```
#PO 841
#10/22/2014
#DATA MANIPULATION, CONT'D

#Today we'll be looking at two data sets; the first is a data set of all US
 Congressional hearings (in both the House and Senate) from 1946-2010. The
 data and codebook are in the Dropbox folder under the sub-folder "Lab 7
 Data."

#Load the car library, which we'll be using:

install.packages('car') #Install it if you haven't yet.
library(car)

#Read in the data file:

library(foreign)
hearings<-read.csv(file.choose())

#Examine the data:

dim(hearings) #Each column represents one variable, while each row
 represents one data point/one hearing (this command tells us that there
 have been -- approximately -- 91,656 hearings over the past 68 years of
 Congressional history)

names(hearings)

#Let's see how many observations there are for each chamber (check the
 codebook: 1=House; 2=Senate; 3=joint committee hearings, both standing and
 ad hoc):

tab1<-table(hearings$Chamber)
tab1

#Let's just work with the data from one chamber, the Senate. Let's create a
 subset with only the hearings that took place in the Senate:

senate<-hearings[hearings$Chamber==2,]
dim(senate)

#Now let's take a look at hearing topics. Say we're interested in observing
 the variation of hearing topics in the Senate from 1946-2010:
```

```
names(senate)
summary(senate$Year)
summary(senate$MajorTopic)

tab2<-table(senate$Year, senate$MajorTopic)
tab2

#Now let's say we just want to observe variations over the past 30 years:

senate$Year<-ifelse(senate$Year %in% c(1946:1979),0,senate$Year) #Here
 we're recoding all years prior to 1980 as 0 and leaving the rest intact.
 There's an easier way to just remove the years you don't want, of course,
 but let's do it this way for pedagogy's sake.
summary(senate$Year)

#Let's create a new data frame for just the year and major topic code for
 Senate hearings:

hearingtopics<-data.frame(senate$Year, senate$MajorTopic)
dim(hearingtopics)
summary(hearingtopics)

#Now let's delete all the values currently coded as "0," i.e., years before
 1980:

topics<-hearingtopics[!hearingtopics$senate.Year == 0,]
dim(topics)
summary(topics)

#Let's look at at table of the number of Senate hearings on ALL major
 topics since 1980:

tab3<-table(topics$senate.Year, topics$senate.MajorTopic)
tab3

#Let's figure out what proportion of total hearings were held on each major
 topic:
tab4<-prop.table(tab3,1)
tab4

#Now, let's visually represent variation in the proportion of Senate
 hearings that were on environmental issues from 1980-2010:

uniqueyears<-unique(sort(topics$senate.Year))
```

```
uniqueyears
propenviron<-tab4[,7]
propenviron
data<-data.frame(uniqueyears, propenviron)
plot(data, xlab="Years",ylab="Proportion of Senate Hearings that Were on
 Environmental Issues", main="Variation in Proportion of Environmental
 Hearings Over 30 Years", pch=19)
lines(data)

#*********************************************************************

#Now let's take a look at another data set -- the Afrobarometer for
 2003-2004, a survey done in 16 African countries. The data and codebook
 are on Dropbox in the subfolder Lab 7 Material.

afro<-read.dta(file.choose(),convert.factors=F)
dim(afro)
names(afro)

#Let's see how many observations there are for each country:

tab5<-table(afro$COUNTRY)
tab5

#Let's say you just wanted to work with the data from Ghana -- let's create
 a subset that only contains observations from Ghana (check codebook for
 how Ghana is coded):

ghana<-afro[afro$COUNTRY==2,]
dim(ghana)

#Let's see what the Party ID variable (from codebook: Q87A) looks like for
 Ghana:

tab6<-table(ghana$Q87A)
tab6

#From the codebook we can figure out what parties the values correspond to.
 Let's create a new, recoded variable with the following values: None, NDC
 (National Democratic Congress), NPP (New Patriotic Party), Other, and NA
 (e.g., for those that refused to answer). We can also assign names rather
 than numbers to our new variable so it's easier to interpret.

partyID<-recode(ghana$Q87A,'0="None"; 120="NPP"; 121="NDC"; 995:999=NA;
```

```
else="Other"') #Note that when we assign certain values as NA, we do NOT
place the NA within quotation marks as we do the other recoded values.
table(partyID)

#NPP is the more conservative party and NDC the more left-wing party, at
least historically. Thus, let's say you want to see how Party ID varies
with income (codebook: Q90):

table(ghana$Q90)

#Let's recode the income variable to eliminate missing values:

decile<-recode(ghana$Q90,'98:99=NA')
table(decile)

#Say we want to observe how party ID varies across each decile of income:

tab7<-table(decile,partyID)
tab7
tab8<-prop.table(tab7,1)
tab8

#The table is a bit hard to interpret with so many income deciles, so let's
just create a dummy variable for "poor" to see if there's a difference in
party affiliation for those in the lowest income decile versus those in
the upper nine:

poor<-decile %in% 0:1 #Let's create a new variable for "poor" rather than
just recoding:
table(poor)
tab9<-table(poor,partyID)
prop.table(tab9,1)

#Just by eyeballing the table, there doesn't seem to be much difference
between the party affiliation breakdown for the lowest income decile
versus the upper-nine deciles (i.e., between "poor" and "not poor"). So
income doesn't appear to explain variation in party ID in Ghana. What else
might account for these differences?

#Let's examine the variable that denotes the region of Ghana in which
people live:

table(ghana$REGION)
```

```
#In Ghana, the Volta region is traditionally known as an NDC stronghold,
 and the Ashanti region as an NPP stronghold. Let's create a variable that
 denotes whether a Ghanian lives in the Volta region, the Ashanti region or
 another region:

region<-recode(ghana$REGION, '128="Volta"; 120="Ashanti"; else="Other"')

#How does party ID vary by region?

tab10<-table(region,partyID)
prop.table(tab10,1)

#There DO appear to be significant differences here! Region appears to help
 explain party ID in Ghana.

# -------------
# USING TAPPLY()
# -------------

#We've worked with the apply(), sapply() and lapply() commands previously.
 Another one, tapply(), is often useful for working with categorical
 variables like those we just created. Say we wanted to find out the median
 decile of the income distribution for identifiers with each party in
 Ghana:

tapply(decile,partyID,median,na.rm=T) #We're telling R to apply the
 function median() to the variable decile and give us a result for each
 separate value of partyID. We see here that the median decile of income is
 2 for both the major political parties in Ghana:

#What if we wanted to know how income differs across regions?

tapply(decile,region,median,na.rm=T) #Here we see that the median decile of
 income for those in the Volta region is higher than those in the Ashanti
 region or in other regions. This suggests that people living in the Volta
 region tend to be slightly more affluent than those living in other
 regions.

median(decile, na.rm=T) #This is just a frame of reference: the median
 decile of income for the entire country is 2.

#What if we wanted to know how the variance of income (the variance is of
 course a measure of income distribution) differs across regions?
```

```
tapply(decile, region,var,na.rm=T)
```