Data 624 Homework 4

Steven Gonzalez

3/2/2025

Load Packages

```r
library(AppliedPredictiveModeling)
library(caret)
library(corrplot)
library(mlbench)
library(tidyverse)
library(VIM)
```

Exercise 1

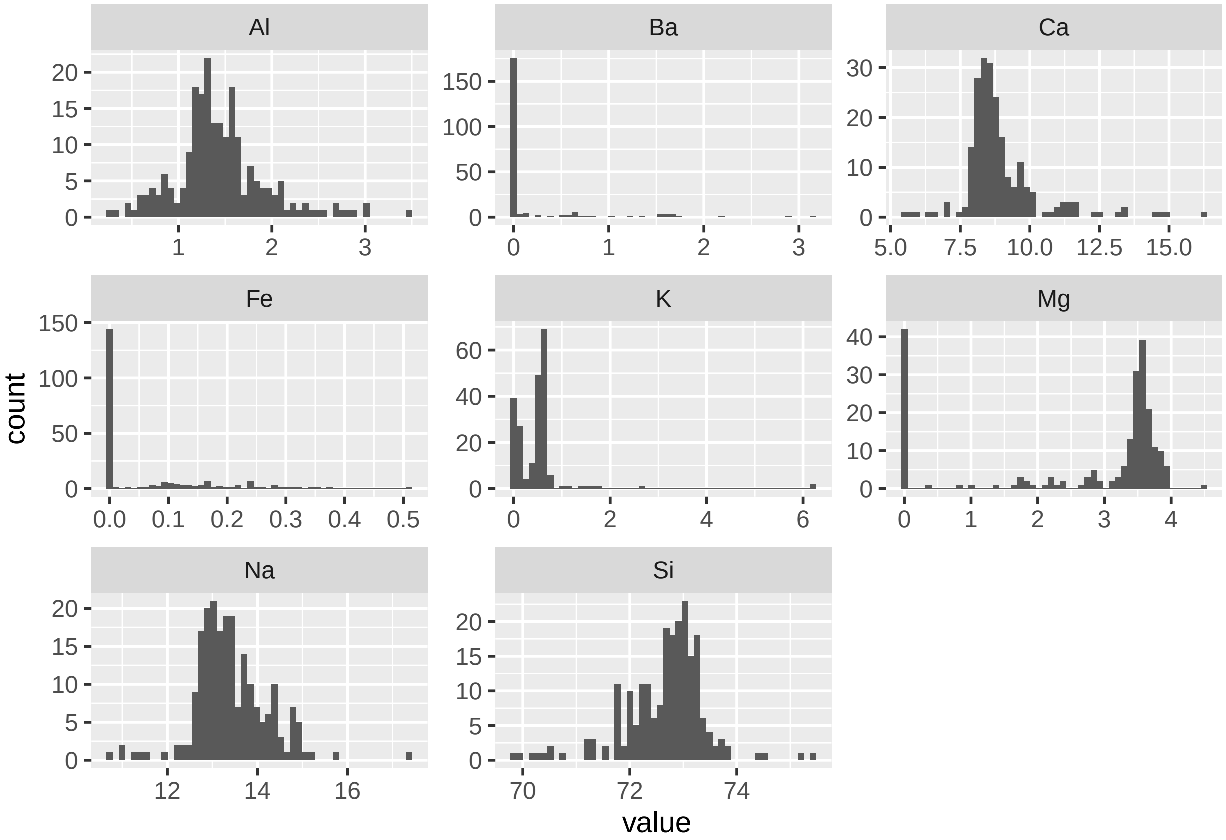
The UC Irvine Machine Learning Repository contains a data set related to glass identification. The data consist of 214 glass samples labeled as one of seven class categories. There are nine predictors, including the refractive index and percentages of eight elements: Na, Mg, Al, Si, K, Ca, Ba, and Fe.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

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

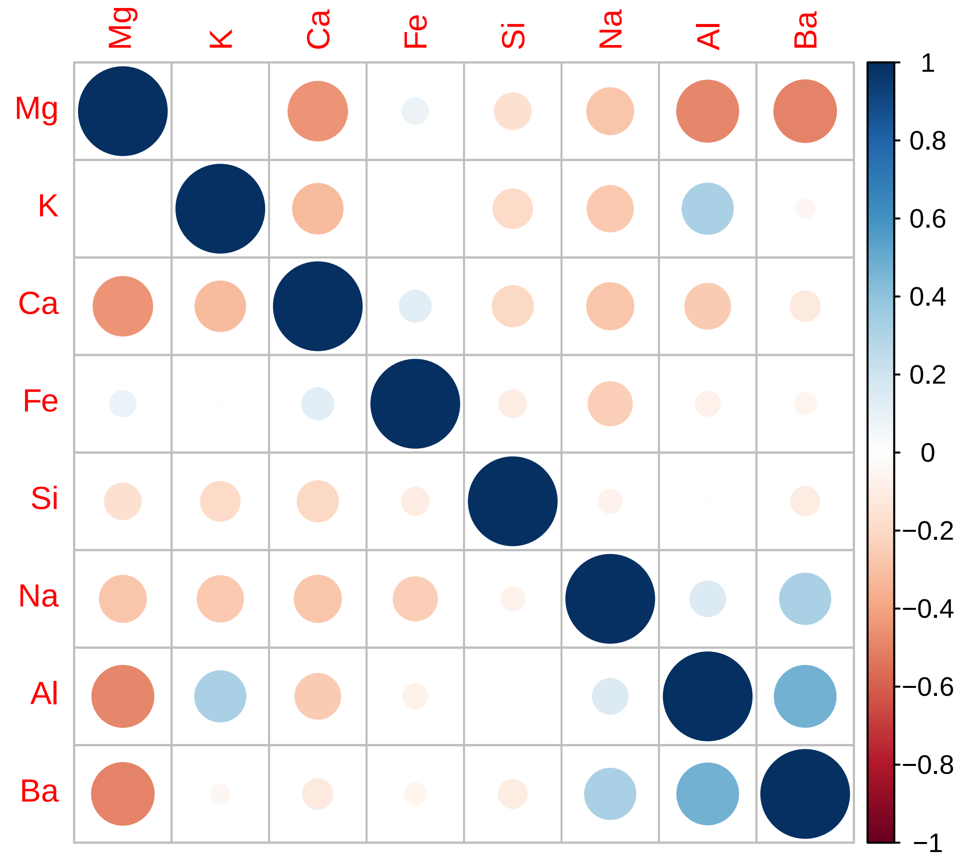
```
## ’data.frame':    214 obs. of  10 variables:
##  $ RI  : num  1.52 1.52 1.52 1.52 1.52 ...
##  $ Na  : num  13.6 13.9 13.5 13.2 13.3 ...
##  $ Mg  : num  4.49 3.6 3.55 3.69 3.62 3.61 3.6 3.61 3.58 3.6 ...
##  $ Al  : num  1.1 1.36 1.54 1.29 1.24 1.62 1.14 1.05 1.37 1.36 ...
##  $ Si  : num  71.8 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 ...
```

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

```r
glass_predictors <- select(Glass, c(2:9))
glass_predictors %>%
  gather() %>%
  ggplot(aes(value)) +
  geom_histogram(bins=50) +
  facet_wrap(~ key, scales = 'free')
```



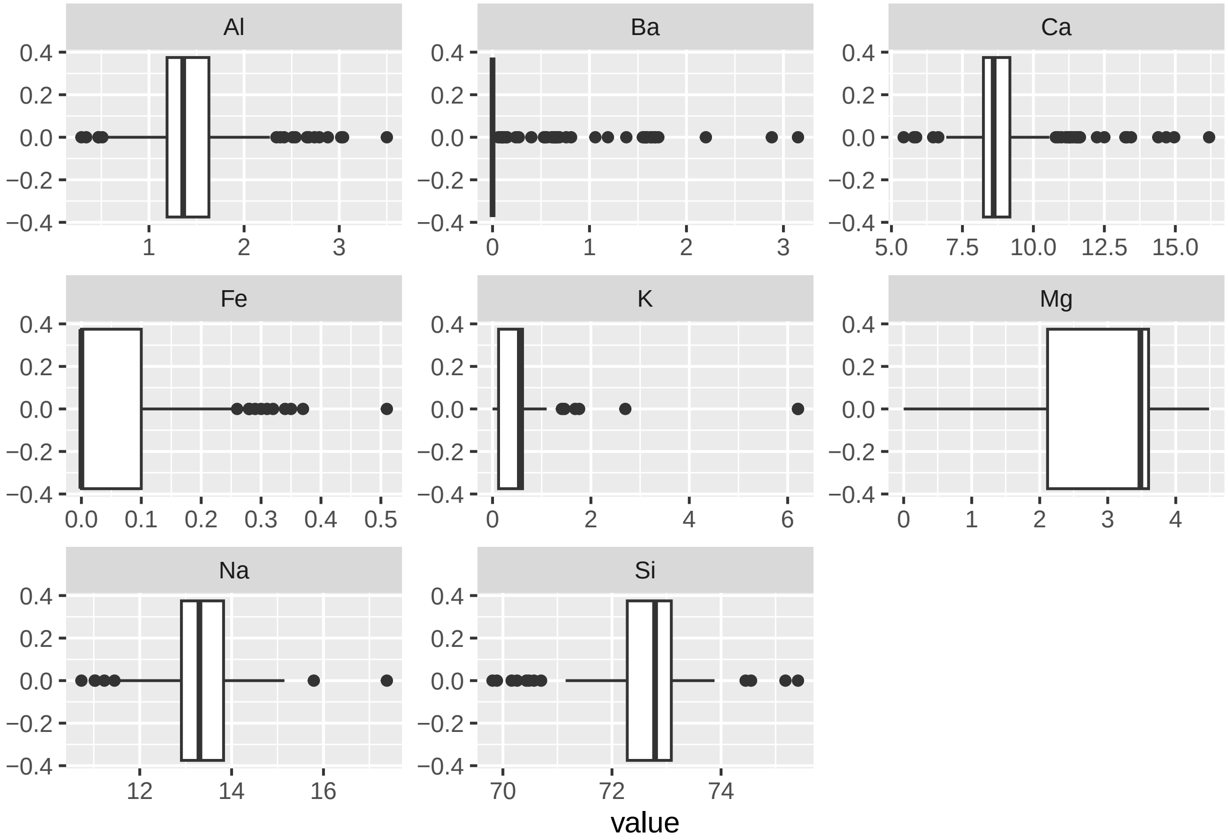
```
predictor_cor <- cor(glass_predictors)
corrplot(predictor_cor, order = 'hclust')
```



As seen from the plots above, the distributions of the individual elements seem to show some skewness. There also seems to be some positive correlation between Al and Ba and negative correlation between Mg and Al, Ca, and Ba.

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

```r
glass_predictors %>%
  gather() %>%
  ggplot(aes(value)) +
  geom_boxplot() +
  facet_wrap(~key, scales = 'free')
```

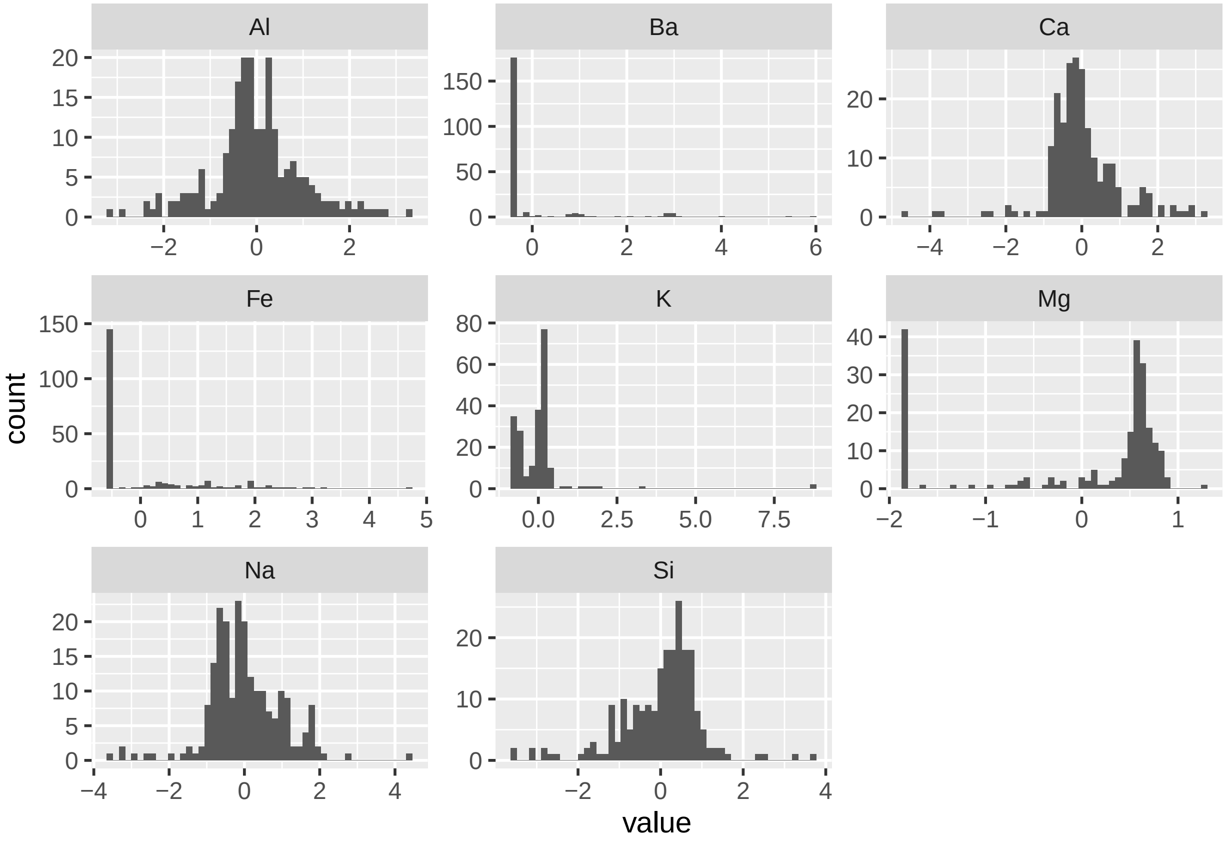


As seen from the plot above, all predictors contain outliers with the exception of Mg. Based on the distribution plots from the previous portion, elements Al, Ca, Na, and Si show minimal skewness while Ba, Fe, and K are skewed to the right and Mg is skewed to the left.

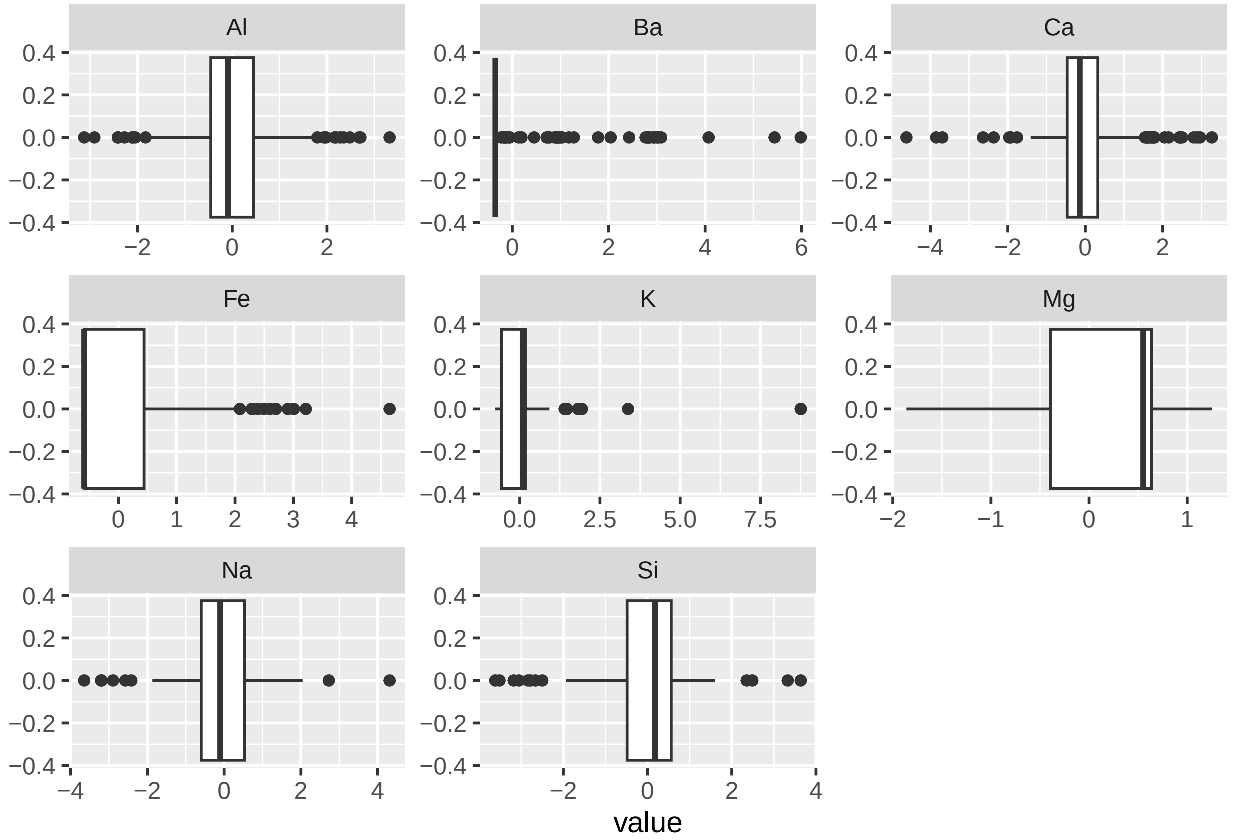
Are there any relevant transformations of one or more predictors that might improve the classification model?

```
bc_transform <- preProcess(glass_predictors,
                           method = c('BoxCox', 'center', 'scale'))
glass_predictors_transform <- predict(bc_transform, glass_predictors)

glass_predictors_transform %>%
  gather() %>%
  ggplot(aes(value)) +
  geom_histogram(bins=50) +
  facet_wrap(~ key, scales = 'free')
```



```
glass_predictors_transform %>%
  gather() %>%
  ggplot(aes(value)) +
  geom_boxplot() +
  facet_wrap(~key, scales = 'free')
```



A BoxCox transformation was performed on the predictors data set and resulted in the pots seen above. The predictors that benefited most from this transformation were Al, Ca, Na, and Si.

Exercise 2

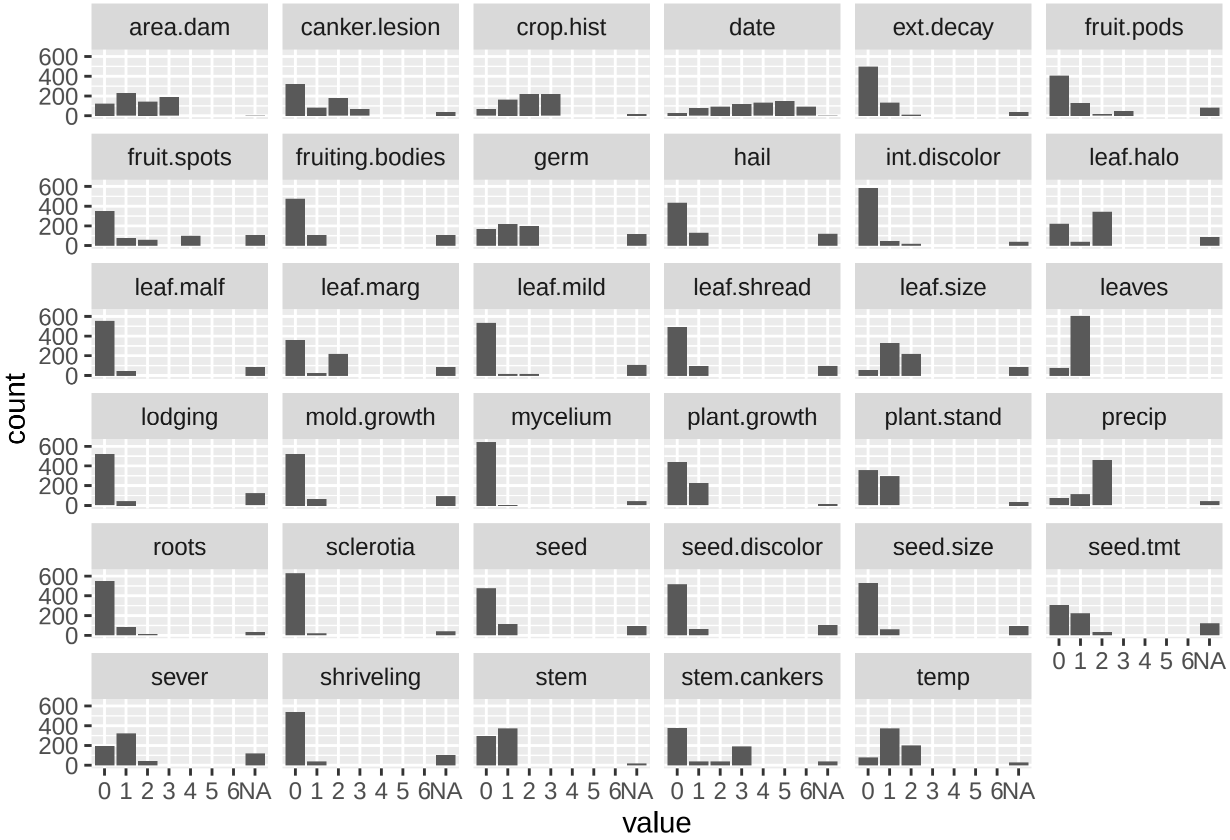
The soybean data can also be found at the UC Irvine Machine Learning Repository. Data were collected to predict disease in 683 soybeans. The 35 predictors are mostly categorical and include information on the environmental conditions (e.g., temperature, precipitation) and plant conditions (e.g., left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

```
data(Soybean)
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 1 ...
##  $ leaf.malf      : Factor w/ 2 levels "0","1": 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 1 ...
##  $ stem           : Factor w/ 2 levels "0","1": 2 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 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 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 1 ...
##  $ mold.growth    : Factor w/ 2 levels "0","1": 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 1 ...
##  $ seed.size      : Factor w/ 2 levels "0","1": 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 1 ...
##  $ roots          : Factor w/ 3 levels "0","1","2": 1 1 1 1 1 1 1 1 1 1 ...
```

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

```
soybean_predictors <- select(Soybean, c(2:36))
soybean_predictors %>%
  gather() %>%
  ggplot(aes(value)) +
  geom_bar() +
  facet_wrap(~ key)
```



```r
nearZeroVar(soybean_predictors, name = TRUE)
```

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

The frequency distributions shown above display signs of degenerate variables. To confirm specifically which, the `nearZeroVar()` function was used resulting in the identification of predictors `leaf.mild`, `mycelium`, and `sclerotia` as degenerate variables.

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?

```r
soybean_predictors %>%
  select(everything()) %>%
  summarize_all(funs(sum(is.na(.)))) %>%
  pivot_longer(everything(),
               names_to = 'Predictor',
               values_to = 'Number of Missing Values')
```

```
## # A tibble: 35 x 2
##    Predictor   ‘Number of Missing Values‘
##    <chr>                            <int>
##  1 date                                 1
##  2 plant.stand                         36
##  3 precip                              38
##  4 temp                                30
##   5 hail                               121
##   6 crop.hist                           16
##   7 area.dam                             1
##   8 sever                              121
##   9 seed.tmt                           121
##  10 germ                               112
## # i 25 more rows
```

```
sum(is.na(soybean_predictors))
```

```
## [1] 2337
```

```
Soybean %>%
  filter_all(any_vars(is.na(.))) %>%
  select(Class) %>%
  group_by(Class) %>%
  summarise(count = n())
```

```
## # A tibble: 5 x 2
##   Class                         count
##   <fct>                         <int>
## 1 2-4-d-injury                     16
## 2 cyst-nematode                    14
## 3 diaporthe-pod-&-stem-blight      15
## 4 herbicide-injury                  8
## 5 phytophthora-rot                 68
```

The tibbles above give us a glimpse into the predictors with the most missing data. These predictors being, `hail`, `sever`, `seed.tmt`, and `lodging` with 121 missing values each. When sifting through for which classes are responsible for this missing data we see that they are found in `2-4-d-injury`, `cyst-nematode`, `diaporthe-pod-&-stem-blight`, `herbicide-injury`, and `phytophthora-rot`.

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

```
soybean_imputed <- kNN(soybean_predictors, k = 3) %>%
  select(c(1:35))

soybean_imputed %>%
  select(everything()) %>%
  summarize_all(funs(sum(is.na(.)))) %>%
  pivot_longer(everything(),
               names_to = 'Predictor',
               values_to = 'Number of Missing Values')
```

```
## # A tibble: 35 x 2
##    Predictor   `Number of Missing Values`
##    <chr>                            <int>
##  1 date                                 0
##  2 plant.stand                          0
##  3 precip                               0
##  4 temp                                 0
##  5 hail                                 0
##  6 crop.hist                             0
##  7 area.dam                              0
##  8 sever                                 0
##  9 seed.tmt                              0
## 10 germ                                  0
## # i 25 more rows
```

```
sum(is.na(soybean_imputed))
```

```
## [1] 0
```

As seen above, imputation was performed on the predictor data set using the KNN model looking up to the third nearest neighbor. This resulted in a total missing value count of 0.