Benchmarks Online

Research and Statistical Support
University of North Texas

RSS Matters

You can link to the last RSS article here: SPSS' Hotfix for Windows Vista - Ed.

R Techniques: Summarizing Data By Grouping Variables

By Dr. Rich Herrington, Academic Computing and User Services, CITC

In this article we examine a few different ways of summarizing data across groups or what is referred to in as "factors" in the R language. R's model for data types consist of several different **"modes"** - numeric, logical, character, and factor (other data types exist, but we concern ourselves here with the type called numeric and factor. Factor variables are used in R to represent class information or "nominal" data. Converting modes, for example character modes such as "M" or "F" or "Single" or "Married", are accomplished by using the **"as."** conversion functions. For example, **"as.factor(gender)"** will convert the variable which consists of numeric 1's and 0's to a factor variable that can be used subset the data for summarization. Finally, we use various combinations of looping techniques, indexing techniques, and the **"split"** function to subset by groups and display the group statistics. Below we present several ways of doing this.

Several Methods for Summarizing on Factor Variables

```
# Create data frame with one variable that is of type "factor"
# to be used as a grouping factor

grp<-c(0,0,0,1,1,1)
grp<-as.factor(grp)

dv1<-c(10,20,30)
dv2<-c(40,50,60)

dataSet<-data.frame(grp, dv1, dv2)

attach(dataSet)

################ Version 1

# First use library "doBy", or alternatively use "split" function

# Using library "doBy"
library(doBy)
summaryBy(dv1~grp, data=dataSet, FUN=c(mean, sd), na.rm=TRUE)
summaryBy(dv2~grp, data=dataSet, FUN=c(mean, sd), na.rm=TRUE)
```

```
# Using "split" function
dataSet.list<-split(dataSet, grp)
lapply(dataSet.list, mean)
lapply(dataSet.list, sd)



################ Version  2

# Using an "apply" function only; Subsetting occurs
# within the apply calling arguments

apply(dataSet[dataSet$grp==0, c("dv1", "dv2")], 2, mean)
apply(dataSet[dataSet$grp==1, c("dv1", "dv2")], 2, sd)


################ Version 3
#
# Using "apply" With a "for loop";
# The subsetting is done by creating "index" objects that
# are then used within the apply function calling arguments

# Create "dv" index object
var.index<-c("dv1", "dv2")

# Create group index object - has the number of levels of outcome
grp.value<-c(0,1)

for (i in grp.value){
 print(i)
 print(apply(dataSet[dataSet$grp==i, var.index], 2, mean))
 print(apply(dataSet[dataSet$grp==i, var.index], 2, sd))
}


#################  Version 4

# Create a summarize function;  set parameters to send to summarize
# function;  call the summarize function in a loop and format output
# before calling summarize function

# Summarize function
my.summarize.function<-function(dataVector, grp.Vector, grp.Value){
  cat("mean\n")
  print(mean(dataVector[grp.Vector==grp.Value]))
  cat("sd\n")
  print(sd(dataVector[grp.Vector==grp.Value]))
        cat("\n\n\n")
}

# Set up the calling parameters to function "my.summarize.function"

# Select names of dv's and grouping variable

# Extract all names
var.index<-names(dataSet)

# Extract matrix with only numeric data vectors (drop grouping variable on
# column 1)
dataVectors<-dataSet[,var.index[-1]]

# Extract grouping variable (vector is column 1)
grp.Vector<-dataSet[,1]

# Declare which value of grouping variable to summarize by
grp.Value<-1


#######     Two different ways of calling my.summarize.function


# Using "apply" function only (cannot print the column names
# or the dv names when the function is called)
apply(dataVectors, 2, my.function, grp.Vector, grp.Value)


# Using a "for loop" that calls function "my.summarize.function"
# We can print the column names and dv names before calling
# "my.summarize.function";  this allows for some formatting of output

for (i in var.index[-1]){
```

```
         cat("group = ")
   cat(paste(as.character(grp.Value)),"\n")
         cat(paste(as.character(i)),"\n")
         my.summarize.function(dataVector=dataVectors[, i], grp.Vector,
grp.Value)
  }
```

Good luck and happy computing until next month. I'll leave you with the following joke that my colleagues here in the office just absolutely groaned at ..... I think they don't have a refined sense of humor.

Q: What is so hilarious about high prices with statistical software in academia?

A: Most people don't get the joke! "arrr..arrrr....arrrr...arrrr....rrrr".

Get it? It's the sound of a pirate laughing. Oh well, I'll keep working at it.

Return to top