Cleaning input data JdG

Epi Lab color code

Software/Packages/Add-ins required

Software/Packages/Add-ins recommended

Description text

R code to copy/paste into console

R code to copy/paste into console that needs adjustment to your personal workspace

Websites where you can download requirements

One of my personal fears about trying to learn a program like R is that it forces you to look at data from a different perspective compared to that of the classic tabular format so distinctive of Microsoft Excel and more importantly for me MS Access. Those programs are geared to allow quick and easy manipulation of raw data to get it to the format you need. In particular Access is powerful when querying your main data tables...needless to say R is quite different. It feels like the data is a lot more "hidden and inaccessible"...but all is not lost and this lab will be one in a series of a few where we show how to look at your data and evaluate it prior to doing any proper analysis. In this lab we purely look at categorical data, so nothing numerical for now. I have taken part of the registration list of our recent SASVEPM congress held in PE as the data source and pulled a few data fields to evaluate. This is a pretty basic example of some of the functions of R...no extra packages are necessary and it wont take very long to get through. While we are going to be editing some data in R I'm certain that you'll find it easier to clean your data prior to importing it into R, and while checking it is always worthwhile before going on to evaluate it, it might be easier to edit the data in your original capture program like MS Access.

Lab #4 requirements

- R - http://cran.r-project.org/bin/windows/base/
- R Studio - www.rstudio.com/ide/download/desktop
- Internet connection

The code

```
sasvepmdata<-read.delim("http://www.jdata.co.za/backpagelabs/backpagelabs_jdg_sasvepmpreclean.txt")
#Here we import a text file of the data and assign it to a variable name called sasvepmdata
#There are a number of ways to evaluate your dataset and here we are going to go through a few that might help you set your data up lets look at the whole data set
sasvepmdata
#our data set here is pretty short (184 rows) but you can imagine that if you had a few thousand rows of data then running this command wont help you get a feel of your data, so lets be more specific
head(sasvepmdata)
#now we just see the first 6 rows of data, if you want to see the top ten then type
head(sasvepmdata,10)
#to view a nice full summary of the dataset go ahead and run this command
summary(sasvepmdata)
# this shows unique values for each column with counts of how many times each variable combination occurs
names(sasvepmdata)
#shows the column or field names of your data - also to see what unique values are in a column ("Designation" in this case) you could type
unique(sasvepmdata$Designation)
#Immediately we can see that it might be worth changing some of the column names - for instance "Designation" and "Field" are confusing - looking at the summary results above we can see what's in those fields and then change the names to something more useful so here we change the 2nd and 3rd column or field names from "Designation" to "workingfield" and "Field" to "institute" respectively
names(sasvepmdata)[c(2,3)]<-c("workingfield","institute")
#Case of letters are very NB in R so lets get all our field names to lowercase - so again - below we select the first column (square brackets) and rename it to title and later the fourth column to "location etc.
names(sasvepmdata)[1]<-"title"
names(sasvepmdata)[4]<-"location"
#"Hotel" and "Presenter" might also be confusing so lets change them as well
names(sasvepmdata)[5]<-"accomreq"
names(sasvepmdata)[6]<-"participation"
#so now we are ready with a solidly labelled dataset - have a look at it now
summary(sasvepmdata)
#lets create another column of data giving a unique number to each row of data so that if we have to edit some data we can select that row out. One quick way to do this is to use the rownames function which essentially returns the number of each row - in the following code we make a new column (callled "id") in the data set sasvepmdata...
#and assign it the value of the row that the line of data is in
sasvepmdata$id<-rownames(sasvepmdata)
#so lets check again our top couple of lines of data to see that an id field has been added
head(sasvepmdata)
```

Continued on next page

```
#you should see that your id field is sitting on the right, lets move it to the first column so that it reads easier
sasvepmdata<-sasvepmdata[,c(7,1,2,3,4,5,6)]
# what we have done above is to re-create sasvepmdata and we have made it from the original sasvepmdata where we have selected all rows (this is the blank prior to the comma in the beginning of the square bracket which refers to rows of data)
#after this we have specified the columns we want from the original data set but we have re-ordered them by creating a small vector of the column numbers and putting the 7th column first and then listed the rest of them - lets look at the result
head(sasvepmdata)
#check again what values are in each column using the summary command,
summary(sasvepmdata)
#lets look at each one to see what unique values are in each data column - do you spot the issue? - have a look at the title column - we have 2 "Dr" levels which must be a mistake - we can guess here that 2 rows have a space before or after the "Dr"
#the last part of this lab is to find the data issue and fix it, we'll continue next month with further data analysis of this set
#we create a vector variable isolating the unique values in the title column
uniquetitle<-unique(sasvepmdata$title)
uniquetitle
#our data issue culprits can be isolated by using the new unique dataset (note that the "Levels" result is just a attribute for the variable looked at - we'll discuss it at the end again)
#the two mismatched "Dr" categories are
uniquetitle[1]
uniquetitle[5]
#or you can look at both by
uniquetitle[c(1,5)]
#And lets check if they are different (we know they must be!) here we test if the 1st unique title is the same as "==" the 5th one even though both look like "Dr"
uniquetitle[1]==uniquetitle[5]
#lets ask if they are different...
uniquetitle[1]!=uniquetitle[5]
#TRUE means they are different, as expected. Now we look at the culprit incorrect data in the “title column” - using for the first time a very powerful function called subset()
subset(sasvepmdata,sasvepmdata$title==uniquetitle[5])
#it is evident that id's number 74 and 142 are the problem - lets look at how the data looks for those ids, and we select a few rows around them to compare with
sasvepmdata[70:76,]
#so here we show rows 70 through 76 and with nothing after the comma we want to view all columns - there is definitely a problem with id 74!
sasvepmdata[140:145,]
#and with 142...we get closer to sorting this out
#lets now change the data in those specific rows to "Dr" without spaces
#first id 74...and then id 142
sasvepmdata$title[sasvepmdata$id==74]<-"Dr"
#what we say here is that make the title field  of sasvepmdata equal to "Dr" for those rows of data that have an id of 74
sasvepmdata$title[sasvepmdata$id==142]<-"Dr"
#now to check that it works we re-look at what our unique values are in the title column
unique(sasvepmdata$title)
summary(sasvepmdata)
#so there is still a residual level of the wrong "Dr ", so lets drop the levels to what they should be
sasvepmdata<-droplevels(sasvepmdata)
#here we have rewritten sasvepmdata excluding all levels not remaining in dataset (so "Dr[space]” in this case)
summary(sasvepmdata)
#so thats our dataset cleaned up a bit and ready for next months analysis
```

The output

```
       id             title             workingfield            institute       location  accomreq    participation
 Length:184         Dr   :141   Veterinarian  :141   Academic        : 19   GP      :72   NO : 36   Attendee :165
 Class :character   Mr   : 27   AHT           : 32   Consultant      :  1   NW      :29   YES:148   Presenter: 19
 Mode  :character   Ms   : 12   Academic      :  3   Industry        :  2   EC      :26
                    Prof:  4   Vet Student   :  3   International   :  2   WC      :21
                               Lab technician:  2   Other           :  1   LP      :18
                               Diplomat      :  1   Private Practice:  1   KZN     : 8
                               (Other)       :  2   State           :158   (Other):10
```

References

R Core Team (2014). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL http://www.R-project.org/