Chapter 3 Exercise KJ3.1 and KJ3.2

Group 3

6/3/2020

```
library(caret)
```

```
## Loading required package: lattice
```

```
## Loading required package: ggplot2
```

```
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------- tidyverse 1.2.1 --
```

```
## v tibble  2.1.3     v purrr   0.3.2
## v tidyr   0.8.3     v dplyr   0.8.3
## v readr   1.3.1     v stringr 1.4.0
## v tibble  2.1.3     v forcats 0.4.0
```

```
## -- Conflicts ------------------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## x purrr::lift()   masks caret::lift()
```

```
library(e1071)
library(knitr)
library(GGally)
```

```
## Warning: package 'GGally' was built under R version 3.6.3
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
##
## Attaching package: 'GGally'
```

```
## The following object is masked from 'package:dplyr':
##
##     nasa
```

```
library(VIM)
```

```
## Warning: package 'VIM' was built under R version 3.6.3
```

```
## Loading required package: colorspace
```

```
## Loading required package: grid

## Loading required package: data.table

##
## Attaching package: 'data.table'

## The following objects are masked from 'package:dplyr':
##
##     between, first, last

## The following object is masked from 'package:purrr':
##
##     transpose

## VIM is ready to use.
##  Since version 4.0.0 the GUI is in its own package VIMGUI.
##
##           Please use the package to use the new (and old) GUI.

## Suggestions and bug-reports can be submitted at: https://github.com/alexkowa/VIM/issues

##
## Attaching package: 'VIM'

## The following object is masked from 'package:datasets':
##
##     sleep
```

Exercise 3.1

3.1. The UC Irvine Machine Learning Repository6 contains a data set related to glass identification. The data consist of 214 glass samples labeled as one of seven class categories. There are nine predictors, including the refractive index and percentages of eight elements: Na, Mg, Al, Si, K, Ca, Ba, and Fe. The data can be accessed via:

```
library(mlbench)
```

```
## Warning: package 'mlbench' was built under R version 3.6.3
```

```
data(Glass)
str(Glass)
```

```
## 'data.frame':    214 obs. of  10 variables:
##  $ RI  : num  1.52 1.52 1.52 1.52 1.52 ...
##  $ Na  : num  13.6 13.9 13.5 13.2 13.3 ...
##  $ Mg  : num  4.49 3.6 3.55 3.69 3.62 3.61 3.6 3.61 3.58 3.6 ...
##  $ Al  : num  1.1 1.36 1.54 1.29 1.24 1.62 1.14 1.05 1.37 1.36 ...
##  $ Si  : num  71.8 72.7 73 72.6 73.1 ...
##  $ K   : num  0.06 0.48 0.39 0.57 0.55 0.64 0.58 0.57 0.56 0.57 ...
##  $ Ca  : num  8.75 7.83 7.78 8.22 8.07 8.07 8.17 8.24 8.3 8.4 ...
##  $ Ba  : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ Fe  : num  0 0 0 0 0 0.26 0 0 0 0.11 ...
##  $ Type: Factor w/ 6 levels "1","2","3","5",..: 1 1 1 1 1 1 1 1 1 1 ...
```

The structure of the Glass data shows that all the predictors are numeric. The dependent variable is factor of 6 levels.

```
head(Glass)
```

```
##        RI    Na   Mg   Al    Si    K   Ca Ba   Fe Type
## 1 1.52101 13.64 4.49 1.10 71.78 0.06 8.75  0 0.00    1
## 2 1.51761 13.89 3.60 1.36 72.73 0.48 7.83  0 0.00    1
## 3 1.51618 13.53 3.55 1.54 72.99 0.39 7.78  0 0.00    1
## 4 1.51766 13.21 3.69 1.29 72.61 0.57 8.22  0 0.00    1
## 5 1.51742 13.27 3.62 1.24 73.08 0.55 8.07  0 0.00    1
## 6 1.51596 12.79 3.61 1.62 72.97 0.64 8.07  0 0.26    1
```

The head function displays the first values of each variables.

(a) Using visualizations, explore the predictor variables to understand their distributions as well as the relationships between predictors.

We remove the target variable

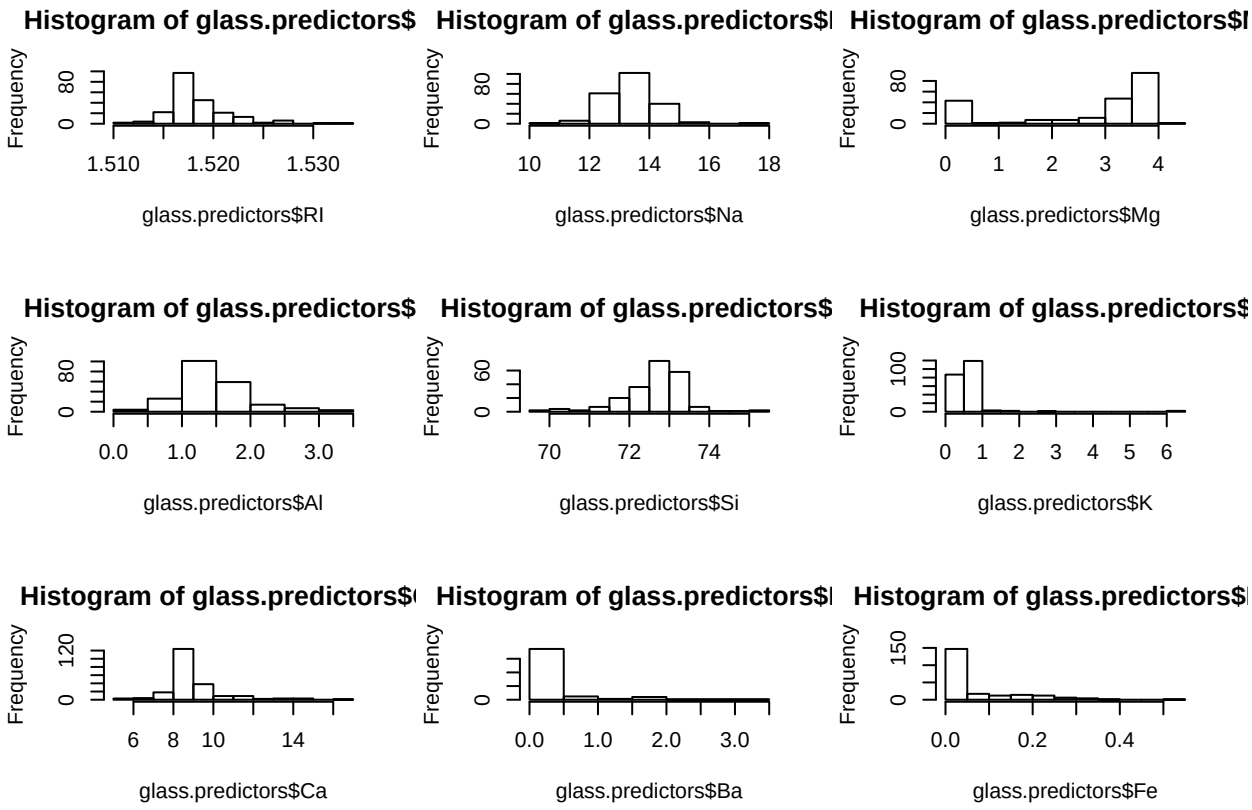
```
glass.predictors <- Glass[,-10]
```

Since all variables are numerical, we can use the skewness function of e1071 to estimated the predictors to represent.

```
library(e1071)
skewValues <- apply(glass.predictors, 2, skewness)
skewValues
```

```
##         RI         Na         Mg         Al         Si          K
##  1.6027151  0.4478343 -1.1364523  0.8946104 -0.7202392  6.4600889
##         Ca         Ba         Fe
##  2.0184463  3.3686800  1.7298107
```

```
par(mfrow = c(3,3))
hist(x = glass.predictors$RI)
hist(x = glass.predictors$Na)
hist(x = glass.predictors$Mg)
hist(x = glass.predictors$Al)
hist(x = glass.predictors$Si)
hist(x = glass.predictors$K)
hist(x = glass.predictors$Ca)
hist(x = glass.predictors$Ba)
hist(x = glass.predictors$Fe)
```



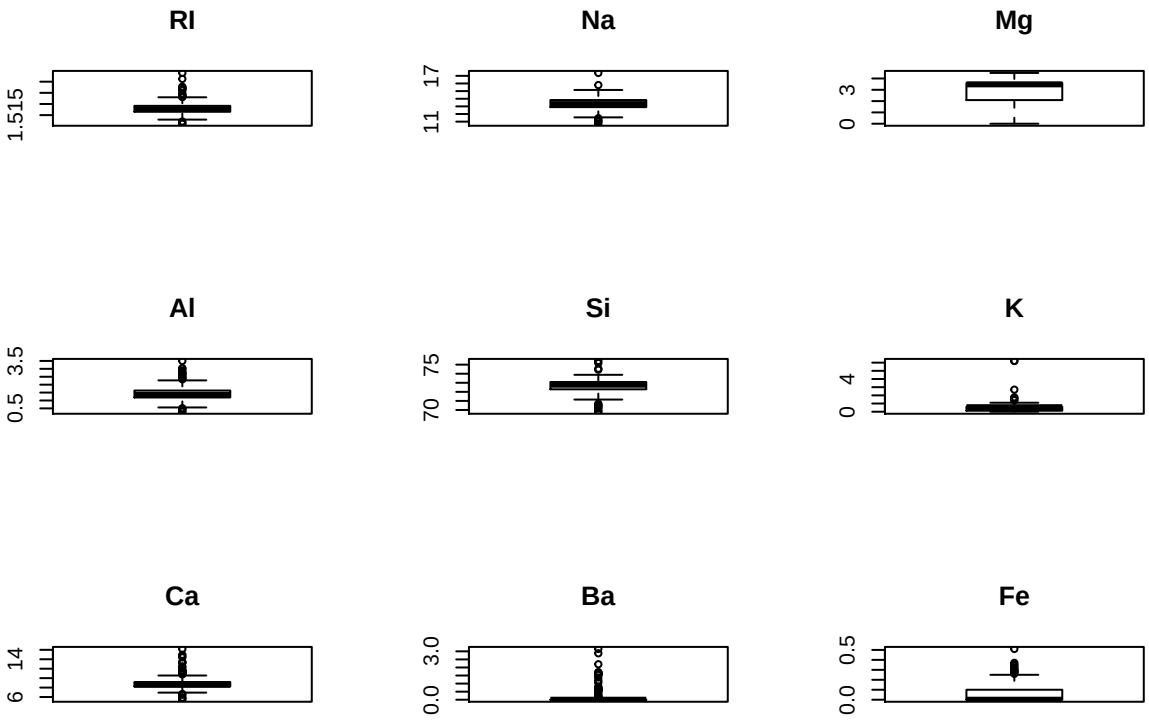
The predictors RI, Na, Al, Si and Ca are normal distributed The predictors K, Ba, and Fe are rigtht skewed. we can aplied the log function on those variable to normalise or Boxcox to centralise, scale and transform.

The Mg predictor need to be centralise and scale. It is neither normal, nor skewed.

(b) Do there appear to be any outliers in the data? Are any predictors skewed?

Looking for outliers

```
par(mfrow = c(3,3))
boxplot(x = glass.predictors$RI, main = "RI")
boxplot(x = glass.predictors$Na, main = "Na")
boxplot(x = glass.predictors$Mg, main = "Mg")
boxplot(x = glass.predictors$Al, main = "Al")
boxplot(x = glass.predictors$Si, main = "Si")
boxplot(x = glass.predictors$K, main = "K")
boxplot(x = glass.predictors$Ca, main = "Ca")
boxplot(x = glass.predictors$Ba, main = "Ba")
boxplot(x = glass.predictors$Fe, main = "Fe")
```



The boxplot graphs shows some outliers with the predictors RI, Na, Al, Si, K, Ca, Ba, and Fe. The outlier of Ba and K are extreme.

```
summary(glass.predictors)
```

```
##        RI              Na              Mg              Al        
##  Min.   :1.511   Min.   :10.73   Min.   :0.000   Min.   :0.290  
##  1st Qu.:1.517   1st Qu.:12.91   1st Qu.:2.115   1st Qu.:1.190  
##  Median :1.518   Median :13.30   Median :3.480   Median :1.360  
##  Mean   :1.518   Mean   :13.41   Mean   :2.685   Mean   :1.445  
##  3rd Qu.:1.519   3rd Qu.:13.82   3rd Qu.:3.600   3rd Qu.:1.630  
##  Max.   :1.534   Max.   :17.38   Max.   :4.490   Max.   :3.500  
##        Si              K                Ca              Ba        
##  Min.   :69.81   Min.   :0.0000   Min.   : 5.430   Min.   :0.000  
##  1st Qu.:72.28   1st Qu.:0.1225   1st Qu.: 8.240   1st Qu.:0.000  
##  Median :72.79   Median :0.5550   Median : 8.600   Median :0.000  
##  Mean   :72.65   Mean   :0.4971   Mean   : 8.957   Mean   :0.175  
##  3rd Qu.:73.09   3rd Qu.:0.6100   3rd Qu.: 9.172   3rd Qu.:0.000  
##  Max.   :75.41   Max.   :6.2100   Max.   :16.190   Max.   :3.150  
##        Fe        
##  Min.   :0.00000  
##  1st Qu.:0.00000  
##  Median :0.00000  
##  Mean   :0.05701  
##  3rd Qu.:0.10000  
##  Max.   :0.51000
```

To visualize the correlation between predictors, we use the corrplot function in the package of the same name.

```
correlations <- cor(glass.predictors)
correlations
```

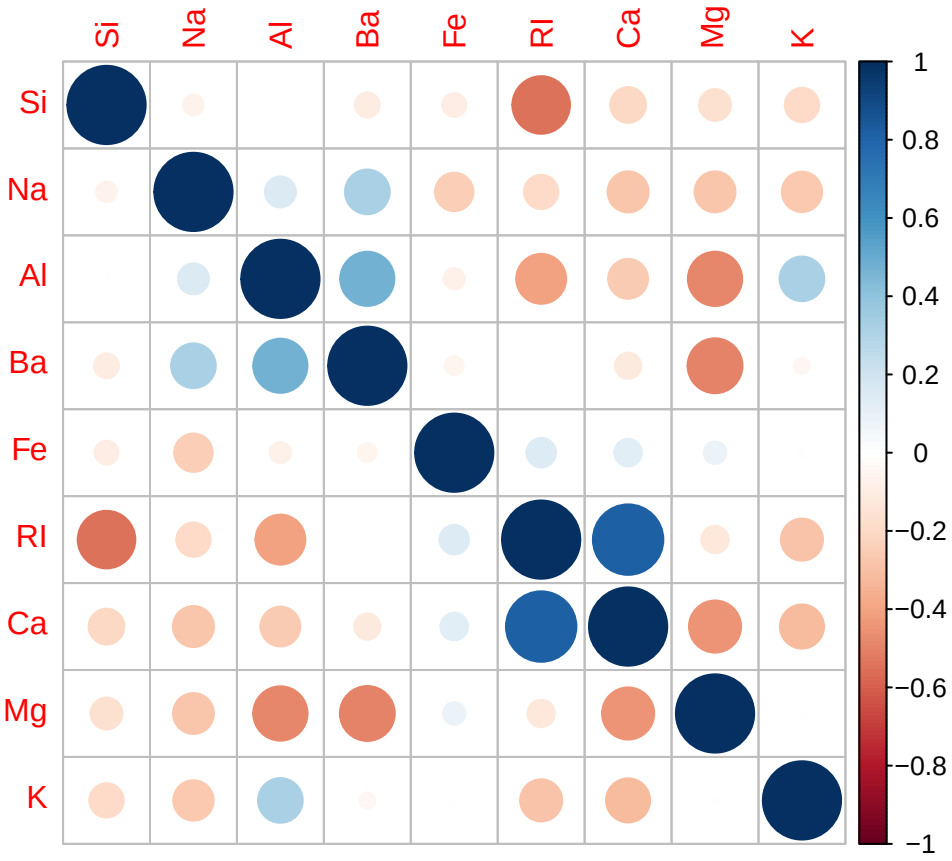
```
##               RI          Na           Mg          Al          Si
## RI  1.0000000000 -0.19188538 -0.122274039 -0.40732603 -0.54205220
## Na -0.1918853790  1.00000000 -0.273731961  0.15679367 -0.06980881
## Mg -0.1222740393 -0.27373196  1.000000000 -0.48179851 -0.16592672
## Al -0.4073260341  0.15679367 -0.481798509  1.00000000 -0.00552372
## Si -0.5420521997 -0.06980881 -0.165926723 -0.00552372  1.00000000
## K  -0.2898327111 -0.26608650  0.005395667  0.32595845 -0.19333085
## Ca  0.8104026963 -0.27544249 -0.443750026 -0.25959201 -0.20873215
## Ba -0.0003860189  0.32660288 -0.492262118  0.47940390 -0.10215131
## Fe  0.1430096093 -0.24134641  0.083059529 -0.07440215 -0.09420073
##                K         Ca            Ba           Fe
## RI -0.289832711  0.8104027 -0.0003860189  0.143009609
## Na -0.266086504 -0.2754425  0.3266028795 -0.241346411
## Mg  0.005395667 -0.4437500 -0.4922621178  0.083059529
## Al  0.325958446 -0.2595920  0.4794039017 -0.074402151
## Si -0.193330854 -0.2087322 -0.1021513105 -0.094200731
## K   1.000000000 -0.3178362 -0.0426180594 -0.007719049
## Ca -0.317836155  1.0000000 -0.1128409671  0.124968219
## Ba -0.042618059 -0.1128410  1.0000000000 -0.058691755
## Fe -0.007719049  0.1249682 -0.0586917554  1.000000000
```

```
library(corrplot)
```

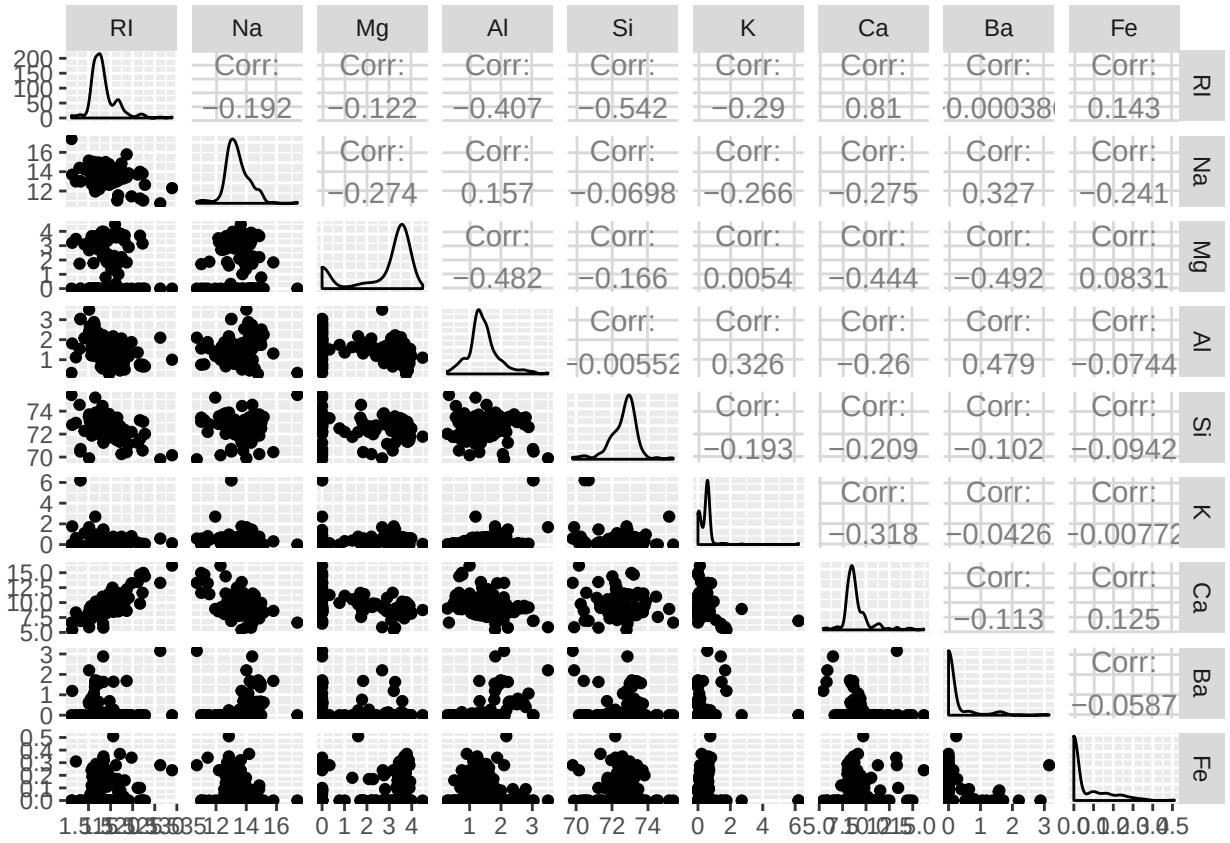
```
## Warning: package 'corrplot' was built under R version 3.6.3
```

```
## corrplot 0.84 loaded
```

```
corrplot(correlations, order = "hclust")
```




```
GGally::ggpairs(as.data.frame(glass.predictors))
```



The only notable correlation is between RI and Ca.

(c) Are there any relevant transformations of one or more predictors that might improve the classification model?

We use the powerTransform of the car package that calculates the Box-Cox transformation. The Box-Cox transformation uses the maximum likelihood approach and returns information on the estimated values along with convenient rounded values that are within 1.96 standard deviations of the maximum likelihood estimate.

```
library(car)
```

```
## Warning: package 'car' was built under R version 3.6.3
```

```
## Loading required package: carData
```

```
## Warning: package 'carData' was built under R version 3.6.3
```

```
##
## Attaching package: 'car'
```

```
## The following object is masked from 'package:dplyr':
##
##     recode
```

```
## The following object is masked from 'package:purrr':
##
##     some
```

```
summary(powerTransform(Glass[,1:9], family="yjPower"))$result[,1:2]
```

```
##      Est Power Rounded Pwr
## RI -25.0853114      -25.09
## Na   1.3755562        1.00
## Mg   1.7699080        2.00
## Al   0.9773267        1.00
## Si  10.9452696       10.95
## K   -0.1441078        0.00
## Ca   0.6774333        0.50
## Ba  -6.8620464       -6.86
## Fe -14.9245600      -14.92
```

The suggested transformations are:

No transformation for RI, Na, Si, and K since lambda=1. Log transformations for Mg, K, Ba, and Fe since lambda =0. Square root transformation for Ca since lambda = 0.5.

Exercise 3.2.

3.2. The soybean data can also be found at the UC Irvine Machine Learning Repository. Data were collected to predict disease in 683 soybeans. The 35 predictors are mostly categorical and include information on the environmental conditions (e.g., temperature, precipitation) and plant conditions (e.g., left spots, mold growth). The outcome labels consist of 19 distinct classes.

The data can be loaded via:

```
library(mlbench)
data(Soybean)
## See ?Soybean for details
str(Soybean)
```

```
## 'data.frame':    683 obs. of  36 variables:
##  $ Class          : Factor w/ 19 levels "2-4-d-injury",..: 11 11 11 11 11 11 11 11 11 11 ...
##  $ date           : Factor w/ 7 levels "0","1","2","3",..: 7 5 4 4 7 6 6 5 7 5 ...
##  $ plant.stand    : Ord.factor w/ 2 levels "0"<"1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ precip         : Ord.factor w/ 3 levels "0"<"1"<"2": 3 3 3 3 3 3 3 3 3 3 ...
##  $ temp           : Ord.factor w/ 3 levels "0"<"1"<"2": 2 2 2 2 2 2 2 2 2 2 ...
##  $ hail           : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 2 1 1 ...
##  $ crop.hist      : Factor w/ 4 levels "0","1","2","3": 2 3 2 2 3 4 3 2 4 3 ...
##  $ area.dam       : Factor w/ 4 levels "0","1","2","3": 2 1 1 1 1 1 1 1 1 1 ...
##  $ sever          : Factor w/ 3 levels "0","1","2": 2 3 3 3 2 2 2 2 2 3 ...
##  $ seed.tmt       : Factor w/ 3 levels "0","1","2": 1 2 2 1 1 1 2 1 2 1 ...
##  $ germ           : Ord.factor w/ 3 levels "0"<"1"<"2": 1 2 3 2 3 2 1 3 2 3 ...
##  $ plant.growth   : Factor w/ 2 levels "0","1": 2 2 2 2 2 2 2 2 2 2 ...
##  $ leaves         : Factor w/ 2 levels "0","1": 2 2 2 2 2 2 2 2 2 2 ...
##  $ leaf.halo      : Factor w/ 3 levels "0","1","2": 1 1 1 1 1 1 1 1 1 1 ...
##  $ leaf.marg      : Factor w/ 3 levels "0","1","2": 3 3 3 3 3 3 3 3 3 3 ...
```

```
##  $ leaf.size      : Ord.factor w/ 3 levels "0"<"1"<"2": 3 3 3 3 3 3 3 3 3 3 ...
##  $ leaf.shread    : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 ...
##  $ leaf.malf      : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 ...
##  $ leaf.mild      : Factor w/ 3 levels "0","1","2": 1 1 1 1 1 1 1 1 1 ...
##  $ stem           : Factor w/ 2 levels "0","1": 2 2 2 2 2 2 2 2 2 ...
##  $ lodging        : Factor w/ 2 levels "0","1": 2 1 1 1 1 1 2 1 1 1 ...
##  $ stem.cankers   : Factor w/ 4 levels "0","1","2","3": 4 4 4 4 4 4 4 4 4 4 ...
##  $ canker.lesion  : Factor w/ 4 levels "0","1","2","3": 2 2 1 1 2 1 2 2 2 2 ...
##  $ fruiting.bodies: Factor w/ 2 levels "0","1": 2 2 2 2 2 2 2 2 2 2 ...
##  $ ext.decay      : Factor w/ 3 levels "0","1","2": 2 2 2 2 2 2 2 2 2 2 ...
##  $ mycelium       : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 ...
##  $ int.discolor   : Factor w/ 3 levels "0","1","2": 1 1 1 1 1 1 1 1 1 1 ...
##  $ sclerotia      : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 ...
##  $ fruit.pods     : Factor w/ 4 levels "0","1","2","3": 1 1 1 1 1 1 1 1 1 1 ...
##  $ fruit.spots    : Factor w/ 4 levels "0","1","2","4": 4 4 4 4 4 4 4 4 4 4 ...
##  $ seed           : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 ...
##  $ mold.growth    : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 ...
##  $ seed.discolor  : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 ...
##  $ seed.size      : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 ...
##  $ shriveling     : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 ...
##  $ roots          : Factor w/ 3 levels "0","1","2": 1 1 1 1 1 1 1 1 1 ...
```

All variables are factors or ordered factors.

```
head(Soybean)
```

```
##                   Class date plant.stand precip temp hail crop.hist
## 1 diaporthe-stem-canker    6           0      2    1    0         1
## 2 diaporthe-stem-canker    4           0      2    1    0         2
## 3 diaporthe-stem-canker    3           0      2    1    0         1
## 4 diaporthe-stem-canker    3           0      2    1    0         1
## 5 diaporthe-stem-canker    6           0      2    1    0         2
## 6 diaporthe-stem-canker    5           0      2    1    0         3
##   area.dam sever seed.tmt germ plant.growth leaves leaf.halo leaf.marg
## 1        1     1        0    0            1      1         0         2
## 2        0     2        1    1            1      1         0         2
## 3        0     2        1    2            1      1         0         2
## 4        0     2        0    1            1      1         0         2
## 5        0     1        0    2            1      1         0         2
## 6        0     1        0    1            1      1         0         2
##   leaf.size leaf.shread leaf.malf leaf.mild stem lodging stem.cankers
## 1         2           0         0         0    1       1           3
## 2         2           0         0         0    1       0           3
## 3         2           0         0         0    1       0           3
## 4         2           0         0         0    1       0           3
## 5         2           0         0         0    1       0           3
## 6         2           0         0         0    1       0           3
##   canker.lesion fruiting.bodies ext.decay mycelium int.discolor sclerotia
## 1             1               1         1        0           0         0
## 2             1               1         1        0           0         0
## 3             0               1         1        0           0         0
## 4             0               1         1        0           0         0
## 5             1               1         1        0           0         0
```

```
## 6             0              1        1       0           0        0
##   fruit.pods fruit.spots seed mold.growth seed.discolor seed.size
## 1          0           4    0           0             0         0
## 2          0           4    0           0             0         0
## 3          0           4    0           0             0         0
## 4          0           4    0           0             0         0
## 5          0           4    0           0             0         0
## 6          0           4    0           0             0         0
##   shriveling roots
## 1          0     0
## 2          0     0
## 3          0     0
## 4          0     0
## 5          0     0
## 6          0     0
```

(a) Investigate the frequency distributions for the categorical predictors. Are any of the distributions degenerate in the ways discussed earlier in this chapter?

```
nearZeroVar(Soybean,saveMetric=TRUE)
```

```
##                 freqRatio percentUnique zeroVar   nzv
## Class            1.010989     2.7818448   FALSE FALSE
## date             1.137405     1.0248902   FALSE FALSE
## plant.stand      1.208191     0.2928258   FALSE FALSE
## precip           4.098214     0.4392387   FALSE FALSE
## temp             1.879397     0.4392387   FALSE FALSE
## hail             3.425197     0.2928258   FALSE FALSE
## crop.hist        1.004587     0.5856515   FALSE FALSE
## area.dam         1.213904     0.5856515   FALSE FALSE
## sever            1.651282     0.4392387   FALSE FALSE
## seed.tmt         1.373874     0.4392387   FALSE FALSE
## germ             1.103627     0.4392387   FALSE FALSE
## plant.growth     1.951327     0.2928258   FALSE FALSE
## leaves           7.870130     0.2928258   FALSE FALSE
## leaf.halo        1.547511     0.4392387   FALSE FALSE
## leaf.marg        1.615385     0.4392387   FALSE FALSE
## leaf.size        1.479638     0.4392387   FALSE FALSE
## leaf.shread      5.072917     0.2928258   FALSE FALSE
## leaf.malf       12.311111     0.2928258   FALSE FALSE
## leaf.mild       26.750000     0.4392387   FALSE  TRUE
## stem             1.253378     0.2928258   FALSE FALSE
## lodging         12.380952     0.2928258   FALSE FALSE
## stem.cankers     1.984293     0.5856515   FALSE FALSE
## canker.lesion    1.807910     0.5856515   FALSE FALSE
## fruiting.bodies  4.548077     0.2928258   FALSE FALSE
## ext.decay        3.681481     0.4392387   FALSE FALSE
## mycelium       106.500000     0.2928258   FALSE  TRUE
## int.discolor    13.204545     0.4392387   FALSE FALSE
```

```
## sclerotia        31.250000     0.2928258   FALSE  TRUE
## fruit.pods        3.130769     0.5856515   FALSE FALSE
## fruit.spots       3.450000     0.5856515   FALSE FALSE
## seed              4.139130     0.2928258   FALSE FALSE
## mold.growth       7.820896     0.2928258   FALSE FALSE
## seed.discolor     8.015625     0.2928258   FALSE FALSE
## seed.size         9.016949     0.2928258   FALSE FALSE
## shriveling       14.184211     0.2928258   FALSE FALSE
## roots             6.406977     0.4392387   FALSE FALSE
```

The predictors that correspond respectively to the position of variables 19, 26, 28 in the datafarme Soybean which are degenerate, are leaf.mild, mycelium and sclerotia.

```
summary(Soybean)
```

```
##                  Class          date     plant.stand  precip      temp
##  brown-spot         : 92   5      :149   0   :354    0   : 74   0   : 80
##  alternarialeaf-spot: 91   4      :131   1   :293    1   :112   1   :374
##  frog-eye-leaf-spot : 91   3      :118   NA's: 36    2   :459   2   :199
##  phytophthora-rot   : 88   2      : 93               NA's: 38   NA's: 30
##  anthracnose        : 44   6      : 90
##  brown-stem-rot     : 44   (Other):101
##  (Other)            :233   NA's   :  1
##    hail     crop.hist  area.dam    sever     seed.tmt     germ
##  0   :435   0   : 65   0   :123   0   :195   0   :305   0   :165
##  1   :127   1   :165   1   :227   1   :322   1   :222   1   :213
##  NA's:121   2   :219   2   :145   2   : 45   2   : 35   2   :193
##             3   :218   3   :187   NA's:121   NA's:121   NA's:112
##             NA's: 16   NA's:  1
##
##
##  plant.growth leaves  leaf.halo  leaf.marg  leaf.size  leaf.shread
##  0   :441     0: 77   0   :221   0   :357   0   : 51   0   :487
##  1   :226     1:606   1   : 36   1   : 21   1   :327   1   : 96
##  NA's: 16             2   :342   2   :221   2   :221   NA's:100
##                       NA's: 84   NA's: 84   NA's: 84
##
##
##
##  leaf.malf  leaf.mild    stem     lodging    stem.cankers canker.lesion
##  0   :554   0   :535   0   :296   0   :520   0   :379     0   :320
##  1   : 45   1   : 20   1   :371   1   : 42   1   : 39     1   : 83
##  NA's: 84   2   : 20   NA's: 16   NA's:121   2   : 36     2   :177
##             NA's:108                         3   :191     3   : 65
##                                              NA's: 38     NA's: 38
##
##
##  fruiting.bodies ext.decay  mycelium   int.discolor sclerotia  fruit.pods
##  0   :473        0   :497   0   :639   0   :581     0   :625   0   :407
##  1   :104        1   :135   1   :  6   1   : 44     1   : 20   1   :130
##  NA's:106        2   : 13   NA's: 38   2   : 20     NA's: 38   2   : 14
##                  NA's: 38              NA's: 38                3   : 48
##                                                                NA's: 84
```

```
##
##
##  fruit.spots   seed     mold.growth seed.discolor seed.size  shriveling
##  0   :345    0   :476   0   :524    0   :513      0   :532   0   :539
##  1   : 75    1   :115   1   : 67    1   : 64      1   : 59   1   : 38
##  2   : 57    NA's: 92   NA's: 92    NA's:106      NA's: 92   NA's:106
##  4   :100
##  NA's:106
##
##
##   roots
##  0   :551
##  1   : 86
##  2   : 15
##  NA's: 31
##
##
##
```

Using the summary of Soybean, the fraction of unique values over the sample size of the predictors is low. There are 2, 3,0r 4 unique values over 683 observations.

The predictors leaf.mild, mycelium and sclerotia have the ratio of the frequency the most prevalent value to the frequency of the second most prevalent very large.

```
imbalance.leaf.mild = 535/20
imbalance.leaf.mild
```

```
## [1] 26.75
```

```
imbalance.mycelium = 639/6
imbalance.mycelium
```

```
## [1] 106.5
```

```
imbalance.sclerotia = 625/20
imbalance.sclerotia
```

```
## [1] 31.25
```

The three predictors have a very strong imbalance. These are near-zero variance predictors

We can observe these large imbalance between uniques values in the plots below.

```
par(mfrow = c(3,3))
plot(x = Soybean$leaves) + title(main = 'leaves')
```

```
## numeric(0)
```

```
plot(x = Soybean$leaf.malf) + title(main = 'leaf.malf')
```

```
## numeric(0)
```

```
plot(x = Soybean$leaf.mild) + title(main = 'leaf.mild')
```

```
## numeric(0)
```

```
plot(x = Soybean$lodging) + title(main = 'lodging')
```

```
## numeric(0)
```

```
plot(x = Soybean$mycelium) + title(main = 'mycelium')
```

```
## numeric(0)
```

```
plot(x = Soybean$int.discolor)+ title(main = 'int.discolor')
```

```
## numeric(0)
```

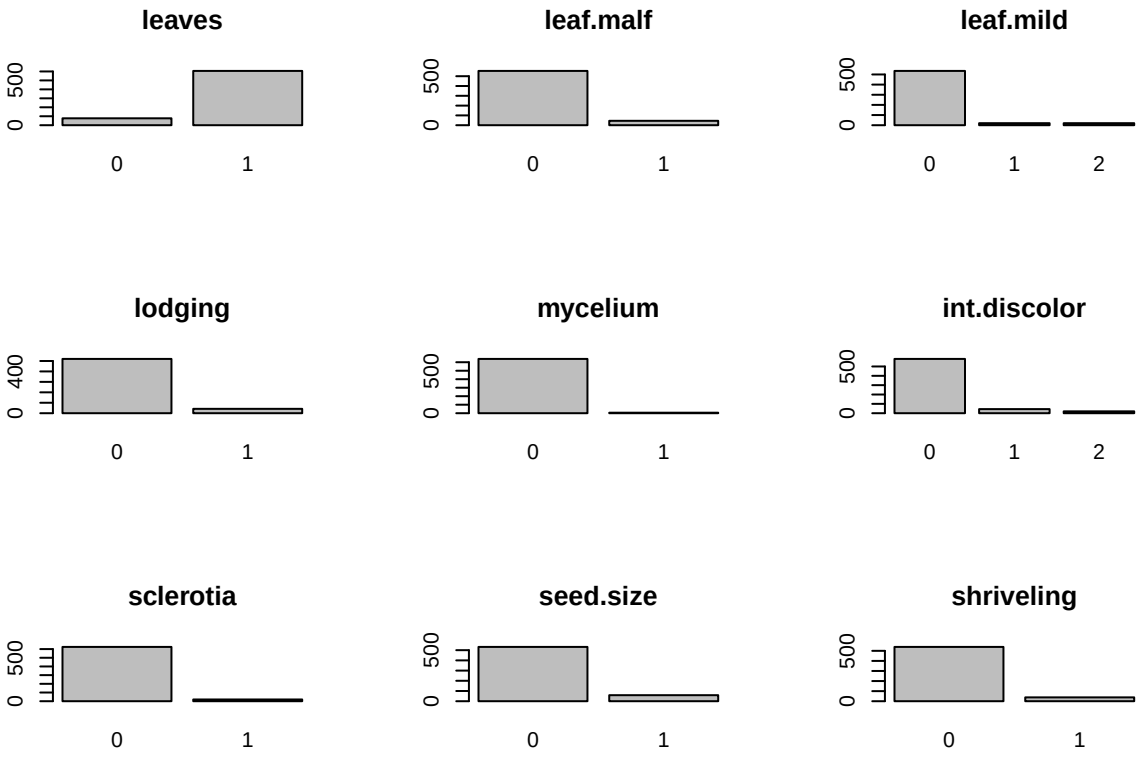
```
plot(x = Soybean$sclerotia) + title(main = 'sclerotia')
```

```
## numeric(0)
```

```
plot(x = Soybean$seed.size) + title(main = 'seed.size')
```

```
## numeric(0)
```

```
plot(x = Soybean$shriveling) + title(main = 'shriveling')
```



```
## numeric(0)
```

(b) Roughly 18% of the data are missing. Are there particular predictors that are more likely to be missing? Is the pattern of missing data related to the classes?

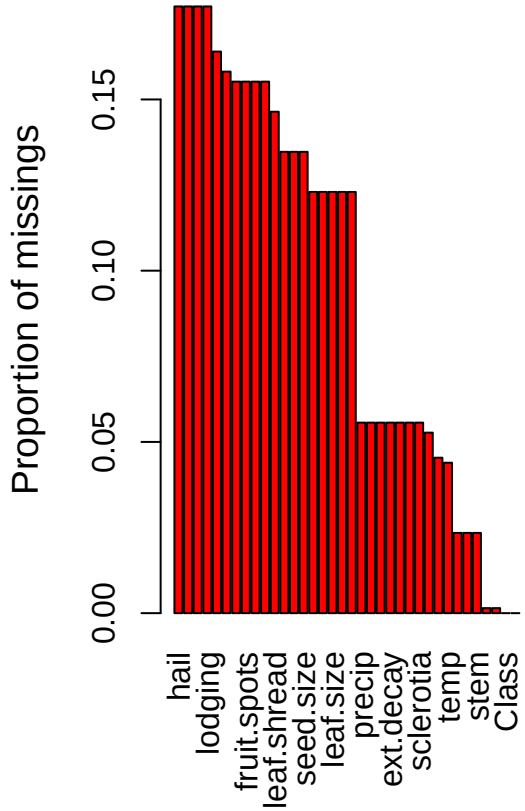
Obsevation of missing values

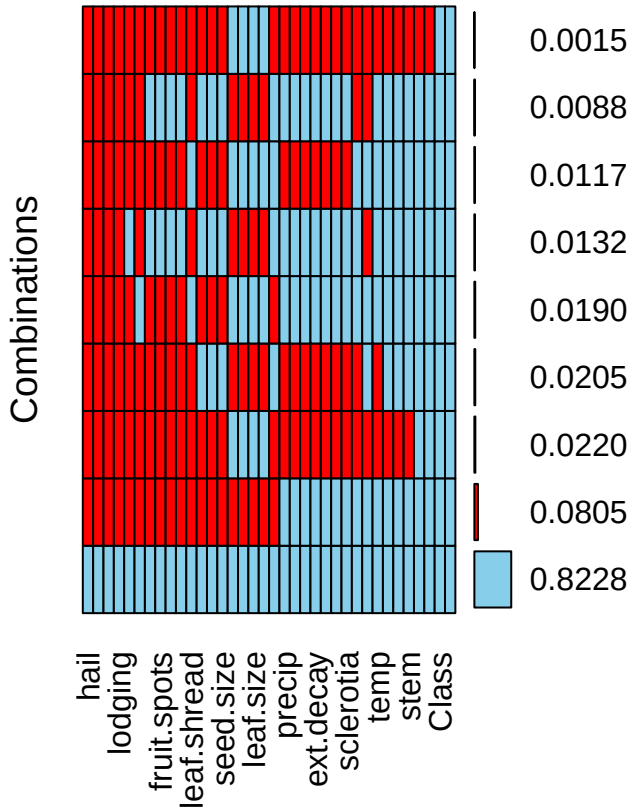
```
colSums(is.na(Soybean))
```

```
##           Class            date     plant.stand          precip
##               0               1              36              38
##            temp            hail       crop.hist        area.dam
##              30             121              16               1
##           sever        seed.tmt            germ    plant.growth
##             121             121             112              16
##          leaves       leaf.halo       leaf.marg       leaf.size
##               0              84              84              84
##     leaf.shread       leaf.malf       leaf.mild            stem
##             100              84             108              16
##         lodging    stem.cankers   canker.lesion fruiting.bodies
##             121              38              38             106
##       ext.decay        mycelium    int.discolor       sclerotia
##              38              38              38              38
##      fruit.pods     fruit.spots            seed     mold.growth
##              84             106              92              92
##   seed.discolor       seed.size      shriveling           roots
##             106              92             106              31
```

The aggr function in the VIM package plots and calculates the amount of missing values in each variable. The dply function is useful for wrangling data into aggregate summaries and is used to find the pattern of missing data related to the classes.

```
aggr(Soybean, prop = c(TRUE, TRUE), bars=TRUE, numbers=TRUE, sortVars=TRUE)
```






```
##
##  Variables sorted by number of missings:
##          Variable       Count
##              hail 0.177159590
##             sever 0.177159590
##          seed.tmt 0.177159590
##           lodging 0.177159590
##              germ 0.163982430
##         leaf.mild 0.158125915
##   fruiting.bodies 0.155197657
##       fruit.spots 0.155197657
##     seed.discolor 0.155197657
##        shriveling 0.155197657
##       leaf.shread 0.146412884
##              seed 0.134699854
##       mold.growth 0.134699854
##         seed.size 0.134699854
##         leaf.halo 0.122986823
##         leaf.marg 0.122986823
##         leaf.size 0.122986823
##         leaf.malf 0.122986823
##        fruit.pods 0.122986823
##            precip 0.055636896
##      stem.cankers 0.055636896
##     canker.lesion 0.055636896
##         ext.decay 0.055636896
```

```
##         mycelium 0.055636896
##     int.discolor 0.055636896
##        sclerotia 0.055636896
##      plant.stand 0.052708638
##            roots 0.045387994
##             temp 0.043923865
##        crop.hist 0.023426061
##     plant.growth 0.023426061
##             stem 0.023426061
##             date 0.001464129
##         area.dam 0.001464129
##            Class 0.000000000
##           leaves 0.000000000
```

The table above and the histograms show that the predictors hail, sever, seed.tmt, and lodging have around 18% of missing data. Other variables that are more likely to be missing are germ(16% of missing values), leaf.mild(16%),fruiting.bodies(15%), fruits.spots(15%), seed.discolor(15%), and shriveling(15%). The grid shows the combination of all with 82% of data not missing in accordance with the problem description (18% missing). The remainder of the grid shows missing data for variable combinations with each row highlighting the missing values for the group of variables detailed in the x-axis. The non-graphical output of the function shows on top the exact proportion of missing values per variable.

Looking for pattern in missing data by classes

```
Soybean %>%
  mutate(Total = n()) %>%
  filter(!complete.cases(.)) %>%
  group_by(Class) %>%
  mutate(Missing = n(), Proportion=Missing/Total) %>%
  select(Class, Missing, Proportion) %>%
  unique()
```

```
## # A tibble: 5 x 3
## # Groups:   Class [5]
##   Class                       Missing Proportion
##   <fct>                         <int>      <dbl>
## 1 phytophthora-rot                 68     0.0996
## 2 diaporthe-pod-&-stem-blight      15     0.0220
## 3 cyst-nematode                    14     0.0205
## 4 2-4-d-injury                     16     0.0234
## 5 herbicide-injury                  8     0.0117
```

Checking if a pattern of missing data related to the classes exists is done by checking if some classes hold most of the incomplete cases. This is accomplished by filtering, grouping, and mutating the data with dplyr. The majority of the missing values are in the phytophthora-rot class which has nearly 10% incomplete cases. The are only four more, out of the eighteen other, variables with incomplete cases. The pattern of missing data is related to the classes. Mostly the phytophthora-rot class however since the other four variables only have between 1% and 2% incomplete cases.

(c) Develop a strategy for handling missing data, either by eliminating predictors or imputation.

The strategy to handle missing data is by using the predictive mean matching method of the mice function to imput data. Next, we create a complete dataset with the function complete() We can previous the new dataset for missing values with aggr from VIM package

```
library(mice)
```

```
## Warning: package 'mice' was built under R version 3.6.3
```

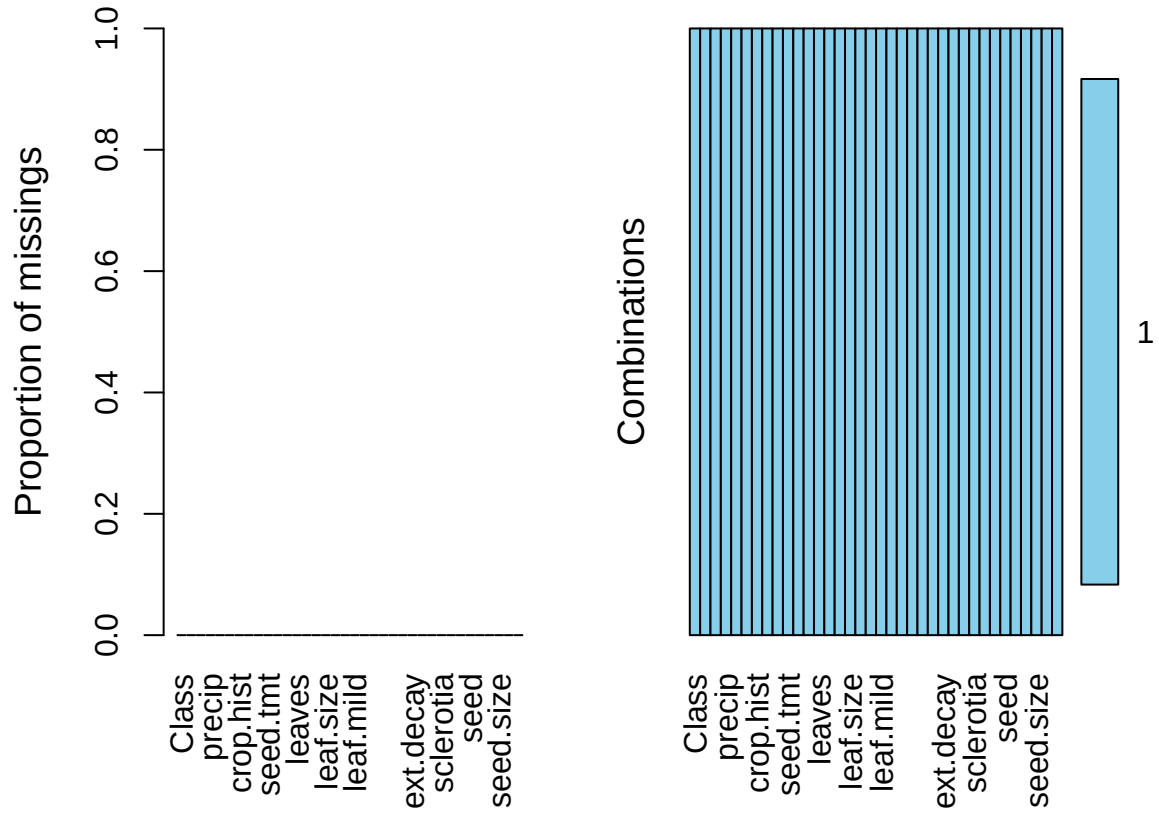
```
##
## Attaching package: 'mice'
```

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

```
MICE <- mice(Soybean, method="pmm", printFlag=FALSE, seed=6)
```

```
## Warning: Number of logged events: 1665
```

```
aggr(complete(MICE), prop = c(TRUE, TRUE), bars=TRUE, numbers=TRUE, sortVars=TRUE)
```



```
##
##  Variables sorted by number of missings:
##         Variable Count
##            Class     0
##             date     0
##      plant.stand     0
##           precip     0
##             temp     0
##             hail     0
##        crop.hist     0
##         area.dam     0
##            sever     0
##         seed.tmt     0
##             germ     0
##     plant.growth     0
##           leaves     0
##        leaf.halo     0
##        leaf.marg     0
##        leaf.size     0
##      leaf.shread     0
##        leaf.malf     0
##        leaf.mild     0
##             stem     0
##          lodging     0
##     stem.cankers     0
##    canker.lesion     0
##  fruiting.bodies     0
##        ext.decay     0
##         mycelium     0
##     int.discolor     0
##        sclerotia     0
##       fruit.pods     0
##      fruit.spots     0
##             seed     0
##      mold.growth     0
##    seed.discolor     0
##        seed.size     0
##       shriveling     0
##            roots     0
```

The strategy we use to deal with missing data is the simple imputation method that uses predictive mean matching (pmm) and "imputes missing values by means of the nearest-neighbor donor with distance based on the expected values of the missing variables conditional on the observed covariates."

After applying the mice function, we realise that there are no missing values in any variable.

References

https://www.otexts.org/fpp/

https://rpubs.com/josezuniga/358605

https://rpubs.com/josezuniga/253955

https://rpubs.com/josezuniga/269297

http://appliedpredictivemodeling.com/