

Intro to Statistical Computing HW 1

Sarah Lotspeich

1 September 2016

Create a Data Set

A data set in R is called a `data.frame`. This particular data set is made of three categorical variables, or factors: `gender`, `smoker`, and `exercise`. In addition `exercise` is an ordered factor. `age` and `los` (length of stay) are continuous variables.

```
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

We can create a model using our data set. In this case I'd like to estimate the association between `los` and all remaining variables. This means `los` is our dependent variable. The other columns will be terms in our model.

The `lm` function will take two arguments, a formula and a data set. The formula is split into two parts, where the vector to the left of `~` is the dependent variable, and items on the right are terms.

```
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 the output, which coefficient seems to have the highest effect on los? (2 points)

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.

This can be tough because it also depends on the scale of the variable. If all the variables are standardized, then this is not the case.

Given that we only have nine observations, it's not really a good idea to include all of our variables in the model. In this case we'd be "over-fitting" our data. Let's only include one term, gender.

Warning

When choosing terms for a model, use prior research, don't just select the variable with the highest coefficient.

2. Create a model using los and gender and assign it to the variable mod. Run the summary function with mod as its argument. (5 points)

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 the model but also the standard errors, t test statistics, and P values.

```
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
```

The summary of our model reports the parameter estimates along with standard errors, test statistics, and p-values. This table of estimates can be extracted with the coef function.

Estimates

3. What is the estimate for the intercept? What is the estimate for gender? Use the coef function. (3 points)

```
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
```

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.

$\text{length of stay} = 3.5 + 4.3(\text{genderM} = 1 \text{ for male, } = 1 \text{ for female})$

4. The second column of coef are standard errors. These can be calculated by taking the sqrt of the diag of the vcov of the summary of mod. Calculate the standard errors. (3 points)

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

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

The third column of coef are test statistics. These can be calculated by dividing the first column by the second column.

The fourth column of coef are p values. This captures the probability of observing a more extreme test statistic. These can be calculated with the pt function, but you will need the degrees-of-freedom. For this model, there are 7 degrees-of-freedom.

5. Use the pt function to calculate the p value for gender. The first argument should be the test statistic for gender. The second argument is the degrees-of-freedom. Also, set the lower.tail argument to FALSE. Finally multiple this result by two. (4 points)

```
(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
```

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
```

Predicted Values

The estimates can be used to create predicted values.

6. It is even easier to see the predicted values by passing the model `mod` to the `predict` or `fitted` functions. Try it out. (2 points)

```
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. `predict` can also use a new data set. Pass `newdat` as the second argument to `predict`. (3 points)

```
(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
```

Residuals

The difference between predicted values and observed values are residuals.

8. Use one of the methods to generate predicted values. Subtract the predicted value from the `x$los` column. (5 points)
9. Try passing `mod` to the `residuals` function. (2 points)

```
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
```

10. Square the residuals, and then sum these values. Compare this to the result of passing `mod` to the deviance function. (6 points)

```
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
```

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

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

```
deviance(mod)
```

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

Remember that our model object has two items in the formula, `los` and `gender`. The residual degrees-of-freedom is the number of observations minus the number of items to account for in the model formula.

This can be seen by passing `mod` to the function `df.residual`.

11. Calculate standard error by dividing the deviance by the degrees-of-freedom, and then taking the square root. Verify that this matches the output labeled “Residual standard error” from `summary(mod)`. (5 points)

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

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

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

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

```
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
```

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

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

T-test

Let's compare the results of our model to a two-sample t-test. We will compare los by men and women.

12. Create a subset of x by taking all records where gender is 'M' and assigning it to the variable men. Do the same for the variable women. (4 points)
13. By default a two-sampled t-test assumes that the two groups have unequal variances. You can calculate variance with the var function. Calculate variance for los for the men and women data sets. (3 points)
14. Call the t.test function, where the first argument is los for women and the second argument is los for men. Call it a second time by adding the argument var.equal and setting it to TRUE. Does either produce output that matches the p value for gender from the model summary? (3 points)

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

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

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

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

```
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(log~gender, dat=x)))[2,4]
```

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