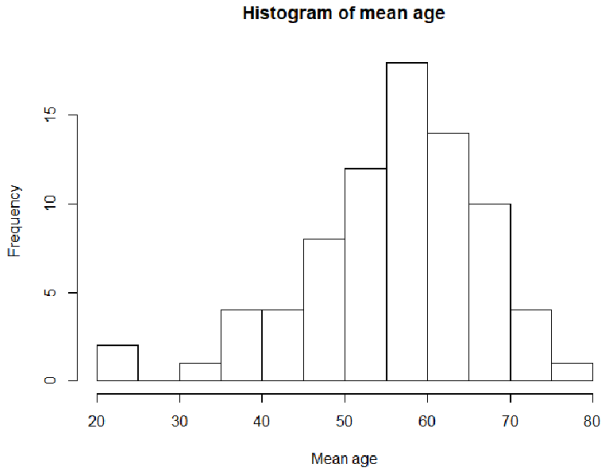
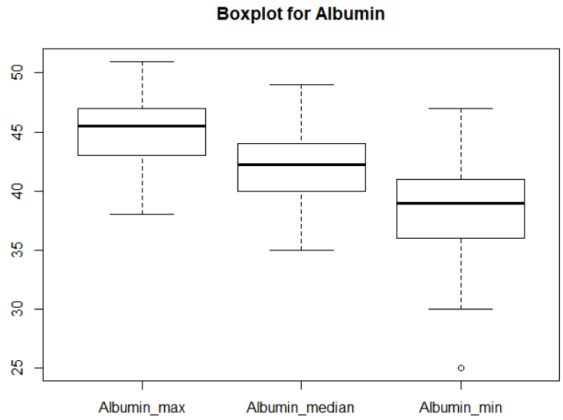
---
title: "HS650_HW2"
author: "Xinchun Li"
date: "2018/1/26"
output:
  pdf_document: default

```{r setup, include=FALSE}
chooseCRANmirror(graphics=FALSE, ind=1)

install.packages("rvest", repos = "http://cran.r-project.org")
install.packages("gmodels")
install.packages("ggplot2")
install.packages("reshape2")
install.packages("plotly")
install.packages("GGally")
install.packages("mi")
install.packages("betareg")
install.packages("corrplot")
install.packages("xtable")

library(rvest)
library(gmodels)
library(ggplot2)
library(reshape2)
library(plotly)
library(MASS)
library(unbalanced)
library(GGally)
library(mi)
library(betareg)
library(corrplot)
library(unbalanced)
library(xtable)

# Q1
# Load the following two datasets, generate summary statistics for all variables, plot some of the features (e.g., histograms,
# box plots, density plots, etc.) of some variables, and save the data locally as CSV files

# Load the ALS Testing Data and TRaining Data
>setwd("C:/Users/xincli/Desktop/HS650/15_ALS_CaseStudy")
```

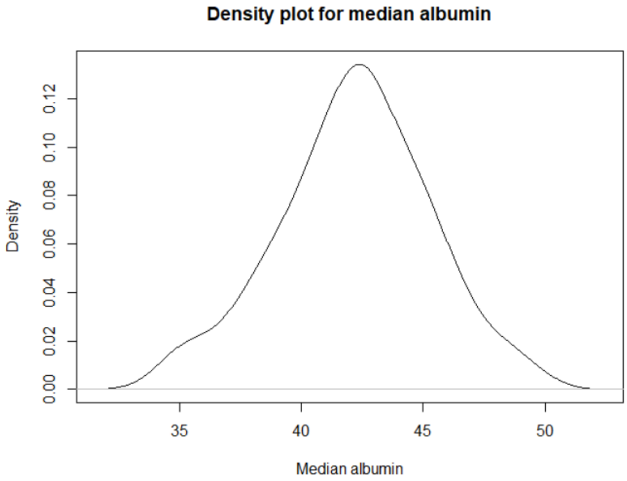
```
>TestingData <- read.csv("ALS_TestingData_78.csv")
>TrainingData <- read.csv("ALS_TrainingData_2223.csv")

#Q1
#Use Summary and plot certain variables with histogram, boxplot,
#density plot for testing data and save the summary statistics to the
#text file

>summary(TestingData) #leave out the output
>hist(TestingData$Age_mean, main = 'Histogram of mean age', xlab
= 'Mean age')
```




```
>boxplot(TestingData[,3:5], main = 'Boxplot for Albumin')
```



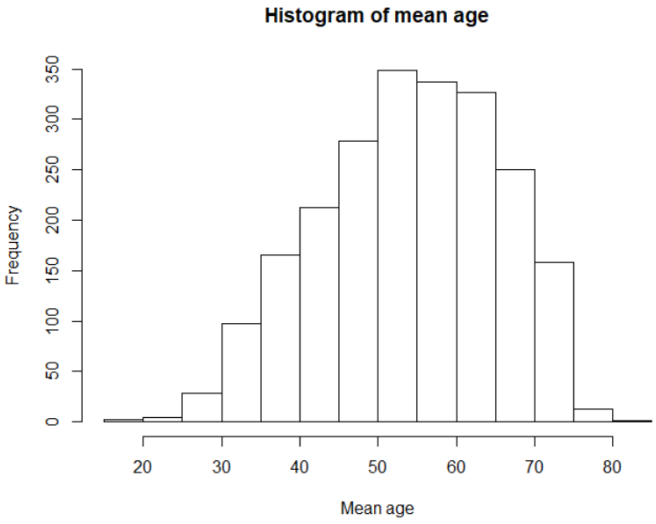

```
>plot(density(TestingData$Albumin_median), main = 'Density plot
for median albumin', xlab = 'Median albumin')
```



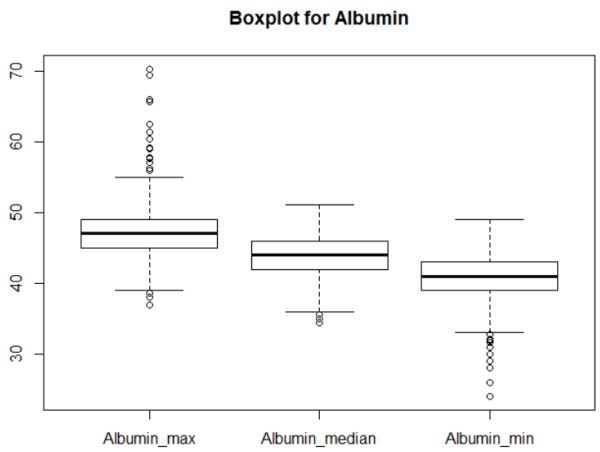
>write.table(summary(TestingData), file = 'C:/Users/xincli/Desktop/HS650/HW2/ALS_TestingData.txt')

#Use Summary and plot certain variables with histogram, boxplot, #density plot for training data and save the summary statistics to the #text file

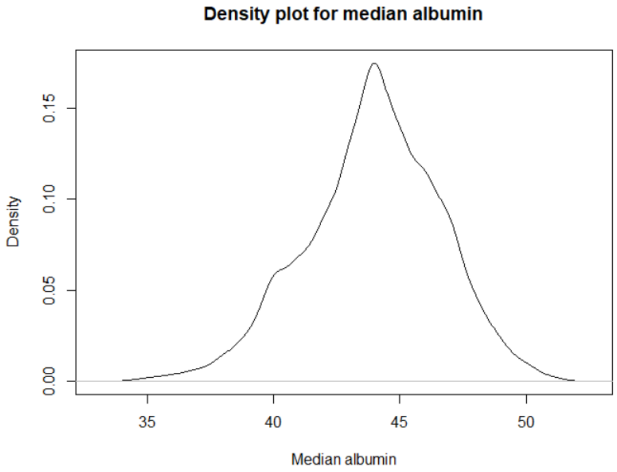
>summary(TrainingData) #leave out the output
>hist(TrainingData$Age_mean, main = 'Histogram of mean age', xlab = 'Mean age')



>boxplot(TrainingData[,3:5], main = 'Boxplot for Albumin')



>plot(density(TrainingData$Albumin_median), main = 'Density plot for median albumin', xlab = 'Median albumin')



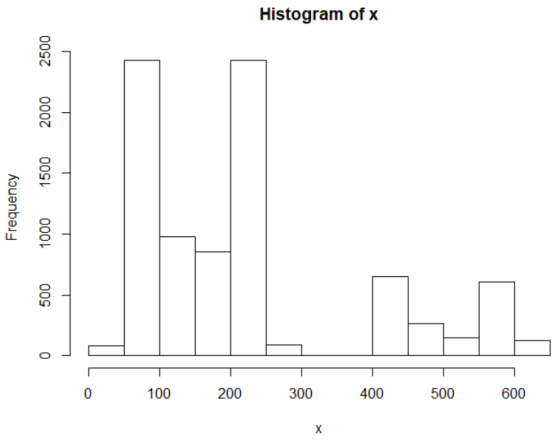
>write.table(summary(TrainingData),file= 'C:/Users/xincli/Desktop/HS650/HW2/ALS_TrainingData.txt')

Load the SOCR Knee Pain Data

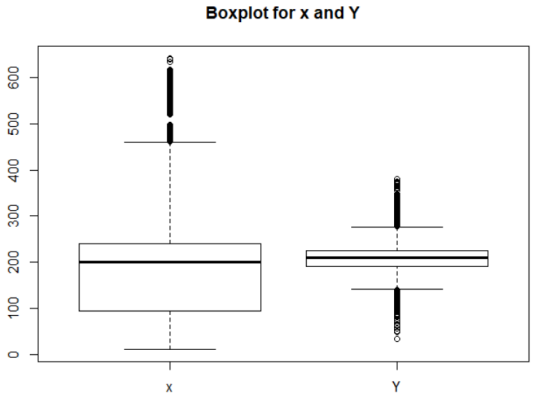
>wiki_url = read_html("http://wiki.socr.umich.edu/index.php/SOCR_Data_Knee
>PainData_041409")
>html_nodes(wiki_url, "#content")
>KneePain = html_table(html_nodes(wiki_url, "table")[[2]])
>KneePainData = as.data.frame(KneePain)

#Summarize, plot the dataset and save the summary statistics to the text file

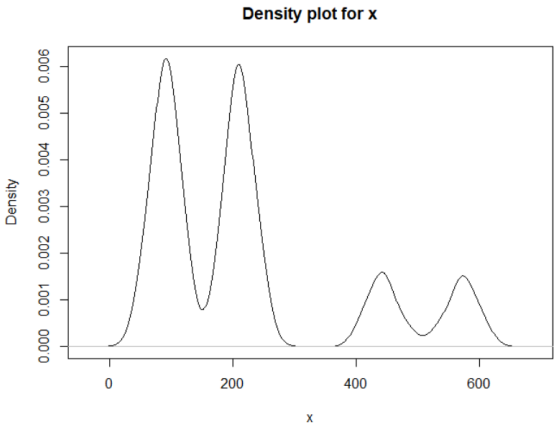
>summary(KneePainData) # leave out the output
>hist(KneePainData$x, main = 'Histogram of x', xlab = 'x')



>boxplot(KneePainData[,1:2], main = 'Boxplot for x and Y')



>plot(density(KneePainData$x), main = 'Density plot for x', xlab = 'x')

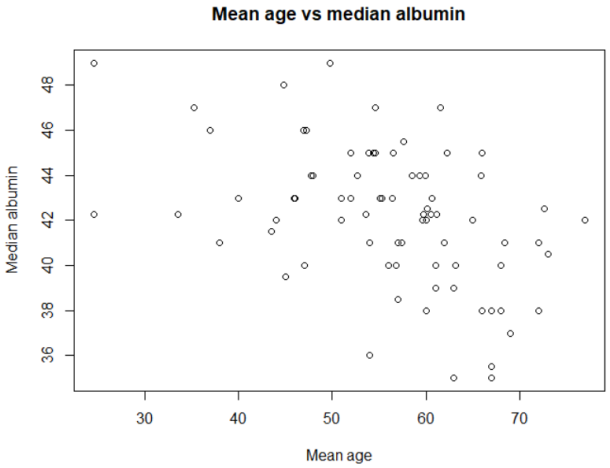


>write.table(summary(KneePainData), file = 'C:/Users/xincli/Desktop/HS650/HW2/SOCR_KneePainData.txt')

Q2
Use ALS case-study data and long-format SOCR Parkinsons Disease
#data(extract rows with Time=0)
to explore some bivariate relations (e.g. bivariate plot, correlation,

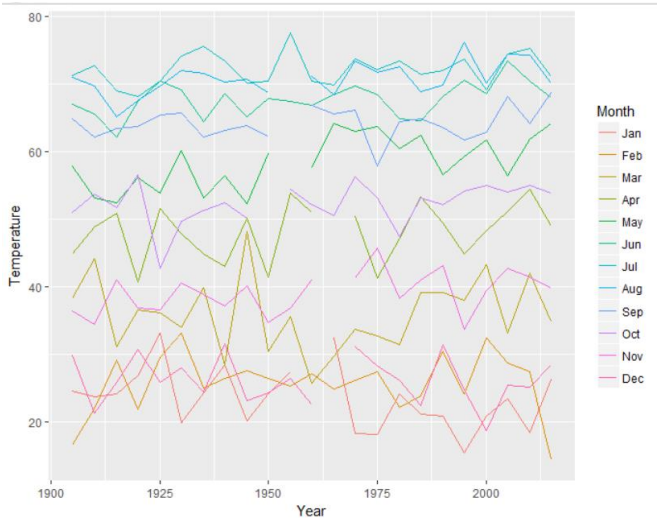
table, crosstable etc.)

```
>plot(x = TestingData$Age_mean, y = TestingData$Albumin_median,
      main = 'Mean age vs median albumin', xlab = 'Mean age', ylab
= 'Median albumin')
```

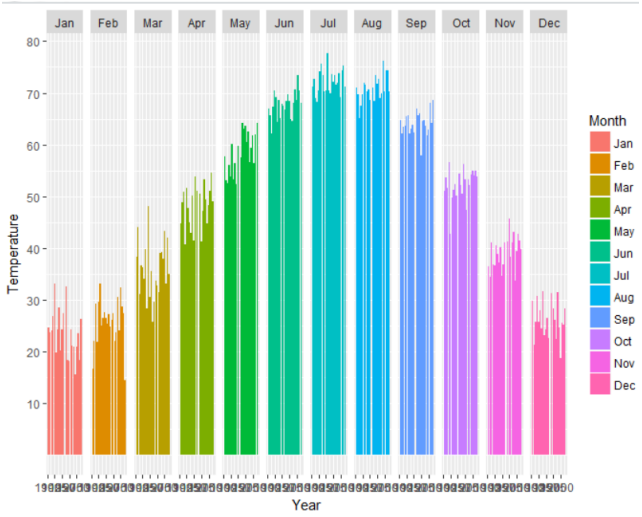



```
>cor(TestingData$Age_mean, TestingData$Albumin_median)
[1] -0.4506721

>t <- table(TestingData$Age_mean, TestingData$Albumin_median)
>head(t)
35 35.5 36 37 38 38.5 39 39.5 40 40.5 41 41.5 42 42.25 42.5 43
  24.58356164   0     0   0   0   0     0   0     0   0     0   0     0   0
0     0   0
  24.60547945   0     0   0   0   0     0   0     0   0     0   0     0   0
1     0   0
  33.51232877   0     0   0   0   0     0   0     0   0     0   0     0   0
1     0   0
  35.2630137    0     0   0   0   0     0   0     0   0     0   0     0   0
0     0   0
  37            0     0   0   0   0     0   0     0   0     0   0     0   0
0     0   0
  38            0     0   0   0   0     0   0     0   0     0   1     0   0
0     0   0

               44 45 45.5 46 47 48 49
  24.58356164   0   0     0   0   0   0   1
  24.60547945   0   0     0   0   0   0   0
  33.51232877   0   0     0   0   0   0   0
  35.2630137    0   0     0   0   1   0   0
  37            0   0     0   1   0   0   0
  38            0   0     0   0   0   0   0
```

```
>CrossTable(TestingData$Age_mean, TestingData$Albumin_median)
# leave out the output, too long

#Use07_UMich_AnnArbor_MI_TempPrecipitation_HistData_1900_20
#15 data to show the relations between
#temperature and time. [Hint: use geom_line and geom_bar]

>aa_temp_data<-
as.data.frame(read.csv("https://umich.instructure.com/files/706163
/download?download_frd=1", header=T, na.strings=c("", ".", "NA",
"NR")))
>b = seq(1, 111, 5)
>aa_temp_data1 = aa_temp_data[b,]
>aa_temp_data_new = melt(aa_temp_data1, id.vars = 'Year')
>colnames(aa_temp_data_new) = c('Year', 'Month', 'Temperature')
>aa_temp_data_new$Month = as.factor(aa_temp_data_new$Month)
>aa_temp_data_new$Temperature                                    =
as.numeric(aa_temp_data_new$Temperature)

>plot = ggplot(aa_temp_data_new, aes(Year, Temperature, group =
Month, color = Month)) + geom_line()
>plot
```




```
>bar = ggplot(aa_temp_data_new, aes(x = Year, y = Temperature,
fill = Month)) + geom_col() + facet_grid(. ~ Month) +
  scale_y_continuous(breaks = seq(10, 80, 10))
>bar
```




```
# Q3
# Introduce (artificially) some missing data, impute the missing
#values and examine the differences between the original,
# incomplete and imputed data in statistics.

> n = 1000
> m = 5
> data = matrix(data = rnorm(5000, 10, 1), 1000, 5)
> miss = sample(1:5000, 500)
> data[miss] = NA
> data = as.data.frame(data)
> summary(data)
       V1                  V2                  V3                  V4
 Min.   : 7.246   Min.   : 6.811   Min.   : 6.738   Min.   : 6.586
 1st Qu.: 9.345   1st Qu.: 9.372   1st Qu.: 9.314   1st Qu.: 9.378
 Median : 9.994   Median :10.001   Median :10.021   Median : 9.995
 Mean   : 9.964   Mean   :10.007   Mean   :10.024   Mean   :10.002
 3rd Qu.:10.618   3rd Qu.:10.636   3rd Qu.:10.766   3rd Qu.:10.700
 Max.   :12.972   Max.   :13.446   Max.   :13.716   Max.   :13.276
 NA's   :101      NA's   :88       NA's   :92       NA's   :105
       V5
 Min.   : 6.942
 1st Qu.: 9.287
 Median : 9.988
 Mean   : 9.998
 3rd Qu.:10.670
```

```
 Max.    :13.848
 NA's    :114
```

```
> mdf = missing_data.frame(data)
> show(mdf)
```

Object of class missing_data.frame with 1000 observations on 5 variables

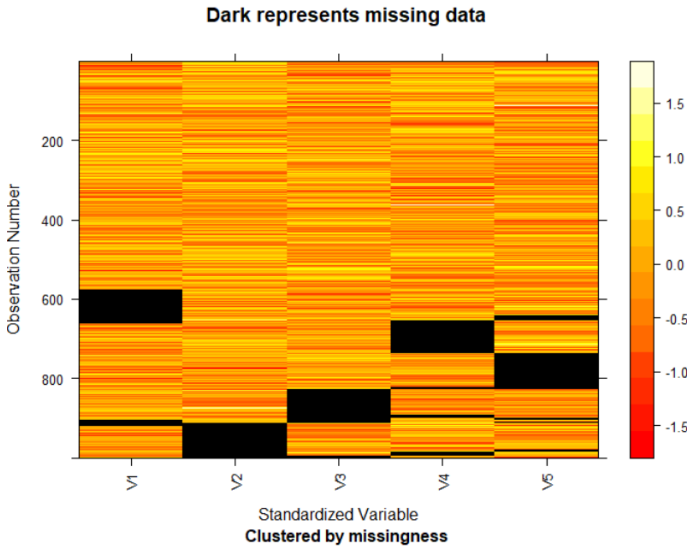
There are 19 missing data patterns

Append '@patterns' to this missing_data.frame to access the corresponding pattern for every observation or perhaps use table()

	type	missing	method	model
V1	continuous	101	ppd	linear
V2	continuous	88	ppd	linear
V3	continuous	92	ppd	linear
V4	continuous	105	ppd	linear
V5	continuous	114	ppd	linear

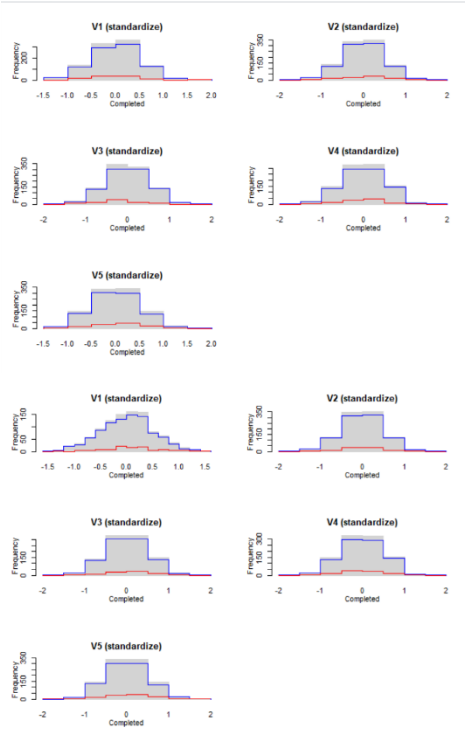
	family	link	transformation
V1	gaussian	identity	standardize
V2	gaussian	identity	standardize
V3	gaussian	identity	standardize
V4	gaussian	identity	standardize
V5	gaussian	identity	standardize

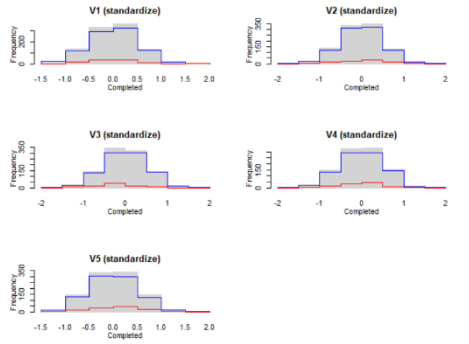
```
> image(mdf)
```




```
> imputations = mi(data, n.iter=5, n.chains=3, verbose=TRUE)
```

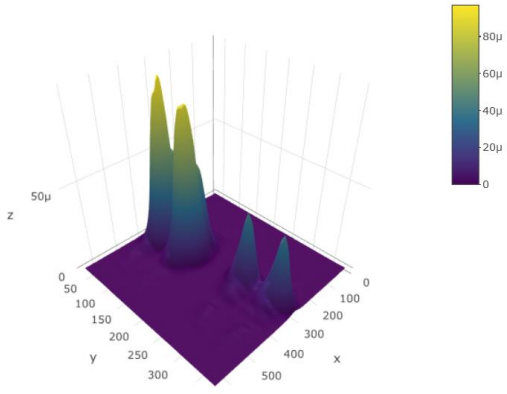
>hist(imputations)





Q4
Generate a surface plot for the SOCR Knee Pain Data illustrating
#the 2D distribution of locations of the patient
reported knee pain (use plotly and kernel density estimation).

>KneePainData$View = as.factor(KneePainData$View)
>kernal_density = with(KneePainData, MASS::kde2d(x, Y, n = 50))
>with(kernal_density, plot_ly(x=x, y=y, z=z, type="surface"))



#Q5
Rebalance the groups of ALS (training data) patients according to
#Age>50and Age≤50 using synthetic minority oversampling (SMOTE)
#to ensure approximately equal cohort sizes.

```
>TrainingData['age_over50'] <- ifelse(TrainingData$Age_mean <= 50, 1, 0)
>balanced_ALS <- ubBalance(X = TrainingData[,], Y = as.factor(TrainingData$age_over50),
                           type = "ubSMOTE", percOver = 100, percUnder = 200, verbose = TRUE)
> table(balanced_ALS[[2]])

   0    1
1576 1576

```