Data Cleaning

Maria Kamenetsky

Friday, October 30, 2015

Data Cleaning

- *Read in Data*

```
#Import from CSV/tab-delimited
df <- read.csv("df.csv")
```

- *Read Out Data*

```
write.table(df,"df.csv", sep=",")
#This creates CSV, but in excel you need to move over the column nms by one cell and then drop first co
```

- *Gather Data/Create Observation Unit*
  - unit_long <- gather(unit, yr, plts, X1900:X2016)
    - This will rectangularize the dataframe to make the obs. units filled with missing where necessary, but it will be panel-shaped then
- *Substring*

```
#Syntax
substr(x, start, stop)

#Example
##this will substring _starting_ at the second element; R is indexed at 1
unitlong\$dec <- substring(unitlong\$yr,2)
```

- *Merge*

```
#Syntax
merge(x, y, by = intersect(nms(x), nms(y)),
      by.x = by, by.y = by, all = FALSE, all.x = all, all.y = all,
      sort = TRUE, suffixes = c(".x",".y"),
      incomparables = NULL, ...)

#Example 1 - Merge two dataframes; you can specify keep using/master/both by working with all/all.x/all
df <- merge(df, unit_long, allby=c("dec","unit"))
```

- *Upper/Lower Case*

```
df\$ct <- toupper(df\$ct)
#Same thing but use *tolower*
```

- *Subsetting Data by Vars and Obs*

```r
#1) Subset DataFrame by dec and Create New DF
dr83 <-subset(dyiR, dec=="1983")

#2) Same subset based on "or"
pre <- subset(df, (dec=="1983" | dec=="1996"))

#3) Subset so you don't have NA's in whole DF
df <- subset(df, !is.na(df$st))

#4) Subset so you don't have NA's based on NA's in a single var
df <- subset(df, subset= !is.na(df$md))

#5) Drop specific observations (and create new dataframe)
df <- df[!(df$z=="30038" & df$stlong=="Georgia"),]

#6) Subset - take only vars that you want
zdf <- subset(df, select=c(...select vars...))

#7) Subset so you don't have Inf
z <- subset(zdf, zdf$sD!=Inf)

#8) Take out obs that are '0'
mds <- mds[which(mds$smd!=0),]
```

- *Changing Values of Specific Obs*

```r
#Example 1: Drop Where a var ==Inf
df$prop[df$prop=="Inf"] <- 0

#Example 2: Change two values of vars based on another var
df$palNew<-NULL
df$palNew[df$plt=="1" | df$plt=="2"] <-"low"
df$palNew[df$plt=="3" | df$plt=="4"] <-"high"

#Example 3: Replace value based on two criterion
df$drDensity[df$unit=="48" & df$dec=="1983"] <- dr83[[2]]

#Example 4: Replacing with value
df$countFortune[df$Classification=="Fortune 500"] <-1
```

- *Dealing with Duplicates*

```r
#Example 1: Create subset with unduplicated data based on specific columns
sub <- df[!duplicated(df[c(1:2,8)]),]

#Example 2: Create subset with unduplicated data based on specific var nms
BA <-df[!duplicated(df[c("dec","unit","spc","ba","ct","smAgg","MedAgg","LargeAgg", "sade")]),]

#Example 3: ~duplicates drop, force in Stata
zdf <- unique(zdf) #keeps only unique; may need to create subset so this works the way you want to; roo
```

- *AggulTate Data by a Var*

```
#Example 1: Aggregate by dec on a Subset, take the mean
dyiR <- aggregatedrDensity~dec, data=subset(df, owr=="IR"), FUN=mean)

#Example 2: Aggregate Based on Two Key Vars (~by group in Stata)
subAgg <- aggregate(sub$drNum, by=list(sub$dec, sub$ct), FUN=sum)

#Example 3: Aggregate Based on Three Key Vars
aggsm <- aggregate(df$sm, by=list(df$dec, df$unit, df$spc, df$ba), FUN=sum)

#Merge Aggureagate Back Into DF Based on Key Var
df <- merge(df, aggsm, by.x=c("dec", "unit","spc","ba"), by.y=c("Group.1","Group.2","Group.3","Group.4"
df$smAgg <-df$x #rename aggregate var in new dataframe from x
df$x <-NULL #get rid of old var

#Merge on two different var nms and keep all from master
df <- merge(df, stnms, by.x="st", by.y="Abbreviation", all.x=TRUE)
```

- *tapply/lapply/sapply*

```
#By ct (ct=long and unique), Sum the area by ct
AC <- tapply(df$Area, df$ct, FUN=sum)

#Merge this summed var back into DF
df$AC <- merge(df, AC, allby="ct")

#Find means and sd by group
tapply(dfnew$trans, dfnew$trt, FUN=mean)
tapply(dfnew$trans, dfnew$trt, FUN=sd)
```

- *Dates in R*

```
df$test <- as.Date(df$Date, "%m/%d/%y")
#For 12/01/1990 format
```

- *ulTular Expressions*

```
df$yr <- as.factor(sub(".*/.*/","",df$Date))
#This takes somethinfr from form "12/01/1990" and takes everything after the second "/"
# * is a wildcard for numbers(?)
```

- *Binning a variable into equal groups and specifying levels*

```
#1) Binning
df$drGroup <- cut(df$drDensity,3)
#Be careful not to have too many groups - overspecification

#2) Specify levels
z$densityR1 <- factor(z$densityR1, levels=c("lo","med","hi"))
## should do this with all factor variables because once you run the regression, it's going to take a l

#3) Relevel levels
df <- within(df, pp <- relevel(pp, ref="Before"))
```

- *Reshape*

```
#Example 1: Reshape Long to Wide
##This uses 'reshape'
zNew <- reshape(z,
                  varying=c("var1", "var2",...),
                  v.nms="wideClassification",
                  timevar="Classification",
                  times=c("var1", "var2",...),
                  direction="long")

#Example 2: Reshape Long to Wide
##This uses 'reshape2'
mds<- melt(zNew, id.vars=c("z","stlong","Classification"),
               measure.vars = c("zmdA","zmdB","zmdC","zmdD","zmdE","zmdF",
                                "zmdG","zmdH","zmdI","zmdJ",
                                "zmdK","zmdL","zmdM","zmdN","zmdO","zmdP",
                                "zmdQ", "zmdR","zmdS","zmdT","zmdU"),
               variable.nm="md",
               value.nm="smd")
```

Exploratory Tricks (not included in other sections)

- *Plot regression fit*

```
plot(smAgg/alT ~ drGroup*plt + dec + ba*spc + unit, data= BA)
#this will create all plots on the different vars specified in regression helps see what's going on
```

- *Sort/Re-order*

```
#Example 1: from smallest to largest
s <- s[order(s$z),]

#Example 2: From largest to smallest
s <- s[order(s$z, decreasing=TRUE),]
```

- *Looking at Quantiles/Modifying Quantiles*

```
quantile(df$s)

upper.limit <- quantile(z$zs)[4] + 1.5*IQR(z$zs)
lower.limit <- quantile(z$zs)[2] - 1.5*IQR(z$zs)
```

- *Table of Top X*

```
#Find Top 20 (by sDst in this case) and print also st, rg, and ulTulation
stsperp <- round(stpop$sDst[1:20],2)
stnm <- stpop$stlong[1:20]
rg <- stpop$rg[1:20]
ulTulation <- stpop$ulT[1:20]
printme <- cbind.data.frame(stnm, stsperp, rg, ulT)
xtable(printme)
```

- *Correlation Matrix*

```r
myvars <- c("var1", "var2",...)
cordata <- z[myvars]
xtable(cor(cordata))
#Correlation matrix can only have numeric values (obvi)
```

- *Winsorizing*

```r
library(robustHD)
z$zsWin <- round(winsorize(z$zs))
stst$sDstW <- round(winsorize(stst$sDst))
```