R-Workshop-James

James LeBreton with Rick Gilmore

2017-08-11 12:35:09

Contents

PART 1: INSTALLATION, SETTINGS, AND DATA MANAGEMENT

TOPIC 1: Projects & Directories in R Studio

```
getwd() #get the current working directory
setwd("~/Dropbox/James Work Files/R Workshop/2017") #change the working directory
```

Since `~/Dropbox/James Work Files/R Workshop/2017` is specific to James' computer, it won't work for others. When using an RStudio project, I don't change my working directory. Instead, I just make sure I give relevant functions information about the directories where other resources can be found.

TOPIC 2: Installing Packages & Loading into Active Library of Resources

Install packages via syntax

```
install.packages("multilevel") #Downloading a package to my computer
#loading packages into working library
library("multilevel")
```

Understanding How R Searches for Information

```
search()
detach(package:multilevel)
search()
```

Obtaining Help

```
#You may inquire about a function using any of the following:
##If you know the exact name:
?search
help(search)

##If want to search by part of the name
apropos("searc")
??sear
```

Another good source of help is StackOverflow.

TOPIC 3: Data Types & Structures in R

Numbers

```r
x <- 2
x
y = c(1:3); y
z = c("Porsche 911", "Porsche 944", "Porsche 911", "BMW 335xi")
z
g=sqrt(x); g

is.numeric(x)
is.numeric(z)
```

Strings

```r
#String Data as character:
z
#String Data as factor:
z2=factor(z)
z2

#Compute the Length of a String (or Numeric) Variable:
nchar(x)
nchar(y)
nchar(y)
nchar(z)
#nchar(z2) Throws error during rendering
```

Logical Data

```r
##Assumes values of TRUE or FALSE
###TRUE is considered equal to 1
###FALSE is considered equal to 0
TRUE*5
sqrt(TRUE)
t=TRUE
# you can test if a variable type is logical using:
is.logical(x)
is.logical(t)
# Logical data types also used as input to functions (see Day 2 examples)
2==2
2==3
```

Vectors

```r
#Vectors - 1 dimensional collections of same type data
v1=1:5; v1 #creating vector of numbers
v2=c(1,2,3,4,5); v2
```

```r
v3=c("Porsche 911", "Ford Mustang GT", "Plymouth Baracuda", "Chevrolet Camaro", "Honda Pilot LX")
v1; v2; v3
#Matrices - 2 dimensional collections of same type data
m=matrix(1:20, nrow=5); m
```

Arrays & Data Frames

```r
#Arrays - multidimensional collection of same type data
#example of 3D array
a=array(1:20, dim=c(2,5,2)); a

#Creating a data frame from vectors
eng=c("Flat-6", "V-8", "V-8", "V-8", "V-6")
doors=c(2,2,2,2,4)
data1=data.frame(v2, v3, eng, doors)

# Viewing content of data framees
#   Look at the "enviroment" tab in the upper left panel
#   Click on one of the data frames listed under Data (e.g., "data1")
#   Or, simply type:

data1

# Obtain a list of the variable names in a data frame
names(data1)
```

```r
# Change the names of the variables in a data frame
data2=data.frame(id=v2, model=v3, eng=eng, doors=doors) #creates a new data frame
data1
data2
data3=data1 #make a copy of the original dataframe
#install.packages("plyr")
library(plyr)
data3=rename(data3, replace=c("v2"="id","v3" = "model"))  #renames specific variables
data3
names(data1)=c("id","model", "eng", "doors")  #replaces names of all variables in existing data frame
data1
```

TOPIC 4: Reading Data Files into R

Reading Data - From R Data Sets

```r
rm(list=ls()) #Clear the Global Environment

##List of avaialble data sets
data()
library(multilevel)
#List data in the multilevel package
```

```r
data(package="multilevel")
#load the univ data frame into R environment
data(univbct, package="multilevel")
d=univbct

#Confirm it is loaded as a data frame
class(d)
```

Saving data frames as comma-separated value (CSV)

```r
#Saving a data frame as a .csv file (to be read into SPSS, Excel, Text Editor, etc.)
write.table(d,file = "d2.csv",sep=",",row.names=F)
write.table(d,"d1.csv",sep=",", row.names=FALSE)

#save the data as a text file to be read into SPSS
install.packages("foreign")
library("foreign")
write.foreign(univbct,
              datafile="univbct.csv",
              codefile="univbct.sps",
              package="SPSS")
file.show("univbct.csv")
file.show("univbct.sps")
```

Reading data from SPSS

```r
library("foreign")
demo1=read.spss(file="../data/demo1.sav",
                use.value.labels=TRUE,
                to.data.frame=TRUE,
                use.missings=TRUE)
summary(demo1)
demo2=read.spss(file="../data/demo2.sav",
                use.value.labels=T,
                to.data.frame=T,
                use.missings=FALSE)
summary(demo2) #oops, GENDER = 999 was a missing values code
demo2=read.spss(file="../data/demo2.sav",
                use.value.labels=T,
                to.data.frame=T,
                use.missings=T)
names(demo1); names(demo2)
#Reading data (csv)
data1=read.csv("../data/data1.csv", header=T)
data2=read.csv("../data/data2.csv")
```

```r
#Now click on "Environment" tab and the "data1" dataframe
#NA (not available) is automatically inserted by R for any missing data
head(data1) # display first 6 cases
tail(data1) # display last 6 cases
summary(data1) # display summary
summary(data2)
```

Handling missing values

```r
#Note: I used 999 to represent missing data for JOBSAT1 COMMIT1 and READY1
#R needs to be told that 999 is not a legitimate value, but is user-defined missing value
data1$JOBSAT1[data1$JOBSAT1==999]=NA #Explain what the heck this means!
data1$COMMIT1[data1$COMMIT1==999]=NA
data1$READY1[data1$READY1==999]=NA
summary(data1)
summary(data2)
```

```r
#The above can be tedious if you have a large number of variables
### it is eaiser if you copy & paste code
#Or, if 999 doens't hold any meaning for ANY of the variables
data1=read.csv("../data/data1.csv", na.strings=c(".", "999","9","-9"))
summary(data1)
#OR, you could write a function
my999isNA=function(x) {x[x==999]=NA; x}
```

```r
#Now we will apply this missing data function to the proper variables in data2
#To do this, we use the "lapply" function which allows us to apply the same function over a list or arr

data1=read.csv("../data/data1.csv") #reread data1 as a data.frame with missing data
names(data1)
summary(data1)
data1[3:5]=lapply(data1[3:5],my999isNA)
summary(data1)
```

TOPIC 5: Merging Data Files

```r
#Merging data by adding variables (e.g, two data.frames, demo1 + data1)
dd1=merge(demo1,data1, by="SUBNUM")
dd1=merge(demo1,data1, by=c("SUBNUM","TIME"), all=TRUE)

dd2=merge(demo2,data2, by=c("SUBNUM","TIME"), all=TRUE)
summary(dd1)
summary(dd2)
```

Merging data by adding rows (subjects)

```
#let's combine dd1 with dd2
#when you have IDENTICAL columns in both data sets you may use rbind
names(dd1); names(dd2)
dd3=rbind(dd1,dd2)
summary(dd3)

#when you have different columns in your data, you can use rbind.fill
#first let's compute some extra variables and add them to dd1
#Computing new variables in an existing data.frame
dd1$STAY=dd1$JSAT+dd1$COMMIT
#dd3=rbind(dd1,dd2) doesn't work because of differing colums
?rbind.fill
dd3=rbind.fill(dd1,dd2)
head(dd3); tail(dd3)
```

Deleting a variable from a data frame

```
#let's delete STAY from the previous dd3 data.frame
names(dd3)
dd4=dd3[c(1,2,3:22)]
names(dd4)

#Renaming a variable in a data.frame
#let's rename HOWLONG to TENURE and MARITAL to STATUS
dd4=rename(dd4, c(HOWLONG="TENURE", MARITAL="STATUS"))
names(dd4)
```

Recoding variables

```
#Categorical Variables: recode sex into a different, dummy variable
#Only "factor" type variables are assigned value labels
dd4$GENDER2=revalue(as.factor(dd4$GENDER), c("1"="male","2"="female"))
dd4$GENDER3=(dd4$GENDER-1)
class(dd4$GENDER)
class(dd4$GENDER2)
class(dd4$GENDER3)

#recode Likert-type items/scales
###let's reverse the overall score on COMMIT so that high scores = more likely to leave
dd4$LEAVE=6-dd4$COMMIT
```

TOPIC 6: Summarizing & Visualizing Data Frames

Central Tendency

```
mean(dd3$JSAT); median(dd3$JSAT)
mean(dd3$JSAT,na.rm=TRUE); median(dd3$JSAT,na.rm=TRUE)
#Dispersion
var(dd3$JSAT,na.rm=T)
sd(dd3$JSAT,na.rm=T)
min(dd3$JSAT, na.rm=T)
max(dd3$JSAT,na.rm=T)
summary(dd3$JSAT,na.rm=T)
quantile(dd3$JSAT,probs=c(.1,.2,.3,.4,.5,.6,.7,.8,.9),na.rm=T)
```

Alternative: Hmisc

```
#install.packages("Hmisc")
library("Hmisc")
describe(dd4)
```

Alternative: psych

```
detach("package:Hmisc")
#install.packages("psych")
library(psych)
describe(dd4,na.rm=T)
describe(dd4,na.rm=F)
describe(na.omit(dd4))
```

Simple Distributions

```
#Frequency Counts
table(dd4$COMPANY)
#Proportions
prop.table(table(dd4$COMPANY))
#Rounding proportions to 3 decimals
round(prop.table(table(dd4$COMPANY)),3)
#Percentages
100*(prop.table(table(dd4$COMPANY)))
```

```
#Cross Tabs & Simple Tables
#install.packages("gmodels")
library(gmodels)
CrossTable(dd4$GENDER,dd4$COMPANY,chisq=TRUE,format="SPSS")
table(dd4$GENDER,dd4$COMPANY)
prop.table(table(dd4$GENDER,dd4$COMPANY))
```

```r
#Histograms
hist(dd4$JSAT)
hist(dd4$JSAT, main="Job Satisfaction Histogram",xlab="Job Satisfaction" )
```

Correlations using cor (part of stats) or rcorr (part of Hmisc)

```r
cor(dd4[,20:22],use="complete.obs")
#install.packages("Hmisc")
library(Hmisc)
rcorr(as.matrix(dd4[,c(20:22)]))
```

Popular Packages

multilevel

lme4 & nlme

plyr

ggplot2

reshape2

Rcmdr

Hmisc