coursera.org

R Programming - Discussions | Coursera

The following are some suggestions based on some common questions and issues that always seem to come up with the hospital assignment. Keep in mind there is no one right method of solving the assignment. The main purpose of the post is to get you thinking about some different ways to approach things.

Reading the Data

If you use read.csv with na.strings="Not Available" and stringsAsFactors=FALSE you shouldn't need to convert any columns to numeric or character. Plus, your sorts will work as expected.

Selecting Columns

You only need at most three columns - hospital name, state, and one of three outcome columns. To make things easier to work with use the names function to rename the three columns. Something like *names(my_data) <- c("hospital", "state", "outcome")*.

When selecting an outcome column, think about how you might go about doing that. Everyone has their preferences but personally I like to do the column subsets with column indexes. One way might be to use a column index variable that's based on the outcome function argument. For this particular data set, heart attack is column 11, heart failure is column 17, and pneumonia is column 23. All you have to do to select columns is use brackets and pass a numeric vector to the right side of the comma. For example, if you named your outcome index column_index, you could select the columns with something like *df[, c(2,7,column_index)]*.

Hint: if you setup a named vector with something like *outcomes <- c("heart attack"=11, "heart failure"=17, "pneumonia"=23)* then you can use that to both test the function argument and select the column. Something like *df[, c(2,7,outcomes[outcome])]*. Also, when you validate the outcome argument instead of using %in% outcomes you'd use %in% names(outcomes).

Removing NA Values

You can use na.omit or complete.cases to remove NA values but don't do that until you've reduced the data to only one output column plus name and state. If you remove NAs immediately after reading the data you can lose data that you need.

Ranking

Think logically about the data and ranking. For example, if you were to sort by state then outcome then hospital name then your data will be in proper rank order including the tie breaker. At that point, all you have to do is take a state subset and use an index to get best, worst, or a particular rank. For best you'd use 1 for the index, for worst you'd use nrow(your_subset) and for just num you'd use that num. Don't worry about testing for num greater than number of rows because if you use an index larger then the number of rows you'll get NA which is what you want anyway.

Processing the Data

If you need to process data in groups consider using one of the apply functions. One of the easiest to work with is lapply. For a walk-through on how lapply works, see this post - https://www.coursera.org/learn/r-programming/module/6BaZ2/discussions/qcv_orZHEeWF2Q53QdZUbw.

Suppose you have data in three columns ordered by state then outcome then hospital. Then suppose you split on state using something like *split(your_data, your_data$state)*. You get a list of data frames ordered by state. Because you already ordered the data everything else is in rank order. Then suppose you only want a list of hospitals by rank. You run the list of data frames through lapply and use a function to store the hospital names in a list.

What happens is lapply runs each item in the list through the function. The data passed to the function is one of the data frames from the list created by split(). The value returned by that function gets stored in the results of lapply() which is a list.

So, if you split the data frame on state then *lapply(split.data, hospitalNameFunction)* would return a list where the name of each item is state and the value is the hospital name. You can get the values as a vector using *unlist(results_of_lapply)*. You can get the states using *names(results_of_lapply)*. You could also use sapply and then you wouldn't need to use unlist.

Then all you have to do is assemble the data frame using the unlisted values and list names using something like this - *data.frame(hospital=unlisted_values, state=list_names, row.names=list_names)*.

Finally

As with everything R there are many ways to do things. These are just a few suggestions. Most of all, have fun and experiment.

Note: This post is now pinned and available to all sessions but has been closed for comments. If you

have questions on any of these methods please post a new thread. Thanks.