Data Input

```
setwd('/voc/work')

titanic <- read.csv('titanic.csv')

print(head(titanic))
```

```
  PassengerId Survived Pclass
1           1        0      3
2           2        1      1
3           3        1      3
4           4        1      1
5           5        0      3
6           6        0      3
                                                 Name    Sex Age SibSp
Parch
1                             Braund, Mr. Owen Harris   male  22     1
0
2 Cumings, Mrs. John Bradley (Florence Briggs Thayer) female  38     1
0
3                              Heikkinen, Miss. Laina female  26     0
0
4        Futrelle, Mrs. Jacques Heath (Lily May Peel) female  35     1
0
5                            Allen, Mr. William Henry   male  35     0
0
6                                    Moran, Mr. James   male  NA     0
0
            Ticket    Fare Cabin Embarked
1        A/5 21171  7.2500              S
2         PC 17599 71.2833   C85        C
3 STON/O2. 3101282  7.9250              S
4           113803 53.1000  C123        S
5           373450  8.0500              S
6           330877  8.4583              Q
```

Checking up on data

```
print(str(titanic))
```

```
'data.frame':   891 obs. of  12 variables:
 $ PassengerId: int  1 2 3 4 5 6 7 8 9 10 ...
 $ Survived   : int  0 1 1 1 0 0 0 0 1 1 ...
 $ Pclass     : int  3 1 3 1 3 3 1 3 3 2 ...
 $ Name       : chr  "Braund, Mr. Owen Harris" "Cumings, Mrs. John
```

```
Bradley (Florence Briggs Thayer)" "Heikkinen, Miss. Laina" "Futrelle,
Mrs. Jacques Heath (Lily May Peel)" ...
 $ Sex         : chr  "male" "female" "female" "female" ...
 $ Age         : num  22 38 26 35 35 NA 54 2 27 14 ...
 $ SibSp       : int  1 1 0 1 0 0 0 3 0 1 ...
 $ Parch       : int  0 0 0 0 0 0 0 1 2 0 ...
 $ Ticket      : chr  "A/5 21171" "PC 17599" "STON/O2. 3101282"
"113803" ...
 $ Fare        : num  7.25 71.28 7.92 53.1 8.05 ...
 $ Cabin       : chr  "" "C85" "" "C123" ...
 $ Embarked    : chr  "S" "C" "S" "S" ...
NULL

is.na(titanic$Age) # returns true and false
```

True + False + False + True + True = 3

```
print(sum(is.na(titanic$Age))) # returns true and false

[1] 177

ismissing <- function (x){
sum(is.na(x))
}

print(sapply(titanic, ismissing))

PassengerId    Survived      Pclass        Name         Sex
Age
          0           0           0           0           0
177
      SibSp       Parch      Ticket        Fare       Cabin
Embarked
          0           0           0           0           0
0

print(sapply(titanic, function(x) {sum(is.na(x))}))

PassengerId    Survived      Pclass        Name         Sex
Age
          0           0           0           0           0
177
      SibSp       Parch      Ticket        Fare       Cabin
Embarked
          0           0           0           0           0
0

print(titanic$Cabin)

"" == FALSE
```

```
# data tranform
# filters for rows where the condition is true
print(titanic[titanic$Cabin == "",])

# get count of rows
print(sum(titanic$Cabin == ""))
```

```
[1] 687
```

they may not always be "", " ", " " additional cleaning use "trim" " " --> ""

```
# replace these empty string with NAs

titanic$Cabin <- sapply(titanic$Cabin, trimws)
titanic[titanic$Cabin == "", 'Cabin']<- NA

sum(is.na(titanic$Cabin))

print(sapply(titanic, ismissing))
```

```
PassengerId    Survived      Pclass        Name         Sex
Age
          0           0           0           0           0
177
      SibSp       Parch      Ticket        Fare       Cabin
Embarked
          0           0           0           0         687
0
```

1. [] : Square brackets --> indexing e.g. titanic[]
2. () : which always follw a function e.g. sum(), sapply(), head()
3. {} : to identify block of code : in function f(x) { block of code}, if (conditon){} loop : {}

Check for duplicates

```
print(sum(duplicated(titanic)))
```

```
[1] 0
```

```
name <- c('John','Allison','Claire','Debra','Jack','Reed')
age <- c(32,25,27,28,22,35)
sex <- c('M','F','F','F','M','F')
height <- c(1.95,1.83, 1.78, 1.92, 1.87,1.75)
weight <- c( 69, 73, 11, 77, 75, 78)
mem <- c(TRUE, FALSE, TRUE, TRUE, F, T)

df_example <- data.frame(name, age, sex, height, weight, mem)

print(df_example)
```

```
     name age sex height weight   mem
1    John  32   M   1.95     69  TRUE
2 Allison  25   F   1.83     73 FALSE
3  Claire  27   F   1.78     11  TRUE
4   Debra  28   F   1.92     77  TRUE
5    Jack  22   M   1.87     75 FALSE
6    Reed  35   F   1.75     78  TRUE
```

```
# row binding
df_example <- rbind(df_example, list(name = 'Allison', age = 25, sex =
'F', height = 1.83, weight = 73, mem = FALSE))
print(df_example)
```

```
     name age sex height weight   mem
1    John  32   M   1.95     69  TRUE
2 Allison  25   F   1.83     73 FALSE
3  Claire  27   F   1.78     11  TRUE
4   Debra  28   F   1.92     77  TRUE
5    Jack  22   M   1.87     75 FALSE
6    Reed  35   F   1.75     78  TRUE
7 Allison  25   F   1.83     73 FALSE
```

```
print(duplicated(df_example))
```

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

```
# row binding
df_example <- rbind(df_example, list(name = 'Reed', age = 25, sex =
'M', height = 1.75, weight = 75, mem = FALSE))
df_example <- rbind(df_example, list(name = 'Reed', age = 25, sex =
'F', height = 1.75, weight = 75, mem = FALSE))
print(df_example)
```

```
     name age sex height weight   mem
1    John  32   M   1.95     69  TRUE
2 Allison  25   F   1.83     73 FALSE
3  Claire  27   F   1.78     11  TRUE
4   Debra  28   F   1.92     77  TRUE
5    Jack  22   M   1.87     75 FALSE
6    Reed  35   F   1.75     78  TRUE
7 Allison  25   F   1.83     73 FALSE
8    Reed  25   M   1.75     75 FALSE
9    Reed  25   F   1.75     75 FALSE
```

```
print(duplicated(df_example))
```

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

```
# to get location of duplicated
print(which(duplicated(df_example)))
```

```
[1] 7
```

```
# if we want to consider the latest value as original
print(duplicated(df_example, fromLast = TRUE))
```

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

```
# if we want to consider the latest value as original
print(which(duplicated(df_example, fromLast = TRUE)))
```

```
[1] 2
```

```
# now, considering only id column for identifyng duplicates
# name and sex

print(duplicated(df_example[c('name', 'sex')]))
```

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

```
print(which(duplicated(df_example[c('name', 'sex')])))
```

```
[1] 7 9
```

```
duplicate_rows <- which(duplicated(df_example[c('name', 'sex')]))
print(df_example[-c(duplicate_rows),])
```

```
     name age sex height weight   mem
1    John  32   M   1.95     69  TRUE
2 Allison  25   F   1.83     73 FALSE
3  Claire  27   F   1.78     11  TRUE
4   Debra  28   F   1.92     77  TRUE
5    Jack  22   M   1.87     75 FALSE
6    Reed  35   F   1.75     78  TRUE
8    Reed  25   M   1.75     75 FALSE
```

```
# for dropping rows
print(df_example[-4, ])
```

```
     name age sex height weight   mem
1    John  32   M   1.95     69  TRUE
2 Allison  25   F   1.83     73 FALSE
3  Claire  27   F   1.78     11  TRUE
5    Jack  22   M   1.87     75 FALSE
6    Reed  35   F   1.75     78  TRUE
7 Allison  25   F   1.83     73 FALSE
8    Reed  25   M   1.75     75 FALSE
9    Reed  25   F   1.75     75 FALSE
```

```
# removing duplicate rows
print(unique(df_example))
```

```
     name age sex height weight   mem
1    John  32   M   1.95     69  TRUE
2 Allison  25   F   1.83     73 FALSE
3  Claire  27   F   1.78     11  TRUE
4   Debra  28   F   1.92     77  TRUE
5    Jack  22   M   1.87     75 FALSE
6    Reed  35   F   1.75     78  TRUE
8    Reed  25   M   1.75     75 FALSE
9    Reed  25   F   1.75     75 FALSE
```

```
# adding a column
df_example$country <- c('US', 'U.S.A.', 'The United States of
America', 'spain', 'SPAIN',
                        'Britain', 'Britain', 'US', 'US')

print(df_example)
```

```
     name age sex height weight   mem                        country
1    John  32   M   1.95     69  TRUE                             US
2 Allison  25   F   1.83     73 FALSE                          U.S.A.
3  Claire  27   F   1.78     11  TRUE The United States of America
4   Debra  28   F   1.92     77  TRUE                            spain
5    Jack  22   M   1.87     75 FALSE                            SPAIN
6    Reed  35   F   1.75     78  TRUE                          Britain
7 Allison  25   F   1.83     73 FALSE                          Britain
8    Reed  25   M   1.75     75 FALSE                             US
9    Reed  25   F   1.75     75 FALSE                             US
```

```
# Check get the unique values form the column
print(unique(df_example$country))
```

```
[1] "US"                             "U.S.A."
[3] "The United States of America" "spain"
[5] "SPAIN"                          "Britain"
```

```
# change the case to uniform --> lowercasing
df_example$country<- tolower(df_example$country)

print(unique(df_example$country))
```

```
[1] "us"                             "u.s.a."
[3] "the united states of america" "spain"
[5] "britain"
```

```
# to get frequency :
print(table(df_example$country))
```

```
                     britain                        spain
                           2                            2
the united states of america                       u.s.a.
                           1                            1
```

```
                          us
                           3

# transform
df_example$country <- sapply(df_example$country,
        function(country) {ifelse((country == 'the united states of
america' )||(country == 'u.s.a.'),
                'us', country)})

print(df_example)

     name age sex height weight   mem country
1    John  32   M   1.95     69  TRUE      us
2 Allison  25   F   1.83     73 FALSE      us
3  Claire  27   F   1.78     11  TRUE      us
4   Debra  28   F   1.92     77  TRUE   spain
5    Jack  22   M   1.87     75 FALSE   spain
6    Reed  35   F   1.75     78  TRUE britain
7 Allison  25   F   1.83     73 FALSE britain
8    Reed  25   M   1.75     75 FALSE      us
9    Reed  25   F   1.75     75 FALSE      us

print(unique(titanic$Embarked))

[1] "S" "C" "Q" ""

# define a function for treatment

replace_with_nas <- function(col){
titanic[titanic[col] == "", col]<- NA
}

# perform for only charcater columns
print(sapply(titanic, class))

PassengerId    Survived      Pclass        Name         Sex
Age
  "integer"   "integer"   "integer" "character" "character"
"numeric"
      SibSp       Parch      Ticket        Fare       Cabin
Embarked
  "integer"   "integer" "character"   "numeric" "character"
"character"

library(dplyr)


Attaching package: ‘dplyr’


The following objects are masked from ‘package:stats’:
```

```
    filter, lag


The following objects are masked from ‘package:base’:

    intersect, setdiff, setequal, union
```

select_if from dplyr --> selects columns which follow the goven condition is.character --> built in function which checks if teh datatype is character or not

```
print(titanic %>%
    select_if(is.character)%>%
    names())
```

```
[1] "Name"     "Sex"      "Ticket"   "Cabin"    "Embarked"
```

```
char_cols <- titanic %>%
    select_if(is.character)%>%
    names()# gets the column names
print(char_cols)
```

```
[1] "Name"     "Sex"      "Ticket"   "Cabin"    "Embarked"
```

```
# define a function for treatment

replace_with_nas <- function(col){
                new_col <- c() # new vector
                for (x in col){ # iterate through the values in the
column
                    if (x == ""){
                        new_col <- c(new_col, NA)
                    }else {
                        new_col <- c(new_col, x)
                    }
                }
                return(new_col)
}

print(sum(is.na(titanic$Embarked)))
```

```
[1] 0
```

```
print(sum(is.na(replace_with_nas(titanic$Embarked))))
```

```
[1] 2
```

```r
new_tit <- titanic %>%
    mutate(across(where(is.character),
    ~ifelse(.x == "", NA, .x)))

print(sapply(new_tit, ismissing))
```

```
PassengerId    Survived      Pclass        Name         Sex
Age
          0           0           0           0           0
177
      SibSp       Parch      Ticket        Fare       Cabin
Embarked
          0           0           0           0         687
2
```

```r
print(titanic %>% mutate(Name = tolower(Name))%>% head())
```

```
  PassengerId Survived Pclass
1           1        0      3
2           2        1      1
3           3        1      3
4           4        1      1
5           5        0      3
6           6        0      3
                                                 Name    Sex Age SibSp
Parch
1                             braund, mr. owen harris   male  22     1
0
2 cumings, mrs. john bradley (florence briggs thayer) female  38     1
0
3                               heikkinen, miss. laina female  26     0
0
4        futrelle, mrs. jacques heath (lily may peel) female  35     1
0
5                             allen, mr. william henry   male  35     0
0
6                                     moran, mr. james   male  NA     0
0
            Ticket    Fare Cabin Embarked
1        A/5 21171  7.2500  <NA>        S
2         PC 17599 71.2833   C85        C
3 STON/O2. 3101282  7.9250  <NA>        S
4           113803 53.1000  C123        S
5           373450  8.0500  <NA>        S
6           330877  8.4583  <NA>        Q
```

new_tit <- titanic %>% mutate(across(where(is.character), function(x) {ifelse(x == "", NA, x)}))

treat missing values

1. Deletion : Rows or columns
   - If there are high percentage of missing values specially for columns
2. Imputation

```
print((sapply(titanic, ismissing))/891* 100)
```

```
PassengerId    Survived      Pclass        Name         Sex
Age
    0.00000     0.00000     0.00000     0.00000     0.00000
19.86532
      SibSp       Parch      Ticket        Fare       Cabin
Embarked
    0.00000     0.00000     0.00000     0.00000    77.10438
0.00000
```

above 30 % is non negotiable you may delet the column

for imputation :

1. constant
2. statistical value : mean, median, mode
3. ML imputers : KNN - K Nearest Neighbours

```
sum(is.na(titanic$Age))

# titanic[is.na(titanic$Age), 'Age'] <- 'constant'

mean(titanic$Age, na.rm = TRUE) # do not use missing in calculation of
mean

print(summary(titanic$Age))
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
   0.42   20.12   28.00   29.70   38.00   80.00     177
```

```
# check age by sex
print(titanic %>%
group_by(Sex)%>%
summarize(Mean = mean(Age, na.rm = TRUE)))

# A tibble: 2 × 2
  Sex     Mean
  <chr>  <dbl>
```

```
1 female  27.9
2 male    30.7
```

```
print(head(titanic$Name))
```

```
[1] "Braund, Mr. Owen Harris"
[2] "Cumings, Mrs. John Bradley (Florence Briggs Thayer)"
[3] "Heikkinen, Miss. Laina"
[4] "Futrelle, Mrs. Jacques Heath (Lily May Peel)"
[5] "Allen, Mr. William Henry"
[6] "Moran, Mr. James"
```