1/26/2010 36-402/608 ADA-II H. Seltman

Breakout #5 Results

Question 1: Figure out how the code works, and what the results are telling us about the missing data in this dataset.

The "mice" multiple imputation package contains a dataset called "boys" which contains measurements of 9 variables for 748 Dutch boys. The first five are "demographics": age, height, weight, body mass index, and head circumference. The last one is region of the country. The other three are measures of the stage of puberty: genital Tanner stage, pubic hair stage, and testicular volume.

```
> library(mice)
> round( 100 * sapply(boys, function(x){mean(is.na(x))}), 1)
age  hgt  wgt  bmi   hc  gen  phb   tv  reg
0.0  2.7  0.5  2.8  6.1 67.2 67.2 69.8  0.4

sapply() takes either a vector of values or a list and applies
a function to each element, and returns the answer in as
simplified form as possible.  (The similar function, lapply(),
always returns the answer as a list with one element for each
element of the input.)  Here the data.frame boys is interpreted
as the list of it's columns, and the function is applied
per-column.  The (anonymous) function I define is to
take its input, in this case a column of data temporarily
referred to as ``x'', and return the mean number of
missing values.

Everyone has age recorded, many people have gen, phb, and tv
missing, and the other variables are missing at low frequency.


> round( 100 * apply(boys, 2, function(x){mean(is.na(x))}), 1)
age  hgt  wgt  bmi   hc  gen  phb   tv  reg
0.0  2.7  0.5  2.8  6.1 67.2 67.2 69.8  0.4

apply() takes a matrix or data.frame and analyzes each row
(if the second argument is 1) or column (if the second
argument is 2).  So we get the same results as above.


> table( apply(boys, 1, function(x){sum(is.na(x))}) )
```

```
  0   1   2   3   4   5   6   7
223  20   1 438  47  17   1   1

The apply() here returns a vector with 748 values for
the 748 boys.  To allow reasonable interpretation,
this vector is used as the argument to table() to
count how many boys are missing various numbers
of data values.

> boys[1,]   # Data for the first boy
   age  hgt  wgt   bmi   hc  gen  phb tv   reg
 0.035 50.1 3.65 14.54 33.7 <NA> <NA> NA south

> is.na(boys[1,])  # Which are missing for boy 1?
   age   hgt   wgt   bmi    hc  gen  phb   tv   reg
 FALSE FALSE FALSE FALSE FALSE TRUE TRUE TRUE FALSE

> 1 + is.na(boys[1,]) # Compute missing=2, present=1
 age hgt wgt bmi hc gen phb tv reg
   1   1   1   1  1   2   2  2   1

> c("P","M")[1+is.na(boys[1,])] # Select P or M for each variable
                                # for boy 1
[1] "P" "P" "P" "P" "P" "M" "M" "M" "P"

# Collapse the P's and M's for boy 1
> paste( c("P","M")[1+is.na(boys[1,])], collapse="")
[1] "PPPPPPMMMP"

# Show the collapsed ``word'' for the first 5 boys:
> apply(boys[1:5,], 1, function(x) {
        paste(c("P","M")[1+is.na(x)], collapse="")})
[1] "PPPPPPMMMP" "PPPPPPMMMP" "PPPPPPMMMP" "PPPPPPMMMP" "PPPPPPMMMP"

# Store the collapsed ``word'' for all of the boys:
> patterns = apply(boys, 1, function(x) {
        paste(c("P","M")[1+is.na(x)], collapse="")})
# Count how many boys have each pattern of missingness:
> table(patterns)
 PMMMMMMMP PMMMMPPPP PMMMPPPPP PMPMMMMMP PMPMPMMMP PPMMPMMMP PPPPMMMMP
         1         1         1         1        16         1        43
 PPPPPMMMM PPPPPMMMP PPPPPMPMP PPPPPPMPP PPPPPPPMP PPPPPPPPP
```

```
         3       437         1         1        19       223

# Count how many boys are missing variables 6 through 8:
> table(substring(patterns,6,8)=="MMM")
FALSE  TRUE
  246   502

# aggregate() has three arguments: The first, a vector or matrix, is
# what is analyzed (per column for a matrix).  The analysis is to
# apply the function that is the third argument, e.g., mean(), length(),
# or a user defined (often anonymous) function.  The second argument,
# which must be a list, contains one or more vectors of the same length
# as the first argument, and each unique value (or set of values for
# a list with more than one element) determines a subset of the
# first argument on which the function is applied.

# Test case:
> aggregate(boys[,2:3], list(under12=boys$age<12, region=boys$reg), mean)
   under12 region      hgt      wgt
1    FALSE  north 179.7451 68.44314
2     TRUE  north       NA 18.25133
3    FALSE   east 174.7308 61.95077
4     TRUE   east       NA 20.02818
5    FALSE   west       NA       NA
6     TRUE   west       NA       NA
7    FALSE  south 175.2263 61.96579
8     TRUE  south       NA 16.33717
9    FALSE   city 170.8655 58.75172
10    TRUE   city       NA 17.67207

Here we divide the boys into (unevenly sized) groups by whether or not
they are under 12 years old and by what region they are from.  The
mean is computed for each group.  Even better is:

aggregate(boys[,2:3], list(under12=boys$age<12, region=boys$reg), mean, na.rm=TRUE)
   under12 region       hgt      wgt
1    FALSE  north 179.74510 68.44314
2     TRUE  north 100.42500 18.25133
3    FALSE   east 174.73077 61.95077
4     TRUE   east 105.77553 20.02818
5    FALSE   west 175.13107 62.52524
6     TRUE   west  93.90157 16.03705
```

```
7    FALSE  south 175.22632 61.96579
8     TRUE  south  95.37455 16.33717
9    FALSE   city 170.86552 58.75172
10    TRUE   city  94.78095 17.67207

> aggregate(boys[,1:5], list(miss3=substring(patterns,6,8)=="MMM"), mean, na.rm=TRUE)
  miss3       age      hgt      wgt      bmi       hc
1 FALSE 14.016533 164.9537 53.64549 19.06549 55.29592
2  TRUE  6.778416 115.6153 29.10494 17.56493 49.47418
```

On average, boys with missing data are younger and smaller than those with no missing data.

```
> table(round(boys$age))
 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21
91 87 59 35 19  5 10  8  9 23 28 30 28 38 48 56 50 32 29 36 23  4

> agecut = cut(boys$age, seq(0,24,4))
> table(agecut)
  (0,4]   (4,8]  (8,12] (12,16] (16,20] (20,24]
    286      37     100     178     135      12
```

I divided the ages into arbitrary groups.

```
> aggregate(substring(patterns,6,8)=="MMM", list(ages=agecut), mean)
     ages         x
1   (0,4] 1.0000000
2   (4,8] 0.9729730
3  (8,12] 0.2200000
4 (12,16] 0.4438202
5 (16,20] 0.5259259
6 (20,24] 0.6666667
```

The puberty questions are mostly skipped below age 8 or above 16.

Here is a more detailed breakdown:

```
> aggregate(substring(patterns,6,8)=="MMM", list(ages=cut(boys$age,8:16)), mean)
     ages         x
1   (8,9] 0.4285714
2  (9,10] 0.1724138
3 (10,11] 0.2333333
```

```
4 (11,12] 0.2058824
5 (12,13] 0.2962963
6 (13,14] 0.4468085
7 (14,15] 0.4693878
8 (15,16] 0.4909091
```

Question 2: Figure out how the code works, and what the results are telling us about predicting body mass index from age, hypertension, and cholesterol.

```
> round( 100 * apply(nhanes, 2, function(x){mean(is.na(x))}), 1)
age bmi hyp chl
  0  36  32  40

Age is complete, but the other variables have 32-40% missing data.

> table( apply(nhanes, 1, function(x){sum(is.na(x))}) )
 0  1  2  3
13  4  1  7

# Seven subjects have only age recorded.

> patterns = apply(nhanes, 1, function(x) {
         paste(c("P","M")[1+is.na(x)], collapse="")})
> table(patterns)
PMMM PMMP PMPP PPPM PPPP
   7    1    1    3   13
The missingess pattern shows that the subjects with
1 missing value are missing bmi or chl, and the one
with two missing values is missing bmi and hyp.

# This creates 5 fill-in datasets with appropriate
# variability in the filled-in data across the 5 datasets:
> nhanes5 = mice(nhanes, 5, printFlag=FALSE)
# This applies linear regression to each of the 5 datasets:
> nhanes5lm = with(nhanes5, lm(bmi ~ age+hyp+chl))
> summary(nhanes5lm)
 ## summary of imputation 1 :
...
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 19.50020    3.20307   6.088 4.85e-06 ***
```

```
age         -3.65876    0.78889  -4.638 0.000142 ***
hyp          0.24298    1.80267   0.135 0.894061
chl          0.07387    0.01700   4.345 0.000285 ***
...
 ## summary of imputation 2 :
...
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 20.25075    3.16735   6.394 2.45e-06 ***
age         -3.69148    0.81501  -4.529 0.000183 ***
hyp          0.62913    1.82420   0.345 0.733614
chl          0.06560    0.01828   3.589 0.001729 **
...

This calculates the best single estimate from combining
the five estimates:
> nhanes5pool = pool(nhanes5lm)
> summary(nhanes5pool)
                    est         se          t        df    Pr(>|t|)
(Intercept) 19.80287222 3.45264062  5.7355730 12.474896 0.000080651
age         -3.84997696 0.98008080 -3.9282240 11.066091 0.002332770
hyp          0.69663077 2.12866914  0.3272612  9.572933 0.750513844
chl          0.06891952 0.01835404  3.7550043 15.430643 0.001828751
                  lo 95      hi 95 missing       fmi
(Intercept)  12.31185362 27.2938908      NA 0.2682756
age          -6.00554946 -1.6944045       0 0.3137279
hyp          -4.07517784  5.4684394       8 0.3675757
chl           0.02989367  0.1079454      10 0.1825114
```

The final estimates suggest that younger age and higher cholesterol predict a higher bmi. Hypertension is not statistically significant. The inverse effect on age is surprising until you realize that the subjects are aged 1 to 3 and the results reflect the loss of baby fat with aging.

```
# Where the above comes from:
> nhanes5pool$qbar
(Intercept)         age         hyp         chl
19.80287222 -3.84997696  0.69663077  0.06891952

# Estimates of coefficient have a variance (uncertainty)
# shown on the diagonal, and correlation between
# pairs of estimates represented by covariances
```

```
# on the off diagonal of this variance-covariance
# matrix:
> round(nhanes5pool$t,4)
            (Intercept)     age     hyp     chl
(Intercept)     11.9207  0.8997 -2.9507 -0.0505
age              0.8997  0.9606 -1.1413 -0.0065
hyp             -2.9507 -1.1413  4.5312 -0.0020
chl             -0.0505 -0.0065 -0.0020  0.0003

> sqrt(diag(nhanes5pool$t))
(Intercept)         age         hyp         chl
 3.45264062  0.98008080  2.12866914  0.01835404

> nhanes5pool$df
(Intercept)         age         hyp         chl
  12.474896   11.066091    9.572933   15.430643
```