Lab 8 Statistical Computing, 36-350

Friday October 30, 2015

Agenda: Writing functions to automate jobs; applying functions; creating new variables.

In the last lab, we worked with one of the classic data sets on the diffusion of innvoations, on the spread of the use of tetracycline among doctors in Illinois in the 1950s. We wrote R expressions to find things like how many of doctor 37's friends had begun prescribing tetracycline by month 5. In this lab, we will turn those particular expressions into general functions, and apply them across the data, to get systematic evidence about whether new practices can spread by peer pressure.

As a reminder, on the class website, you will find two data files, $http://www.stat.cmu.edu/~ryantibs/statcomp/labs/ckm_network.dat$. Read them in with

```
ckm_nodes = read.csv(url("http://www.stat.cmu.edu/~ryantibs/statcomp/labs/ckm_nodes.csv"))
ckm_network = as.matrix(read.table("http://www.stat.cmu.edu/~ryantibs/statcomp/labs/ckm_network.dat"))
```

The former has information about each individual doctor in the four towns. The latter records which doctors knew each other. You are free to re-use any relevant work from the last lab.

Part I

- 1. In the object ckm_nodes, the adoption_date column records the month in which the doctor began prescribing tetracycline, counting from November 1953. If the doctor did not begin prescribing it by month 17, i.e., February 1955, when the study ended, this is recorded as Inf. If it's not known when or if a doctor adopted tetracycline, their value is NA. Create a vector which records the index numbers of doctors for whom adoption_date is not NA. Check that this vector has length 125. Create a data frame, cleaned_nodes, which contains only those rows of ckm_nodes. (Do not drop rows if they have a value for adoption_date but are NA in some other column.) Use cleaned_nodes, rather than ckm_nodes, for the rest of the lab.
- 2. Write a function, adopters, which takes two arguments, month, with no default value, and not.yet, defaulting to FALSE. If not.yet is FALSE, adopters should return a Boolean vector, indicating the doctors who began prescribing tetracycline in that month. If not.yet is TRUE, then adopters should return the vector indicating the doctors who began prescribing *after* that month (or never did). Check that adopters(2) indicates 9 doctors began prescribing in month 2, and that adopters(month=14,not.yet=TRUE) indicates that 23 doctors began prescribing after month 14, or never did.
- 3. The object ckm_network is a binary matrix; the entry in row *i*, column *j* is 1 if doctor number *i* said that doctor *j* was a friend or close professional contact, and 0 otherwise. Create a new matrix, clean_network, which drops the rows and columns corresponding to doctors with missing adoption_date values. Check that the result has 125 rows and columns. Use this reduced matrix, and its row and column numbers, for the rest of the lab.
- 4. Create a vector which stores the number of contacts each doctor has. Use an apply function rather than a loop. Check that doctor number 41 had 3 contacts.

Part II

- 5. Counting Peer Pressure
 - a. Write a function, count_peer_pressure, which takes in the index number of a doctor and a month, and returns the number of doctors whom that doctor named as contacts, *and* had begun prescribing tetracycline by that month or earlier. If it is working properly, doctor number 37 and month 5 should return 3.
 - b. Write a function, prop_peer_pressure, which takes in the index number of a doctor and a month, and returns the proportion of the doctor's contacts who are already prescribing tetracycline by that month. If a doctor has no contacts, your function should return NaN. Check that doctor 37, month 5 returns a proportion of 0.6, but doctor 102 in month 14 returns NaN. Your function should call, not repeat, your function from (5a), and use your vector from (4).
- 6. Averaging Peer Pressure
 - a. Write a function which takes in a month, and returns a vector of length two. For the first element, consider doctors who *began* prescribing in that month, then return the average proportion of prescribers among their contacts. For the second element, , consider doctors who *began* prescribing after that month (or never), then return the average proportion of prescribers among their contacts. Call your code from (2) and (5); use an apply function rather than a loop if at all possible. *Hint*: mean has an na.rm option.
 - b. Compute the average proportions from (6a) for each month in the study. Use an **apply** function rather than a loop if you can. Plot the two average proportions from (6a) over time, and in a second plot show their difference. Do the doctors who adopt in a given month consistently have more contacts who are already prescribing than the non-adopters?

Behind the scenes: See Lab 7 for source notes.