STAT 215: Mini Project 0 (AKA Lab 6)

Colin Reimer Dawson

Last Revised October 13, 2017

The goal of this mini-project 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). I am not providing step-by-step instructions, but I have included a quick-reference guide for the most important R commands.

You should turn in a Markdown report with (at a minimum) the components indicated below in "Components to Include" by the start of the lab on Friday 10/27.

I have provided reference material on the following pages for essential R commands, and for the Markdown workflow (in case you have not been using it much).

The Goal

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 try to model the relationship between the life expectancy of a country and the other variables in the data.** We have not yet discussed models with multiple predictors, so for now your models will be limited to one predictor at a time. However, you may want to consider transformation of the response variable and/or the explanatory variable, as well as "composite" variables that are computed using more than one of the other variables.

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 text interspersed between code chunks in your RMarkdown document.

Components to Include

The following are suggested elements to include in your analysis, though you might want to include other things as well!

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 (e.g., it may be a transformed variable, a difference or ratio between two other variables, etc.))
3. Diagnostic plots that allow you to assess modeling conditions, detect outliers, etc., 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 (ANOVA) 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 overall interpretation of the results.

Essential R commands

(Most of the following require `mosaic`)

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

```
filter(DataName, CONDITION) %>%
    some.function(..., data = .)
NewDataName <-
    mutate(DataName, NewVariableName = sqrt(OldVariable1) / OldVariable2)
##   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,
       type = c("p","r"), 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
plot(my.model, which = 4)  ## Cook's Distance 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)
rstandard(my.model); rstudent(mymodel)
```

Using the model (after fitting):

```r
## 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)
```