Missing Data and Imputations

Irfan Kanat

July 5, 2017

In this learning activity we will learn about various ways of handling missing data.

Introducing the Data

Here we will use the dataset from previous learning activity (Combining Datasets). It is a mix of UN and WB data on number of internet users and number of servers.

```
itInfrastructure <- read.csv("data/itInfrastructure.csv")
head(itInfrastructure)
```

```
##   iso3 Year        name iso2     Users      Servers
## 1  ABW 2006       Aruba   AW 28.000000 357.03659625
## 2  ABW 2007       Aruba   AW 30.900000 385.30696121
## 3  ADO 2006        <NA> <NA> 48.936847 419.80017512
## 4  ADO 2007        <NA> <NA> 70.870000 435.91979076
## 5  AFG 2006 Afghanistan   AF  2.107124   0.07941672
## 6  AFG 2007 Afghanistan   AF  1.900000   0.23186126
```

Missing Data

Handling missing data can be as simple as ignoring it, and there are many occasions where ignoring missing data is absolutely justified. But the correct approach depends on reasons of missingness and what kind of model/question you have.

Let us talk about reasons of missingness, briefly:

Missing Completely at Random (MCAR) There is no pattern to missingness. The missingness, does not result variable itself or other variables we can control for. An example would be interference from passing radio sources on sensor equipment. Car radios are not predictable and they are not related to what the sensors are measuring, or anything we are modelling.

Missing at Random (MAR) Missing values are not a result of variable itself but can be related to levels of other variables. An example would be school attendance over the years. You are more likely to not have observations for children from low income families. Their missingness from the data is a factor of a variable you can observe.

Not Missing at Random (NMAR) The missing value is missing due to the level of the value. An example may be income level. High income groups are less likely to report their income on surveys.

In all these examples, we try to see if the missing data is result of a systematic mechanism. If the children of low income families are not represented in certain parts of our data, do we get reliable results? Can we generalize from a dataset that is missing observations from certain segments of population? Depends on the goal of the study of course, but in all likelihood your results will be biased.

If you have data missing completely at random, you can just ignore it. Drop the observations with missing data and carry on as usual.

If your data is missing at random, you need to make a call. What is the best way to reduce the bias? Remove observations where missingness occurs? Impute some value into those missing observations?

If your data is not missing at random, the problem is a bit more serious. Some methods used in MAR can help. You may also look at pre-processing the data to obtain a balanced sample with an approach like Coarsened Exact Matching.

Another consideration is where the data is missing. It is usually not that big of a deal if the data is missing in independent variables. But if you are missing the variable you are trying to estimate (dependent variable) that is a problem. This is somewhat related to self selection bias. For example, people who buy optional insurance are generally risk averse. In all likelihood, you won't observe lots of risky people in indentity theft insurance data. Can you make generalizations on the broader public from a dataset of risk averse people?

Inspect Your Data for Missingness

A simple inspection of your data can tell you something, let us use the summary function.

```r
summary(itInfrastructure)
```

```
##        iso3          Year                              name          iso2    
##  ABW    :  2   Min.   :2006   Afghanistan        :  2   AE     :  2  
##  ADO    :  2   1st Qu.:2006   Albania            :  2   AF     :  2  
##  AFG    :  2   Median :2006   Algeria            :  2   AG     :  2  
##  AGO    :  2   Mean   :2006   Angola             :  2   AL     :  2  
##  ALB    :  2   3rd Qu.:2007   Antigua and Barbuda:  2   AM     :  2  
##  ARE    :  2   Max.   :2007   (Other)            :392   (Other):390  
##  (Other):422                  NA's               : 32   NA's   : 34  
##      Users          Servers        
##  Min.   : 0.00   Min.   :   0.0124  
##  1st Qu.: 4.29   1st Qu.:   0.7826  
##  Median :16.00   Median :   9.4600  
##  Mean   :24.64   Mean   : 145.1513  
##  3rd Qu.:40.27   3rd Qu.:  93.7699  
##  Max.   :90.60   Max.   :2145.0143  
##  NA's   :33      NA's   :62
```

Under some variables you can see NA's reported. NA is how R reports missingness.

You can see we have some missingness in country name and iso2 numbers. Those don't concern me, basically this is a matter of coding. An inspection reveals missing values are from microstates like Andorra or territories like American Samoa. Knowing how the data is created, most probably these were not included in codes dataset, and can still be used as names or iso2 are not variables we can not do without (iso3 is perfect substitute for both).

The real missingness, we should be worried about is in Users and Servers. Especially on Servers there is a lot more missingness. Simply looking at number of missing observations in a column will not however tell you if the missingness is due to some common factor or not. It won't tell you of the mechanism.

A more involved way of looking at missingness is to use a command like md.pattern() from mice package.

```r
# If this doesn't work try to guess why it doesn't work.
md.pattern(itInfrastructure[, 5:6])
```

```
##     Users Servers   
## 365     1       1  0
##   7     0       1  1
##  36     1       0  1
##  26     0       0  2
##        33      62 95
```

The results mean 365 observations have complete data for both Users and iso2. 7 observations are just missing users, and 36 observations are missing just the servers. 26 observations are missing both variables.

26/33 missing values from users are also missing from servers. This may mean the missingness is due to a common factor.

At this stage, you need to go out and and think about where data comes from, how it is collected, how it is put together. An inspection of data, reveals that the missingness is focused on developing/third world countries. The data collection in these places were not as rigorous and created patterns of missingness. This means there is a systematic pattern to missingness.

Now you know a thing or two about investigating missingness.

Deletion

Deleting the missing observations is the simplest thing you can do. In fact, many modelling functions automatically remove observations with missing values. Inspect the lm() manual page.

```
?lm
```

See that bit about na.action, it says the default is na.omit. Leaving it to lm() function has the added benefit of not discarding observations for variables not included in your model.

If you are dead set on manually handling missing data you can use one of the various commands from na.*() function family. I often use na.exclude().

```
itInfrastructureNARm <- na.exclude(itInfrastructure)
```

Above command will remove all rows with missing observations. Note the effect on sample size. itInfrastructure had 434 observations. itInfrastructureNARm has 349 rows. na.rm() removed 85 rows.

We can rein in the removal of rows by limiting the activity to just the columns we will use in models. As discussed above, there is no point in removing a row just because it's missing country name.

There are a number of ways of doing this. Since you are familiar with filtering, I will demonstrate filtering.

```
itInfrastructureNARm1 <- itInfrastructure[!(is.na(itInfrastructure$Users) &
                                             is.na(itInfrastructure$Servers)), ]
nrow(itInfrastructureNARm1)
```

```
## [1] 408
```

That simple trick saved us over 50 observations.

The advantage of this approach is that it is simple and is pretty much the standard approach. In regression models, simply deleting rows does not bias results as long as data is MCAR or MAR.

The disadvantage is the impact on sample size. If your sample size is small, it may reduce the statistical power of your analysis. Nowadays having too much data is a bigger problem than not having enough. So, in all likelihood you should be fine by just deleting observations as long as you have over a couple hundred observations.

Simple Imputations

Another approach is to write in a reasonable value instead of missing observations. People regularly use average value of the variable.

Let us see how this can be achieved. We will use two techniques you are already familiar with; (1) filtering and (2) assignment. What we want to do is select the rows where there is missingness in a column and assign a certain value into that column. Here is how:

```r
# I dont want to overwrite the original dataset, so I create a new one to work on
itInfrastructureImpute <- itInfrastructure

itInfrastructureImpute[is.na(itInfrastructureImpute$Users), "Users"] <-
  mean(itInfrastructureImpute$Users, na.rm = T)
summary(itInfrastructureImpute)
```

```
##       iso3          Year                       name          iso2    
##  ABW    :  2   Min.   :2006   Afghanistan        :  2   AE     :  2  
##  ADO    :  2   1st Qu.:2006   Albania            :  2   AF     :  2  
##  AFG    :  2   Median :2006   Algeria            :  2   AG     :  2  
##  AGO    :  2   Mean   :2006   Angola             :  2   AL     :  2  
##  ALB    :  2   3rd Qu.:2007   Antigua and Barbuda:  2   AM     :  2  
##  ARE    :  2   Max.   :2007   (Other)            :392   (Other):390  
##  (Other):422                  NA's               : 32   NA's   : 34  
##      Users           Servers        
##  Min.   : 0.000   Min.   :   0.0124  
##  1st Qu.: 4.754   1st Qu.:   0.7826  
##  Median :19.100   Median :   9.4600  
##  Mean   :24.640   Mean   : 145.1513  
##  3rd Qu.:36.200   3rd Qu.:  93.7699  
##  Max.   :90.600   Max.   :2145.0143  
##                   NA's   :62
```

Let us see what is going on here:

`itInfrastructureImpute[is.na(itInfrastructureImpute$Users), "Users"] <- mean(itInfrastructureImpute$Use`

I will break this code into 3 parts, so you can follow easily.

First is the index *([is.na(itInfrastructureImpute$Users), "Users"])* we use to summon rows from itInfrastructureImpute. I want you to understand what is going on clearly.

1. is.na(): this is a function that returns logical evaluation of our variable vector. If an observation is missing, there will be a TRUE, and if it is not missing there will be a FALSE. Observe:

```r
is.na(c(1, NA, 3))
```

```
## [1] FALSE  TRUE FALSE
```

When you pass an array of true/false values into the index of a data.frame you get only rows that have TRUE. In case of is.na() those are the rows with observations missing.

2. then I specify the column I am interested in *("Users")*, I want to replace NA's in this column with mean value.

Second part is the data.frame and the assignment operator *(itInfrastructureImpute[is.na(itInfrastructureImpute$Users), "Users"] <-)*, we know what the index will do. It will bring us all observations from Users column that has missing values. With the assignment operator we are writing over those values.

Third and final part is the mean() function. I ask R to calculate the mean value of Users so it can be used to overwrite missing values. Since there are missing values I needed to use na.rm=T parameter. This instructs mean to calculate the mean, disregarding missing observations (otherwise mean function will fail).

The advantage of this method is that it preserves the sample size, you don't discard data. Disadvantage is that it reduces variance, and may result in biased results. You have to have a justification for imputing data and how you impute it.

More Advanced Techniques

What I have covered so far should get you through 99% of the analysis tasks you will face. I just want to discuss some advanced approaches to missing values.

If your variables are related, that is you can estimate value of one variable from other variables, you can fit a simple model (like multiple regression) to estimate the missing values.

In more advanced cases, you create multiple datasets with such models. This gives you a range of estimates and does not unnecessarily reduce the variance. This technique is called multiple imputations. In R there is a great package called Amelia II for multiple imputations. While it is beyond the scope of this course, I highly encourage you to explore it if you ever find yourselves imputing a large amount of data.

Another approach is to pre-process data to balance out the effect of NMAR. Coarsened Exact Matching (CEM) can be used to pre-process dataset to have a more balanced dataset, thus reducing bias due to model. While the original purpose of CEM is not to address missingness, it can still be used to get unbiased results.