

Experiment 2

Prof Meenakshi Garg

7/29/2020

Data Preprocessing Techniques

This reference Material is created for Mumbai university MCA Course for ADBMS. Topics Covered are- Implementation of Data preprocessing techniques like,

1. Naming and Renaming variables, adding a new variable.
2. Dealing with missing data.
3. Dealing with categorical data.
4. Data reduction using subsetting

```
setwd("E:/R Orientation") getwd()
```

```
my_data<-mtcars  
head(my_data,5)
```

```
##           mpg cyl  disp  hp  drat    wt  qsec vs  am  gear  carb  
## Mazda RX4      21.0   6  160 110  3.90  2.620 16.46  0  1    4    4  
## Mazda RX4 Wag  21.0   6  160 110  3.90  2.875 17.02  0  1    4    4  
## Datsun 710     22.8   4  108  93  3.85  2.320 18.61  1  1    4    1  
## Hornet 4 Drive  21.4   6  258 110  3.08  3.215 19.44  1  0    3    1  
## Hornet Sportabout 18.7   8  360 175  3.15  3.440 17.02  0  0    3    2
```

```
#my_data  
my_data1 <- my_data[1:6,1:5]  
my_data1
```

```
##           mpg cyl  disp  hp  drat  
## Mazda RX4      21.0   6  160 110  3.90  
## Mazda RX4 Wag  21.0   6  160 110  3.90  
## Datsun 710     22.8   4  108  93  3.85  
## Hornet 4 Drive  21.4   6  258 110  3.08  
## Hornet Sportabout 18.7   8  360 175  3.15  
## Valiant        18.1   6  225 105  2.76
```

```
## Renaming columns with dplyr::rename()  
require(dplyr)  
my_data1 = rename(my_data1, horse_power = hp)  
my_data1
```

```
##           mpg cyl  disp horse_power drat  
## Mazda RX4      21.0   6  160      110 3.90  
## Mazda RX4 Wag  21.0   6  160      110 3.90  
## Datsun 710     22.8   4  108       93 3.85  
## Hornet 4 Drive  21.4   6  258      110 3.08  
## Hornet Sportabout 18.7   8  360      175 3.15  
## Valiant        18.1   6  225      105 2.76
```

```
## Adding new variable  
my_data1$new_hp1 <- my_data1$horse_power * 0.5  
colnames(my_data1)
```

```
## [1] "mpg"      "cyl"      "disp"     "horse_power" "drat"
```

```
## [6] "new_hp1"
```

```
my_data1
```

```
##           mpg cyl disp horse_power drat new_hp1
## Mazda RX4      21.0   6  160          110 3.90    55.0
## Mazda RX4 Wag  21.0   6  160          110 3.90    55.0
## Datsun 710      22.8   4  108           93 3.85    46.5
## Hornet 4 Drive  21.4   6  258          110 3.08    55.0
## Hornet Sportabout 18.7   8  360          175 3.15    87.5
## Valiant        18.1   6  225          105 2.76    52.5
```

```
#naming variable
```

```
#Reading with read.table() assumes no headers by default. First few lines :
```

```
data2 = read.table(file="E:/R Orientation/missing_coll.csv", sep = ",")
```

```
data2
```

```
##      V1      V2      V3      V4      V5
## 1     1     Rick 623.30 01/01/2012      IT
## 2     2      Dan 515.20 23/09/2013 Operations
## 3     3 Michelle 611.00 15/11/2014      IT
## 4     4     Ryan 729.00 11/05/2014      HR
## 5    NA     Gary 843.25 27/03/2015  Finance
## 6     6     Nina      NA 21/05/2013      IT
## 7     7    Simon 632.80 30/07/2013 Operations
## 8     8     Guru 722.50 17/06/2014  Finance
## 9     9     John      NA 21/05/2012
## 10    10    Rock 600.80 30/07/2013      HR
## 11    11    Brad 1032.80 30/07/2013 Operations
## 12    12    Ryan 729.00 11/05/2014      HR
```

```
#V1, V2, V3.. are given as default names (titles) by R
```

```
data2 = read.csv(file="E:/R Orientation/missing_coll.csv", col.names=c("Sno", "NAME", "SALARY", "DateOfJoin", "Department"))
```

```
data2
```

```
##      Sno      NAME  SALARY DateOfJoin Department
## 1      2      Dan  515.20 23/09/2013 Operations
## 2      3 Michelle  611.00 15/11/2014      IT
## 3      4     Ryan  729.00 11/05/2014      HR
## 4     NA     Gary  843.25 27/03/2015  Finance
## 5      6     Nina      NA 21/05/2013      IT
## 6      7    Simon  632.80 30/07/2013 Operations
## 7      8     Guru  722.50 17/06/2014  Finance
## 8      9     John      NA 21/05/2012
## 9     10    Rock  600.80 30/07/2013      HR
## 10     11    Brad 1032.80 30/07/2013 Operations
## 11     12    Ryan  729.00 11/05/2014      HR
```

Error Detection and Correction

NA : Not Available - Known as missing values

Works as a place holder for something that is 'missing'

Most basic operations(addition, subtraction, multiplication, etc.) in R deal with it without crashing and return NA if one of the inputs is NA

is.na(VALUE) is used to check if the input value is NA or not. Returns a TRUE/FALSE vector Whereas in case of Excel like utilities for numeric computations it's assumed to be 0

```

# Operation with NA
NA + 4

## [1] NA

# Create a vector V with 1 NA value
V <- c(1,2,NA,3)
# Median with and without NA (remove NA)
median(V)

## [1] NA

# On removing NAs
median(V, na.rm = T)

## [1] 2

# Apply is.na() to vector
is.na(V)

## [1] FALSE FALSE TRUE FALSE

# Removing the NA values by using logical indexing
naVals <- is.na(V)
# Get values that are not NA
V[!naVals]

## [1] 1 2 3

# Subsetting with complete cases - values that are not NA
V[complete.cases(V)]

## [1] 1 2 3

# Subsetting a data frame with complete cases
# Complete Data of Prime Ministers. Notice NAs
dataC <- read.csv(file = "E:/R Orientation/na_data.csv", na.strings = "")
dataC

##      X1      Rick X623.3 X01.01.2012      IT
## 1    2      Dan  515.20 23/09/2013 Operations
## 2    3 Michelle  611.00 15/11/2014      IT
## 3    4      Ryan  729.00 11/05/2014      HR
## 4   NA      Gary  843.25 27/03/2015  Finance
## 5    6      Nina    NA 21/05/2013      IT
## 6    7    Simon  632.80 30/07/2013 Operations
## 7    8      Guru  722.50 17/06/2014  Finance
## 8    9      John    NA 21/05/2012    <NA>
## 9   10      Rock  600.80 30/07/2013      HR
## 10 11      Brad 1032.80 30/07/2013 Operations
## 11 12      Ryan  729.00 11/05/2014      HR

# Subset only the rows without NA
dataCompleteCases <- dataC[complete.cases(dataC),]
dataCompleteCases

##      X1      Rick X623.3 X01.01.2012      IT
## 1    2      Dan  515.2 23/09/2013 Operations
## 2    3 Michelle  611.0 15/11/2014      IT
## 3    4      Ryan  729.0 11/05/2014      HR

```

```
## 6 7 Simon 632.8 30/07/2013 Operations
## 7 8 Guru 722.5 17/06/2014 Finance
## 9 10 Rock 600.8 30/07/2013 HR
## 10 11 Brad 1032.8 30/07/2013 Operations
## 11 12 Ryan 729.0 11/05/2014 HR
```

Imputation

The process of estimating or deriving missing values

There are various methods for imputation

- Imputation of the mean
- Imputation of the median
- Imputation using linear regression models
 - Package Hmisc implements many imputation methods, few examples :

```
library(Hmisc)
## create a vector
x = c(1,2,3,NA,4,4,NA)
# mean imputation - from package, mention name of function to be used
x <- impute(x, fun = mean)
x
```

```
## 1 2 3 4 5 6 7
## 1.0 2.0 3.0 2.8* 4.0 4.0 2.8*
```

```
#median imputation
x <- impute(x, fun = median)
x
```

```
## 1 2 3 4 5 6 7
## 1.0 2.0 3.0 2.8* 4.0 4.0 2.8*
```

```
## 1 2 3 4 5 6 7
## 1.0 2.0 3.0 2.8* 4.0 4.0 2.8*
```

**** Categorical Data ****

Factors are variables in R which take on a limited number of different values; such variables are often referred to as categorical variables.

```
#Convert Character into Factor(categorical data)
# Create gender vector
gender_vector <- c("Male", "Female", "Female", "Male", "Male")
class(gender_vector)
```

```
## [1] "character"
```

```
# Convert gender_vector to a factor
factor_gender_vector <- factor(gender_vector)
class(factor_gender_vector)
```

```
## [1] "factor"
```

```
# Create Ordinal categorical vector
day_vector <- c('evening', 'morning', 'afternoon', 'midday', 'midnight', 'evening')
# Convert `day_vector` to a factor with ordered level
factor_day <- factor(day_vector, order = TRUE, levels =c('morning', 'midday', 'afternoon', 'evening', 'midnight'))
# Print the new variable
factor_day
```

```
## [1] evening morning afternoon midday midnight evening
```

```
## Levels: morning < midday < afternoon < evening < midnight
# Convert Numeric to Factor
# Creating vectors
age <- c(40, 49, 48, 40, 67, 52, 53)
salary <- c(103200, 106200, 150200, 10606, 10390, 14070, 10220)
gender <- c("male", "male", "transgender",
            "female", "male", "female", "transgender")

# Creating data frame named employee
employee<- data.frame(age, salary, gender)
employee

##   age salary      gender
## 1  40 103200         male
## 2  49 106200         male
## 3  48 150200 transgender
## 4  40  10606         female
## 5  67  10390         male
## 6  52  14070         female
## 7  53  10220 transgender

# Creating a factor corresponding to age with labels
wfact = cut(employee$age, 3, labels=c('Young', 'Medium', 'Aged'))
table(wfact)

## wfact
##  Young Medium   Aged
##      4      2      1
```