Shiny Applications

Created By Matt Branan on Wednesday, October 22, 2014 1:41:05 AM MDT

Why Shiny Applications?

Shiny applications with R. It sounds intriguing by name already! Hans Rosling's discussion of financial development of countries over time showed many not involved in statistics that statistics can be visually engaging. Shiny is a packages that implements an interface in R to help the user create interactive web-based applications. I see this type of interactive implementation of R to have a lot of potential in the classroom, in certain consulting contexts, and in the informing of the general population of statistical ideas. For example, in some of the introductory statistics courses, we already use a JavaScript application to help students better understand power caluclations and their implications on sample size using Russel Lenth's power applet. Sometimes calculations can go awry even with this small-ish applet and if this is not appealing to you, you can pretty quickly develop your own using R!

Menu-driven software as well as applets like that of Dr. Lenth are great teaching tools in my experience not only in the classroom but possibly with consultees. Let's say that we are working with a consultee on finding a good linear model to model baseball salaries using the simulated annealing approach from Homework 4 from this class. There were a slew of parameters that could be tweaked in order to arrive at an algorithm with satisfying convergence properties. Suppose that our consultee is not comfortable programming raw R but we would nevertheless like their input in the proper choice of these convergence-altering parameters (such as initial temperature, cooling schedule, intra-temperature iteration update schedule, total number of iterations, etc.). We could very quickly write up a menu-based interactive plot that could allow the client to manipulate the various parameters to achieve the convergence that their expertise agrees with. Although it is a simple example, it goes to demonstrate the empowerment given to the client. They could put in their opinions without feeling excluded by the raw computing environment of ordinary R.

How do you Create a Shiny Application?

How can we actually build and run a Shiny application? First, we will need to install the shiny package in R and load it into our current working directory:

```
install.packages("shiny",dependencies=TRUE)
library(shiny)
```

Once this is done, we will need to create two files in a directory:

1. The ui.R file: this file contains the necessary information to create the input/output buttons, sliders, data input windows, and print the necessary graphics and tables that you would like to include in your application.
2. The server.R file: this file contains the information to generate the data to feed the buttons, sliders, and graphics that are printed to the user.

As far as I know, these two files must be called exactly "ui.R" and "server.R" in order to be found by the shiny package functions. Let's say that you have these two files in a folder on your desktop called "ShinyApp". Then, using regular R syntax, you can begin to construct your application. I created an application (see the linked files above) that generated random data from distributions of various shapes and plotted the observations in both a histogram and a normal QQ plot so that students could play around to get a feel of what information was contained in the normal QQ plots.

Actually writing the two files doesn't take very long after you know what tools are available to use. A great introduction can be found on the shiny package website. Some of the user interface tools you can weave into your application are:

- Sliders, radial buttons, checkboxes, and drop-down menus to choose values of variables to be used
- Simple data uploading and downloading
- Simple raw data exploration and subsetting in real-time

As a brief view into how shiny integrates interactivity into the R-based code, shiny focuses on what are dubbed "reactive" objects. When one might ordinarily write interactive applications in JavaScript, HTML, or CSS, you would have to write blocks of code that specifically track when user input has changed and specify when and what to do when those values change. In shiny via R, one only has to specify what the user will input and what happens to the application for the values the user can change that variable to and declare that a "reactive" object to denote

that the user could change the variable's value at any time. For example, the reactive object in the server.R file for the QQ plot application is defined as:

```
distrSpecs=reactive({
input$goButton #Make the distribution randomize upon the go button
chosendata=switch(input$distrshape, #Choose from a variety of distributions with n observations from user input
normal=rnorm(input$n),
skewedleft=rbeta(input$n,5,2),
skewedright=rbeta(input$n,2,5),
heavytailed=rt(input$n,1),
shorttailed=rbeta(input$n,2,2),
bimodal=rep(c(0,1),c(input$n/2,input$n/2))*rchisq(input$n,4)+rep(c(1,0),c(input$n/2,input$n/2))*rnorm(input$n,15,3),
multimodal=rnorm(input$n,c(1:input$modes)^2+10,rep(1,input$modes))
)
})
```

Notice, the variable "distrSpecs" is reactive because the user can change it's input value using a dropdown menu with seven options for the shape of the distribution they would like to observe. Then, the switch statement simply tells shiny what to do for each possible shape the user could choose. Specifically, if the user wanted a normal distribution, then the "distrSpecs" variable would then be declared as a vector of "n" observations from the standard normal distribution. The changes to the input values by the user are updated in real-time (or by the press of a button if there is computational expense to running your code) using intelligent passing of arguments from the user interface to the reactive objects and back to the user in the form of a graphic or other output.

How do I share my Shiny Application?

Now that you have your two application scripts written or someone else has them written, how do you run the code? There are two choices: (1) you need a server from which to host your web-based application, or (2) your users must have R and the shiny package installed on their own computers. I don't know much about (1), so I'll focus on (2) but let us know if you can shed some light on hosting your application on your own server!

As for (2), you can make your files privately available on the web somewhere such as a Dropbox or Gist. Then, your user must open up R, load the shiny library, and run code such as:

```
shiny::runGist('https://gist.github.com/anonymous/3f3a2a167712b4e91ea5')
```

And the application stored at that Gist URL will be run in the user's browser. Run the above code to run the QQ plot example alluded to above. If you visit that URL, you'll find there is just the two ui.R and server.R files that were given above. This can be run on *any* computer with R, shiny, and access to the internet!