Untitled

Vinoth Aryan Nagabosshanam

May 20, 2017

step to create a list

```
list1<-list(1,2,3,4,5,6,6,6,7)

print(typeof(list1))
```

```
## [1] "list"
```

to create a martic

```
a<-c(1,2,3,4,5)
b<-c(6,7,8,9,10)

m<-matrix(c(a,b),nrow=2,byrow=T)
print(m)
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    1    2    3    4    5
## [2,]    6    7    8    9   10
```

create a data frame

```
a<-c(1,2,3,4,5)
b<-c('a','b','c','d','e')
c<-c('vino','in','gggg','jjj','hhh')

df<-data.frame(a,b,c)
print(typeof(df))
```

```
## [1] "list"
```

```
df
```

```
##   a b    c
## 1 1 a vino
## 2 2 b   in
## 3 3 c gggg
## 4 4 d  jjj
## 5 5 e  hhh
```

sequence its give the print space sequence

```
v<-seq(1,10,1)
v
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10
```

```
v1<-seq(1,20, 2)
v1
```

```
##  [1]  1  3  5  7  9 11 13 15 17 19
```

```
v2<-seq(1,10,3)
v2
```

```
## [1]  1  4  7 10
```

square

```
a1<-c(10,11,12)
sqrt(a1)
```

```
## [1] 3.162278 3.316625 3.464102
```

defeine a function

```
cube<-function(x){x*x*x}
cube(3)
```

```
## [1] 27
```

step import the csv

```
mba<-read.csv("B:\\data science course of udemy\\Datasets_BA 2\\mba.csv")
# to get name of columms in list
ls(mba)
```

```
## [1] "datasrno" "gmat"     "workex"
```

```
fraudata<-read.csv("B:\\data science course of udemy\\Datasets_BA 2\\fraudData.csv")
# to get the first 10 lines of data
head(fraudata)
```

```
##   custID gender state cardholder balance numTrans numIntlTrans creditLine
## 1      1      1    35          1    3000        4           14          2
## 2      2      2     2          1       0        9            0         18
## 3      3      2     2          1       0       27            9         16
## 4      4      1    15          1       0       12            0          5
## 5      5      1    46          1       0       11           16          7
## 6      6      2    44          2    5546       21            0         13
##   fraudRisk
```

```
## 1         0
## 2         0
## 3         0
## 4         0
## 5         1
## 6         0
```

```
#to list names
ls(fraudata)
```

```
## [1] "balance"      "cardholder"   "creditLine"   "custID"
## [5] "fraudRisk"    "gender"       "numIntlTrans" "numTrans"
## [9] "state"
```

```
#summary of data set

summary(fraudata)
```

```
##      custID          gender          state          cardholder
##  Min.   : 1.0   Min.   :1.000   Min.   : 2.00   Min.   :1.000
##  1st Qu.: 3.5   1st Qu.:1.000   1st Qu.: 6.50   1st Qu.:1.000
##  Median : 6.0   Median :1.000   Median :23.00   Median :1.000
##  Mean   : 6.0   Mean   :1.364   Mean   :23.45   Mean   :1.111
##  3rd Qu.: 8.5   3rd Qu.:2.000   3rd Qu.:39.50   3rd Qu.:1.000
##  Max.   :11.0   Max.   :2.000   Max.   :46.00   Max.   :2.000
##                                                 NA's   :2
##     balance        numTrans      numIntlTrans      creditLine
##  Min.   :   0   Min.   : 4.00   Min.   : 0.000   Min.   : 1.0
##  1st Qu.:   0   1st Qu.:10.00   1st Qu.: 0.000   1st Qu.: 4.5
##  Median :2000   Median :18.00   Median : 3.000   Median : 6.0
##  Mean   :2145   Mean   :20.09   Mean   : 9.818   Mean   : 9.0
##  3rd Qu.:3800   3rd Qu.:24.00   3rd Qu.:12.000   3rd Qu.:14.5
##  Max.   :6016   Max.   :54.00   Max.   :56.000   Max.   :22.0
##
##    fraudRisk
##  Min.   :0.0000
##  1st Qu.:0.0000
##  Median :0.0000
##  Mean   :0.2727
##  3rd Qu.:0.5000
##  Max.   :1.0000
##
```

```
# to get the type of colunms in data set
str(fraudata)
```

```
## 'data.frame':    11 obs. of  9 variables:
##  $ custID      : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ gender      : int  1 2 2 1 1 2 1 1 2 1 ...
##  $ state       : int  35 2 2 15 46 44 3 10 32 23 ...
##  $ cardholder  : int  1 1 1 1 1 2 NA 1 1 1 ...
##  $ balance     : int  3000 0 0 0 0 5546 2000 6016 2428 0 ...
##  $ numTrans    : int  4 9 27 12 11 21 41 20 4 18 ...
##  $ numIntlTrans: int  14 0 9 0 16 0 0 3 10 56 ...
##  $ creditLine  : int  2 18 16 5 7 13 1 6 22 5 ...
##  $ fraudRisk   : int  0 0 0 0 1 0 0 0 0 1 ...
```

basic function can done in the data set

```
or<-order(fraudata$creditLine)

# to arrange the creditline in order maner we use this command
fraudata[or,]
```

```
##    custID gender state cardholder balance numTrans numIntlTrans creditLine
## 7       7      1     3         NA    2000       41            0          1
## 1       1      1    35          1    3000        4           14          2
## 11     11      1    46         NA    4601       54            0          4
## 4       4      1    15          1       0       12            0          5
## 10     10      1    23          1       0       18           56          5
## 8       8      1    10          1    6016       20            3          6
## 5       5      1    46          1       0       11           16          7
## 6       6      2    44          2    5546       21            0         13
## 3       3      2     2          1       0       27            9         16
## 2       2      2     2          1       0        9            0         18
## 9       9      2    32          1    2428        4           10         22
##    fraudRisk
## 7          0
## 1          0
## 11         1
## 4          0
## 10         1
## 8          0
## 5          1
## 6          0
## 3          0
## 2          0
## 9          0
```

```
#to arrange int descending order
fraudata[rev(order(fraudata$creditLine)),]
```

```
##    custID gender state cardholder balance numTrans numIntlTrans creditLine
## 9       9      2    32          1    2428        4           10         22
## 2       2      2     2          1       0        9            0         18
## 3       3      2     2          1       0       27            9         16
## 6       6      2    44          2    5546       21            0         13
## 5       5      1    46          1       0       11           16          7
## 8       8      1    10          1    6016       20            3          6
## 10     10      1    23          1       0       18           56          5
## 4       4      1    15          1       0       12            0          5
## 11     11      1    46         NA    4601       54            0          4
## 1       1      1    35          1    3000        4           14          2
## 7       7      1     3         NA    2000       41            0          1
##    fraudRisk
## 9          0
## 2          0
## 3          0
## 6          0
## 5          1
## 8          0
```

```
## 10         1
## 4          0
## 11         1
## 1          0
## 7          0
```

combine the different data set

```
plasma<-read.csv("B:\\data science course of udemy\\Datasets_BA 2\\Plasma.csv")
diec<-read.csv("B:\\data science course of udemy\\Datasets_BA 2\\Diabetes.csv")
pd<-cbind(plasma,diec)
```

if its un equal

```
tran_h<-read.csv("B:\\data science course of udemy\\Datasets_BA 2\\hour_transaction.csv")
tran_d<-read.csv("B:\\data science course of udemy\\Datasets_BA 2\\transaction_data.csv")
tt<-rbind(tran_h,tran_d)
```

if both unequal we use merge function

```
all_tran<-read.csv("B:\\data science course of udemy\\Datasets_BA 2\\all_transactions.csv")
cre<-read.csv("B:\\data science course of udemy\\Datasets_BA 2\\creditdata.csv")

slc<-merge(all_tran,cre)
```

to read text file by using following command

```
vitcims<-readLines("B:\\data science course of udemy\\Datasets_BA 2\\victims.txt")
vitcims
```

```
##  [1] "%% Data on the titanic victims" "Anthony,1870,1912"
##  [3] "Ernest,1892,NA"                 "Eugene,1871,1912"
##  [5] "Rossmore,1856,1912"             "Samuel,1902,NA"
##  [7] "William,1876,1912"              "John,1904,1912"
##  [9] "Robert,1863,1912"               "Percy,1910,1912"
## [11] "% Names, birth and death dates"
```

```
dfv<-data.frame(vitcims)
dfv
```

```
##                           vitcims
## 1  %% Data on the titanic victims
## 2               Anthony,1870,1912
## 3                  Ernest,1892,NA
## 4                Eugene,1871,1912
## 5              Rossmore,1856,1912
```

```
## 6                  Samuel,1902,NA
## 7               William,1876,1912
## 8                  John,1904,1912
## 9                 Robert,1863,1912
## 10                 Percy,1910,1912
## 11 % Names, birth and death dates
```

```
# grepl which used to find extact sentence
com<-grepl("^%", vitcims)
com
```

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

```
# we are viewing without those element
text<-vitcims[!com]
text
```

```
## [1] "Anthony,1870,1912"  "Ernest,1892,NA"     "Eugene,1871,1912"
## [4] "Rossmore,1856,1912" "Samuel,1902,NA"     "William,1876,1912"
## [7] "John,1904,1912"     "Robert,1863,1912"   "Percy,1910,1912"
```

```
# code used to split lines in columms
out <- strsplit(text,',')

# step convert in data fram
nl<-matrix(unlist(out),nrow=length(out),byrow = T)
# assign col names
colnames(nl)<-c("name","birthyear","deathyear")
ti_victims<-as.data.frame(nl)
summary(nl)
```

```
##       name     birthyear deathyear
##  Anthony:1   1856   :1   1912:7
##  Ernest :1   1863   :1   NA  :2
##  Eugene :1   1870   :1
##  John   :1   1871   :1
##  Percy  :1   1876   :1
##  Robert :1   1892   :1
##  (Other):3   (Other):3
```

```
# to conver year into numer
ti_victims$birthyear <-as.numeric(ti_victims$birthyear)
ti_victims$deathyear<-as.numeric(ti_victims$deathyear)
summary(ti_victims)
```

```
##       name     birthyear   deathyear
##  Anthony:1   Min.   :1   Min.   :1.000
##  Ernest :1   1st Qu.:3   1st Qu.:1.000
##  Eugene :1   Median :5   Median :1.000
##  John   :1   Mean   :5   Mean   :1.222
##  Percy  :1   3rd Qu.:7   3rd Qu.:1.000
##  Robert :1   Max.   :9   Max.   :2.000
##  (Other):3
```

```
str(ti_victims)
```

```
## 'data.frame':    9 obs. of  3 variables:
##  $ name     : Factor w/ 9 levels "Anthony","Ernest",..: 1 2 3 7 8 9 4 6 5
```

```
##  $ birthyear: num  3 6 4 1 7 5 8 2 9
##  $ deathyear: num  1 2 1 1 2 1 1 1 1
```

```
telecalls<-read.csv("B:\\data science course of udemy\\Datasets_BA 2\\telecomCalls.csv")
```

replace negative values in colums with na values

```
upd_tel<-apply(telecalls,MARGIN = 2,
               function(a)
                 {ifelse (a==99|a==-99, NA,a)})
upd_tel
```

```
##       AccountID numberVmail total_daycalls total_localcalls
##  [1,]     45090           7             10                6
##  [2,]     45091           8             14                8
##  [3,]     45093           3              6                3
##  [4,]     45089          12              1                1
##  [5,]     46588          NA              8                5
##  [6,]     46578           5              3                2
##  [7,]     44015           6             12                5
##  [8,]     44067           2             13                7
##  [9,]     43890          NA             NA                4
## [10,]     45099          11              2                0
##       customerService_calls
##  [1,]                     4
##  [2,]                     3
##  [3,]                     0
##  [4,]                    NA
##  [5,]                     1
##  [6,]                     2
##  [7,]                    NA
##  [8,]                    NA
##  [9,]                    NA
## [10,]                     2
```

now clear the NA with mean valur

```
apply(upd_tel,MARGIN = 2,function(a){mean(a,na.rm = T)})
```

```
##             AccountID           numberVmail        total_daycalls
##          45060.000000              6.750000              7.666667
##      total_localcalls customerService_calls
##              4.100000              2.000000
```

```
library(zoo)
```

```
## Warning: package 'zoo' was built under R version 3.3.3
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
```

```
##
##     as.Date, as.Date.numeric
```

```
na.aggregate(upd_tel)
```

```
##       AccountID numberVmail total_daycalls total_localcalls
##  [1,]     45090        7.00      10.000000                6
##  [2,]     45091        8.00      14.000000                8
##  [3,]     45093        3.00       6.000000                3
##  [4,]     45089       12.00       1.000000                1
##  [5,]     46588        6.75       8.000000                5
##  [6,]     46578        5.00       3.000000                2
##  [7,]     44015        6.00      12.000000                5
##  [8,]     44067        2.00      13.000000                7
##  [9,]     43890        6.75       7.666667                4
## [10,]     45099       11.00       2.000000                0
##       customerService_calls
##  [1,]                     4
##  [2,]                     3
##  [3,]                     0
##  [4,]                     2
##  [5,]                     1
##  [6,]                     2
##  [7,]                     2
##  [8,]                     2
##  [9,]                     2
## [10,]                     2
```

install lattice package

```
library("lattice")
data(barley)
tapply(barley$yield,barley$site, mean)
```

```
##    Grand Rapids          Duluth University Farm          Morris
##        24.93167        27.99667        32.66667        35.40000
##       Crookston          Waseca
##        37.42000        48.10833
```

how to pload othe rthe stastical package

```
library("foreign")

can<-read.spss("B:\\data science course of udemy\\Datasets_BA 2\\cancer.sav")

cans<-as.data.frame(can)
str(cans)
```

```
## 'data.frame':    25 obs. of  9 variables:
##  $ ID      : num  1 5 6 9 11 15 21 26 31 35 ...
##  $ TRT     : num  0 0 0 0 0 0 0 0 0 0 ...
```

```
##  $ AGE     : num  52 77 60 61 59 69 67 56 61 51 ...
##  $ WEIGHIN : num  124 160 136 180 176 ...
##  $ STAGE   : num  2 1 4 1 2 1 1 3 1 1 ...
##  $ TOTALCIN: num  6 9 7 6 6 6 6 6 6 6 ...
##  $ TOTALCW2: num  6 6 9 7 7 6 11 11 9 4 ...
##  $ TOTALCW4: num  6 10 17 9 16 6 11 15 6 8 ...
##  $ TOTALCW6: num  7 9 19 3 13 11 10 15 8 7 ...
```

```
summary(cans)
```

```
##        ID             TRT              AGE           WEIGHIN
##  Min.   : 1.0   Min.   :0.00   Min.   :27.00   Min.   :124.0
##  1st Qu.:12.0   1st Qu.:0.00   1st Qu.:52.00   1st Qu.:160.0
##  Median :24.0   Median :0.00   Median :60.00   Median :172.8
##  Mean   :25.6   Mean   :0.44   Mean   :59.64   Mean   :178.3
##  3rd Qu.:39.0   3rd Qu.:1.00   3rd Qu.:67.00   3rd Qu.:187.0
##  Max.   :58.0   Max.   :1.00   Max.   :86.00   Max.   :261.4
##
##      STAGE         TOTALCIN        TOTALCW2        TOTALCW4
##  Min.   :0.00   Min.   : 4.00   Min.   : 4.00   Min.   : 6.00
##  1st Qu.:1.00   1st Qu.: 6.00   1st Qu.: 7.00   1st Qu.: 8.00
##  Median :1.00   Median : 6.00   Median : 8.00   Median :10.00
##  Mean   :1.88   Mean   : 6.52   Mean   : 8.28   Mean   :10.36
##  3rd Qu.:2.00   3rd Qu.: 7.00   3rd Qu.:10.00   3rd Qu.:12.00
##  Max.   :4.00   Max.   :12.00   Max.   :16.00   Max.   :17.00
##
##     TOTALCW6
##  Min.   : 3.000
##  1st Qu.: 7.000
##  Median : 9.000
##  Mean   : 9.478
##  3rd Qu.:11.000
##  Max.   :19.000
##  NA's   :2
```

```
# step to fill the na value with median
cans$TOTALCW6[is.na(cans$TOTALCW6)]<-median(cans$TOTALCW6,na.rm=TRUE)
summary(cans)
```

```
##        ID             TRT              AGE           WEIGHIN
##  Min.   : 1.0   Min.   :0.00   Min.   :27.00   Min.   :124.0
##  1st Qu.:12.0   1st Qu.:0.00   1st Qu.:52.00   1st Qu.:160.0
##  Median :24.0   Median :0.00   Median :60.00   Median :172.8
##  Mean   :25.6   Mean   :0.44   Mean   :59.64   Mean   :178.3
##  3rd Qu.:39.0   3rd Qu.:1.00   3rd Qu.:67.00   3rd Qu.:187.0
##  Max.   :58.0   Max.   :1.00   Max.   :86.00   Max.   :261.4
##      STAGE         TOTALCIN        TOTALCW2        TOTALCW4
##  Min.   :0.00   Min.   : 4.00   Min.   : 4.00   Min.   : 6.00
##  1st Qu.:1.00   1st Qu.: 6.00   1st Qu.: 7.00   1st Qu.: 8.00
##  Median :1.00   Median : 6.00   Median : 8.00   Median :10.00
##  Mean   :1.88   Mean   : 6.52   Mean   : 8.28   Mean   :10.36
##  3rd Qu.:2.00   3rd Qu.: 7.00   3rd Qu.:10.00   3rd Qu.:12.00
##  Max.   :4.00   Max.   :12.00   Max.   :16.00   Max.   :17.00
##     TOTALCW6
##  Min.   : 3.00
```

```
##  1st Qu.: 7.00
##  Median : 9.00
##  Mean   : 9.44
##  3rd Qu.:11.00
##  Max.   :19.00
```

```
cor(cans)
```

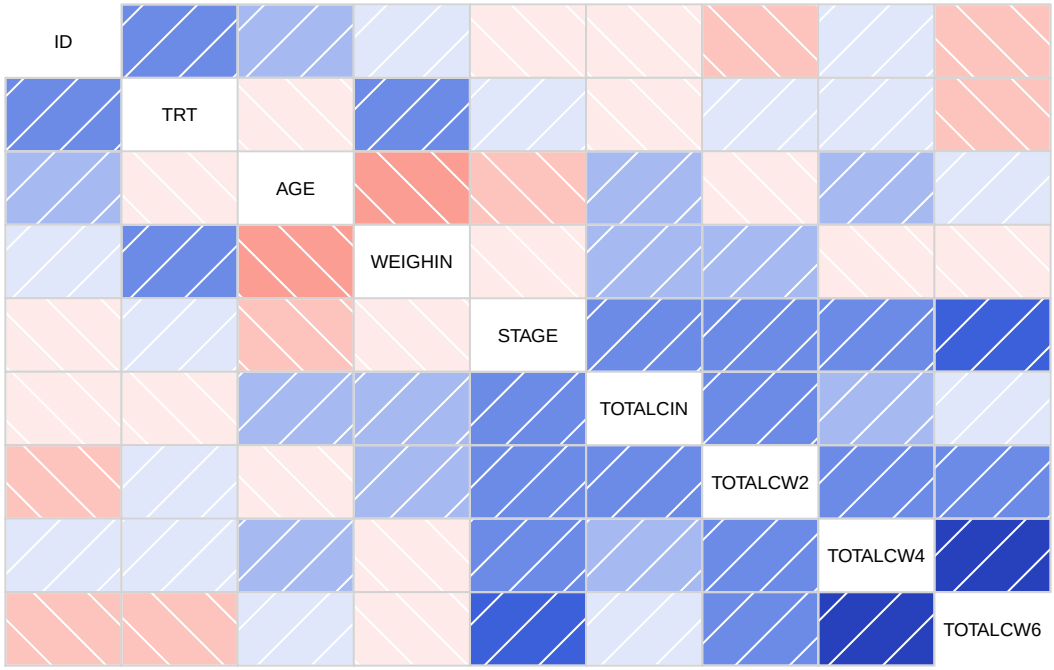
```
##                    ID         TRT         AGE       WEIGHIN         STAGE
## ID         1.000000000  0.36004884  0.21758198  0.0221147940 -0.0459017034
## TRT        0.360048840  1.00000000 -0.01297371  0.3879107651  0.0878629625
## AGE        0.217581984 -0.01297371  1.00000000 -0.2875867746 -0.2166421989
## WEIGHIN    0.022114794  0.38791077 -0.28758677  1.0000000000 -0.0001687351
## STAGE     -0.045901703  0.08786296 -0.21664220 -0.0001687351  1.0000000000
## TOTALCIN  -0.001335122 -0.03868294  0.25610664  0.1696136194  0.2987165612
## TOTALCW2  -0.165045936  0.06212823 -0.10582417  0.2740960772  0.2897697337
## TOTALCW4   0.061466803  0.07195550  0.16249437 -0.0950352938  0.3889831276
## TOTALCW6  -0.196880372 -0.16829404  0.02309480 -0.0750329227  0.4672978897
##              TOTALCIN    TOTALCW2    TOTALCW4    TOTALCW6
## ID        -0.001335122 -0.16504594  0.06146680 -0.19688037
## TRT       -0.038682941  0.06212823  0.07195550 -0.16829404
## AGE        0.256106644 -0.10582417  0.16249437  0.02309480
## WEIGHIN    0.169613619  0.27409608 -0.09503529 -0.07503292
## STAGE      0.298716561  0.28976973  0.38898313  0.46729789
## TOTALCIN   1.000000000  0.31442098  0.22184588  0.09185218
## TOTALCW2   0.314420979  1.00000000  0.33724339  0.36252927
## TOTALCW4   0.221845885  0.33724339  1.00000000  0.64588142
## TOTALCW6   0.091852175  0.36252927  0.64588142  1.00000000
```

to check the correlation and covariance between the dataset by using corrgram

```
#install.packages("corrgram")
library(corrgram)
```

```
## Warning: package 'corrgram' was built under R version 3.3.3
```

```
corrgram(cans)
```




```
# red color incidates the negative correlation
# blue color incidates the postive correlation
```