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)    3.500      1.099   3.186   0.0154 *
## genderM        4.300      1.474   2.917   0.0224 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.197 on 7 degrees of freedom
## Multiple R-squared:  0.5487, Adjusted R-squared:  0.4842
## F-statistic:  8.51 on 1 and 7 DF,  p-value: 0.02243
```

If we're interested in pulling just the coefficients out of this new model, we can do so via the coef() function.

```r
coef(summary(mod))
```

```
##             Estimate Std. Error  t value   Pr(>|t|)
## (Intercept)      3.5   1.098701 3.185581 0.01537082
## genderM          4.3   1.474061 2.917110 0.02243214
```

3. From this additional output, it is straightforward that the estimate for the y-intercept of mod is 3.5 and for the effect of gender is 4.3.

$lengt\hat{h}ofstay = 3.5 + 4.3(genderM = 1 formale, = 1 forfemale)$

4. While the coef() function displays the standard errors, we can calculate these by hand as follows:

```r
sqrt(diag(vcov(summary(mod))))
```

```
## (Intercept)     genderM
##    1.098701    1.474061
```

From here, we can calculate the t test statistics on our own by dividing the coefficient estimates by their respective standard errors.

```r
(mod.c <- coef(summary(mod))) #save the coefficients
```

```
##             Estimate Std. Error  t value   Pr(>|t|)
## (Intercept)      3.5   1.098701 3.185581 0.01537082
## genderM          4.3   1.474061 2.917110 0.02243214
```

```
(testStats <- mod.c[,1]/mod.c[,2]) #calculate test statistics = est / std error
```

```
## (Intercept)     genderM
##    3.185581    2.917110
```

5. Then carry out the t test for gender to find the P value and draw conclusions about significance.

```
(genderP <- pt(q = testStats[2],df = 7, lower.tail = FALSE)) #calculated p value
```

```
##    genderM
## 0.01121607
```

```
2*genderP #calculate two-tailed p value
```

```
##    genderM
## 0.02243214
```

6. Practice predicting values from mod using predict() and fitted() functions.

```
predict(mod)
```

```
##   1   2   3   4   5   6   7   8   9
## 7.8 7.8 3.5 7.8 3.5 3.5 7.8 3.5 7.8
```

```
fitted(mod)
```

```
##   1   2   3   4   5   6   7   8   9
## 7.8 7.8 3.5 7.8 3.5 3.5 7.8 3.5 7.8
```

7. Try using mod to make predictions based on a new data set.

```
(newdat <- data.frame(gender=c("F","M","F")))
```

```
##   gender
## 1      F
## 2      M
## 3      F
```

```
predict(object = mod, newdata = newdat)
```

```
##   1   2   3
## 3.5 7.8 3.5
```

8. Then I added the predicted values to the original dataframe, along with their respective residuals (calculated both by hand and by way of the residuals() function), for easy comparison.

```r
x$predicted <- predict(mod)
x$residCalc <- x$los - x$predicted
x$residFunc <- residuals(mod)
x
```

```
##   gender age smoker exercise los predicted residCalc residFunc
## 1      M  34     no moderate   4       7.8      -3.8      -3.8
## 2      M  64    yes frequent   8       7.8       0.2       0.2
## 3      F  38     no     some   1       3.5      -2.5      -2.5
## 4      M  63     no     some  10       7.8       2.2       2.2
## 5      F  40    yes moderate   6       3.5       2.5       2.5
## 6      F  73     no     none   3       3.5      -0.5      -0.5
## 7      M  27     no     none   9       7.8       1.2       1.2
## 8      F  51     no moderate   4       3.5       0.5       0.5
## 9      M  47    yes moderate   8       7.8       0.2       0.2
```

9. Additionally, I created a sqResid column equal to the residuals^2, which could then be easily summed to calculate the SSR. After executing the deviance() function on mod, I was relieved to find that my homemade calculation for the sum of the squared residuals was equal.

```r
x$sqResid <- x$residFunc^2
x
```

```
##   gender age smoker exercise los predicted residCalc residFunc sqResid
## 1      M  34     no moderate   4       7.8      -3.8      -3.8   14.44
## 2      M  64    yes frequent   8       7.8       0.2       0.2    0.04
## 3      F  38     no     some   1       3.5      -2.5      -2.5    6.25
## 4      M  63     no     some  10       7.8       2.2       2.2    4.84
## 5      F  40    yes moderate   6       3.5       2.5       2.5    6.25
## 6      F  73     no     none   3       3.5      -0.5      -0.5    0.25
## 7      M  27     no     none   9       7.8       1.2       1.2    1.44
## 8      F  51     no moderate   4       3.5       0.5       0.5    0.25
## 9      M  47    yes moderate   8       7.8       0.2       0.2    0.04
```

```r
(SSR <- sum(x$sqResid))
```

```
## [1] 33.8
```

```r
deviance(mod)
```

```
## [1] 33.8
```

10. Using the df.residual() function in conjunction with the deviance() function, I calculated the residual standard error manually and then compared it to the residual standard error in the model summary. Success!

```r
df.residual(mod)
```

```
## [1] 7
```

```r
(calcStdErr <- sqrt(deviance(mod)/df.residual(mod)))
```

```
## [1] 2.197401
```

```r
summary(mod)
```

```
##
## Call:
## lm(formula = los ~ gender, data = x)
##
## Residuals:
##    Min     1Q Median     3Q    Max
##   -3.8   -0.5    0.2    1.2    2.5
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)    3.500      1.099   3.186   0.0154 *
## genderM        4.300      1.474   2.917   0.0224 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.197 on 7 degrees of freedom
## Multiple R-squared:  0.5487, Adjusted R-squared:  0.4842
## F-statistic:  8.51 on 1 and 7 DF,  p-value: 0.02243
```

```r
predict(mod, se.fit=TRUE)$residual.scale
```

```
## [1] 2.197401
```

11. Finally, I conducted a two-sample t-test comparing the length of stay (los) sample means between men and women. Since the two-sample t-test assumes the two samples are of unequal variance, I began by comparing the variance of the los variable in the men and women groups. Inspired by this variance output, I ran the t.test() with and without the assumption that the variances are equal. The p-value from the t.test with equal variance matched the p-value for the genderM variable in the model summary.

```r
men <- subset(x, gender=="M")
women <- subset(x, gender=="F")
var(men$los)
```

```
## [1] 5.2
```

```r
var(women$los)
```

```
## [1] 4.333333
```

```r
t.test(women$los, men$los)
```

```
##
##  Welch Two Sample t-test
##
## data:  women$los and men$los
## t = -2.9509, df = 6.8146, p-value = 0.02205
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -7.7647486 -0.8352514
## sample estimates:
## mean of x mean of y
##       3.5       7.8
```

```
t.test(women$los, men$los, var.equal=TRUE)
```

```
##
##  Two Sample t-test
##
## data:  women$los and men$los
## t = -2.9171, df = 7, p-value = 0.02243
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -7.7856014 -0.8143986
## sample estimates:
## mean of x mean of y
##       3.5       7.8
```

```
t.test(los~gender, dat=x, var.equal=TRUE)
```

```
##
##  Two Sample t-test
##
## data:  los by gender
## t = -2.9171, df = 7, p-value = 0.02243
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -7.7856014 -0.8143986
## sample estimates:
## mean in group F mean in group M
##             3.5             7.8
```

```
#alternative way
t.test(los~gender, dat=x, var.equal=TRUE)$p.value
```

```
## [1] 0.02243214
```

```
coef(summary(lm(los~gender, dat=x)))[2,4]
```

```
## [1] 0.02243214
```