A *new* recommended way of dealing with multiple missing values: Using missForest for all your imputation needs.

As published in Benchmarks RSS Matters, July 2014

http://web3.unt.edu/benchmarks/issues/2014/07/rss-matters

Jon Starkweather, PhD

Jon Starkweather, PhD
jonathan.starkweather@unt.edu
Consultant
Research and **S**tatistical **S**upport

http://www.unt.edu

http://www.unt.edu/rss

RSS hosts a number of “Short Courses”.
A list of them is available at:
http://www.unt.edu/rss/Instructional.htm

Those interested in learning more about R, or how to use it, can find information here:
http://www.unt.edu/rss/class/Jon/R_SC

A *new* recommended way of dealing with multiple missing values: Using missForest for all your imputation needs.

A couple of months ago we provided an article tutorial for using the 'rrp' package for multiple missing value imputation. The 'rrp' package has consistently been our most recommended tool for dealing with missing values. However, the 'rrp' package has not been updated (i.e. adapted) to new versions of R since the release of R-2.15.1 (over a year ago). This has presented challenges to its utility – basically necessitating the install of R-2.15.0 in order to use the 'rrp' package. Very recently it was discovered that the 'rrp' package is no longer available (even from R-forge) for any Windows install of R. This prompted us to find a new *go-to* package for missing value imputation.

The good news is this: we have now found a satisfactory replacement for the beloved 'rrp' package. The 'missForest' package (Stekhoven, 2013, Stekhoven, 2012) provides not only a function for conducting multiple imputation of mixed data (numeric and factor variables in one data frame), but it also has a utility to parallelize the process of doing such imputations. Below we offer a quick example of how to use the function with a simple data set. Please keep in mind, the function is not terribly fast and when applied to large data sets it may take a considerable amount of time to complete the imputations (even when using the parallelize argument).

First, import some (simulated) example data. Notice we are importing the same data set twice; one version with no missing values and one version with missing values (Missing Completely At Random [MCAR]). Note the data files can be imported directly from the RSS URLs provided (i.e. simply copy the script and paste into your R console to follow along).

```
no.miss <- read.table(
     "http://www.unt.edu/rss/class/Jon/R_SC/Module4/missForest_noMiss.txt",
     header=TRUE, sep=",", na.strings="NA", dec=".", strip.white=TRUE)
wi.miss <- read.table(
     "http://www.unt.edu/rss/class/Jon/R_SC/Module4/missForest_Miss.txt",
     header=TRUE, sep=",", na.strings="NA", dec=".", strip.white=TRUE)
```

Next, we may want to take a look at the proportion of missing values (cells) which are present in the data file (i.e. number of missing cells divided by the product of the number of rows multiplied by number of columns). Below, we see only 4.37% (1181) of the total cells (6 columns * 4500 rows = 27000 cells) are missing values (i.e. 1181 / 27000 = .0437).

```
ncol(wi.miss); nrow(wi.miss)
[1] 6
[1] 4500
length(which(is.na(wi.miss) == "TRUE")) / (nrow(wi.miss)*ncol(wi.miss))
[1] 0.04374074
```

Next, we need to load the required package (missForest) and its dependencies (i.e. randomForest, foreach, itertools, & iterators).

```
library(missForest)
Loading required package: randomForest
randomForest 4.6-7
Type rfNews() to see new features/changes/bug fixes.
Loading required package: foreach
foreach: simple, scalable parallel programming from Revolution Analytics
Use Revolution R for scalability, fault tolerance and more.
http://www.revolutionanalytics.com
Loading required package: itertools
Loading required package: iterators
```

Apply the 'missForest' function with all arguments set to default values. The function returns a list object with 3 elements: "ximp" which is the imputed data, "OOBerror" which is the estimated (out of bag) imputation error, and "error" which is the true imputation error (the "error" is only returned when an 'xtrue' value is provided). Please note: the function **does** accept a data frame; the package documentation states that the data must be in a matrix (all numeric); however that is not the case.

```
im.out.1 <- missForest(xmis = wi.miss, maxiter = 10, ntree = 100,
            variablewise = FALSE,
            decreasing = FALSE, verbose = FALSE,
            mtry = floor(sqrt(ncol(wi.miss))), replace = TRUE,
            classwt = NULL, cutoff = NULL, strata = NULL,
            sampsize = NULL, nodesize = NULL, maxnodes = NULL,
            xtrue = NA, parallelize = "no")
  missForest iteration 1 in progress...done!
  missForest iteration 2 in progress...done!
  missForest iteration 3 in progress...done!
  missForest iteration 4 in progress...done!
  missForest iteration 5 in progress...done!
  missForest iteration 6 in progress...done!
  missForest iteration 7 in progress...done!
  missForest iteration 8 in progress...done!
  missForest iteration 9 in progress...done!
  missForest iteration 10 in progress...done!
```

To extract only the imputed data from the output (list), we use the familiar "$" operator to index the output object and retrieve the 'ximp' data frame. We can then compare the summaries of the original (no missing) data to the missing data and the imputed data.

```
im.miss.1 <- im.out.1$ximp
summary(no.miss)
       id          region             city.names      gender
 Min.   :    858   I  :1713   New Jork     : 457   female:2241
 1st Qu.: 245659   II :1167   Los Angelinas: 438   male  :2259
 Median : 499423   III:1620   San Francis  : 393
 Mean   : 501929              Bahston      : 356
 3rd Qu.: 758180              Astin        : 352
```

```
 Max.   :1012027              Carlot       : 346
                              (Other)      :2158
      age          education
 Min.   :18.00   Min.   : 2.00
 1st Qu.:29.00   1st Qu.: 9.00
 Median :33.00   Median :11.00
 Mean   :34.69   Mean   :11.12
 3rd Qu.:39.00   3rd Qu.:13.00
 Max.   :82.00   Max.   :22.00
summary(wi.miss)
       id           region             city.names      gender
 Min.   :    858   I   :1627   New Jork     : 434   female:2119
 1st Qu.: 245659   II  :1090   Los Angelinas: 415   male  :2143
 Median : 499423   III :1541   San Francis  : 373   NA's  : 238
 Mean   : 501929   NA's: 242   Bahston      : 334
 3rd Qu.: 758180               Astin        : 332
 Max.   :1012027               (Other)      :2387
                               NA's         : 225
      age          education
 Min.   :18.00   Min.   : 2.00
 1st Qu.:29.00   1st Qu.: 9.00
 Median :33.00   Median :11.00
 Mean   :34.66   Mean   :11.12
 3rd Qu.:39.00   3rd Qu.:13.00
 Max.   :80.00   Max.   :22.00
 NA's   :234     NA's   :242
summary(im.miss.1)
       id           region             city.names      gender
 Min.   :    858   I  :1713   New Jork     : 457   female:2239
 1st Qu.: 245659   II :1167   Los Angelinas: 438   male  :2261
 Median : 499423   III:1620   San Francis  : 393
 Mean   : 501929              Bahston      : 356
 3rd Qu.: 758180              Astin        : 352
 Max.   :1012027              Carlot       : 346
                              (Other)      :2158
      age          education
 Min.   :18.00   Min.   : 2.00
 1st Qu.:29.00   1st Qu.: 9.00
 Median :34.00   Median :11.00
 Mean   :34.67   Mean   :11.11
 3rd Qu.:39.00   3rd Qu.:13.00
 Max.   :80.00   Max.   :22.00
```

The OOBerror rates are returned as two statistics; the first number returned is the normalized root mean squared error (NRMSE; for continuous variables) and the second is the proportion falsely classified (PFC; for categorical variables). The OOBerror rates can be retrieved by using the familiar "$" from the output object. Others (Waljee, et al.; 2013) have compared the 'missForest' function to other imputation

methods and found “it [missForest] had the least imputation error for both continuous and categorical variables ... and it had the smallest prediction difference [error]...” (p.1).

```
im.out.1$OOBerror
       NRMSE          PFC
0.0000187039 0.1652550716
```

One of the major benefits of the ‘missForest’ function is that it has an argument for utilizing multiple cores (i.e. processors) of a computer in *parallel*. Below we repeat the example from above showing how to exploit this functionality. Keep in mind, the larger the data set, the greater the benefit achieved by parallelizing the imputation. First, we need to load the ‘doParallel’ package and its dependency (i.e. the ‘parallel’ package).

```
library(doParallel)
Loading required package: parallel
```

Next, we need to register the number of cores (or processors) of our computer.

```
registerDoParallel(cores = 2)
```

Now we can apply the ‘missForest’ function while breaking the work down into equal numbers of ‘variables’ or ‘forests’ for each core to work on (here we break the number of variables).

```
im.out.2 <- missForest(xmis = wi.miss, maxiter = 10, ntree = 100,
            variablewise = FALSE,
            decreasing = FALSE, verbose = FALSE,
            mtry = floor(sqrt(ncol(wi.miss))), replace = TRUE,
            classwt = NULL, cutoff = NULL, strata = NULL,
            sampsize = NULL, nodesize = NULL, maxnodes = NULL,
            xtrue = NA, parallelize = "variables")
  missForest iteration 1 in progress...done!
  missForest iteration 2 in progress...done!
  missForest iteration 3 in progress...done!
  missForest iteration 4 in progress...done!
```

Again, extract only the imputed data from the output (list) using the familiar “$” operator to index the ‘ximp’ data frame. We can then compare the summaries of the original (no missing) data to the missing data and the imputed data.

```
im.miss.2 <- im.out.2$ximp
summary(no.miss)
       id          region             city.names      gender
 Min.   :    858   I  :1713   New Jork     : 457   female:2241
 1st Qu.: 245659   II :1167   Los Angelinas: 438   male  :2259
 Median : 499423   III:1620   San Francis  : 393
 Mean   : 501929              Bahston      : 356
 3rd Qu.: 758180              Astin        : 352
```

```
 Max.   :1012027             Carlot       : 346
                             (Other)      :2158
      age          education
 Min.   :18.00   Min.   : 2.00
 1st Qu.:29.00   1st Qu.: 9.00
 Median :33.00   Median :11.00
 Mean   :34.69   Mean   :11.12
 3rd Qu.:39.00   3rd Qu.:13.00
 Max.   :82.00   Max.   :22.00
summary(wi.miss)
       id           region             city.names      gender
 Min.   :    858   I   :1627   New Jork     : 434   female:2119
 1st Qu.: 245659   II  :1090   Los Angelinas: 415   male  :2143
 Median : 499423   III :1541   San Francis  : 373   NA's  : 238
 Mean   : 501929   NA's: 242   Bahston      : 334
 3rd Qu.: 758180               Astin        : 332
 Max.   :1012027               (Other)      :2387
                               NA's         : 225
      age          education
 Min.   :18.00   Min.   : 2.00
 1st Qu.:29.00   1st Qu.: 9.00
 Median :33.00   Median :11.00
 Mean   :34.66   Mean   :11.12
 3rd Qu.:39.00   3rd Qu.:13.00
 Max.   :80.00   Max.   :22.00
 NA's   :234     NA's   :242
summary(im.miss.2)
       id           region             city.names      gender
 Min.   :    858   I  :1713   New Jork     : 457   female:2244
 1st Qu.: 245659   II :1167   Los Angelinas: 438   male  :2256
 Median : 499423   III:1620   San Francis  : 393
 Mean   : 501929              Bahston      : 356
 3rd Qu.: 758180              Astin        : 352
 Max.   :1012027              Carlot       : 346
                              (Other)      :2158
      age          education
 Min.   :18.00   Min.   : 2.00
 1st Qu.:29.00   1st Qu.: 9.00
 Median :34.00   Median :11.00
 Mean   :34.67   Mean   :11.12
 3rd Qu.:39.00   3rd Qu.:13.00
 Max.   :80.00   Max.   :22.00
```

1 Conclusions

RSS has previously been recommending the use of the 'rrp' package for multiple missing value imputation, primarily because unlike alternatives, the 'rrp.impute' function could accept and impute categorical variables as well as continuous (or nearly continuous) variables. However, the 'rrp' package has not been consistently maintained since the release of R-2.15.1. The 'missForest' package, which recently came to our attention, offers the same benefit and is available for the most recent release of R (R-3.1.0). Furthermore, the 'missForest' function contains the added benefit of parallelization for larger data sets. It is for these reasons (i.e. benefits) which RSS recommends its usage in any missing value situation. For those interested in learning more about what R can do; please visit the Research and Statistical Support *Do-It-Yourself Introduction to R*[1] course page.

Until next time; document everything, audio is good but audio with video is better.

[1] `http://www.unt.edu/rss/class/Jon/R_SC/`

2 References & Resources

Little, R. J. A., & Rubin, D. B. (1985). *Statistical Analysis with Missing Data*. New York:John Wiley & Sons.

Stekhoven, D., J. (2013). Package missForest: Nonparametric missing value imputation using random forest. Package documentation available at CRAN: http://cran.r-project.org/web/packages/missForest/index.html

Stekhoven, D., J. (2012). MissForest – Non-parametric missing value imputation for mixed-type data. *Bioinformatics, 28*(1), 112 - 118.

Waljee, A. K., Mukherjee, A., Singal, A. G., Zhang, Y., Warren, J., Balis, U., Marrero, J., Zhu, J., & Higgins, P. DR. (2013). Comparison of imputation methods for missing laboratory data in medicine. *BMJ Open, 3*(8), 1 - 7. DOI: 10.1136/1136-bmjopen-2013-002847

This article was last updated on June 23, 2014.

This document was created using LaTeX