STAT 213: Simple Linear Regression Lab

Colin Reimer Dawson

Last Revised February 22, 2016

The goal of this lab is to carry out the process of choosing, fitting, assessing, and using a regression model from beginning to end, documenting the process and writing about the results in a reproducible format (i.e., RMarkdown).

There is a dataset on my website stored in the file `CountryData.csv`

Data URL: `http://colinreimerdawson.com/data/CountryData.csv`
Code book URL: `http://colinreimerdawson.com/data/CountryData-CodeBook.txt`

with several macro-level variables on 100 randomly selected world countries. **Your goal is to identify and fit a linear model to predict life expectancy using (admittedly artificially) a single predictor variable. That predictor variable may, however, be a transformation and/or combination of more than one of the variables in the data.**

You should carry out the CHOOSE-FIT-ASSESS-USE process to identify, fit and test your model, and to make predictions about life expectancy in countries not included in the data. Document your process (both code and text description of your thought process and results) in an RMarkdown document.

What to include (at a minimum)

1. Plots of the data using various predictors under consideration, together with the accompanying linear model.
2. Mathematical statements of the models that you consider, in the $Y = \beta_0 + \beta_1 \cdot X + \varepsilon$ form, where X may be a more complicated expression than just the name of a single variable.
3. Residual plots that allow you to assess modeling conditions, along with a verbal interpretation of what they show.
4. Confidence interval and hypothesis test of the slopes of your linear models (describe in text; not just by referring to computer output, and interpret what the slope tells us).
5. R^2 for your models (describe in text; not just by referring to computer output)
6. F tests of your models.
7. Plot of confidence and prediction bands with a text description of what they show (for example, pick an X value and describe what the corresponding intervals are and what they mean).
8. An explanation of how you chose among competing models.
9. An overall interpretation of the results.

R Markdown Workflow

1. From the RStudio File menu, select New File > R Markdown.... Select From Template, and pick mosaic fancy. You will want to delete the example stuff in the middle, but leave in place the first two chunks and the last chunk, as well as the bulleted list with session info.
2. The first thing you should do after creating a new Markdown document is to save the file. Do File > Save As... and give your document a name.
3. Before editing anything, press the Knit HTML button in the document toolbar. If on the server you may get a message about a blocked pop-up. Allow it to show.
4. The first thing you should edit is the Title and Author fields at the top of the document. Give the document a title and put your name inside the quotes on the Author line. Re-Knit to see your change reflected in the document.
5. In general, every time you make a change, it is a good idea to Knit again. That way, if you get an error message, you know what caused it. As you get more experienced you may find yourself going longer between re-Knits, but at first you should do it very frequently.
6. You should **not** include any lines of code in the Markdown document that (1) access documentation for packages or functions (e.g., `?histogram`) (2) otherwise cause external windows (other than plots) to pop up, or (3) cause new packages to be downloaded and installed.
7. You **must** include lines of code in the Markdown document that (1) cause packages or datasets to become *visible* (for example, `library()` and `data()` lines), (2) read datasets from files (e.g, `read.file()`), or (3) define variables that you use in later lines.
8. In general, include all and only code that is needed to produce the output you want, beginning from a fresh R session at a computer on which needed packages are already installed.

Essential R Commands for Regression (most require `mosaic`)

Case selection / defining new variables (e.g., ratio of two others)

```
filter(DATANAME, CONDITION) %>%
    some.function(..., data = .)
mutate(DATANAME, NewVariableName = OldVariable1 / OldVariable2) %>%
    some.function(..., data = .)
##   Note: CONDITION is an expression like
##     `Meltdown == 0`, or `Height >= 60`, or
##     `Color == "red" & Size < 100` (without the quotes)
##     (It is also possible to assign the output of filter() or mutate()
##     to a new variable, if, e.g., you want to use it repeatedly,
##     instead of piping it directly to plotting/modeling functions)
```

Scatterplots

```
xyplot(Y ~ X, data = DATANAME, groups = Z, auto.key = TRUE)
##   Note: the groups= argument will create different plotting
##     symbols for different levels of the categorical variable Z.
##     Can also create a binary (TRUE/FALSE) variable on the fly by
##     doing something like: groups = (Z <= 100).
##     auto.key = TRUE displays a legend showing the symbol meanings
```

Fitting, describing and assessing a regression model

```
## Fit (can also apply transformations, e.g., log(Y) ~ X)
my.model <- lm(Y ~ X, data = DATANAME)
## T-tests, F-test, R^2; ANOVA table
summary(my.model); anova(my.model)
## To get LEVEL confidence intervals for the parameters
confint(my.model, level = LEVEL)
## Residual plots
plot(my.model, which = 1)  ## fitted by residuals
plot(my.model, which = 2)  ## quantile-quantile plot
## Getting parts of the model as variables
## (Note: can also use these in plots, e.g., histogram())
coef(my.model); residuals(my.model); fitted.values(my.model)
```

More Essential R Commands for Regression (most require `mosaic`)

Using the model (after fitting):

```
## Create a prediction function
f.hat <- makeFun(my.model)

## Plot the line over the data
xyplot(Y ~ X, data = DATANAME)
plotFun(f.hat(X) ~ X, add = TRUE)

## Get the predicted Y value for a new X
f.hat(X = 3)
## with LEVEL confidence/prediction intervals
f.hat(X = 3, interval = "confidence", level = LEVEL)
f.hat(X = 3, interval = "prediction", level = LEVEL)

## Plot the data with LEVEL confidence and prediction bands
xyplot(Y ~ X, data = DATANAME, panel = panel.lmbands, level = LEVEL)
```