Week 6 - Class Worksheet

Scan: Missing Values

Dr. Anil Dolgun

Last updated: 18 June, 2020

Required Packages

The following packages will be required or may come in handy.

```
library(readr)
library(dplyr)
library(readxl)
library(gdata)
library(rvest)
library(tidyr)
library(knitr)
library(deductive)
library(validate)
library(Hmisc)
library(stringr)
```

Exercises

Private Consumption Data

The following exercises (exercise 1-3) will be based on Private Consumption Data, pr.RDS (../data/pr.RDS) located at http://www.oecd-ilibrary.org/economics/data/main-economic-indicators_mei-data-en (http://www.oecd-ilibrary.org/economics/data/main-economic-indicators_mei-data-en) containing 44 observations of countries' seasonally adjusted private consumption which is one of the main economic indicators. Variables are self explanatory however it is expected to do checks on the type of the data and using the suitable transformations if necessary.

1 Identify NAs in full data frame and for each column, use `print()` to see the results. Find out the location of NAs in each column using the appropriate function. Identify the count of NAs in each column using `sum()` , then use `colSums()` for the same task.

2 Create two data frames using `complete.cases()` and subsetting with `!` operator to get incomplete cases respectively. Use `na.omit()` to get complete cases. Find out the country that has NAs for every column using `rowSums()` nested with `is.na()`

then remove that country from the data frame with the same method with reversing the calculation.

3 Recode the missing values using the mean values for quarterly values for 2017 with `rowMeans()` , use `na.rm=TRUE` argument. To complete this, simply use the mean values of the quarterly columns for 2017, recode the NA values using `ifelse()` function. When you complete recoding the missing values, check for the existence of `NaN` values within the data frame, refer to the lecture notes for `is.notanumber()` function given below (using `na.rm=TRUE` will create `NaN` values).

```
is.notanumber <- function(x){ if (is.numeric(x)) is.nan(x) }
```

Population by Country Time Series Data

The following exercises (exercise 4-6) will be based on Population by Country Time Series Data, popbycountry.csv (../data/popbycountry.csv) sourced from U.S Department of Energy, located at https://openei.org/doe-opendata/dataset/population-by-country-1980-2010 (https://openei.org/doe-opendata/dataset/population-by-country-1980-2010) containing 232 observations of countries' population in millions per year from 1980 to 2010. Variables are self explanatory however it is expected to do checks on the type of the data and using the suitable transformations if necessary.

4 Investigate the `NA` values in the `popbycountry.csv` data set. Have you noticed `--` values? Replace them with `NA` . Remove the countries if they have NAs for each column. Now identify the NA values. Don't forget to check types of data using `str()` or `typeof()` functions, make appropriate adjustments.

5 Use `str_detect()` function from `stringr` package to `dplyr::filter()` strings that has `Germany` in it and save this as a data frame. Replace the NA values in `Germany` row with the column sums of `Germany, East` and `Germany, West` .

6 Repeat task 5 in the original dataset, this time do the calculation without subsetting `Germany` . Remove `Germany, East` , `Germany, West` and the countries that have no data for any year.

7 **Data Challenge**: Use the data frame you created in Exercise 5 and impute the NAs with the mean values using `impute()` function from `Hmisc` package. Use `is.imputed()` to see values imputed once you finish. Work with the transpose of the data frame with simply using `t()` . The challenge is creating a loop since there are many columns with NAs. Once you take the transpose, don't forget to check the type of the columns! Understand the difference of `[ ]` and `[[ ]]` . Here is an example to get you started:

```
df <- data.frame(var1=c(1,3,NA,10),
               var2=c(NA,2,5,9))

vars <- c("var1", "var2")

for (i in vars ) { df[[i]]<-mean(df[[i]], na.rm=TRUE) #replaces every value with the mean of the
column
                  }
```

Bank Marketing Data set

Bonus exercise will be based on randomly sampled bank marketing data (revisiting from Week 1), banksim.csv (../data/banksim.csv) which is manipulated for the purpose of the task,located at UCI Machine Learning Repository https://archive.ics.uci.edu/ml/datasets/Bank+Marketing (https://archive.ics.uci.edu/ml/datasets/Bank+Marketing) containing the variables:

age: Numerical variable

marital: Categorical variable with three levels (married,single,divorced where widowed counted as divorced)

education: Categorical variable with three levels (primary, secondary, tertiary)

job: Categorical variable containing type of jobs

balance: Numerical variable, balance in the bank account

day: Numerical variable, last contacted month of the day

month: Categorical variable, last contacted month

duration: Numerical variable, duration of the contact time

8 **Bonus Exercise**: Check for obvious inconsistencies or errors in the banksim.csv data using your own choice of package. Share your own story with your code on the discussion board. Best solution(s) will be immortalised as example solutions in this worksheet.

Finished?

If you have finished the above tasks, work through the weekly list of tasks posted on the Canvas announcement page.

Return to Course Website (../index.html)