LAB Manual

PART A

(PART A : TO BE REFFERED BY STUDENTS)

**Experiment No.03**

**A.1 Aim:**

To study and practice Data cleaning using R

**A.2 Prerequisite:**

Understanding of Statistics and basic function of R Studio. Major steps in Data Pre-processing.

**A.3 Outcome:**

**After successful completion of this experiment students will be able to**

1. Apply appropriate data cleaning techniques and improve data quality and to make it complete and consistent.

**A.4 Theory**

1. **Data Preprocessing:**

Real-world databases arehighly susceptible to noisy, missing, and inconsistent data due to their typically huge size (often several gigabytes or more) and their likely origin from multiple, heterogenous sources.

There are several data preprocessing techniques. *Data cleaning* can be applied to remove noise and correct inconsistencies in data. *Data integration* merges data from multiple sources into a coherent data store such as a data warehouse. *Data reduction* can reduce data size by, for instance, aggregating, eliminating redundant features, or clustering. *Data transformations* (e.g., normalization) may be applied, where data are scaled to fall within a smaller range like 0.0 to 1.0. This can improve the accuracy and efficiency of mining algorithms involving distance measurements. These techniques are not mutually exclusive; they may work together. For example, data cleaning can involve transformations to correct wrong data, such as by transforming all entries for a *date* field to a common format.

1. **Study the working of following commands in R from R documentation by typing them in the ‘help’ tab**
   1. **read.csv**
2. **Prepare working environment for the Lab and load data files**
3. Set the working directory to where we have stored the data.
4. Read customer.csv dataset using read.csv command

**df<- read.csv("customer.csv", stringsAsFactors = FALSE)**

1. Display structure of dataframe df

**str(df)**

1. Display summary of dataframe to know five point summary of each attribute
2. Install package Harrell Miscellaneous (hmisc)

**install.packages("Hmisc")**

**require(Hmisc)**

This package contains many functions useful for data analysis, high-level graphics, utility operations, functions for computing sample size and power, importing and annotating datasets, imputing missing values, advanced table making, variable clustering, character string manipulation, conversion of R objects to LaTeX and html code, and recoding variables.

require(package) load the namespace of the package with name package and attach it on the search list. require is designed for use inside other functions; it returns FALSE and gives a warning (rather than an error as library() does by default) if the package does not exist.

1. Generate a concise statistical description of dataframe using describe command

1. Display monthlycharges column
2. Check if any missing values are there in column

is.na(df$MonthlyCharges)

1. Find how many missing values are there

sum(is.na(df$MonthlyCharges))

1. Install package tidyr and dplyr

install.packages("tidyr") #for pipe operator

library(tidyr)

install.packages("dplyr") #for distinct function

library(dplyr)

1. find unique values in MonthlyCharges column

df %>% distinct(MonthlyCharges)

1. summarise distinct values

df %>% summarise(n= n\_distinct(MonthlyCharges))

## ‘n\_distinct’ Efficiently count the number of unique values in a set of vector. This is a faster and more concise equivalent of length(unique(x))

## doing multiple things in summarise

## df%>% summarise(n=n\_distinct(MonthlyCharges),

## count = sum(is.na(MonthlyCharges)),

## M = mean(MonthlyCharges, na.rm=TRUE))

## replace missing values with median

## df <- df %>% mutate(MonthlyCharges =replace(MonthlyCharges,is.na(MonthlyCharges),median(MonthlyCharges,na.rm = TRUE)))

## checking for nonstandard missing values:

## is.na(df$TotalCharges) #detects only single null value

## df%>% summarise(n=sum(is.na(TotalCharges)))

## change the 'na' and 'N/A' values to NA in TotalCharges column. Then count and display null values in totalcharges column

## Display all values in totalcharges column

## convert Totalcharges to Numeric using as.numeric command

## Describe structure of dataframe

## replace the missing values with mean value in Totalcharges column and display totalcharges column. Ignore null values while calculating mean value.

## check ‘paymentmethod’ column for null values and comment on the result.

## Replace ‘- -‘ by ‘NA’ and null value by ‘unavailable’

## Add new column ‘percentagecharges’ using ‘mutate’ command

## Display dataframe

**PART B**

(PART B: TO BE COMPLETED BY STUDENTS)

*(Students must submit the soft copy as per the following segments within two hours of the practicals. The soft copy must be uploaded on Blackboard LMS or emailed to the concerned Lab in charge Faculties at the end of practical; in case Blackboard is not accessible)*

|  |  |
| --- | --- |
| Roll No B032 | Name Naman Garg |
| Class: BTech CS B | Batch B2 |
| Date of Experiment 17 – 07 - 2020 | Date of Submission |
| Grade |  |

## B.1 Work done by student

## Code :

## setwd("C:/Users/naman/Desktop/LECTURES/LECTURES-sem5/dm/Experiment 3")

## df<- read.csv("customer.csv", stringsAsFactors = FALSE)

## str(df)

## summary(df)

## install.packages("Hmisc")

## require(Hmisc)

## describe(df)

## df$MonthlyCharges

## is.na(df$MonthlyCharges)

## sum(is.na(df$MonthlyCharges))

## install.packages("tidyr") #for pipe operator

## library(tidyr)

## install.packages("dplyr") #for distinct function

## library(dplyr)

## df %>% distinct(MonthlyCharges)

## df %>% summarise(n= n\_distinct(MonthlyCharges))

## df <- df %>% mutate(MonthlyCharges =replace(MonthlyCharges,is.na(MonthlyCharges),median(MonthlyCharges,na.rm = TRUE)))

## is.na(df$TotalCharges) #detects only single null value

## df%>% summarise(n=sum(is.na(TotalCharges)))

## 

## df %>% distinct(PaymentMethod)

## df <- df %>%

## mutate(TotalCharges = replace(TotalCharges, (TotalCharges == "na"||TotalCharges == "N/A"), NA))

## df%>% summarise(n=sum(is.na(TotalCharges)))

## df$TotalCharges

## df <- df %>%

## mutate(TotalCharges = replace(TotalCharges, TotalCharges == "na", NA)) %>%

## mutate(TotalCharges = replace(TotalCharges, TotalCharges == "N/A", NA))

## df$TotalCharges<- as.numeric(df$TotalCharges)

## df$TotalCharges

## str(df)

## df<-df %>% mutate\_all(~ifelse(is.na(.x), mean(.x, na.rm = TRUE), .x))

## df$PaymentMethod

## df%>% summarise(n=sum(is.null(df$PaymentMethod)))

## df <- df %>%

## mutate(PaymentMethod = replace(PaymentMethod, PaymentMethod == "--", NA)) %>%

## mutate(PaymentMethod = replace(PaymentMethod, PaymentMethod == "", 'unavailabe'))

## df <- df %>%

## mutate(PercentChanges =(MonthlyCharges/TotalCharges)\*100 )

## df

## Screenshots:

1. Set the working directory to where we have stored the data.

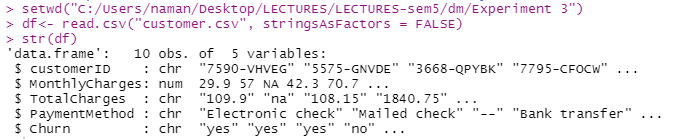
setwd("C:/Users/naman/Desktop/LECTURES/LECTURES-sem5/dm/Experiment 3")

1. Read customer.csv dataset using read.csv command

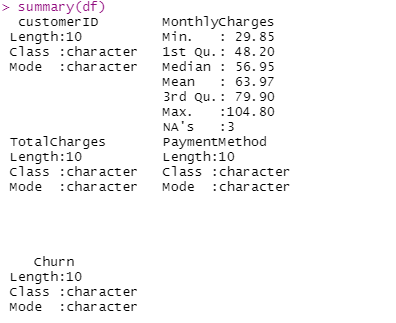
**df<- read.csv("customer.csv", stringsAsFactors = FALSE)**

1. Display structure of dataframe df

**str(df)**

****

1. Display summary of dataframe to know five point summary of each attribute



1. Install package Harrell Miscellaneous (hmisc)

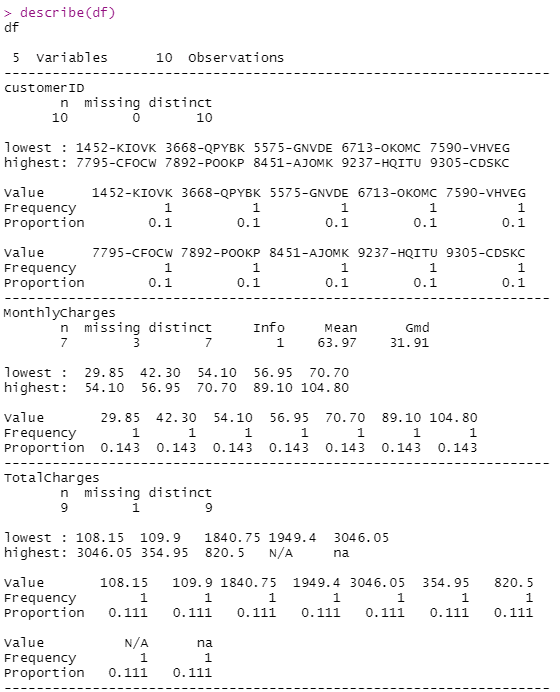
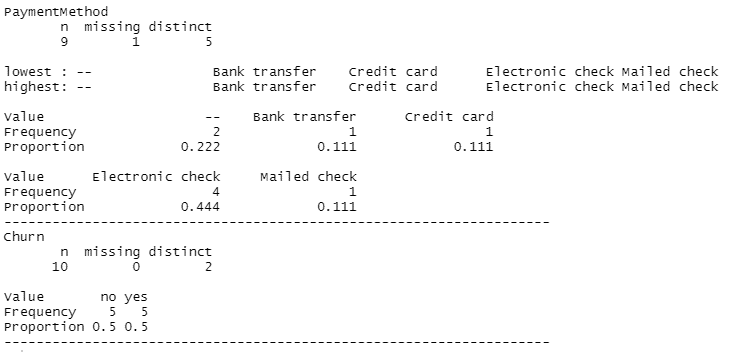
**install.packages("Hmisc")**

**require(Hmisc)**

This package contains many functions useful for data analysis, high-level graphics, utility operations, functions for computing sample size and power, importing and annotating datasets, imputing missing values, advanced table making, variable clustering, character string manipulation, conversion of R objects to LaTeX and html code, and recoding variables.

require(package) load the namespace of the package with name package and attach it on the search list. require is designed for use inside other functions; it returns FALSE and gives a warning (rather than an error as library() does by default) if the package does not exist.

1. Generate a concise statistical description of dataframe using describe command

1. Display monthlycharges column



1. Check if any missing values are there in column

is.na(df$MonthlyCharges)

1. Find how many missing values are there

sum(is.na(df$MonthlyCharges))

1. Install package tidyr and dplyr

install.packages("tidyr") #for pipe operator

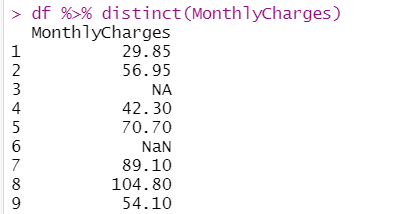
library(tidyr)

install.packages("dplyr") #for distinct function

library(dplyr)

1. find unique values in MonthlyCharges column

df %>% distinct(MonthlyCharges)



1. summarise distinct values

df %>% summarise(n= n\_distinct(MonthlyCharges))

## ‘n\_distinct’ Efficiently count the number of unique values in a set of vector. This is a faster and more concise equivalent of length(unique(x))

## 

## doing multiple things in summarise

## df%>% summarise(n=n\_distinct(MonthlyCharges),

## count = sum(is.na(MonthlyCharges)),

## M = mean(MonthlyCharges, na.rm=TRUE))

## replace missing values with median

## df <- df %>% mutate(MonthlyCharges =replace(MonthlyCharges,is.na(MonthlyCharges),median(MonthlyCharges,na.rm = TRUE)))

## 

## checking for nonstandard missing values:

## is.na(df$TotalCharges) #detects only single null value

## df%>% summarise(n=sum(is.na(TotalCharges)))

## 

## change the 'na' and 'N/A' values to NA in TotalCharges column. Then count and display null values in totalcharges column

## df <- df %>% mutate(TotalCharges = replace(TotalCharges, TotalCharges == "na", NA)) %>%

## mutate(TotalCharges = replace(TotalCharges, TotalCharges == "N/A", NA))

## Display all values in totalcharges column

## 

## convert Totalcharges to Numeric using as.numeric command

## 

## Describe structure of dataframe

## 

## replace the missing values with mean value in Totalcharges column and display

## 

## totalcharges column. Ignore null values while calculating mean value.

## na.rm was used to achieve this effect as shown in the above ss

## check ‘paymentmethod’ column for null values and comment on the result.

## 

## So there is one null value and some -- values

## Replace ‘- -‘ by ‘NA’ and null value by “unavailable”

## 

## Add new column ‘percentagecharges’ using ‘mutate’ command

## 

## Display dataframe

## 

## B.2 Conclusion

## By the end of this lab session, we were able to:

## sting as parameter used for dfs to read muddy data

## Usage of Hmisc library

## Filtering data using isna

## Pipelining functions to make code more readable using %>%

## Making changes to the dataframe using mutate function

## This was a rudimentary exercise in cleaning nan na NA n/A values since if we receive this data from different sources they might handle not available values/sources differently we need to be able to clean and organize this data