Lecture 4,5

IAQS | INSTITUTE OF ACTUARIAL & QUANTITATIVE STUDIES

Class: SY BSc

Subject : Statistical Modelling in R - 1

Chapter: Unit I

Chapter Name: Data Management

Grouping data in R

```
> df_sales=read.csv("d:/sales.csv")
> df_sales
  Region SalesMan_Code Sales_Volume
1  North          S01        20000
2  South          S02       400000
3   East          S03        25000
4   West          S04        50000
5  North          S02        50000
```

Print the average sales made by each sales man

```
> nrow(df_sales)
[1] 12
```

To count the total number of records

```
> df_sales %>%
+     group_by(SalesMan_Code)%>%
+     summarise(mean(Sales_Volume))
# A tibble: 4 x 2
  SalesMan_Code `mean(Sales_Volume)`
  <fct>                        <dbl>
1 S01                          40333.
2 S02                         165000
3 S03                          64000
4 S04                          54000
```

Print the average sales made region-wise

```
> df_sales %>% group_by(Region) %>% summarise(mean(Sales_Volume))
# A tibble: 4 x 2
  Region `mean(Sales_Volume)`
  <fct>                <dbl>
1 East                38333.
2 North               45667.
3 South              192667.
4 West                46667.
```

Print the total sales made region-wise by each sales man

```
> df_sales %>% group_by(Region,SalesMan_Code) %>%
summarise(sum(Sales_Volume))
# A tibble: 12 x 3
# Groups:   Region [4]
   Region SalesMan_Code `sum(Sales_Volume)`
   <fct>  <fct>                        <int>
 1 East   S01                          56000
 2 East   S03                          25000
 3 East   S04                          34000
 4 North  S01                           20000
 5 North  S02                           50000
```

Print the sales_volume and salesman_Code of only East region

```
> df_sales[df_sales$Region=="East",c(2,3)]
   SalesMan_Code Sales_Volume
3            S03        25000
7            S04        34000
11            S01         56000
```

Print the total sales_volume of east and west region

```
> df_sales %>%
+ filter(Region=="East" | Region=="West")%>%
+ group_by(Region)%>%
+ summarise(sum(Sales_Volume))
# A tibble: 2 x 2
  Region `sum(Sales_Volume)`
  <fct>              <int>
1 East              115000
2 West               140000
```

Apply functions in R

- Apply functions apply the same aggregate summary functions across all elements of a matrix.
- They act on an input list, matrix or array and apply a named function with one or several optional arguments.
- The called function could be:
- An aggregating function, like for example the mean,
- or the sum (that return a number or scalar);
- Other transforming or subsetting functions;

Rowwise and columnwise mean

```
> mat_a=matrix(seq(from=2,to=20,by=2),nrow=5,ncol=2)
> mat_a
     [,1] [,2]
[1,]    2   12
[2,]    4   14
[3,]    6   16
[4,]    8   18
[5,]   10   20
> apply(mat_a,1,mean)
[1]  7  9 11 13 15
> apply(mat_a,2,mean)
[1]  6 16
```

Lapply function

- You want to apply a given function to every element of a list and obtain a list as a result.
- It can be used for other objects like data frames, lists or vectors; and
- The output returned is a list (which explains the “l” in the function name),
- which has the same number of elements as the object passed to it.
- > li=list(1:20)
- > lapply(li,mean)
- [[1]]
- [1] 10.5

Finding mean of multiple lists

```
> li1=c(1,5,6,4,8,9,10,12)
> li2=2:20
> li3=seq(from=5,to=50,by=5)
> lapply(list(li1,li_2,li3),mean)
[[1]]
[1] 6.875
[[2]]
[1] 11
[[3]]
[1] 27.5
```

Sapply function

- sapply() function takes list, vector or data frame as input
- and gives output in vector or matrix.
- It is useful for operations on list objects
- and returns a list object of same length of original set.
- sapply() function does the same job as lapply() function
- but returns a vector.

Sapply function

```
> sapply(list(li1),median)
[1] 7
> lapply(list(li1),median)
[[1]]
[1] 7
```

tapply() function

- tapply() computes a measure (mean, median, min, max, etc..)
- or a function for each factor variable in a vector.
- It is a very useful function that lets you create a subset of a vector
- and then apply some functions to each of the subset.

tapply() function

- tapply(X, INDEX, FUN = NULL)
- Arguments:
- -X: An object, usually a vector
- -INDEX: A list containing factor
- -FUN: Function applied to each element of x

tapply() function on IRIS dataset

```
> tapply(iris$Petal.Length,iris$Species,mean)
    setosa versicolor  virginica
     1.462      4.260      5.552
> sales=read.csv("D:/sales.csv")
> tapply(sales$Sales_Volume,sales$Region,mean)
     East     North     South      West
 38333.33  45666.67 192666.67  46666.67
```

tapply() function on IRIS and Sales dataset

```
> tapply(iris$Petal.Length,iris$Species,mean)
    setosa versicolor  virginica
     1.462      4.260      5.552
> sales=read.csv("D:/sales.csv")
> tapply(sales$Sales_Volume,sales$Region,mean)
     East     North     South      West
 38333.33  45666.67 192666.67  46666.67
```

Case Study

A Japanese automobile company Geely Auto aspires to enter the Indian market by setting up their manufacturing unit there and producing cars locally to give competition to their Indian counterparts.

They have contracted an automobile consulting company to understand the factors on which the pricing of cars depends. Specifically, they want to understand the factors affecting the pricing of cars in the Indian market, since those may be very different from the Japanese market.

The company wants to know:

- **Which variables are significant in predicting the price of a car**
-**How well those variables describe the price of a car**

Explain the approach taken to fulfill the company's requirement.

Fields in the dataset

aspiration	aspiration type of engine
boreratio	ratio between engine cyinder bore diameter and piston stroke length
car_ID	unique id provided to each car
carbody	car body types
carheight	height of a car in inches
carlength	car length in inches
CarName	Company's car with model name
carwidth	car widthin in inches
citympg	city travel miles per gallon
compressionratio	ratio of relative volumes of combustion chamber and cylinder
curbweight	total mass of a vehicle in kg
cylindernumber	number of cylinders in the car
doornumber	door number
drivewheel	drive train
enginelocation	location of the engine in the car

Fields in the dataset

enginesize	size of engine in cc
enginetype	types of car engines
fuelsystem	combination of parts responsible for delivering fuel to system
fueltype	type of fuel used in engine
highwaympg	indiactes long distance travel miles per gallon
horsepower	power of engine in horsepower
peakrpm	peak revolutions per minute
price	price of car
stroke	stroke length
symboling	risk fator symbol associated with its price
wheelbase	distance between centers of front and rear wheels

Proceeding with the case study

Data cleaning and management form part of data preparation and data visualization helps in understanding the data.

```
library(readxl)
data=read_excel("C:/Users/lokesh/Desktop/iaqs/iaqs R/carpriceprediction.xlsx")
head(data)
```

```
> head(data)
# A tibble: 6 x 26
  car_ID symboling CarName fueltype aspiration doornumber carbody drivewheel
   <dbl>     <dbl> <chr>   <chr>    <chr>      <chr>      <chr>   <chr>
1      1         3 alfa-r~ gas      std        two        conver~ rwd
2      2         3 alfa-r~ gas      std        two        conver~ rwd
3      3         1 alfa-r~ gas      std        two        hatchb~ rwd
4      4         2 audi 1~ gas      std        four       sedan   fwd
5      5         2 audi 1~ gas      std        four       sedan   4wd
6      6         2 audi f~ gas      std        two        sedan   fwd
# ... with 18 more variables: enginelocation <chr>, wheelbase <dbl>, carlength <dbl>,
#   carwidth <dbl>, carheight <dbl>, curbweight <dbl>, enginetype <chr>,
#   cylindernumber <chr>, enginesize <dbl>, fuelsystem <chr>, boreratio <dbl>,
#   stroke <dbl>, compressionratio <dbl>, horsepower <dbl>, peakrpm <dbl>,
#   citympg <dbl>, highwaympg <dbl>, price <dbl>
```

Proceeding with the case study

```
> str(data)
Classes 'tbl_df', 'tbl' and 'data.frame':       205 obs. of  26 variables:
 $ car_ID           : num  1 2 3 4 5 6 7 8 9 10 ...
 $ symboling        : num  3 3 1 2 2 2 1 1 1 0 ...
 $ CarName          : chr  "alfa-romero giulia" "alfa-romero stelvio" "alfa-romero Quadrif
oglio" "audi 100 ls" ...
 $ fueltype         : chr  "gas" "gas" "gas" "gas" ...
 $ aspiration       : chr  "std" "std" "std" "std" ...
 $ doornumber       : chr  "two" "two" "two" "four" ...
 $ carbody          : chr  "convertible" "convertible" "hatchback" "sedan" ...
 $ drivewheel       : chr  "rwd" "rwd" "rwd" "fwd" ...
 $ enginelocation   : chr  "front" "front" "front" "front" ...
 $ wheelbase        : num  88.6 88.6 94.5 99.8 99.4 ...
 $ carlength        : num  169 169 171 177 177 ...
 $ carwidth         : num  64.1 64.1 65.5 66.2 66.4 66.3 71.4 71.4 71.4 67.9 ...
 $ carheight        : num  48.8 48.8 52.4 54.3 54.3 53.1 55.7 55.7 55.9 52 ...
 $ curbweight       : num  2548 2548 2823 2337 2824 ...
 $ enginetype       : chr  "dohc" "dohc" "ohcv" "ohc" ...
 $ cylindernumber   : chr  "four" "four" "six" "four" ...
 $ enginesize       : num  130 130 152 109 136 136 136 136 131 131 ...
 $ fuelsystem       : chr  "mpfi" "mpfi" "mpfi" "mpfi" ...
 $ boreratio        : num  3.47 3.47 2.68 3.19 3.19 3.19 3.19 3.19 3.13 3.13 ...
 $ stroke           : num  2.68 2.68 3.47 3.4 3.4 3.4 3.4 3.4 3.4 3.4 ...
 $ compressionratio: num  9 9 9 10 8 8.5 8.5 8.5 8.3 7 ...
 $ horsepower       : num  111 111 154 102 115 110 110 110 140 160 ...
 $ peakrpm          : num  5000 5000 5000 5500 5500 5500 5500 5500 5500 5500 ...
 $ citympg          : num  21 21 19 24 18 19 19 19 17 16 ...
 $ highwaympg       : num  27 27 26 30 22 25 25 25 20 22 ...
 $ price            : num  13495 16500 16500 13950 17450 ...
```

Data Management

- Data management is the process of ingesting, storing, organizing and maintaining the data created and collected by an organization.
- Managing data effectively requires having a data strategy and reliable methods to access, integrate, cleanse, govern, store and prepare data for analytics.

- In our digital world, data pours into organizations from many sources – operational and transactional systems, scanners, sensors, smart devices, social media, video and text. Data from different sources is integrated in a data warehouse or data lake for analysis.

- Data quality checks are done to identify data errors and inconsistencies, so they can be resolved via data cleansing tasks.

- Data models are created to map workflows and the relationships in data sets so that information can be organized to meet business needs.

Viewing the column names

```
> colnames(df)
 [1] "car_ID"           "symboling"        "CarName"          "fueltype"
 [5] "aspiration"       "doornumber"       "carbody"          "drivewheel"
 [9] "enginelocation"   "wheelbase"        "carlength"        "carwidth"
[13] "carheight"        "curbweight"       "enginetype"       "cylindernumber"
[17] "enginesize"       "fuelsystem"       "boreratio"        "stroke"
[21] "compressionratio" "horsepower"       "peakrpm"          "citympg"
[25] "highwaympg"       "price"
> |
```

To count each carbody types

```
library(plyr)
> count(df$carbody)
           x freq
1 convertible    6
2     hardtop    8
3   hatchback   70
4       sedan   96
5       wagon   25
```

To count each types of fuelsystem

```
library(plyr)
count(df$fuelsystem)
     x freq
1 1bbl   11
2 2bbl   66
3 4bbl    3
4  idi   20
5  mfi    1
6 mpfi   94
7 spdi    9
8 spfi    1
```

To print the mean price for each type of carbody

```
library(dplyr)
> df %>%
+ group_by(carbody) %>%
+ summarise(mean(price))
# A tibble: 5 x 2
  carbody     `mean(price)`
  <chr>               <dbl>
1 convertible        21890.
2 hardtop            22208.
3 hatchback          10377.
4 sedan              14344.
5 wagon              12372.
```

To print the average mileage for each type of fuelsystem

```
df %>%
+ group_by(fuelsystem)%>%
+ summarise(mean(highwaympg))
# A tibble: 8 x 2
  fuelsystem `mean(highwaympg)`
  <chr>                   <dbl>
1 1bbl                     36.5
2 2bbl                     36.0
3 4bbl                     23
4 idi                     34.8
5 mfi                      24
6 mpfi                     26.2
7 spdi                     27.3
8 spfi                     29
```

To print the average mileage for each type of fuelsystem

```
df %>%
+ group_by(fuelsystem)%>%
+ summarise(mean(highwaympg))
# A tibble: 8 x 2
  fuelsystem `mean(highwaympg)`
  <chr>                   <dbl>
1 1bbl                     36.5
2 2bbl                     36.0
3 4bbl                     23
4 idi                     34.8
5 mfi                      24
6 mpfi                     26.2
7 spdi                     27.3
8 spfi                     29
```

To print the average mileage for each type of fuel system

```
df %>%
+ group_by(fuelsystem)%>%
+ summarise(mean(highwaympg))
# A tibble: 8 x 2
  fuelsystem `mean(highwaympg)`
  <chr>                   <dbl>
1 1bbl                     36.5
2 2bbl                     36.0
3 4bbl                     23
4 idi                     34.8
5 mfi                      24
6 mpfi                     26.2
7 spdi                     27.3
8 spfi                     29
```

#to print the carname and price of cars where fuel is diesel

```
df[df$fueltype=="diesel",c("carname","price")]
```

```
# A tibble: 20 x 2
   CarName                   price
   <chr>                     <dbl>
 1 mazda glc deluxe          10795
 2 mazda rx-7 gs             18344
 3 buick electra 225 custom  25552
 4 buick century luxus (sw)  28248
 5 buick century             28176
 6 buick skyhawk             31600
 7 nissan gt-r                7099
 8 peugeot 304               13200
 9 peugeot 504               13860
10 peugeot 604sl             16900
11 peugeot 505s turbo diesel 17075
12 peugeot 504               17950
13 toyota corona              7898
14 toyota corolla             7788
15 toyota celica gt          10698
16 vokswagen rabbit           7775
17 volkswagen model 111       7995
18 volkswagen super beetle    9495
19 volkswagen rabbit custom  13845
20 volvo 246                 22470
```

#print unique carnames

```
> unique(df$CarName)
 [1] "alfa-romero giulia"            "alfa-romero stelvio"
 [3] "alfa-romero Quadrifoglio"      "audi 100 ls"
 [5] "audi 100ls"                    "audi fox"
 [7] "audi 5000"                     "audi 4000"
 [9] "audi 5000s (diesel)"           "bmw 320i"
[11] "bmw x1"                        "bmw x3"
[13] "bmw z4"                        "bmw x4"
[15] "bmw x5"                        "chevrolet impala"
[17] "chevrolet monte carlo"         "chevrolet vega 2300"
[19] "dodge rampage"                 "dodge challenger se"
[21] "dodge d200"                    "dodge monaco (sw)"
[23] "dodge colt hardtop"            "dodge colt (sw)"
[25] "dodge coronet custom"          "dodge dart custom"
[27] "dodge coronet custom (sw)"     "honda civic"
[29] "honda civic cvcc"              "honda accord cvcc"
[31] "honda accord lx"               "honda civic 1500 gl"
[33] "honda accord"                  "honda civic 1300"
[35] "honda prelude"                 "honda civic (auto)"
[37] "isuzu MU-X"                    "isuzu D-Max"
[39] "isuzu D-Max V-Cross"           "jaguar xj"
[41] "jaguar xf"                     "jaguar xk"
[43] "maxda rx3"                     "maxda glc deluxe"
[45] "mazda rx2 coupe"               "mazda rx-4"
[47] "mazda glc deluxe"              "mazda 626"
[49] "mazda glc"                     "mazda rx-7 gs"
```

Data Cleaning

Data cleaning is the process of detecting and correcting(or removing) corrupt, inaccurate records from a data frame.
Incorrect or inconsistent data leads to false conclusions.

The criteria for data quality:

- Validity
- Accuracy
- Completeness
- Consistency
- Uniqueness
- Relevancy

Fixing invalid values

1. Splitting company name from CarName

Since **CarName** is not a categorical value, we convert into factor variable with levels for convenience in working with the field.

While separating, we have stored the data in a new variable named 'car'

```
#splittng Carname
library(dplyr)
library(tidyr)
```

```
> head(unique(data$CarName))
[1] "alfa-romero giulia"      "alfa-romero stelvio"      "alfa-romero Quadrifoglio"
[4] "audi 100 ls"             "audi 100ls"               "audi fox"
```

Fixing invalid values

```
> car=separate(data,CarName,into=c("CompanyName","carname"),sep=" ")
Warning messages:
1: Expected 2 pieces. Additional pieces discarded in 65 rows [4, 10, 20, 21, 23, 25, 26, 27, 28, 29,
 30, 32, 34, 35, 36, 37, 39, 43, 46, 52, ...].
2: Expected 2 pieces. Missing pieces filled with `NA` in 2 rows [139, 142].
```

```
> head(car)
# A tibble: 6 x 28
  car_ID symboling CompanyName carname fueltype aspiration doornumber carbody
   <dbl>     <dbl> <chr>       <chr>   <chr>    <chr>      <chr>      <chr>
1      1         3 alfa-romero giulia  gas      std        two        conver~
2      2         3 alfa-romero stelvio gas      std        two        conver~
3      3         1 alfa-romero Quadri~ gas      std        two        hatchb~
4      4         2 audi        100     gas      std        four       sedan
5      5         2 audi        100ls   gas      std        four       sedan
6      6         2 audi        fox     gas      std        two        sedan
# ... with 20 more variables: drivewheel <chr>, enginelocation <chr>, wheelbase <dbl>,
#   carlength <dbl>, carwidth <dbl>, carheight <dbl>, curbweight <dbl>,
#   enginetype <chr>, cylindernumber <chr>, enginesize <dbl>, fuelsystem <chr>,
#   boreratio <dbl>, stroke <dbl>, compressionratio <dbl>, horsepower <dbl>,
#   peakrpm <dbl>, citympg <dbl>, highwaympg <dbl>, price <dbl>, fueleconomy <dbl>
```

Fixing invalid values

2. Identifying spelling errors in **CompanyName**

```
> unique(car$CompanyName)
 [1] "alfa-romero" "audi"       "bmw"        "chevrolet"  "dodge"
 [6] "honda"       "isuzu"      "jaguar"     "maxda"      "mazda"
[11] "buick"       "mercury"    "mitsubishi" "Nissan"     "nissan"
[16] "peugeot"     "plymouth"   "porsche"    "porcshce"   "renault"
[21] "saab"        "subaru"     "toyota"     "toyouta"    "vokswagen"
[26] "volkswagen"  "vw"         "volvo"
```

Fixing invalid values

We need to change the values in the column to make it consistent.

```
#correcting spelling errors in CompanyName
car$CompanyName[car$CompanyName=="mazda"]="maxda"
car$CompanyName[car$CompanyName=="nissan"]="Nissan"
car$CompanyName[car$CompanyName=="porcshce"]="porsche"
car$CompanyName[car$CompanyName=="toyouta"]="toyota"
car$CompanyName[car$CompanyName=="vw"]="volkswagen"
car$CompanyName[car$CompanyName=="vokswagen"]="volkswagen"
car$CompanyName=as.factor(car$CompanyName)
```

We will check if the changes have happened accurately.

filter() is used to select a subset of rows from the data frame based on conditional statements.

```
> #checking for corrected values
> filter(car,car$CompanyName=="mazda"& CompanyName =="maxda")
# A tibble: 0 x 28
# ... with 28 variables: car_ID <dbl>, symboling <dbl>, CompanyName <fct>,
#   carname <chr>, fueltype <chr>, aspiration <chr>, doornumber <chr>, carbody <chr>,
#   drivewheel <chr>, enginelocation <chr>, wheelbase <dbl>, carlength <dbl>,
#   carwidth <dbl>, carheight <dbl>, curbweight <dbl>, enginetype <chr>,
#   cylindernumber <chr>, enginesize <dbl>, fuelsystem <chr>, boreratio <dbl>,
#   stroke <dbl>, compressionratio <dbl>, horsepower <dbl>, peakrpm <dbl>,
#   citympg <dbl>, highwaympg <dbl>, price <dbl>, fueleconomy <dbl>
```

```
> filter(car,CompanyName=="vokswagen")
# A tibble: 0 x 28
# ... with 28 variables: car_ID <dbl>, symboling <dbl>, CompanyName <fct>,
#   carname <chr>, fueltype <chr>, aspiration <chr>, doornumber <chr>, carbody <chr>,
#   drivewheel <chr>, enginelocation <chr>, wheelbase <dbl>, carlength <dbl>,
#   carwidth <dbl>, carheight <dbl>, curbweight <dbl>, enginetype <chr>,
#   cylindernumber <chr>, enginesize <dbl>, fuelsystem <chr>, boreratio <dbl>,
#   stroke <dbl>, compressionratio <dbl>, horsepower <dbl>, peakrpm <dbl>,
#   citympg <dbl>, highwaympg <dbl>, price <dbl>, fueleconomy <dbl>
```

Since the values don't exist in the dataset, R doesn't return fields with that data. We have successfully corrected the spelling errors.

IAQS INSTITUTE OF ACTUARIAL & QUANTITATIVE STUDIES

Missing values in R

Why Prepare or clean Data?

- Some data preparation is needed for all mining tools
- The purpose of preparation is to transform data sets so that their information content is best exposed to the mining tool
- Error prediction rate should be lower (or the same) after the preparation as before it

Data preparation

- Preparing data also prepares the miner so that when using prepared data ,the miner produces better models, faster
- GIGO - good data is a prerequisite for producing effective models of any type

Missing values

- Data need to be formatted for a given software tool
- Data need to be made adequate for a given method
- Data in the real world is dirty
- incomplete:
- lacking attribute value
- lacking certain attributes of interest
- containing only aggregate data

Missing values

- **Incomplete**
- e.g., occupation=“”
- **noisy: containing errors or outliers**
- e.g.
- Salary=“-10”
- Age=“222”
- **inconsistent: containing discrepancies in codes or names**
- E.g Age= 42
- Birthday= 03/07/1997 “42”
- Birthday=“03/07/1997”
- Was rating “1,2,3”
- now rating “A, B, C”

Inconsistency

- Vivo smartphones - Extra upto Rs 6000 off on exchange | No Cost EMI
- DEAL OF THE DAY
- ₹7,990 - ₹35,990
- Ends in 16:25:33
- Vivo smartphones - Extra upto Rs 6000 off on exchange | No Cost E...
- Avg. Customer Review 995
- View Deal
- Honor 7X (Blue, 4GB RAM + 32GB Memory)
- DEAL OF THE DAY
- ₹9,999
- M.R.P.: ₹13,999 (29% off)
- Ends in 16:20:35

Lecture information table

Lec_Room_No	+ Lec_Date + PK	Lec_Time	Room_size	Programme_ Code	Lec_Course_code	Lect_T_Name
601	20/08/2013	7.00am-9.00am	50	MScIT-I	PP	Anita Ghosal
601	20-08-13	9.30a.m-10.30a.m	50	MScIT-I	DS	Manisha P
601	25/08/2013	11.00a.m-1.00p.m	50		DS	Manisha P
601	21/08/2013	9.00-11.00		MScIT-I	IT	AparnaPanigrahy
602	21/08/2013	7.00-9.00	100	MScIT-II	AI	Kavita P
602	20/08/2013	9.00-11.00	100	MScIT-II	DW	Vaishali Mishra
603	21/09/2013	9.00am-11.00am	70	TYIT	DAT	IS
603	25/09/2013	12.00 noon-2.00pm	70	TYIT	DAT	Mahesh Naik
603	21st sept 2013	11.30-1.30	70	TYIT	AC	Indrani S
603	22nd september	9.00-11.00	70	TYIT	AC	Indrani Sen

IAQS INSTITUTE OF ACTUARIAL & QUANTITATIVE STUDIES

Why data goes missing?

Different types of missing values in dataset

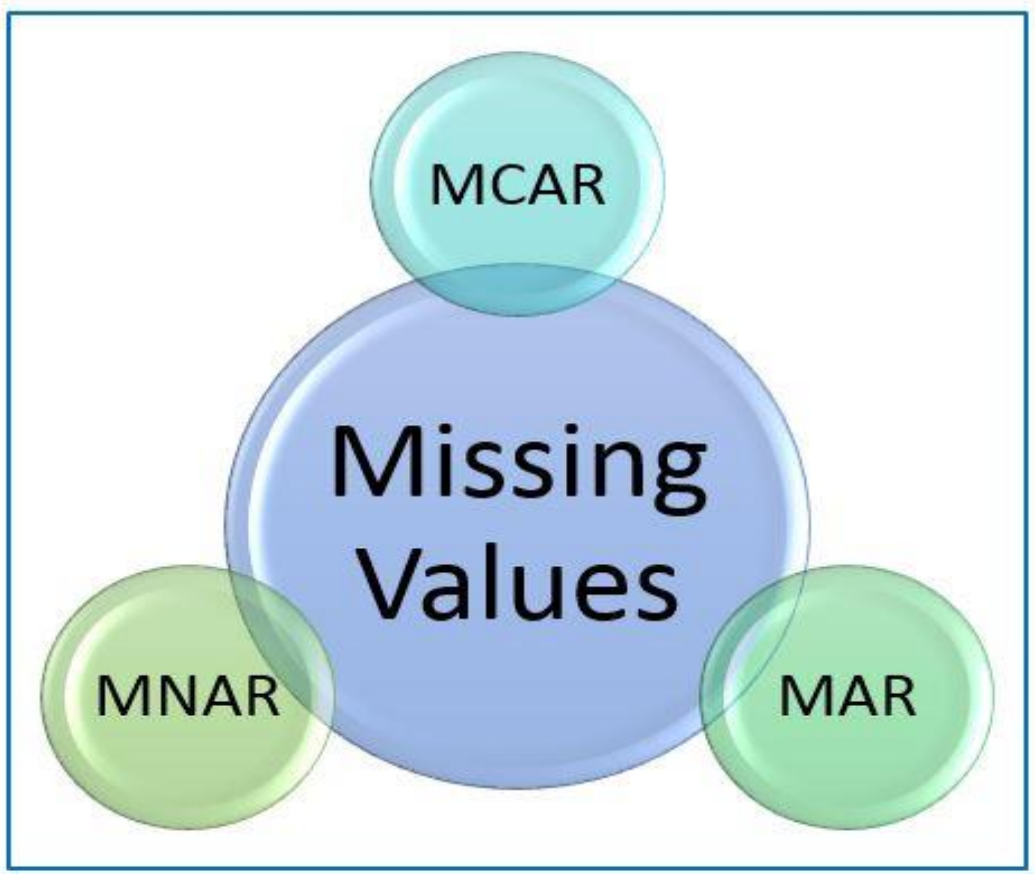

Figure 1 - Different Types of Missing Values in Datasets

Missing at Random (MAR):

1. Missing at random means that the propensity for a data point to be missing is not related to the missing data, but it is related to some of the observed data.
2. This means that the missing variable is missing due to some other attribute value.
3. The probability of missing data on Y is unrelated to the value of Y after controlling for other variables in the analysis (say X). on another hand, Missing value (y) depends on x, but not y.
4. E.g Price of a house is missing but it depends on the locality

Missing completely at random(MCAR)

- There is no relationship between the data missing and any of the observed values of the other attributes
- Its completely independent.
- E.g While crawling data from the web some mobile prices were not crawled and were missing in the dataset.

Missing not at Random (MNAR):

- Two possible reasons are that the missing value depends on the hypothetical value and the value is left missing on purpose
- E.g if gender=female, they generally don’t fill age attribute value
- Salary of the higher officials above band 5 were confidential.

IAQS INSTITUTE OF ACTUARIAL & QUANTITATIVE STUDIES

Missing values in the vector

Adding missing values in the vector

```
> x=c(10,20,14,NA,23,45)
> is.na(x)
[1] FALSE FALSE FALSE  TRUE FALSE FALSE
> sum(is.na(x))
[1] 1
> x[is.na(x)==TRUE]
[1] NA
```

Removing NA values from the vector

```
> x
[1] 10.0 20.0 14.0 22.4 23.0 45.0
> y=x[is.na(x)==FALSE]
> y
[1] 10.0 20.0 14.0 22.4 23.0 45.0
>
```

Replacing na values with mean

```
> x[is.na(x)==TRUE]=mean(x,na.rm=TRUE)
> x
[1] 10.0 20.0 14.0 22.4 23.0 45.0
```

IAQS INSTITUTE OF ACTUARIAL & QUANTITATIVE STUDIES

Missing values in carprediction dataset

Fixing invalid values

3. Check the dataset for duplicate values and missing values

```
> #checking for duplicates
> car$car_ID[duplicated(car$car_ID)]
numeric(0)
```

```
> #checking missing values
> any(is.na(car))
[1] TRUE
> any(is.na(car$carname))
[1] TRUE
```

```
> is.na(car$carname)
```

```
[118] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[131] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE  TRUE FALSE
[144] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[157] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[170] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
```

This section of output shows two ‘TRUE’ values indicating two missing values.

Fixing invalid values

```
> #removing na
> car$carname[is.na(car$carname)]=0
> any(is.na(car))
[1] FALSE
```

Na values are replaced with ‘0’ .

Thank You