ShovanBiswas-DATA24_Homework_04

Shovan Biswas

9/26/2020

Libraries

```
library(tidyverse)
library(kableExtra)
library(corrplot)
library(reshape2)
library(caret)
library(Amelia)
library(dlookr)
```

Exercise 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)
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 ...
```

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

Sanity check of Glass (prsence of null and NA):

```
any(is.null(Glass))
```

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

```
any(is.na(Glass))
```

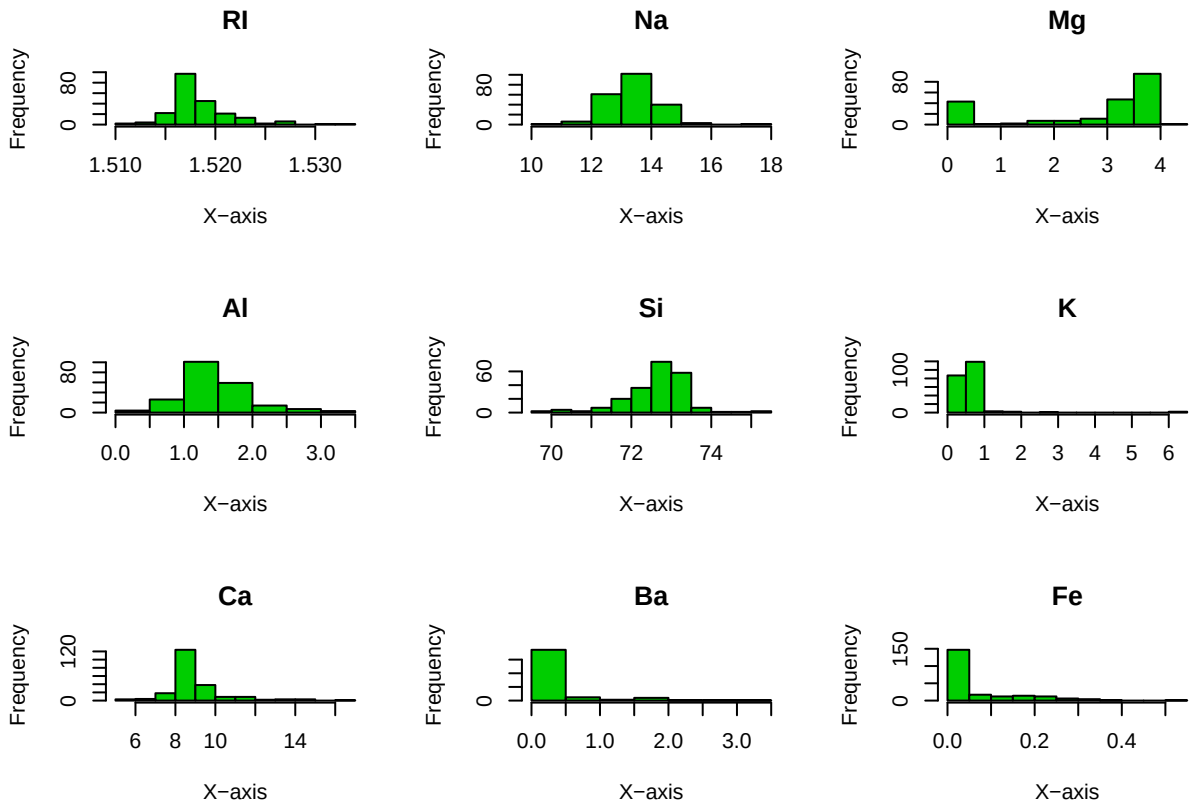
```
## [1] FALSE
```

Both are false, so there doesn't appear to be any missing values. In oder to be doubly sure, I did View(Glass) at the console – looks good.

In the following, I'll use histograms, to understand distribution of predictor variables.

```
Glass_subset <- subset(Glass, select = -Type)
predictors <- colnames(Glass_subset)
```

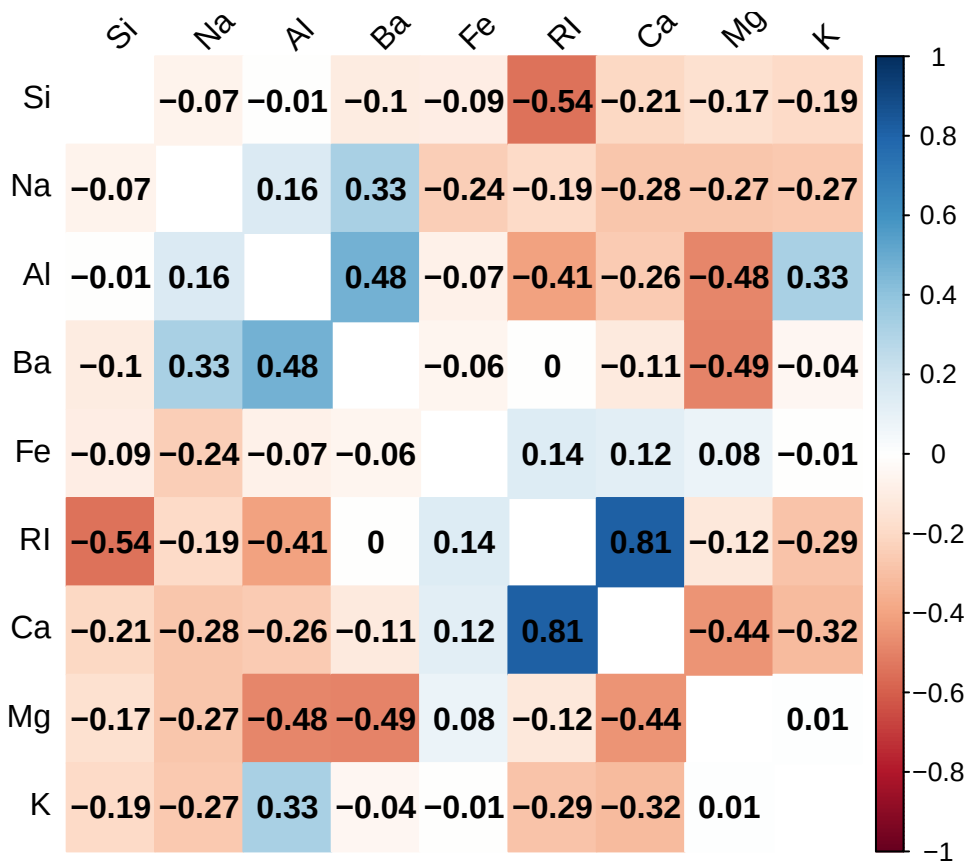
```
par(mfrow = c(3, 3))
for(i in 1:9) {
  hist(Glass_subset[,i], main = predictors[i], col = 3, xlab = "X-axis")
}
```



There are high concentrations of Si (Silicon), Na (Sodium) and Al (Aluminium).

To explore the relationships between the predictors, I'll find their correlations.

```
corrplot(cor(Glass[,1:9]), method="color", order="hclust", addCoef.col = "black", tl.col="black", tl.sr
```



A total of $\binom{9}{2} = 36$ predictor-wise correlations are possible. Out of this, 26 are negative. Si has negative correlation with all other elements and nowhere too close to -1.

About 6 of the predictors (Si, RI, -0.54), (Al, Ba, 0.48), (Al, RI, -0.41), (Al, Mg, -0.48), (Ba, Mg, -0.49), (Ca, Mg, -0.44) are less in the neighborhood of 1 or -1. So, they are not too strong. Most of the others are even less strongly correlated.

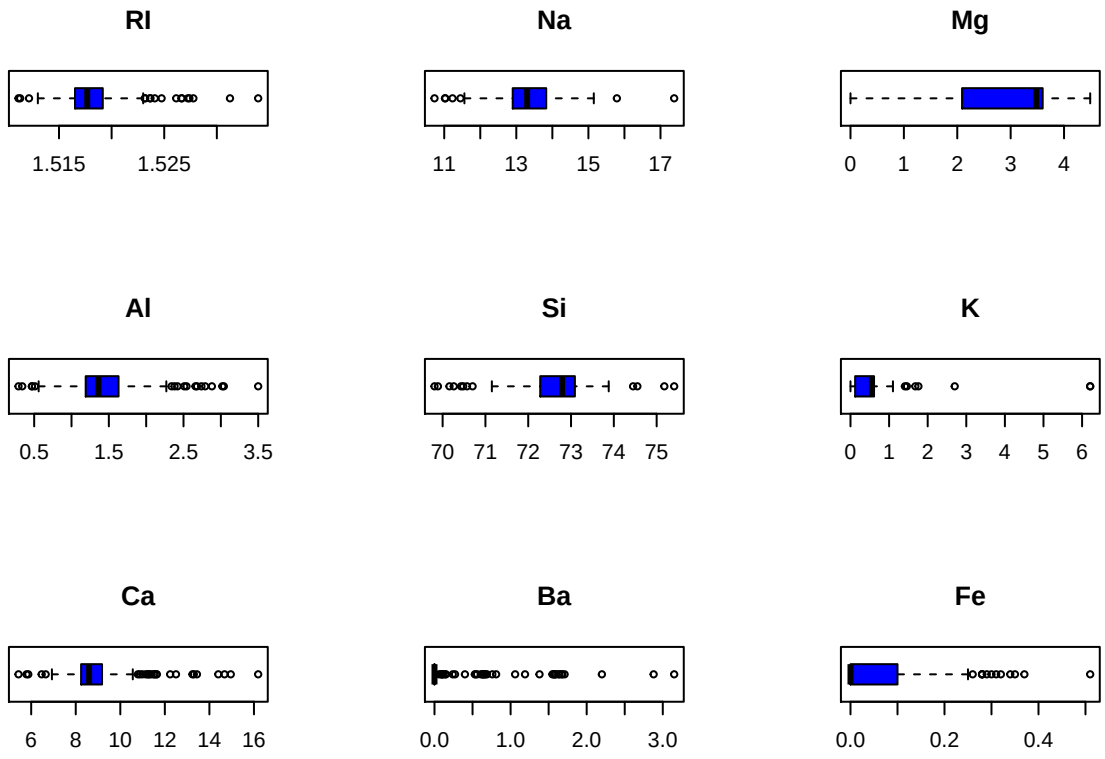
Only (RI, Ca, 0.81) stands out as a relatively stronger correlation. So, I would think adding more Calcium increases the refractive index – guessing.

However, this abstract (https://www.researchgate.net/publication/222625637_Hardness_and_Refractive_Index_of_Ca-Si-O-N_Glasses) says that there is no significant dependence on Ca content.

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

Although the histograms give a clue, I'll run boxplot() function to be sure of outliers.

```
par(mfrow = c(3, 3))
for(i in 1:9) {
  boxplot(Glass_subset[,i], main = predictors[i], col = 4, horizontal = TRUE)
}
```



Outliers: We observe that every predictor, except Mg has outlier. Skewed: The histogram in (a) tells us that Mg (with some bimodality), K, Ca, Ba and Fe are very skewed and Si is also slightly skewed.

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

Yes, there are several transformations that could improve the classification model, which I'll state below:

- Transformation to resolve Skewness: If data is determined to be skewed (a thumb rule is if the ratio to highest value skewed data to lowest value is greater than 20), then replacing the data with log, square or inverse might help remove the skew. Alternatively, Box and Cox propose a set of transformations (known as Box-Cox transformations) that are indexed by a parameter λ. If $\lambda = 0$, then $log(x)$, else $x^* = \frac{x^\lambda - 1}{\lambda}$. In addition to log transformation, this set of transformation can identify square transformation $\lambda = 2$, square root $\lambda = 0.5$ and inverse $\lambda = -1$ and others in between.
- Transformation to resolve Outliers: There several predictive models that are resistant to outliers. So, if a model is considered sensitive to outliers, then the transformation, called Spatial Sign can reduce the problem. In this scheme, the predictor data is centered and scaled before transforming. The predictor values are projected onto an n-dimensional sphere. So, all the data are at the same distance from the center.
- Data Reduction: This method reduces the number of predictors to a smaller set of predictors that seek to capture the majority of information in the original set of variables.
- Missing values: Outliers could occur, because intermediate values bridging the outliers to the center of gravity, could be missing. If values are missing, the first round of analysis should be to try to diagnose why the data are missing. Missing data could be imputed – very popular imputation is KNN Imputation.

Exercise 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:

```
data(Soybean)
```

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

Sanity check of Soybean (prsence of null and NA):

```
dim(Soybean)
```

```
## [1] 683  36
```

```
any(is.null(Soybean))
```

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

```
any(is.na(Soybean))
```

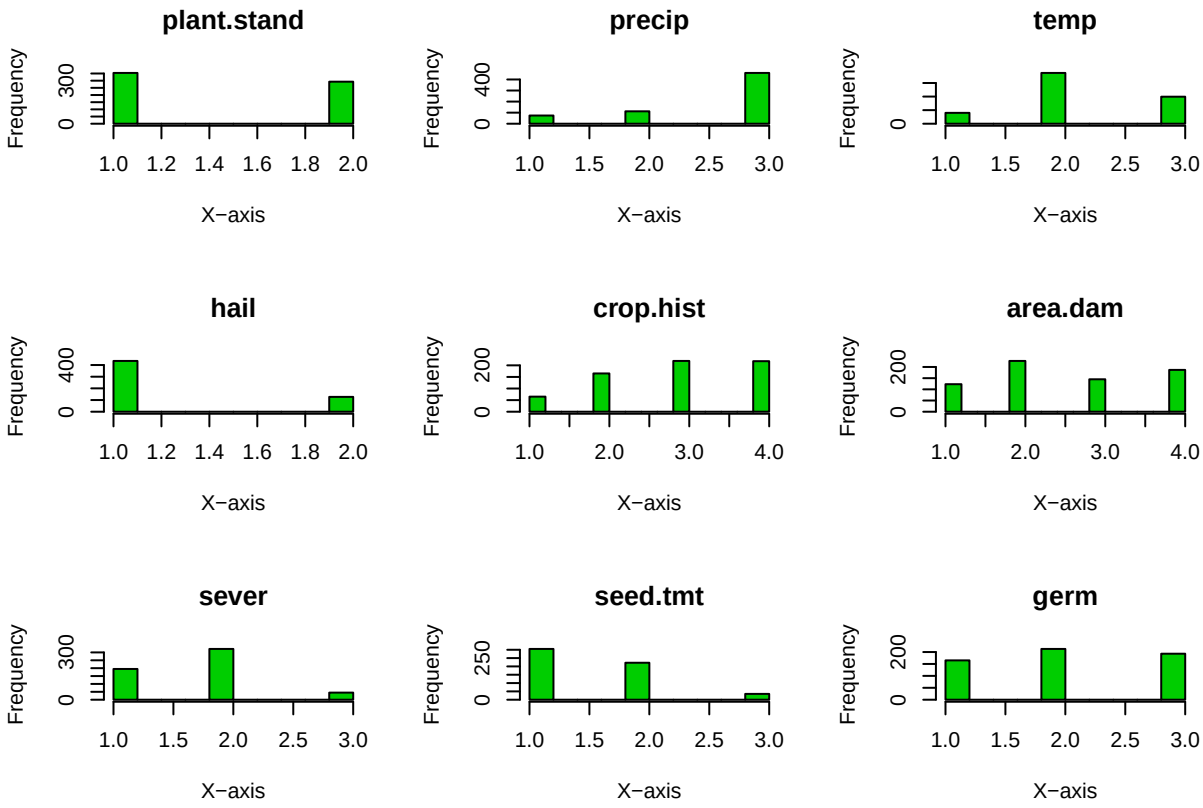
```
## [1] TRUE
```

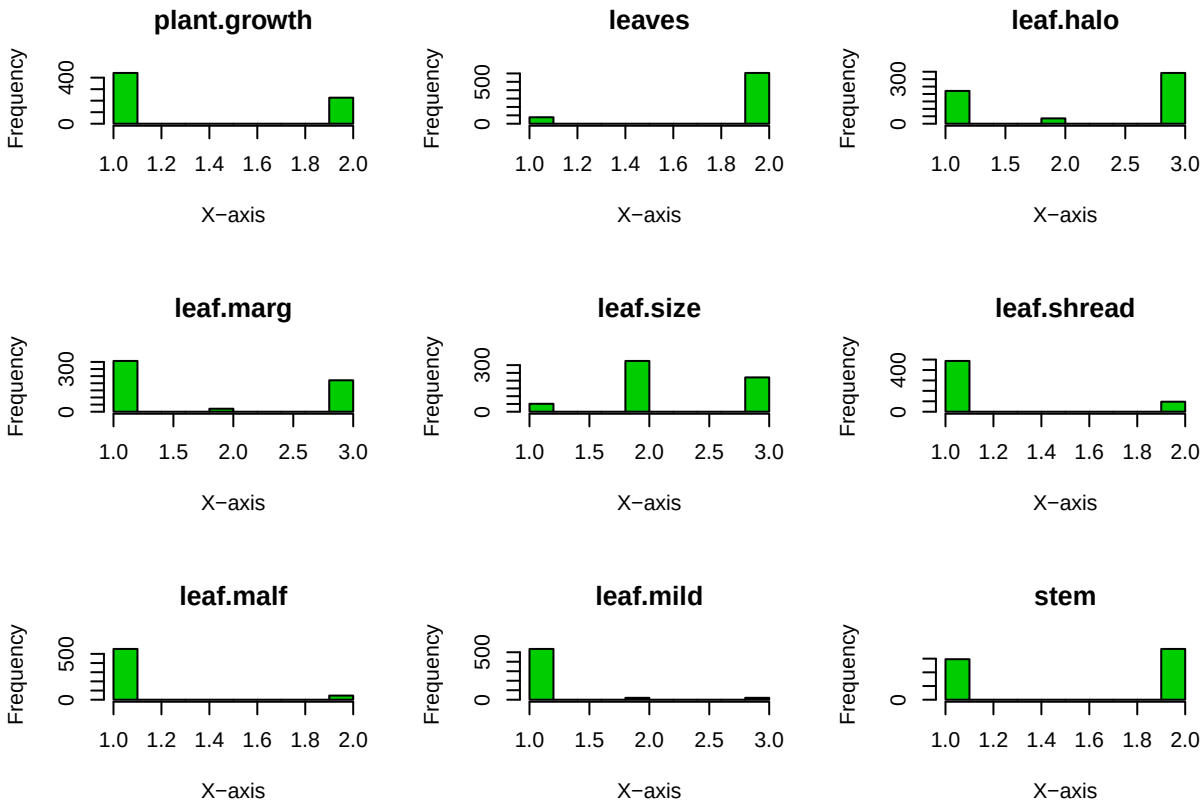
Fact: There are no null values, but there are missing values.

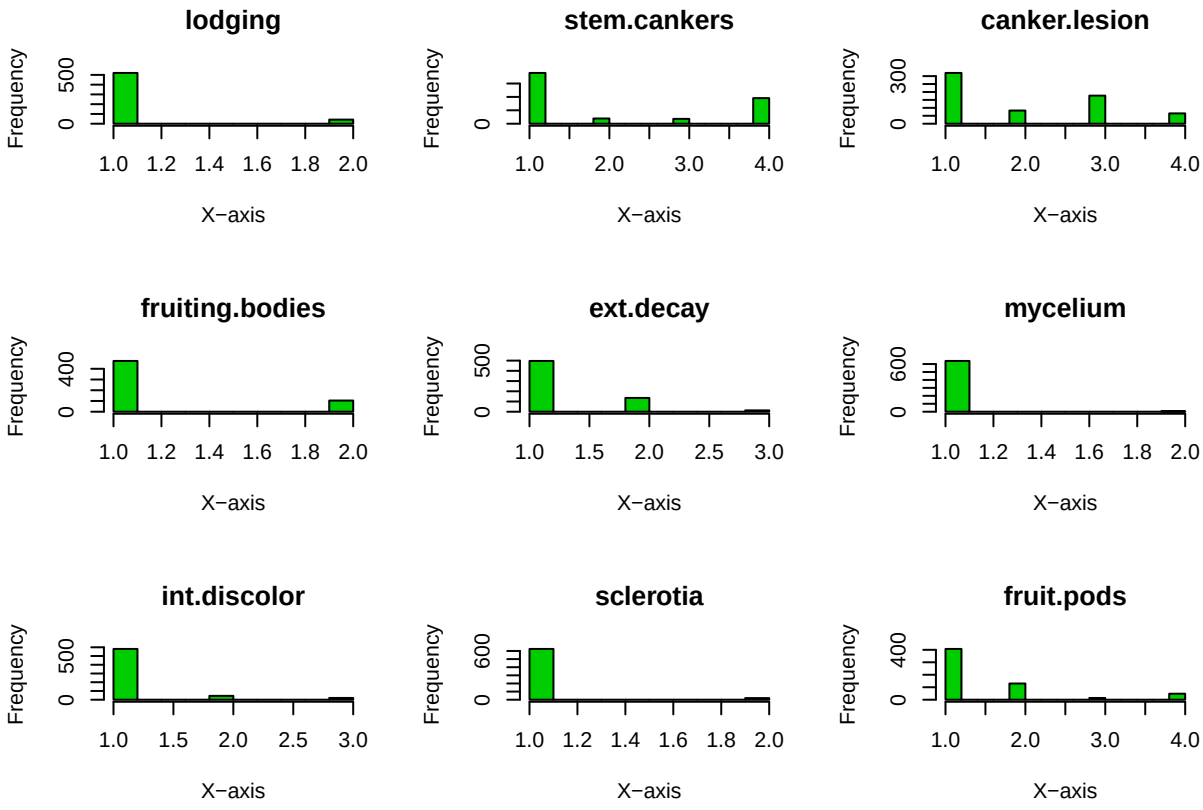
Frequency distribution for the significant predictors are shown below:

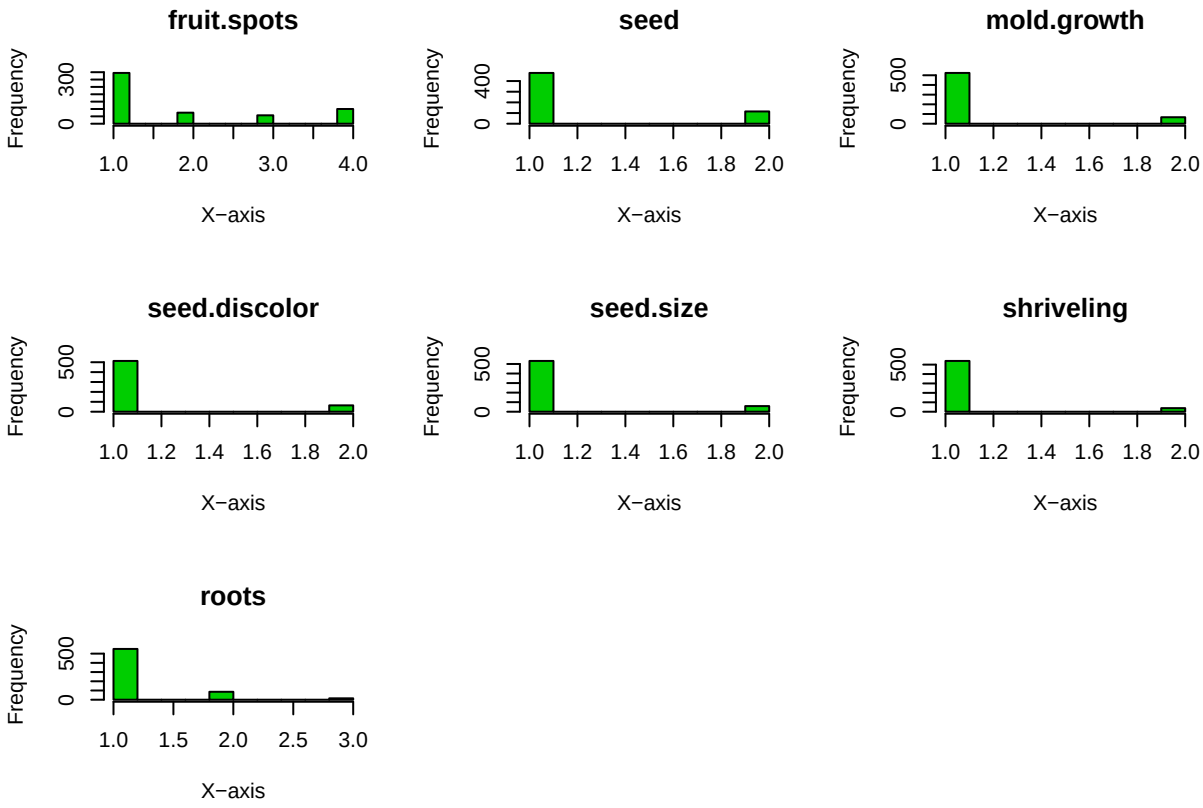
```
Soybean_subset <- subset(Soybean, select = -c(Class, date))
predictors <- colnames(Soybean_subset)
```

```
par(mfrow = c(3, 3))
for(i in 1:34) {
  hist(as.numeric(Soybean_subset[,i]), main = predictors[i], col = 3, xlab = "X-axis")
}
```


plant.stand
Frequency
0
300
1.0
1.2
1.4
1.6
1.8
2.0
X−axis
precip
Frequency
0
400
1.0
1.5
2.0
2.5
3.0
X−axis
temp
Frequency
0
1.0
1.5
2.0
2.5
3.0
X−axis
hail
Frequency
0
400
1.0
1.2
1.4
1.6
1.8
2.0
X−axis
crop.hist
Frequency
0
200
1.0
2.0
3.0
4.0
X−axis
area.dam
Frequency
0
200
1.0
2.0
3.0
4.0
X−axis
sever
Frequency
0
300
1.0
1.5
2.0
2.5
3.0
X−axis
seed.tmt
Frequency
0
250
1.0
1.5
2.0
2.5
3.0
X−axis
germ
Frequency
0
200
1.0
1.5
2.0
2.5
3.0
X−axis


plant.growth
Frequency
0
400
1.0
1.2
1.4
1.6
1.8
2.0
X−axis
leaves
Frequency
0
500
1.0
1.2
1.4
1.6
1.8
2.0
X−axis
leaf.halo
Frequency
0
300
1.0
1.5
2.0
2.5
3.0
X−axis
leaf.marg
Frequency
0
300
1.0
1.5
2.0
2.5
3.0
X−axis
leaf.size
Frequency
0
300
1.0
1.5
2.0
2.5
3.0
X−axis
leaf.shread
Frequency
0
400
1.0
1.2
1.4
1.6
1.8
2.0
X−axis
leaf.malf
Frequency
0
500
1.0
1.2
1.4
1.6
1.8
2.0
X−axis
leaf.mild
Frequency
0
500
1.0
1.5
2.0
2.5
3.0
X−axis
stem
Frequency
0
1.0
1.2
1.4
1.6
1.8
2.0
X−axis


lodging
Frequency
0
500
1.0
1.2
1.4
1.6
1.8
2.0
X−axis
stem.cankers
Frequency
0
1.0
2.0
3.0
4.0
X−axis
canker.lesion
Frequency
0
300
1.0
2.0
3.0
4.0
X−axis
fruiting.bodies
Frequency
0
400
1.0
1.2
1.4
1.6
1.8
2.0
X−axis
ext.decay
Frequency
0
500
1.0
1.5
2.0
2.5
3.0
X−axis
mycelium
Frequency
0
600
1.0
1.2
1.4
1.6
1.8
2.0
X−axis
int.discolor
Frequency
0
500
1.0
1.5
2.0
2.5
3.0
X−axis
sclerotia
Frequency
0
600
1.0
1.2
1.4
1.6
1.8
2.0
X−axis
fruit.pods
Frequency
0
400
1.0
2.0
3.0
4.0
X−axis



To determine the degenracy of the distributions, I'll use the thumb rule for detecting *near-zero variance* predictors on page 44 of *Applied Predictive Modeling*. The thumb rules are:

- The fraction of unique values over the sample size is low (say 10%).
- The ratio of the frequency of the most prevalent value to the frequency of the second most prevalent value is large (say around 20).

If both of the above thumb rules are true, it may be advantageous to remove the variable from the model.

I'll use function nearZeroVAr() (of package Caret) to detect the degenerate cases and then verify them.

```
names(Soybean)[caret::nearZeroVar(Soybean)]
```

```
## [1] "leaf.mild" "mycelium"  "sclerotia"
```

Verification of leaf.mild:

```
summary(Soybean$leaf.mild)
```

```
##    0    1    2 NA's
##  535   20   20  108
```

We observe that leaf.mild has 3 unique values 0, 1 & 2. Therefore, the fraction of unique values is: (3 / 683) x 100 = 0.43 % $<$ 10 %. So, leaf.mild satisfies thumb rule 1.

We also observe that the most frequent value is 0, which occurs 535 times. The next highest frequency is 20. Therefore, the ratio is: 535 / 20 = 26.75 > 20. So, leaf.mild satisfies thumb rule 2.

Verification of mycelium:

```
summary(Soybean$mycelium)
```

```
##    0    1 NA's
##  639    6   38
```

We observe that mycelium has 2 unique values 0 & 1. Therefore, the fraction of unique values is: (2 / 683) x 100 = 0.29 % < 10 %. So, mycelium satisfies thumb rule 1.
We also observe that the most frequent value is 0, which occurs 639 times. The next highest frequency is 6. Therefore, the ratio is: 639 / 6 = 106.5 > 20. So, mycelium satisfies thumb rule 2.

Verification of sclerotia:

```
summary(Soybean$sclerotia)
```

```
##    0    1 NA's
##  625   20   38
```

We observe that sclerotia has 2 unique values 0 & 1. Therefore, the fraction of unique values is: (2 / 683) x 100 = 0.29 % < 10 %. So, sclerotia satisfies thumb rule 1.
We also observe that the most frequent value is 0, which occurs 625 times. The next highest frequency is 20. Therefore, the ratio is: 625 / 20 = 31.25 > 20. So, sclerotia satisfies thumb rule 2.

(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?

Here's a sorted list of predictors having missing data. The top ones have more missing data. I would think these predictors are more likely to be missing.

```
sort(Soybean[-1] %>% apply(2, is.na) %>% apply(2, sum, na.rm=T), decreasing=TRUE)
```

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

The following table answers the second question *Is the pattern of missing data related to the classes?*. There are missing data related to classes and the top five of them are shown here.

```
Soybean$na_count <- apply(Soybean, 1, function(x) sum(is.na(x)))
(Soybean %>% select(Class, na_count) %>% group_by(Class) %>% summarise(na_count = sum(na_count)) %>% ar
```

```
## # A tibble: 19 x 2
##    Class                        na_count
##    <fct>                           <int>
##  1 phytophthora-rot                 1214
##  2 2-4-d-injury                      450
##  3 cyst-nematode                     336
##  4 diaporthe-pod-&-stem-blight       177
##  5 herbicide-injury                  160
##  6 alternarialeaf-spot                 0
##  7 anthracnose                         0
##  8 bacterial-blight                    0
##  9 bacterial-pustule                   0
## 10 brown-spot                          0
## 11 brown-stem-rot                      0
## 12 charcoal-rot                        0
## 13 diaporthe-stem-canker               0
## 14 downy-mildew                        0
## 15 frog-eye-leaf-spot                  0
## 16 phyllosticta-leaf-spot              0
## 17 powdery-mildew                      0
## 18 purple-seed-stain                   0
## 19 rhizoctonia-root-rot                0
```

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

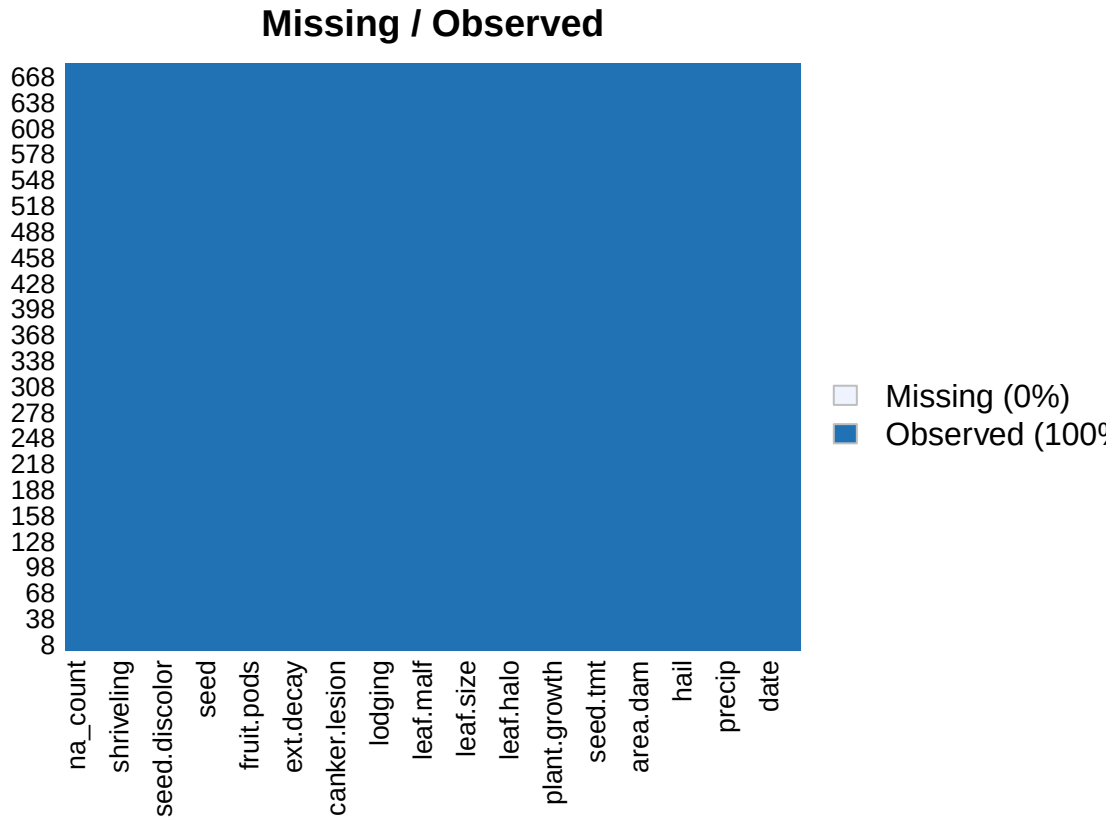
In order to handle missing data, I'll remove the three nearzero predictors and impute values for the rest of the predictors. For this reason, *dlookr* package was installed and enlisted in the libraries at the begining of this RMD.

```
Soybean_complete <- Soybean %>%
  mutate(
    date = imputate_na(Soybean, date, Class, method = "rpart", no_attrs = TRUE),
    plant.stand = imputate_na(Soybean, plant.stand, Class, method = "rpart", no_attrs = TRUE),
    precip = imputate_na(Soybean, precip, Class, method = "rpart", no_attrs = TRUE),
    temp = imputate_na(Soybean, temp, Class, method = "rpart", no_attrs = TRUE),
    hail = imputate_na(Soybean, hail, Class, method = "rpart", no_attrs = TRUE),
    crop.hist = imputate_na(Soybean, crop.hist, Class, method = "rpart", no_attrs = TRUE),
    area.dam = imputate_na(Soybean, area.dam, Class, method = "rpart", no_attrs = TRUE),
    sever = imputate_na(Soybean, sever, Class, method = "rpart", no_attrs = TRUE),
    seed.tmt = imputate_na(Soybean, seed.tmt, Class, method = "rpart", no_attrs = TRUE),
    germ = imputate_na(Soybean, germ, Class, method = "rpart", no_attrs = TRUE),
    plant.growth = imputate_na(Soybean, plant.growth, Class, method = "rpart", no_attrs = TRUE),
    leaf.halo = imputate_na(Soybean, leaf.halo, Class, method = "rpart", no_attrs = TRUE),
    leaf.marg = imputate_na(Soybean, leaf.marg, Class, method = "rpart", no_attrs = TRUE),
    leaf.size = imputate_na(Soybean, leaf.size, Class, method = "rpart", no_attrs = TRUE),
    leaf.shread = imputate_na(Soybean, leaf.shread, Class, method = "rpart", no_attrs = TRUE),
    leaf.malf = imputate_na(Soybean, leaf.malf, Class, method = "rpart", no_attrs = TRUE),
```

```
    stem = imputate_na(Soybean, stem, Class, method = "rpart", no_attrs = TRUE),
    lodging = imputate_na(Soybean, lodging, Class, method = "rpart", no_attrs = TRUE),
    stem.cankers = imputate_na(Soybean, stem.cankers, Class, method = "rpart", no_attrs = TRUE),
    canker.lesion = imputate_na(Soybean, canker.lesion, Class, method = "rpart", no_attrs = TRUE),
    fruiting.bodies = imputate_na(Soybean, fruiting.bodies, Class, method = "rpart", no_attrs = TRUE),
    ext.decay = imputate_na(Soybean, ext.decay, Class, method = "rpart", no_attrs = TRUE),
    int.discolor = imputate_na(Soybean, int.discolor, Class, method = "rpart", no_attrs = TRUE),
    fruit.pods = imputate_na(Soybean, fruit.pods, Class, method = "rpart", no_attrs = TRUE),
    seed = imputate_na(Soybean, seed, Class, method = "rpart", no_attrs = TRUE),
    mold.growth = imputate_na(Soybean, mold.growth, Class, method = "rpart", no_attrs = TRUE),
    seed.discolor = imputate_na(Soybean, seed.discolor, Class, method = "rpart", no_attrs = TRUE),
    seed.size = imputate_na(Soybean, seed.size, Class, method = "rpart", no_attrs = TRUE),
    shriveling = imputate_na(Soybean, shriveling, Class, method = "rpart", no_attrs = TRUE),
    fruit.spots = imputate_na(Soybean, fruit.spots, Class, method = "rpart", no_attrs = TRUE),
    roots = imputate_na(Soybean, roots, Class, method = "rpart", no_attrs = TRUE)) %>%
  select(-leaf.mild, -mycelium, -sclerotia)
```

Now, I'll check the effect of the ellimination.

```
Soybean_complete %>% arrange(Class) %>% missmap(main = "Missing / Observed")
```



So, there are no missing observations.