Homework 4_data624

Heleine Fouda

2024-09-25

Set up the environment: Load necessary libraries

Exercise 3.1

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

```r
# Load the Glass dataset
data(Glass)
```

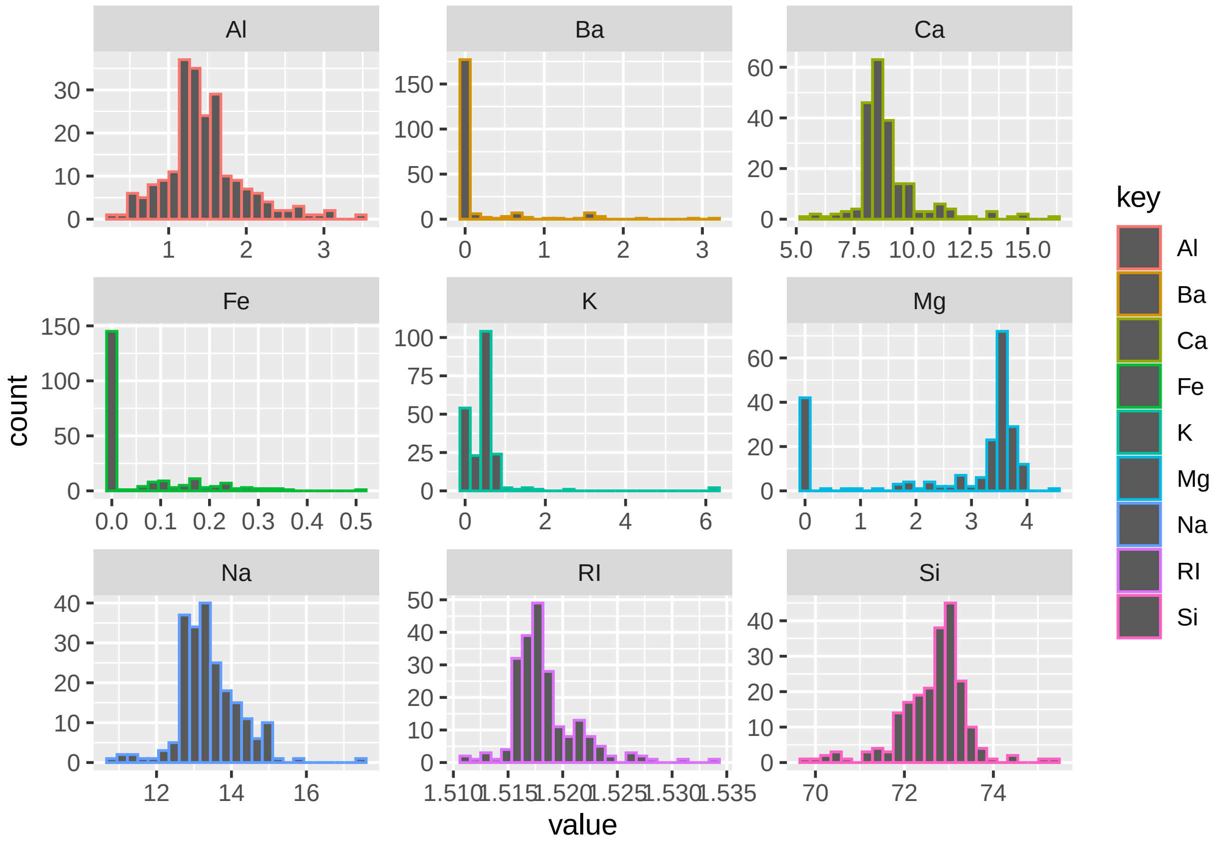
```r
# Checking the structure of the data
str(Glass)
```

```
## '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 ...
```

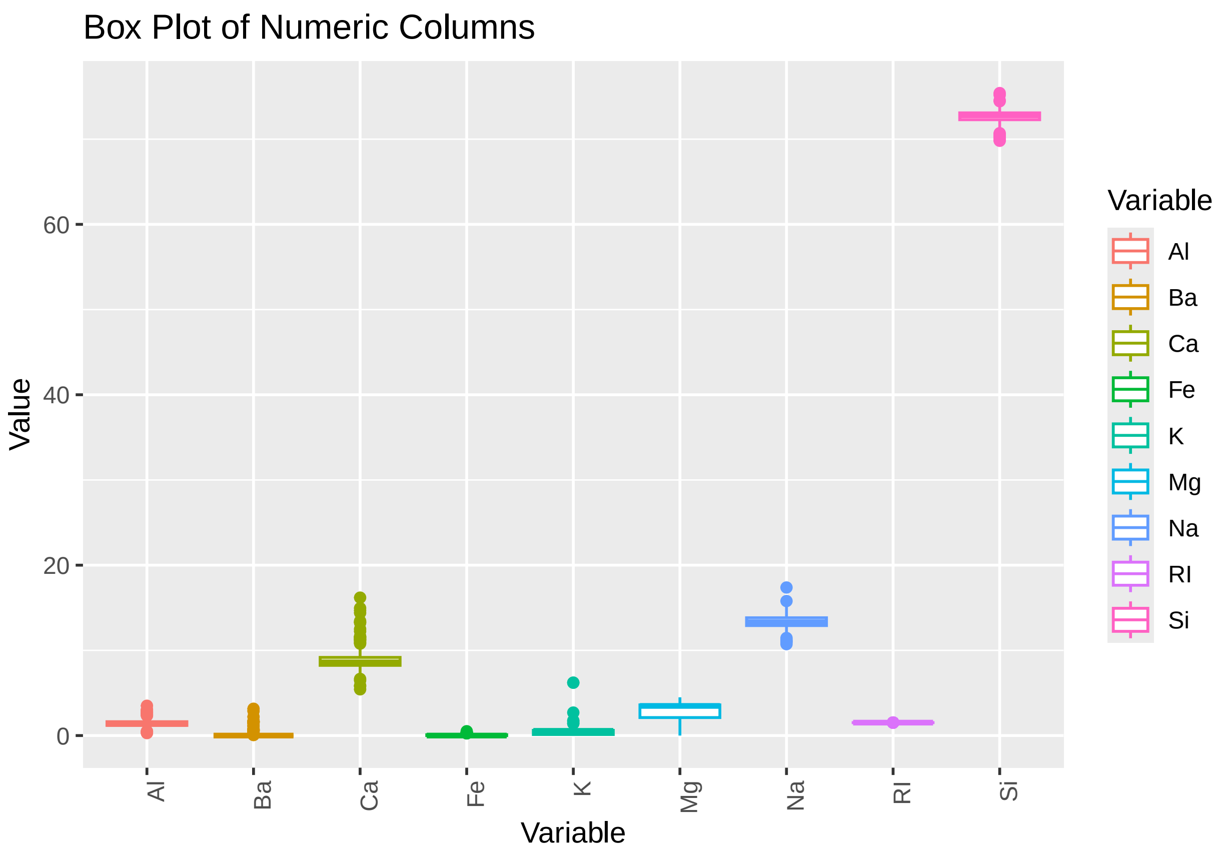
```r
# Checking the names of the variables
names(Glass)
```

```
##  [1] "RI"   "Na"   "Mg"   "Al"   "Si"   "K"    "Ca"   "Ba"   "Fe"   "Type"
```

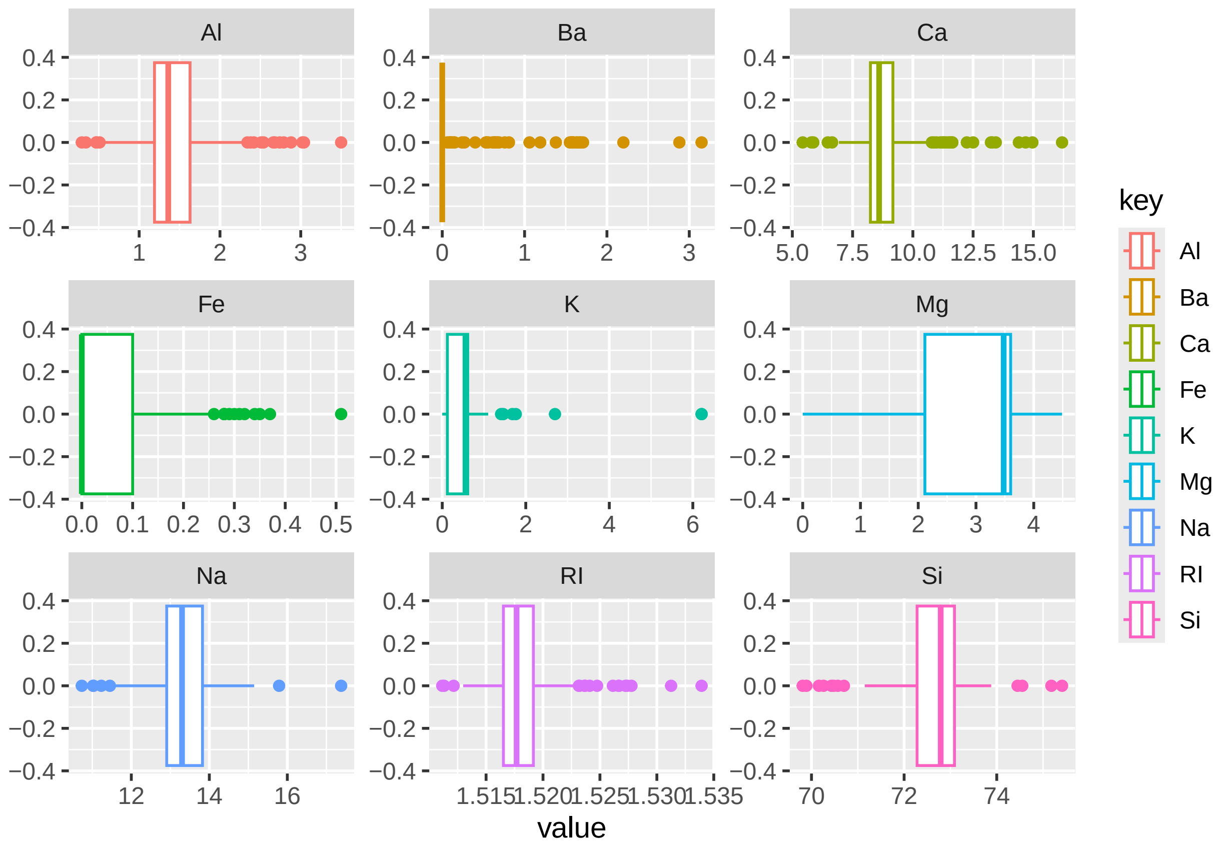
```r
# Visualizing the distribution of the predictors using pair using ggplot() and facet_wrap()
data(Glass)
glass <- data.frame(Glass)

numeric_cols <- glass[-c(10)]

numeric_cols %>%
  gather() %>%
  ggplot(aes(value,color = key))+
  geom_histogram(bins=25)+
  facet_wrap(~key,scales='free')
```



```
# Visualizing the distribution of the predictors using a boxplot
numeric_cols <- glass[-c(10)]

# Convert data to long format and create box plots
numeric_cols %>%
  gather(key = "Variable", value = "Value") %>%
  ggplot(aes(x = Variable, y = Value, color = Variable)) +
  geom_boxplot() +
  labs(title = "Box Plot of Numeric Columns", x = "Variable", y = "Value") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```

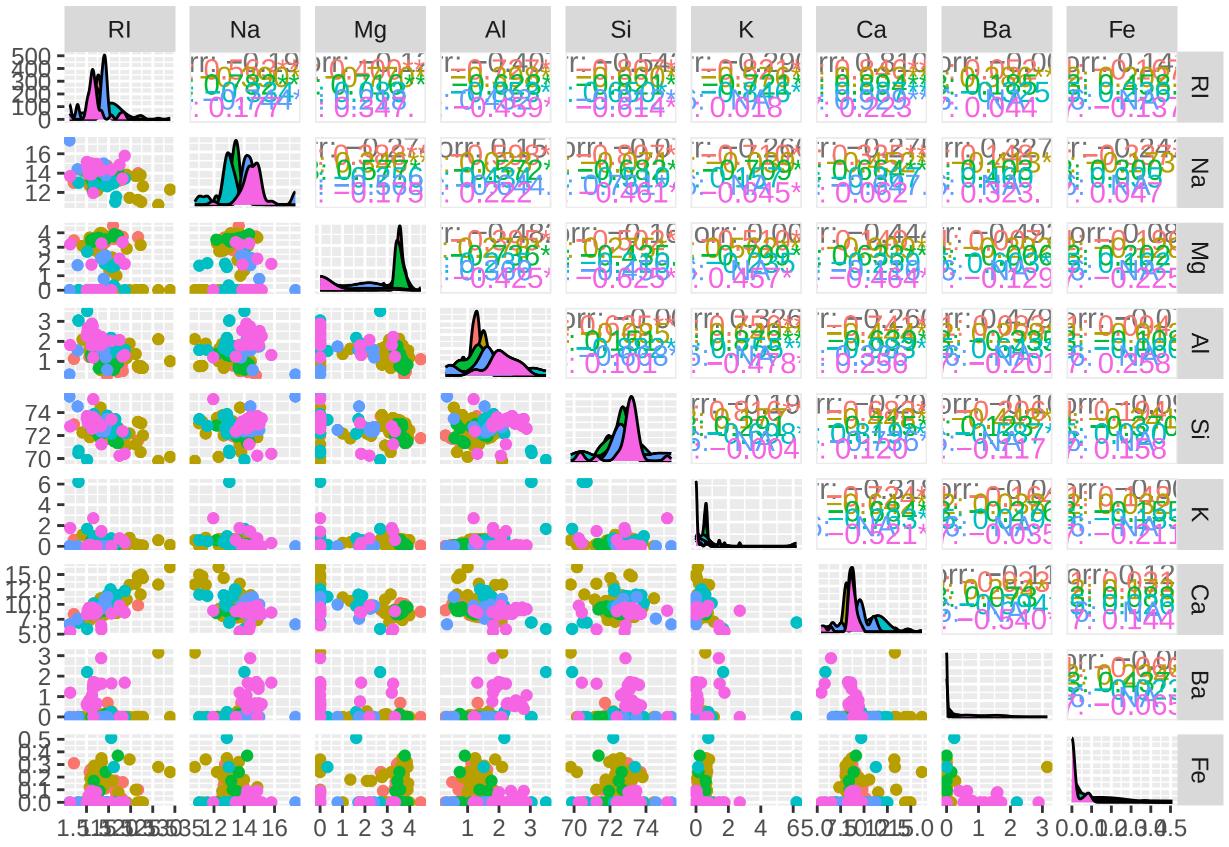


```
numeric_cols %>%
  gather() %>%
  ggplot(aes(value,color = key))+
  geom_boxplot()+
  facet_wrap(~key,scales='free')
```



```r
# Visualizing the distribution of the predictors using pair plot
ggpairs(Glass, columns=1:9, aes(color = Type))
```

```
## Warning in cor(x, y): the standard deviation is zero

## Warning in cor(x, y): the standard deviation is zero

## Warning in cor(x, y): the standard deviation is zero

## Warning in cor(x, y): the standard deviation is zero

## Warning in cor(x, y): the standard deviation is zero

## Warning in cor(x, y): the standard deviation is zero

## Warning in cor(x, y): the standard deviation is zero

## Warning in cor(x, y): the standard deviation is zero

## Warning in cor(x, y): the standard deviation is zero

## Warning in cor(x, y): the standard deviation is zero

## Warning in cor(x, y): the standard deviation is zero

## Warning in cor(x, y): the standard deviation is zero
## Warning in cor(x, y): the standard deviation is zero

## Warning in cor(x, y): the standard deviation is zero

## Warning in cor(x, y): the standard deviation is zero

## Warning in cor(x, y): the standard deviation is zero

## Warning in cor(x, y): the standard deviation is zero

## Warning in cor(x, y): the standard deviation is zero

## Warning in cor(x, y): the standard deviation is zero

## Warning in cor(x, y): the standard deviation is zero

## Warning in cor(x, y): the standard deviation is zero
```



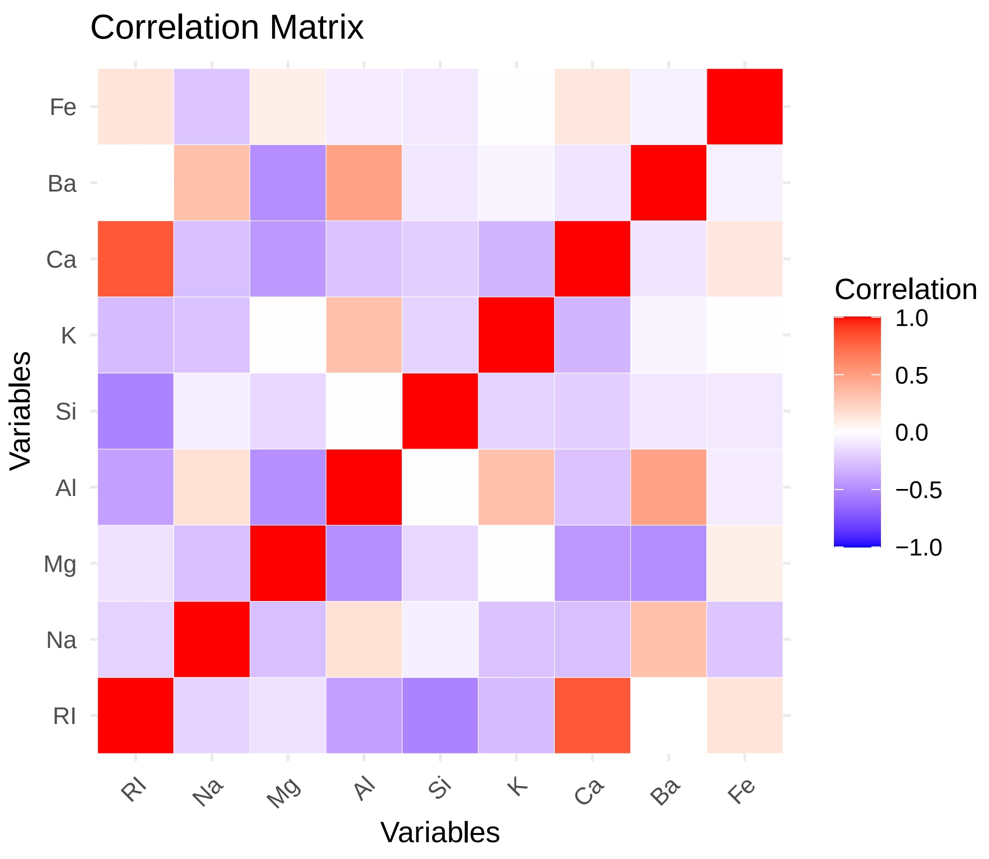
```r
# Visualizing using a correlation matrix
numeric_cols <- glass[-c(10)]

# Calculate the correlation matrix
cor_matrix <- cor(numeric_cols, use = "complete.obs")
# Convert the correlation matrix into long format
cor_data <- melt(cor_matrix)
```

```
## Warning in melt(cor_matrix): The melt generic in data.table has been passed a
## matrix and will attempt to redirect to the relevant reshape2 method; please
## note that reshape2 is deprecated, and this redirection is now deprecated as
## well. To continue using melt methods from reshape2 while both libraries are
## attached, e.g. melt.list, you can prepend the namespace like
## reshape2::melt(cor_matrix). In the next version, this warning will become an
## error.
```

```r
# Display the first few rows to check
head(cor_data)
```

```
##   Var1 Var2      value
## 1   RI   RI  1.0000000
## 2   Na   RI -0.1918854
## 3   Mg   RI -0.1222740
## 4   Al   RI -0.4073260
## 5   Si   RI -0.5420522
## 6    K   RI -0.2898327
```

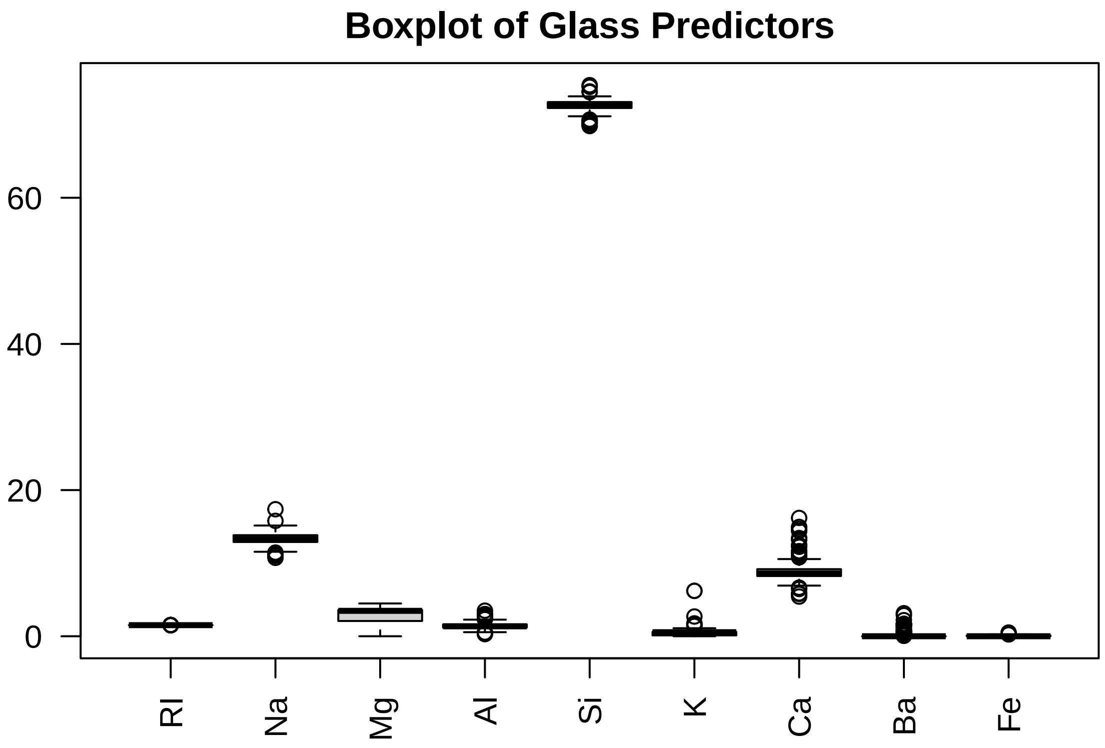
```r
# Create the correlation matrix plot
ggplot(cor_data, aes(Var1, Var2, fill = value)) +
  geom_tile(color = "white") +  # Use tiles to represent correlations
  scale_fill_gradient2(low = "blue", high = "red", mid = "white",
                       midpoint = 0, limit = c(-1,1), space = "Lab",
                       name = "Correlation") +
  theme_minimal() +  # Use a minimal theme
  theme(axis.text.x = element_text(angle = 45, vjust = 1, hjust = 1)) +
  coord_fixed() +  # Keep the squares proportional
  labs(title = "Correlation Matrix", x = "Variables", y = "Variables")
```



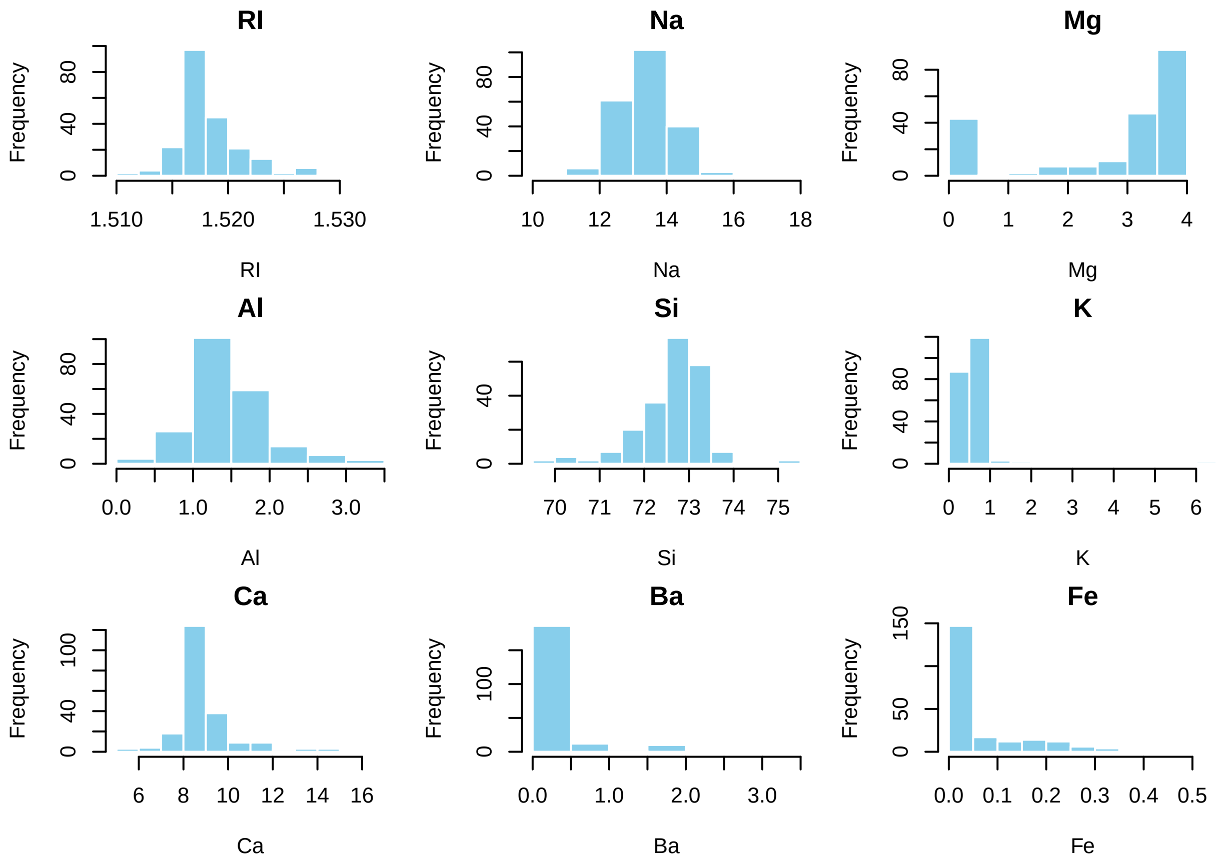
Do there appear to be any outliers in the data? Are any predictors skewed?

ANSWER: YES, Both the boxplot (which helps detect outliers),the histogram, and the pair plot(which here reveals some variables with non-linear trends), indicate the need for transformations. The histogram and the boxplot clearly show that most of the predictors are either right skewed or left skewed. All these plots reveal the presence of outliers in a number of the predictors.

```r
# Boxplot to check for outliers
par(mar=c(5,4,2,2))  # Adjust margins
boxplot(Glass[,1:9], main="Boxplot of Glass Predictors", las=2)
```



```r
# Histograms to check for skewness
par(mfrow=c(3,3), mar=c(4,4,2,1))  # Adjust the plotting area and margins
for(i in 1:9) {
  hist(Glass[,i], main=colnames(Glass)[i], xlab=colnames(Glass)[i], col='skyblue', border='white')
}
```

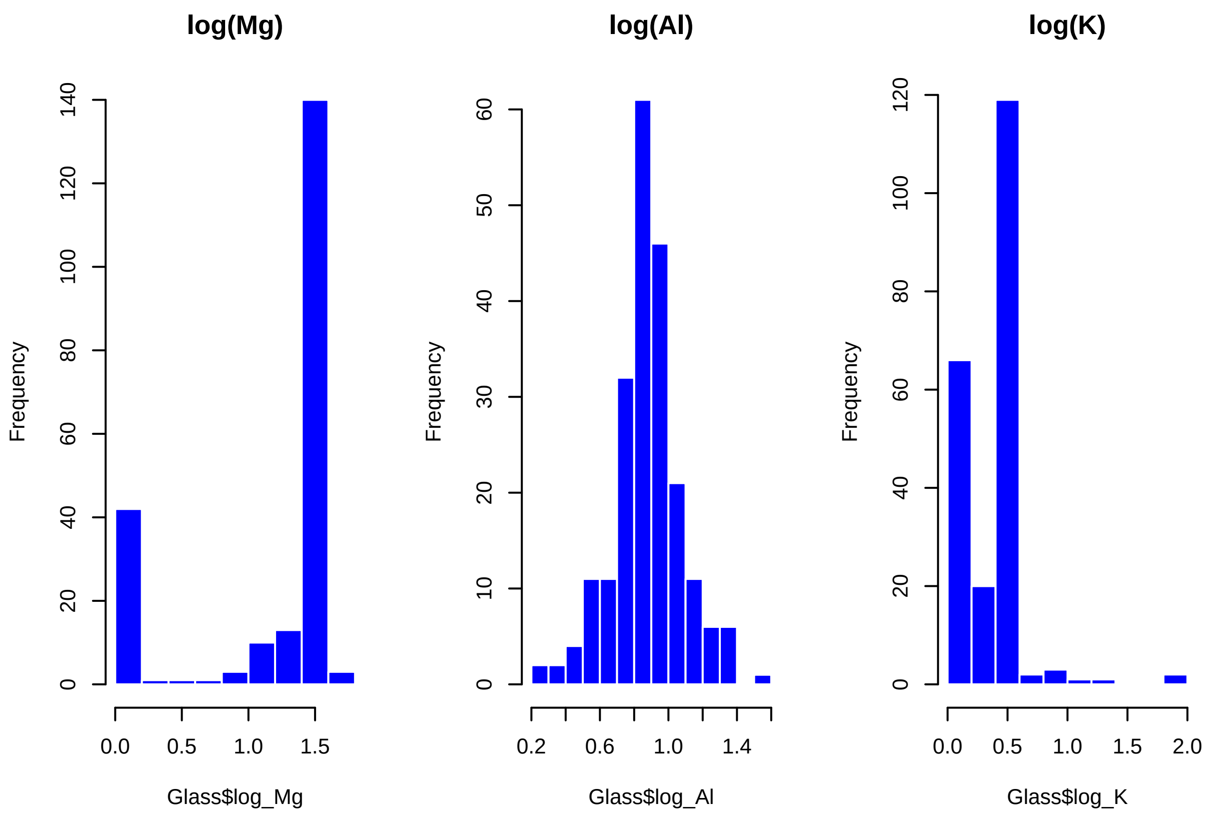


```r
# Reset the graphical parameters to default after plotting
par(mfrow=c(1,1))
```

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

YES, the predictors that are highly skewed (such as Fe,Mg, Al K) can benefit from log transformations as shwon below. Guerrero lambda could akso be helpful for variables that exhibit non-constant variance with skewed distributions or outliers, such as Mg or Fe. Scaling to standardize the variables and bring predictors to a comparable scale can be helpful for Na, Al, Ca) because of the difference in their ranges. Lastly, it can be useful to remove or combine highly correlated variables detected in a pair plot

```r
# Apply log transformation to skewed predictors
Glass$log_Mg <- log(Glass$Mg + 1)  # Adding 1 to avoid log(0)
Glass$log_Al <- log(Glass$Al + 1)
Glass$log_K <- log(Glass$K + 1)

# Visualize the transformed data
par(mfrow=c(1,3))
hist(Glass$log_Mg, main="log(Mg)", col='blue', border='white')
hist(Glass$log_Al, main="log(Al)", col='blue', border='white')
hist(Glass$log_K, main="log(K)", col='blue', border='white')
```



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:

```r
library(mlbench)
data(Soybean)
```

```r
head(Soybean)
```

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

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

```r
# Inspect the dataset
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 ...
```

```r
# Step 1: Investigate the frequency distributions of each column

frequency_distributions <- lapply(Soybean, table)

# Print the frequency distributions for each column
head(frequency_distributions)
```

```
## $Class
##
##                2-4-d-injury         alternarialeaf-spot
##                          16                          91
##                 anthracnose            bacterial-blight
##                          44                          20
##           bacterial-pustule                  brown-spot
##                          20                          92
##              brown-stem-rot                charcoal-rot
##                          44                          20
##               cyst-nematode diaporthe-pod-&-stem-blight
##                          14                          15
##       diaporthe-stem-canker                downy-mildew
##                          20                          20
##          frog-eye-leaf-spot            herbicide-injury
##                          91                           8
##      phyllosticta-leaf-spot            phytophthora-rot
##                          20                          88
##              powdery-mildew           purple-seed-stain
##                          20                          20
##        rhizoctonia-root-rot
##                          20
##
## $date
##
##   0   1   2   3   4   5   6
##  26  75  93 118 131 149  90
##
## $plant.stand
##
##   0   1
## 354 293
##
## $precip
##
##   0   1   2
##  74 112 459
##
## $temp
```

```
##
##   0   1   2
##  80 374 199
##
## $hail
##
##   0   1
## 435 127
```

```
# Step 2: Identify degenerate columns (those with only one unique value)
degenerate_columns <- Soybean %>%
  summarise(across(everything(), ~ length(unique(.)) == 1))
# Find the column indices that are degenerate
degenerate_columns <- which(degenerate_columns == TRUE)
degenerate_columns
```

```
## integer(0)
```

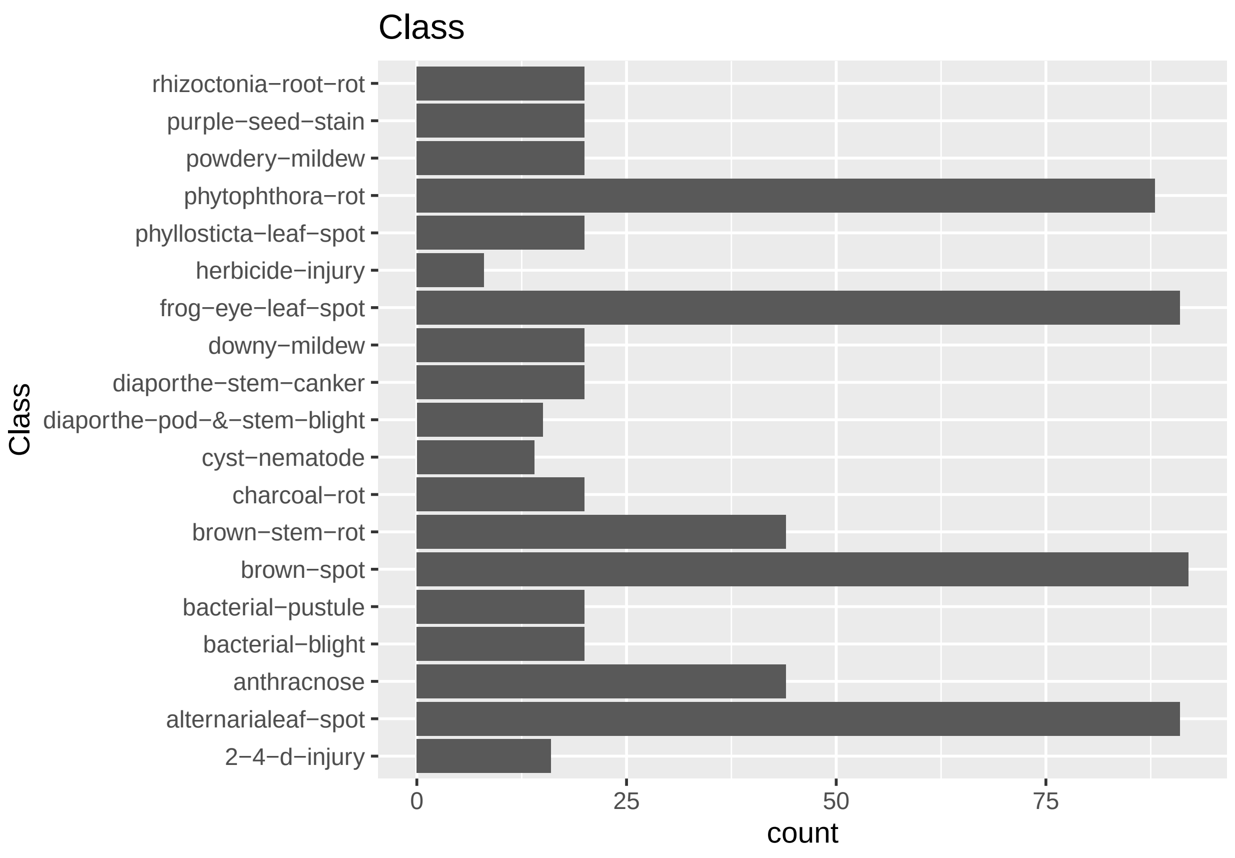
```
# Print the degenerate columns (if any)
if (length(degenerate_columns) > 0) {
  print(paste("Degenerate columns:", degenerate_columns))
} else {
  print("No degenerate columns found.")
}
```

```
## [1] "No degenerate columns found."
```

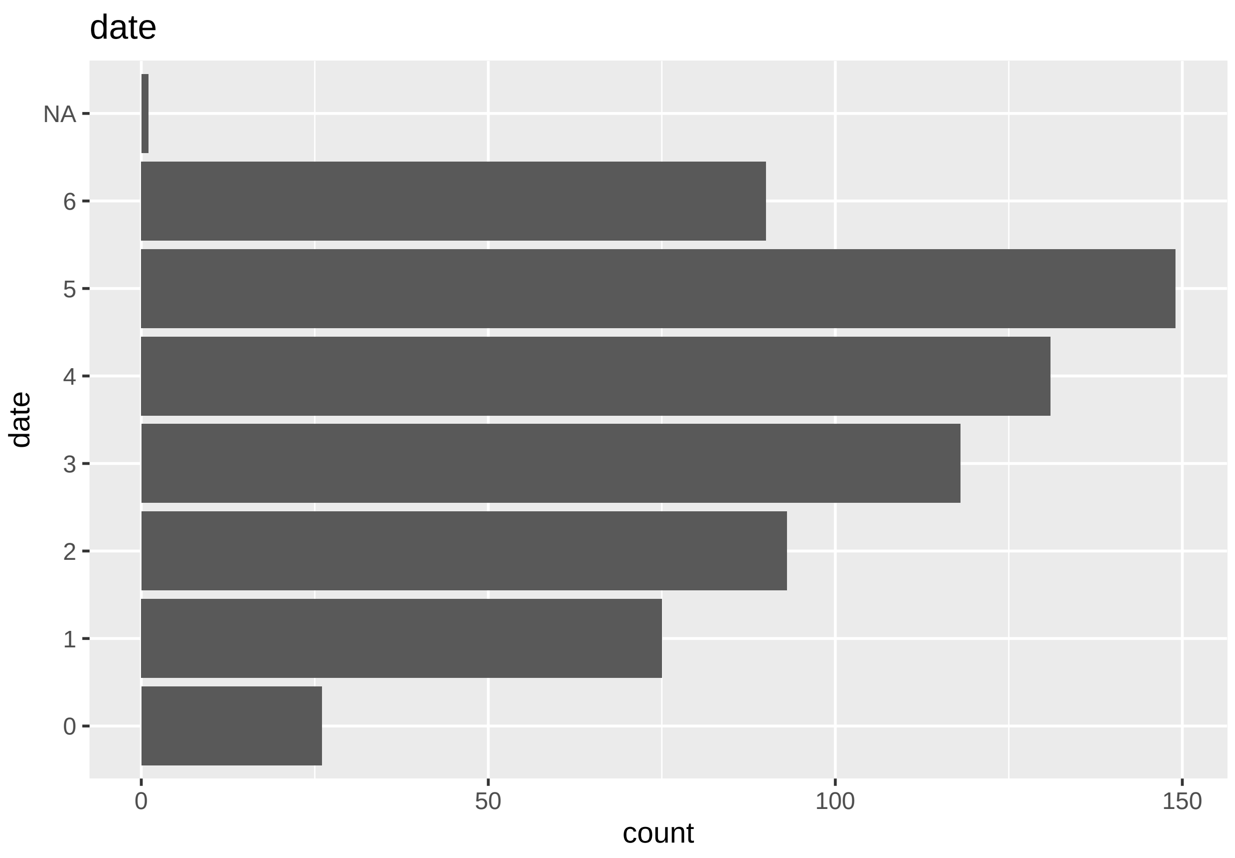
```
data(Soybean)

columns <- colnames(Soybean)

lapply(columns,
  function(col) {
    ggplot(Soybean,
           aes_string(col)) + geom_bar() + coord_flip() + ggtitle(col)})
```

```
## Warning: `aes_string()` was deprecated in ggplot2 3.0.0.
## i Please use tidy evaluation idioms with `aes()`.
## i See also `vignette("ggplot2-in-packages")` for more information.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

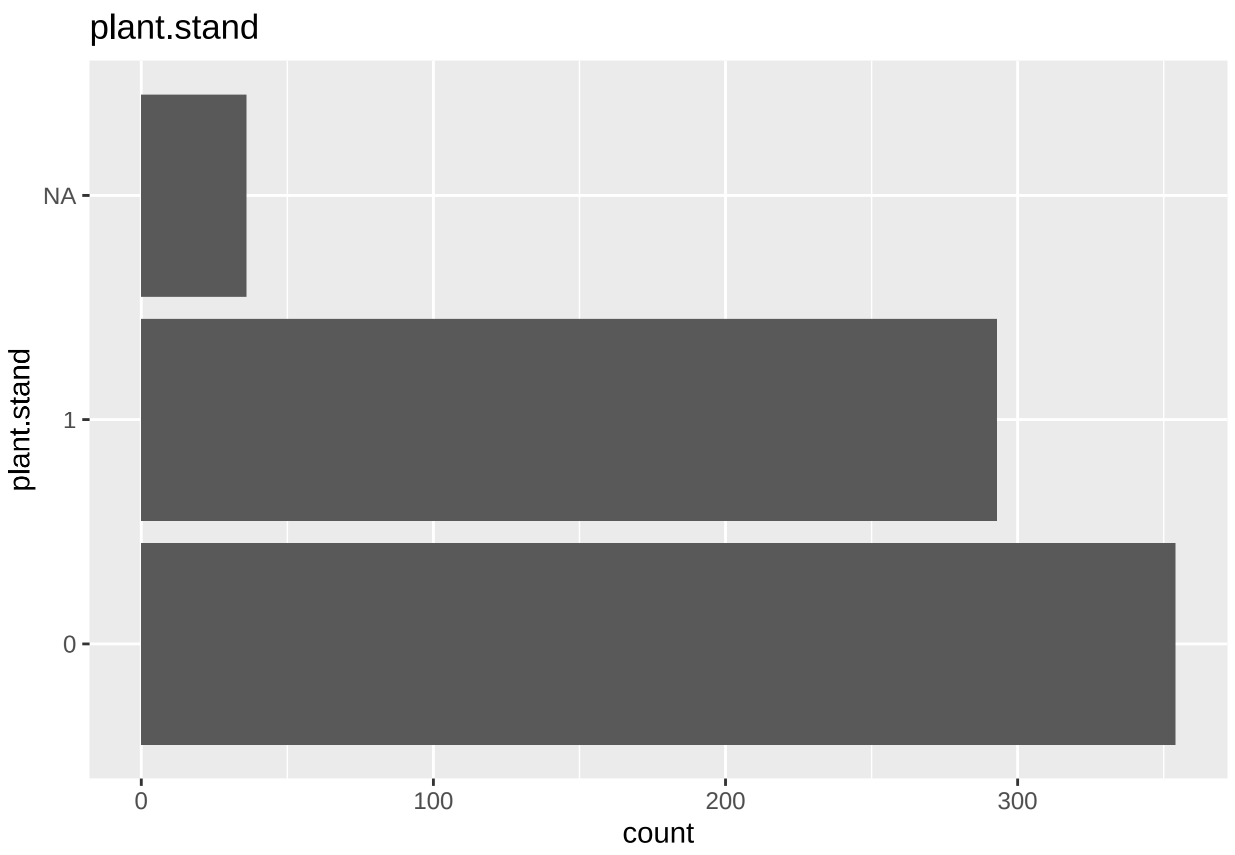
```
## [[1]]
```


Class
rhizoctonia−root−rot
purple−seed−stain
powdery−mildew
phytophthora−rot
phyllosticta−leaf−spot
herbicide−injury
frog−eye−leaf−spot
downy−mildew
diaporthe−stem−canker
diaporthe−pod−&−stem−blight
cyst−nematode
charcoal−rot
brown−stem−rot
brown−spot
bacterial−pustule
bacterial−blight
anthracnose
alternarialeaf−spot
2−4−d−injury
Class
0
25
50
75
count

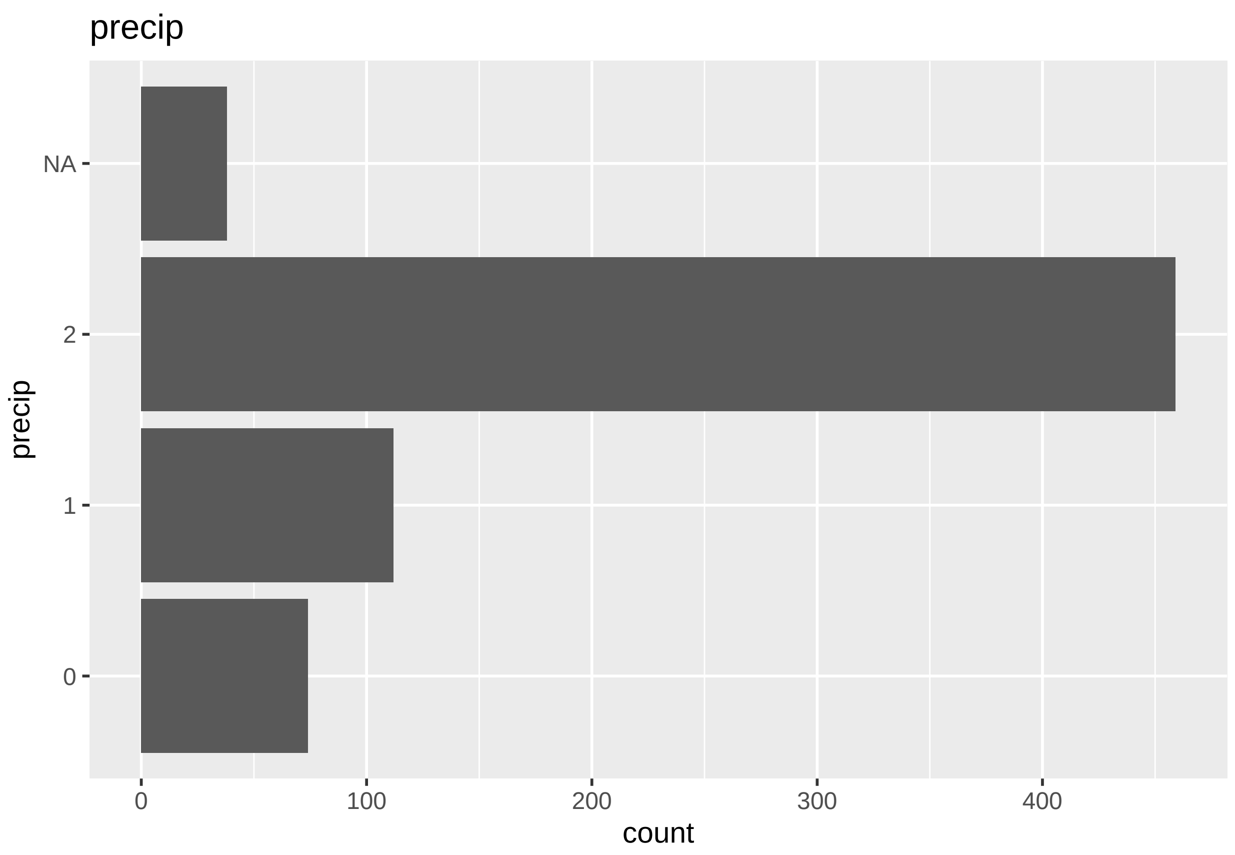
```
##
## [[2]]
```



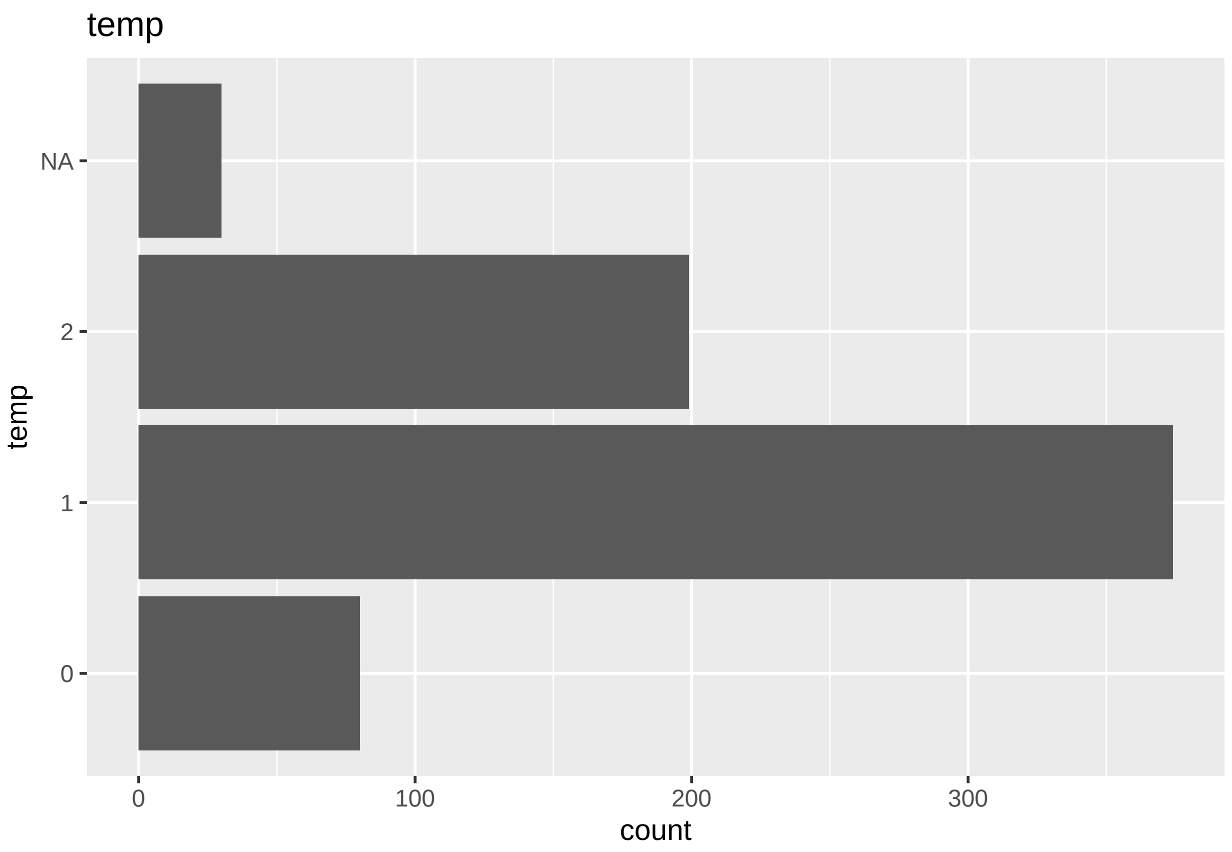
```
##
## [[3]]
```



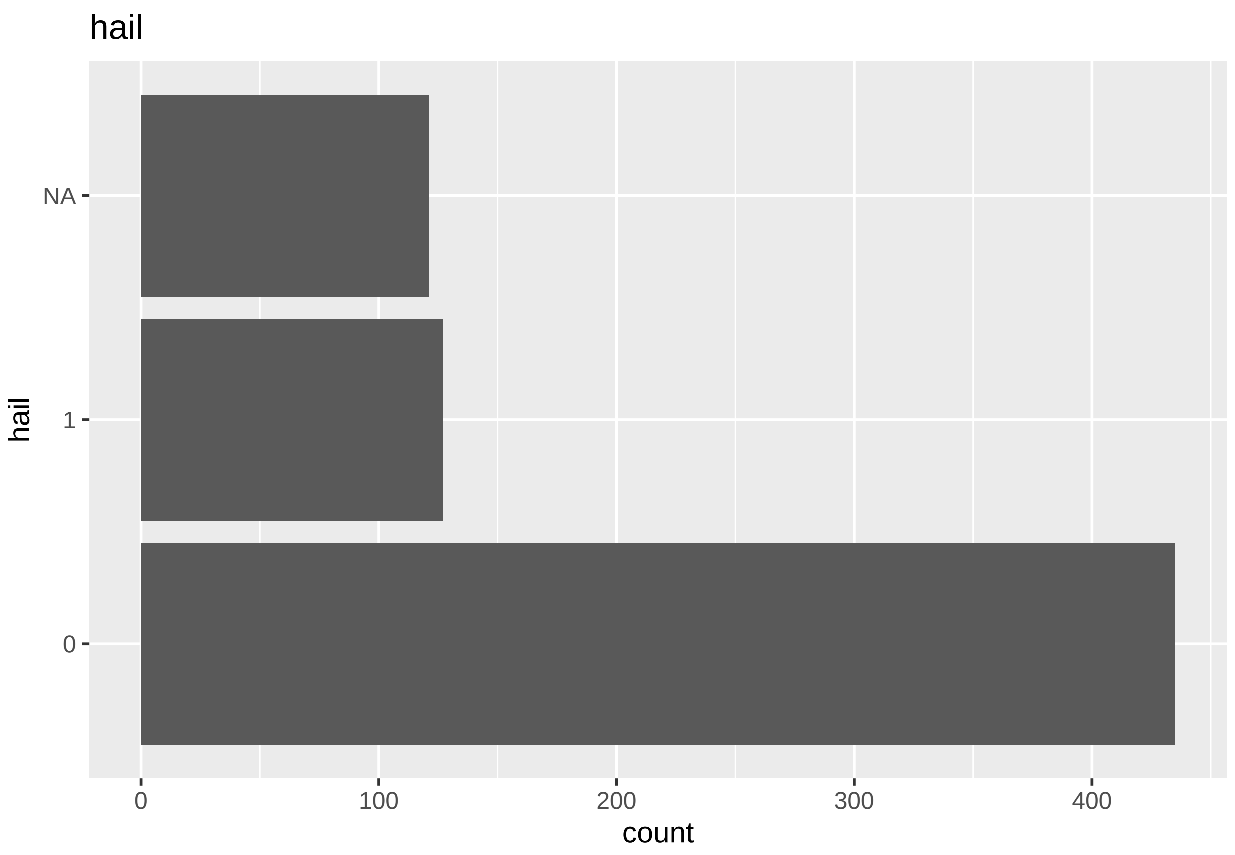
```
##
## [[4]]
```



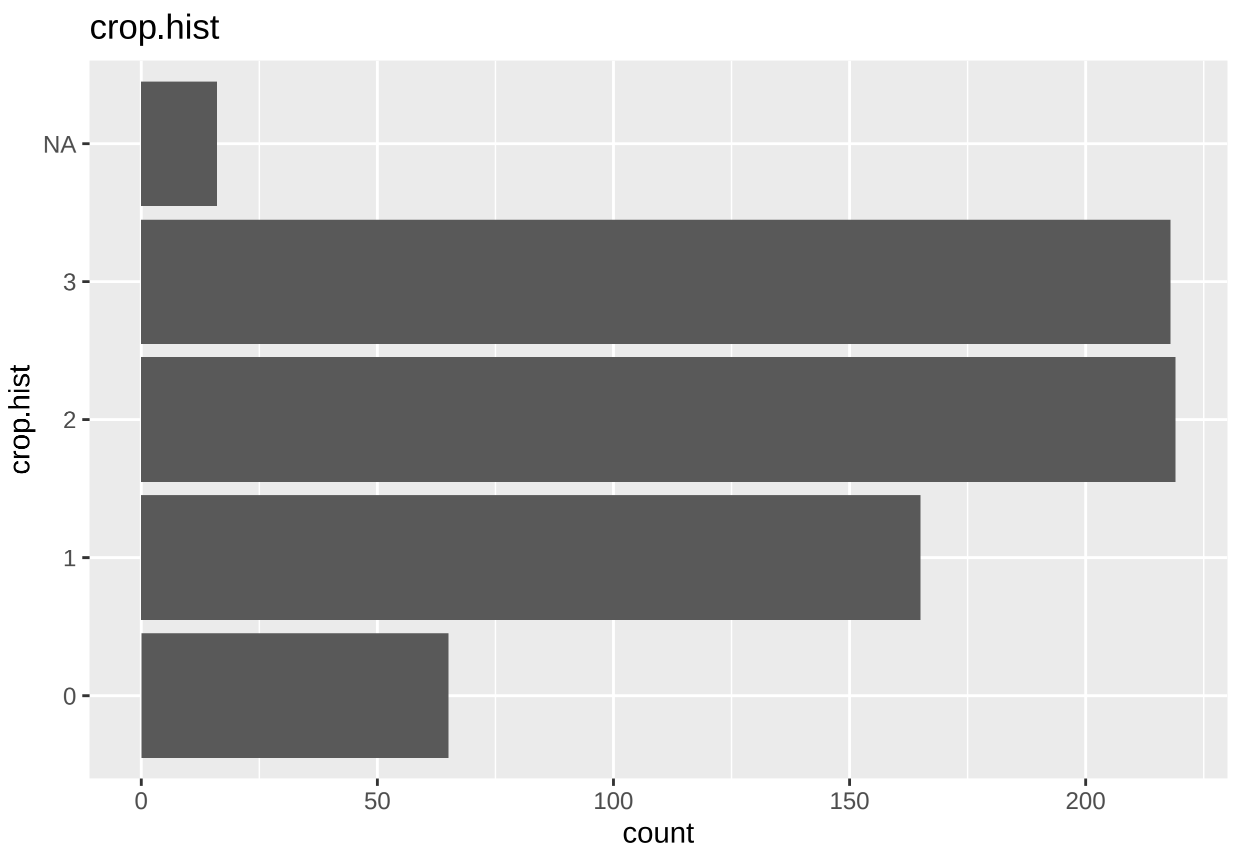
```
##
## [[5]]
```



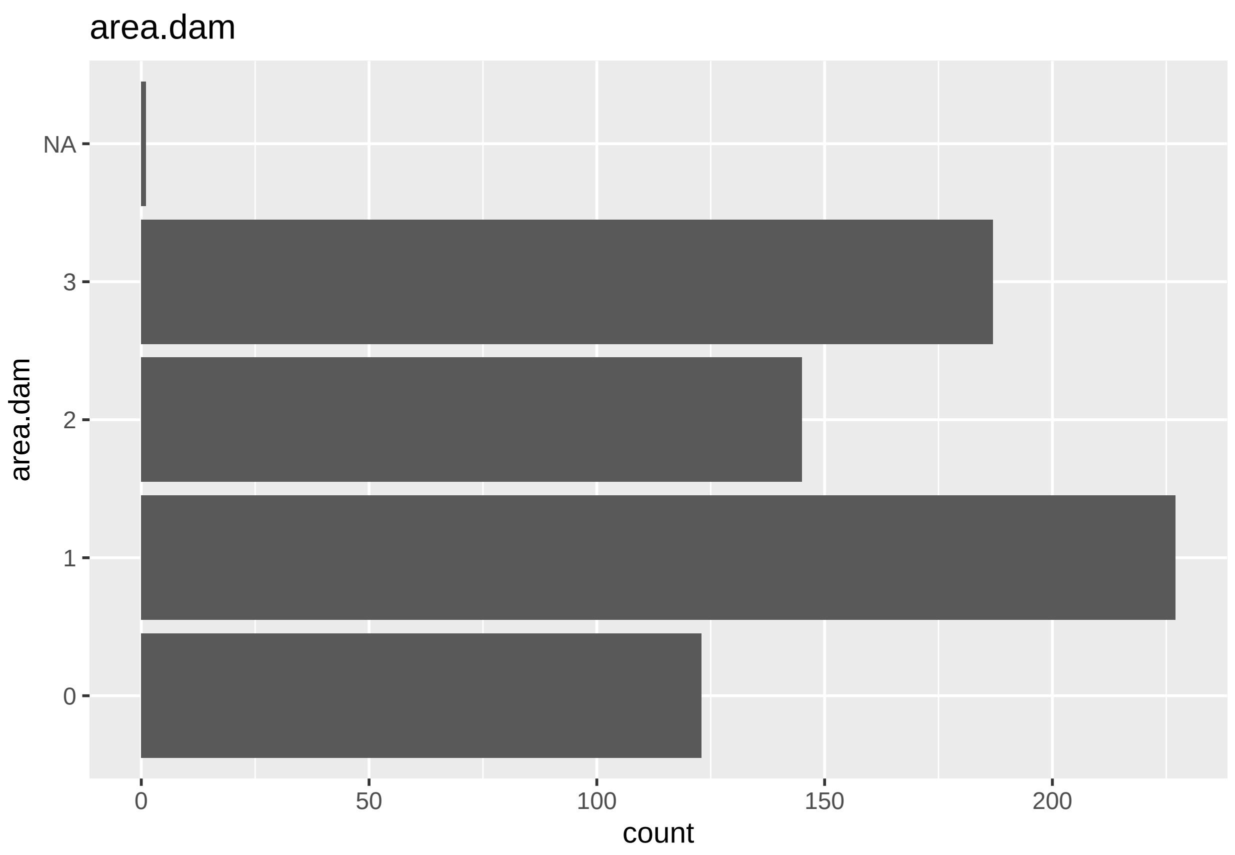
```
##
## [[6]]
```



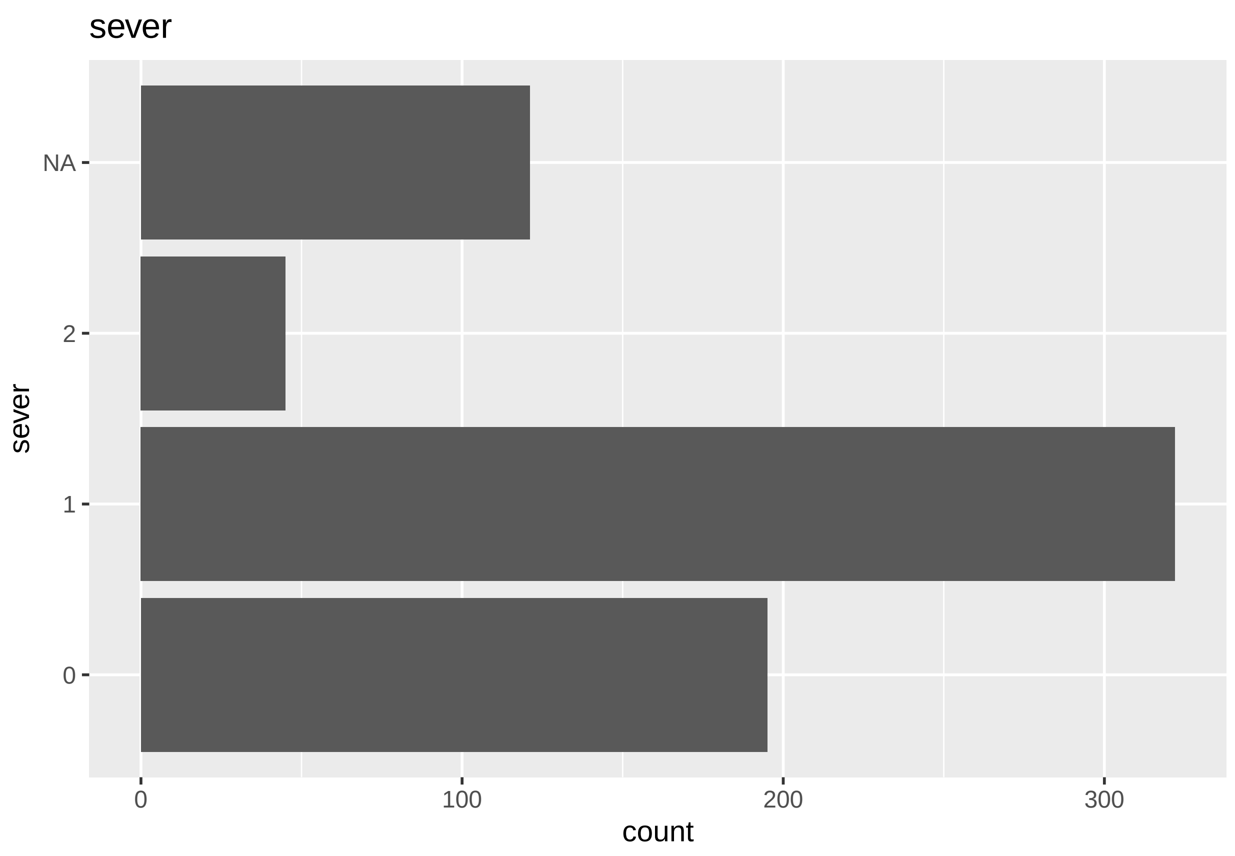
```
##
## [[7]]
```



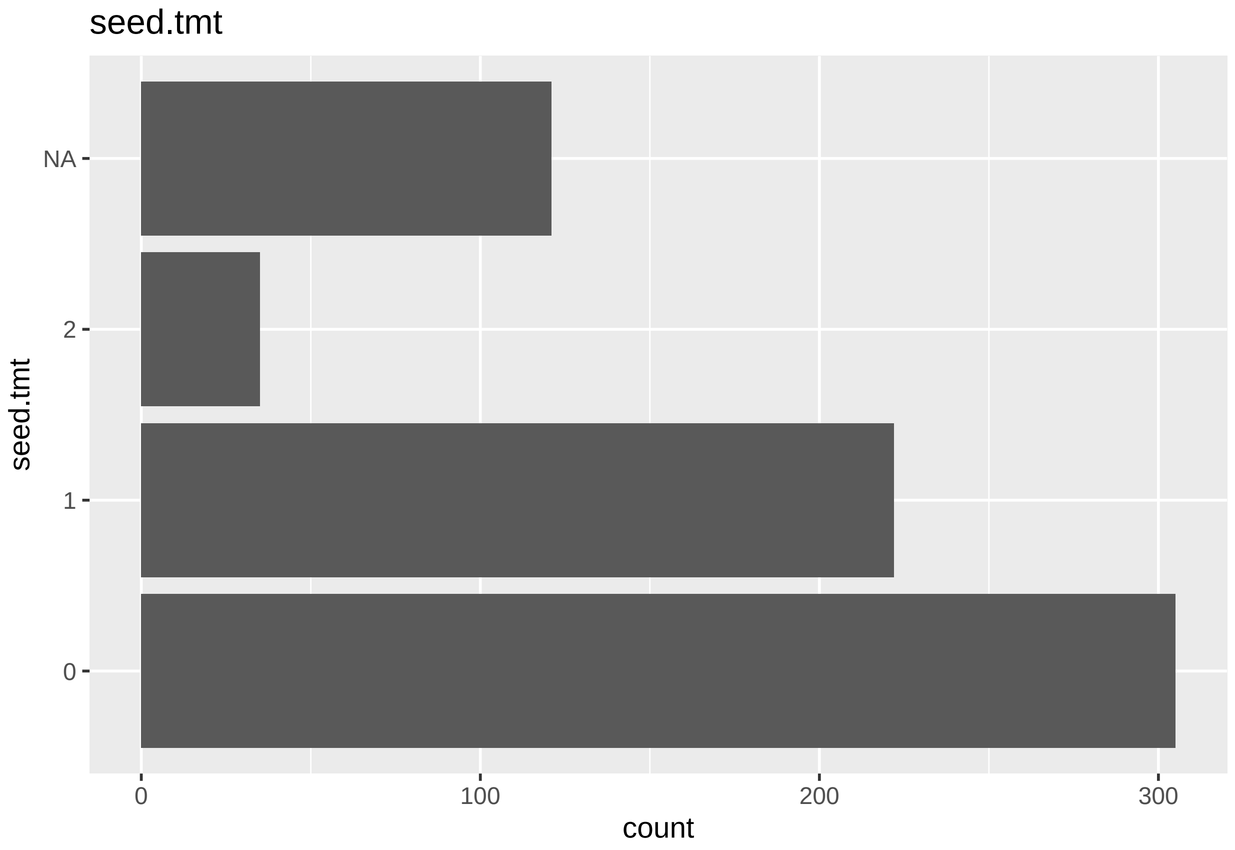
```
##
## [[8]]
```



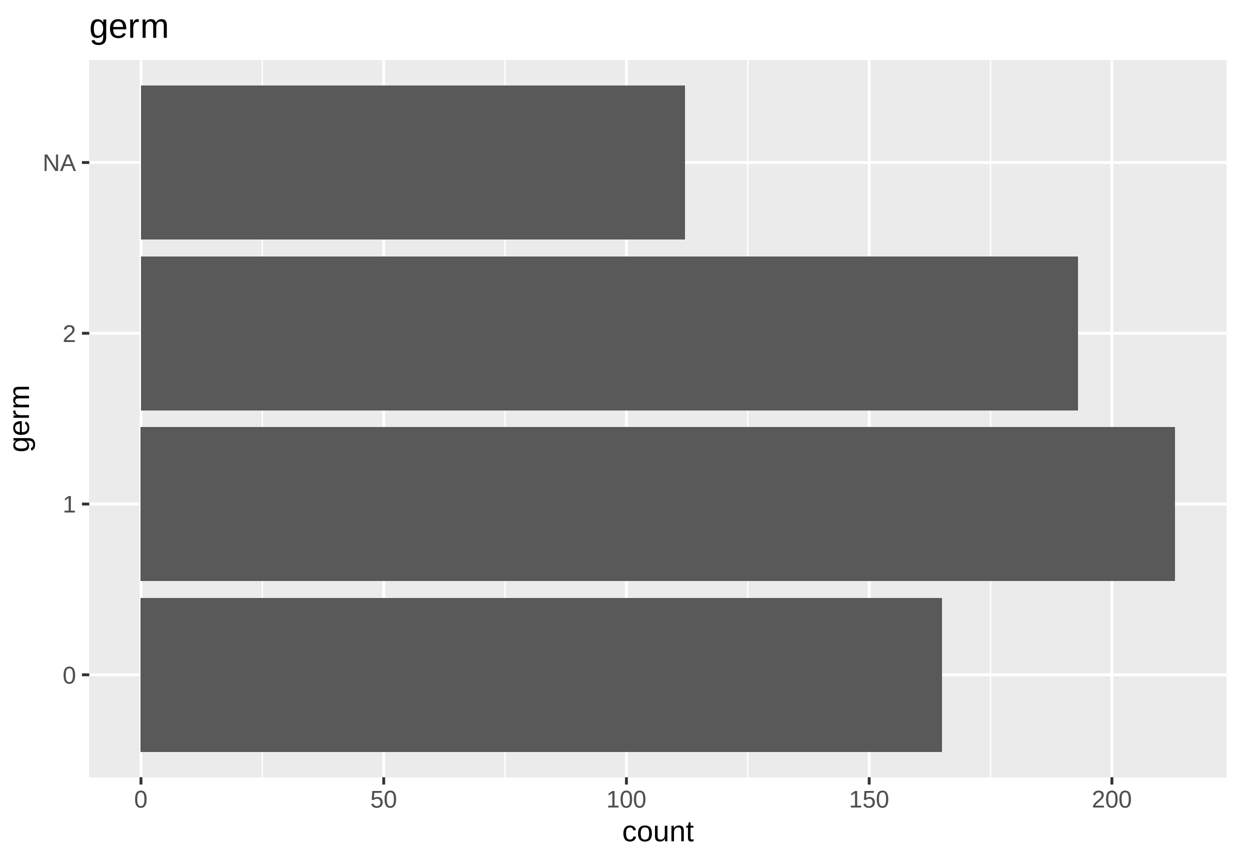
```
##
## [[9]]
```



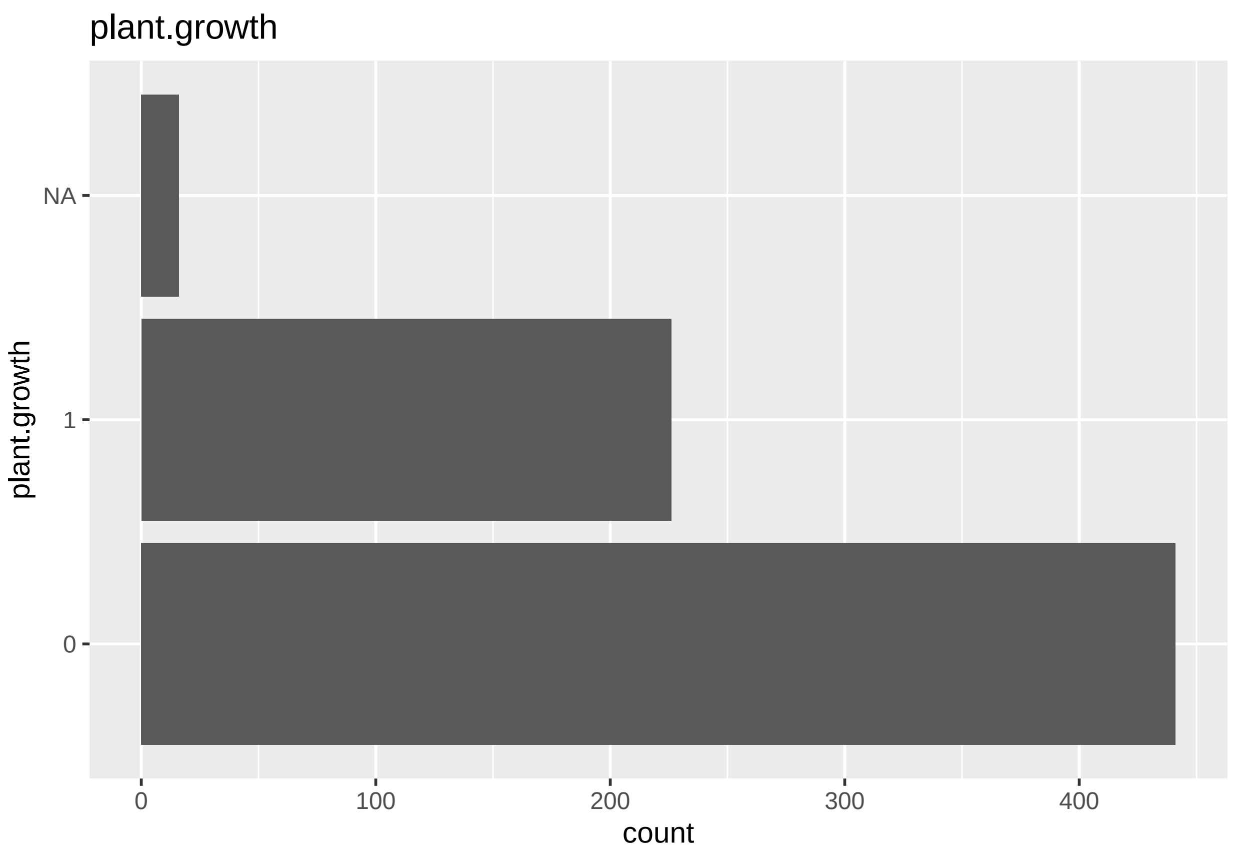
```
##
## [[10]]
```



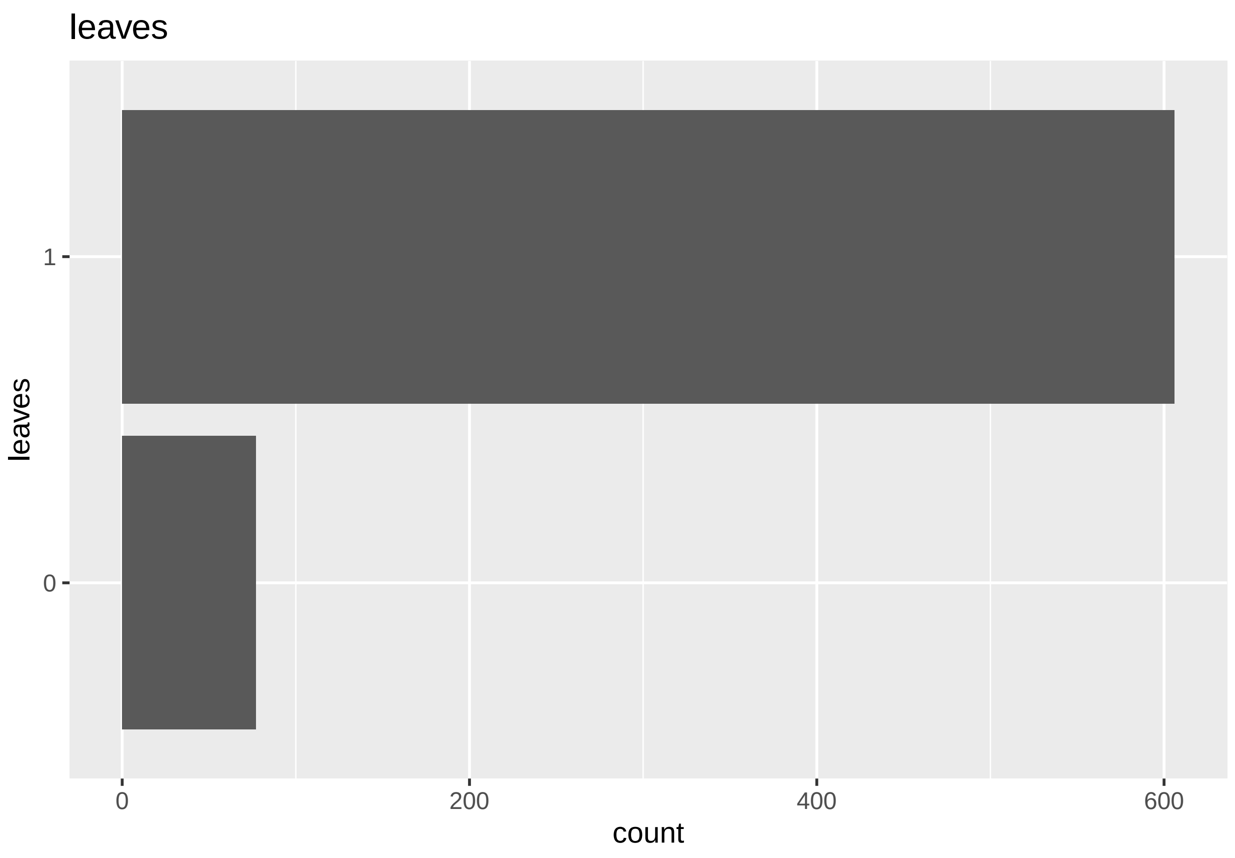
```
##
## [[11]]
```



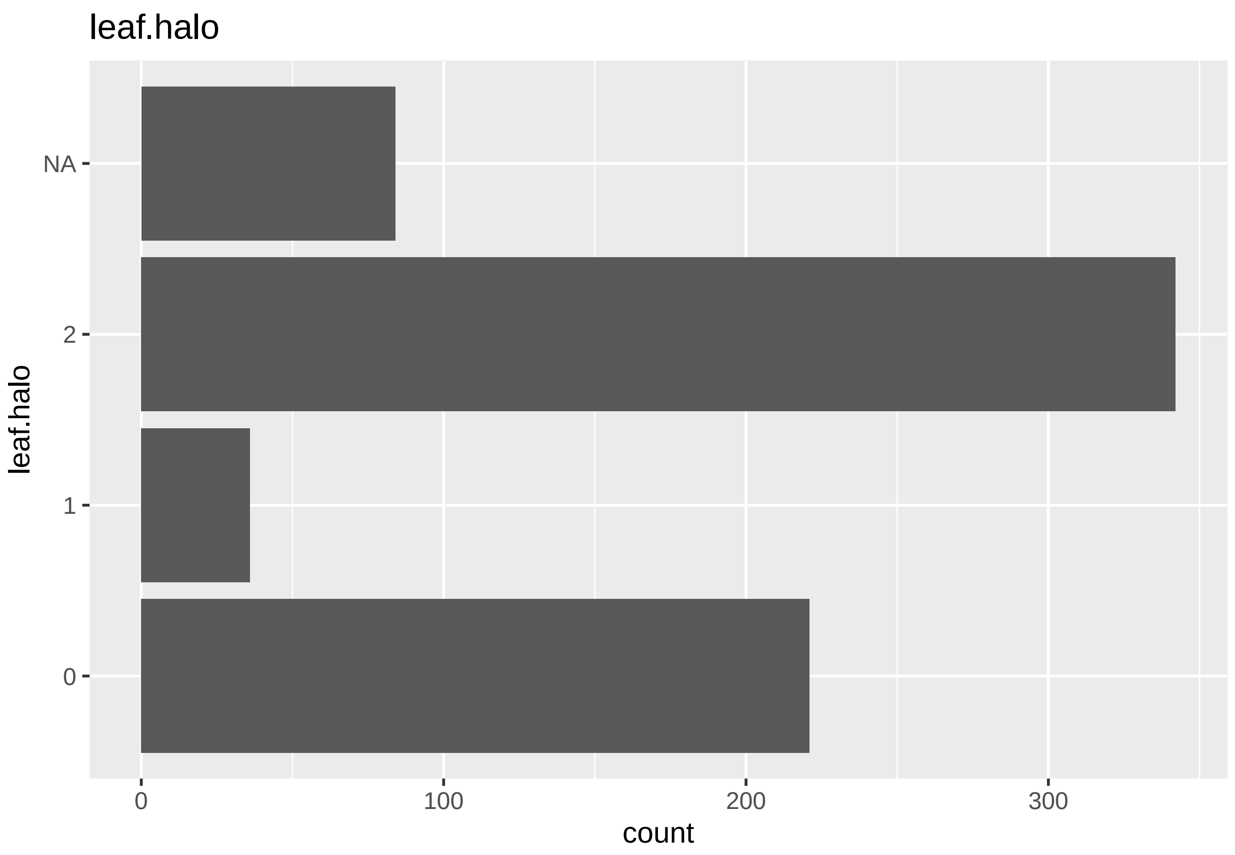
```
##
## [[12]]
```



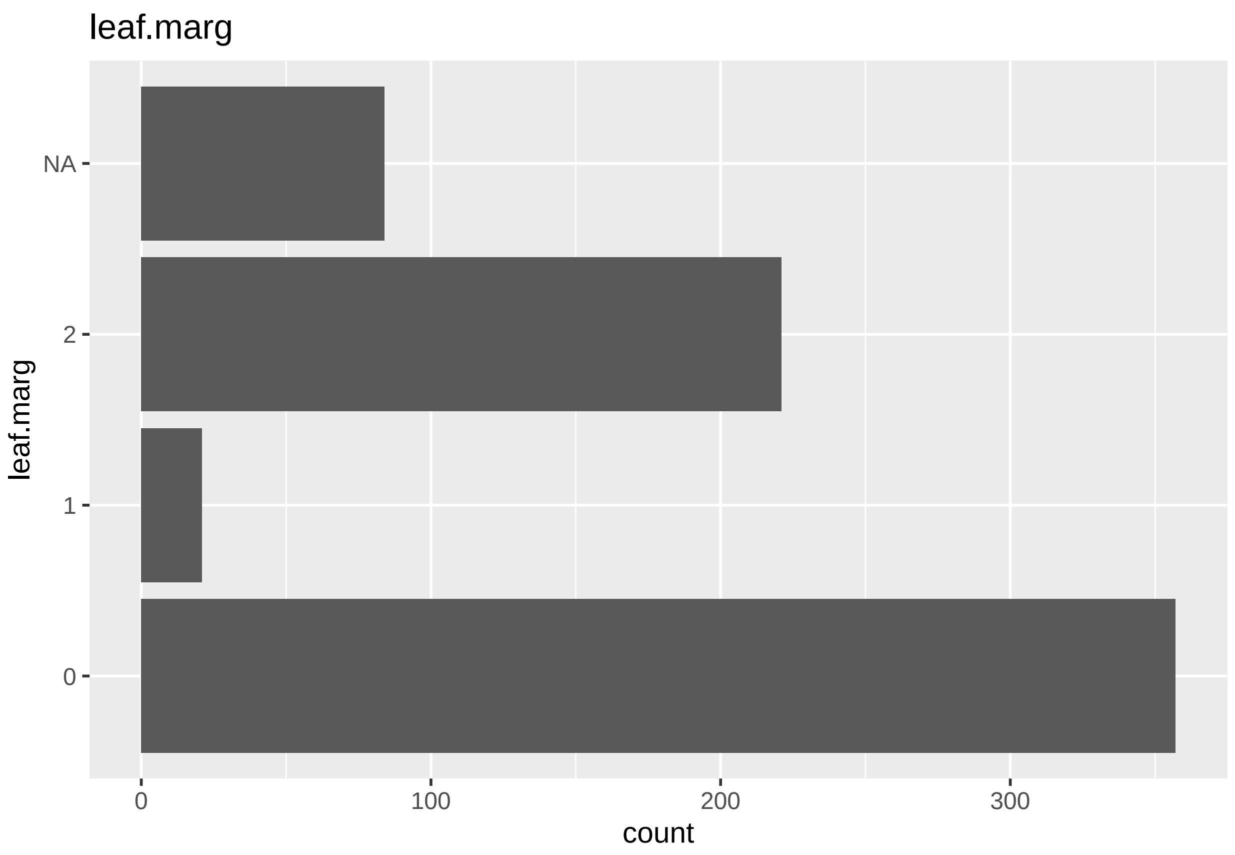
```
##
## [[13]]
```



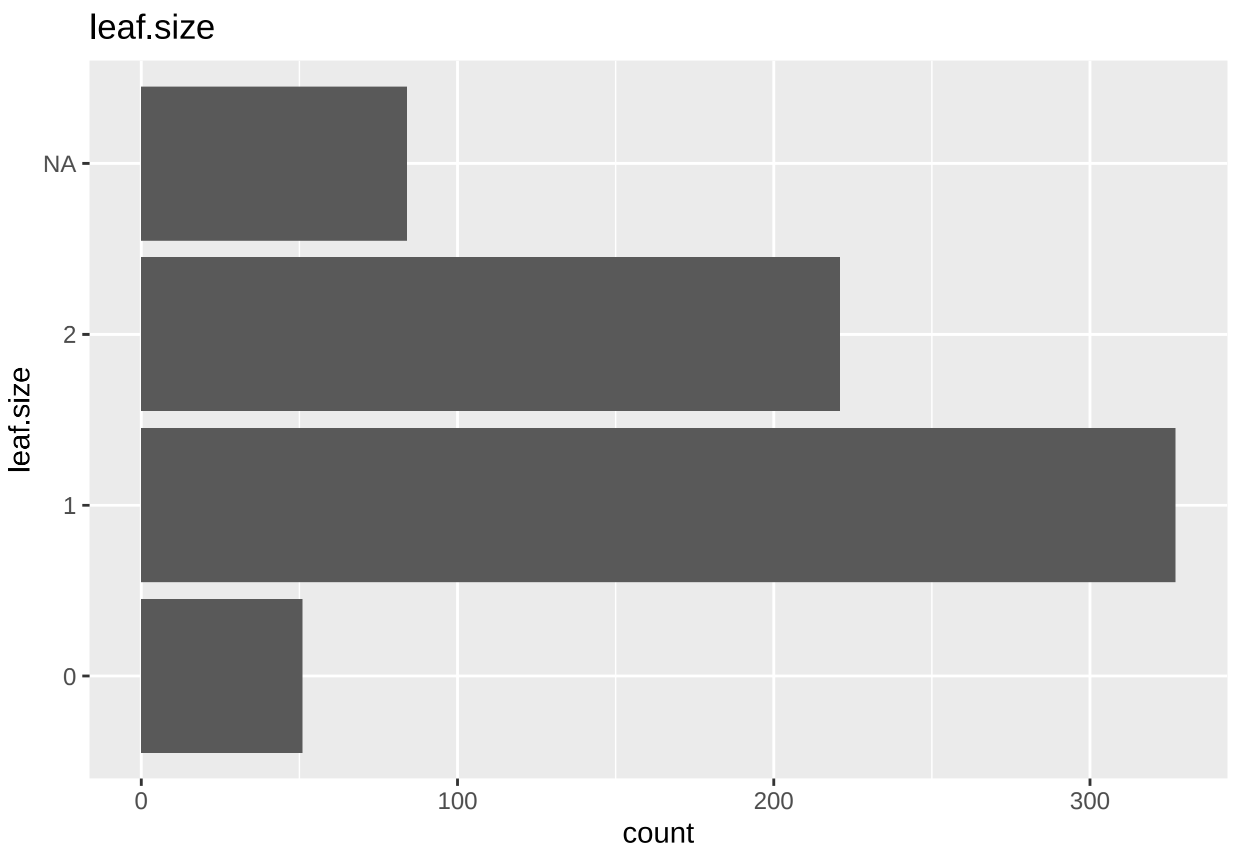
```
##
## [[14]]
```



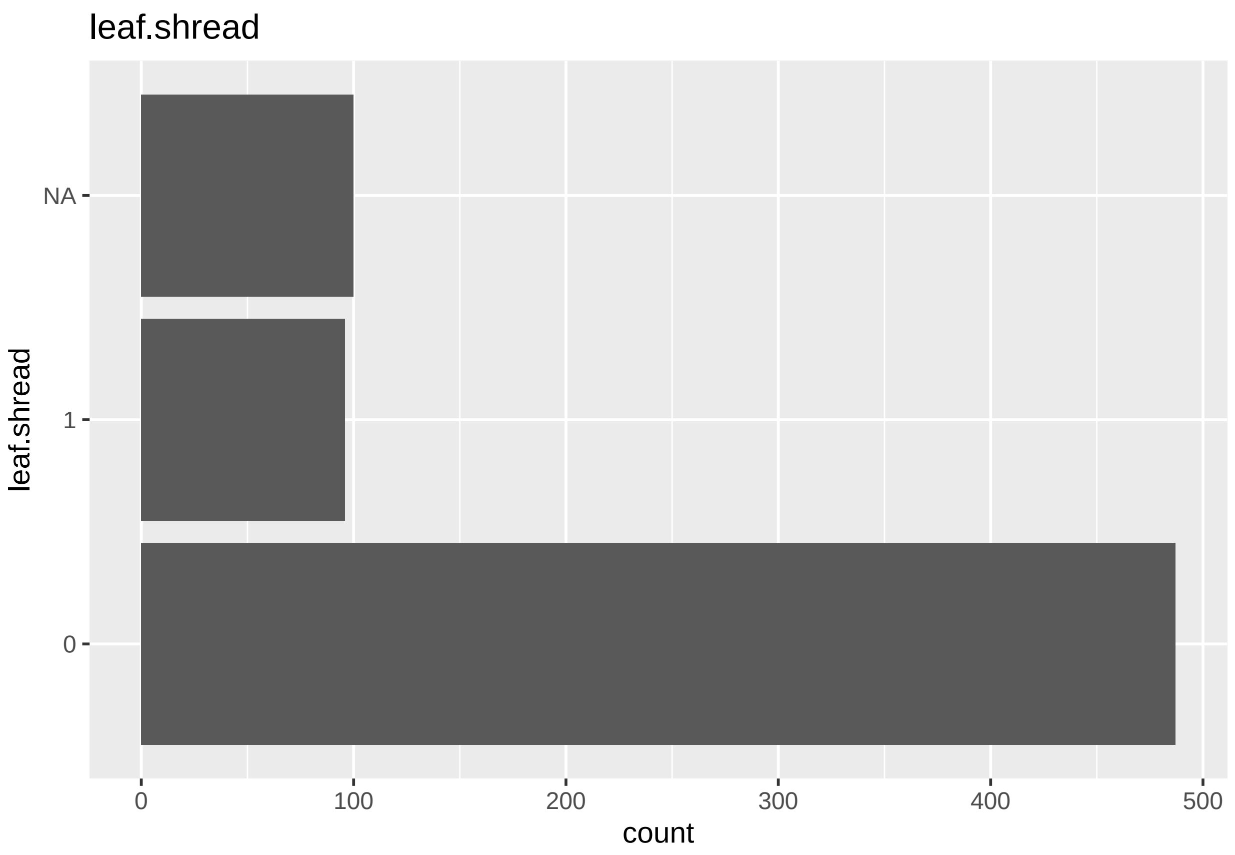
```
##
## [[15]]
```



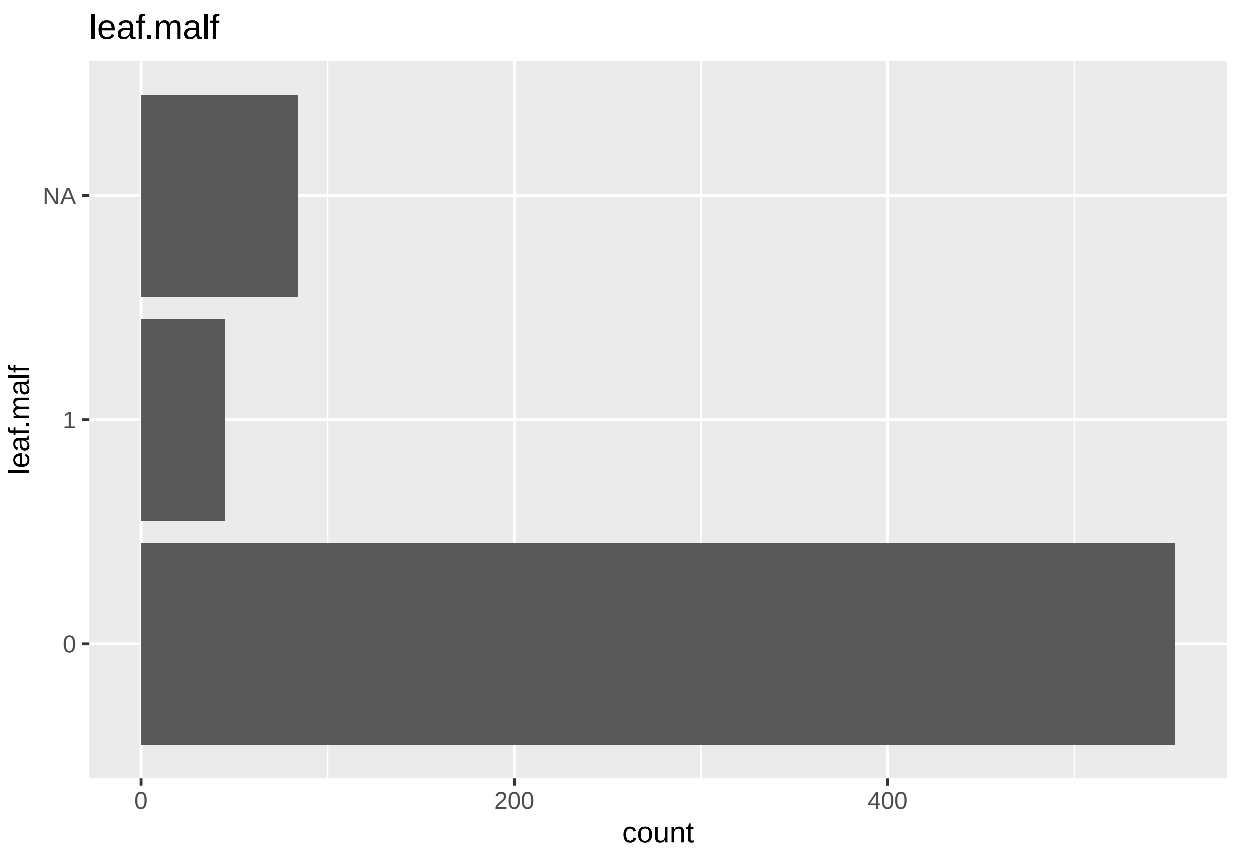
```
##
## [[16]]
```



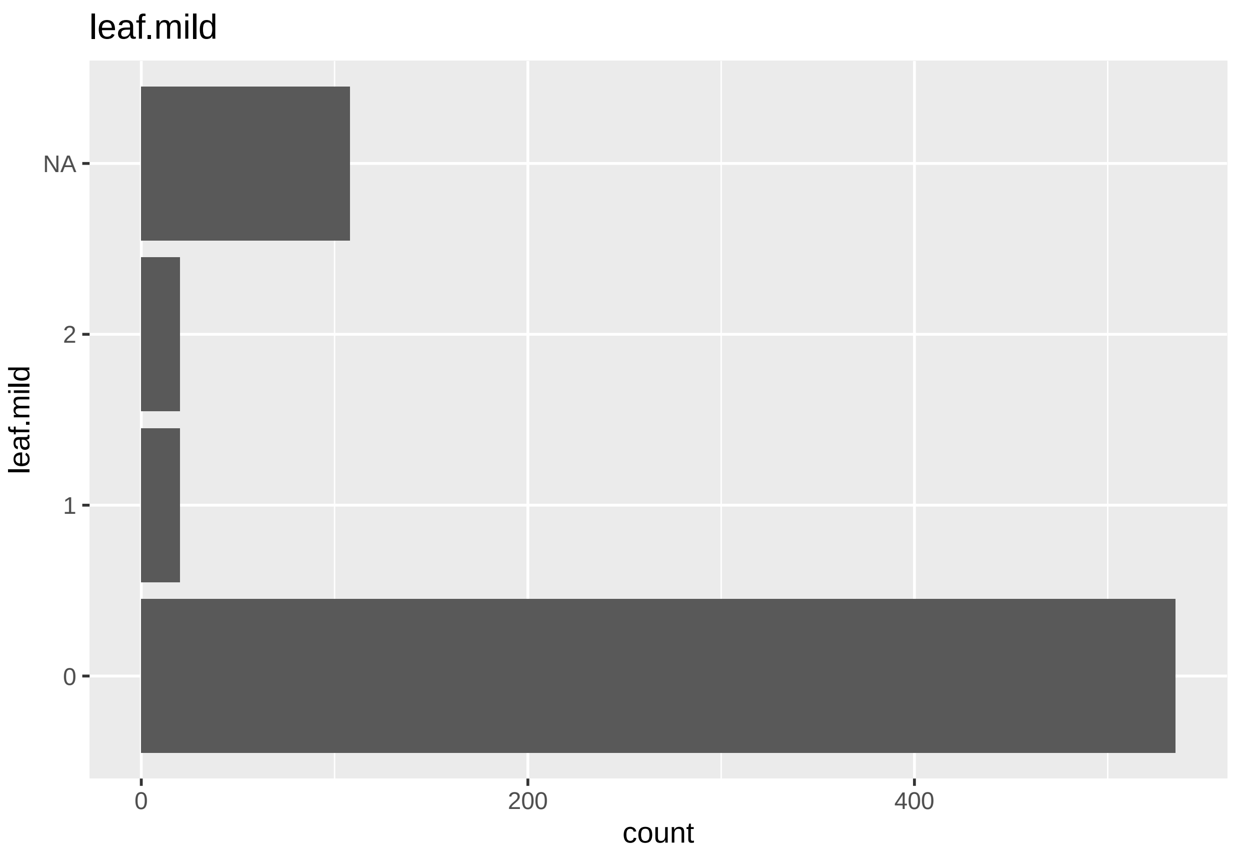
```
##
## [[17]]
```



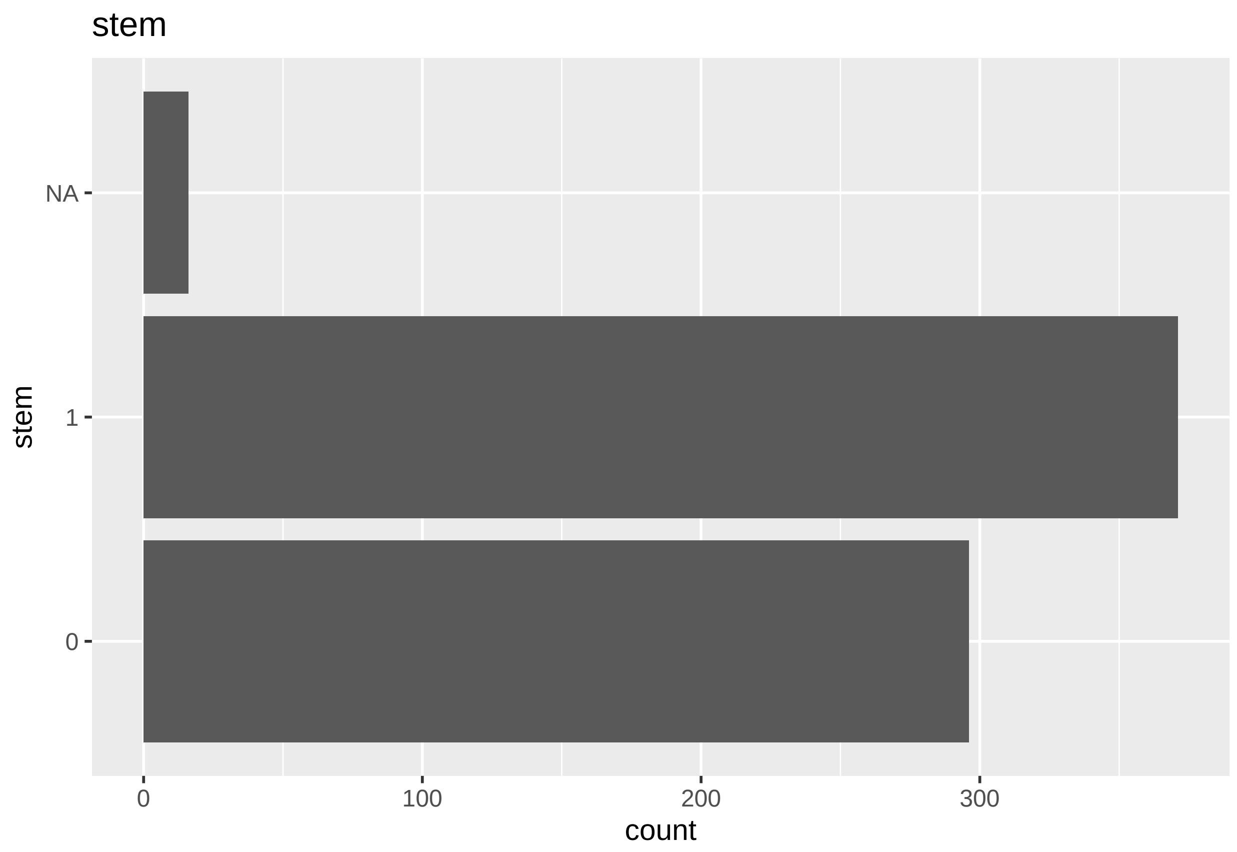
```
##
## [[18]]
```



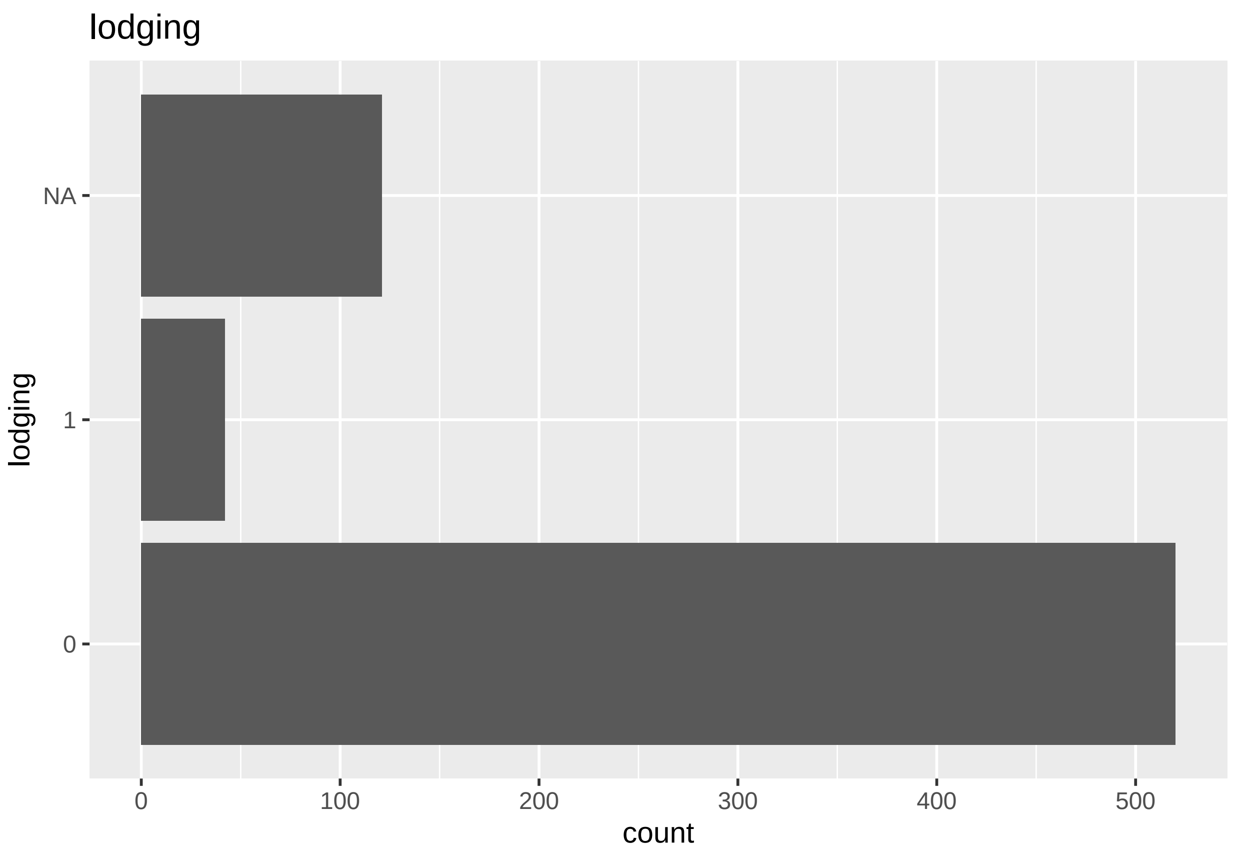
```
##
## [[19]]
```



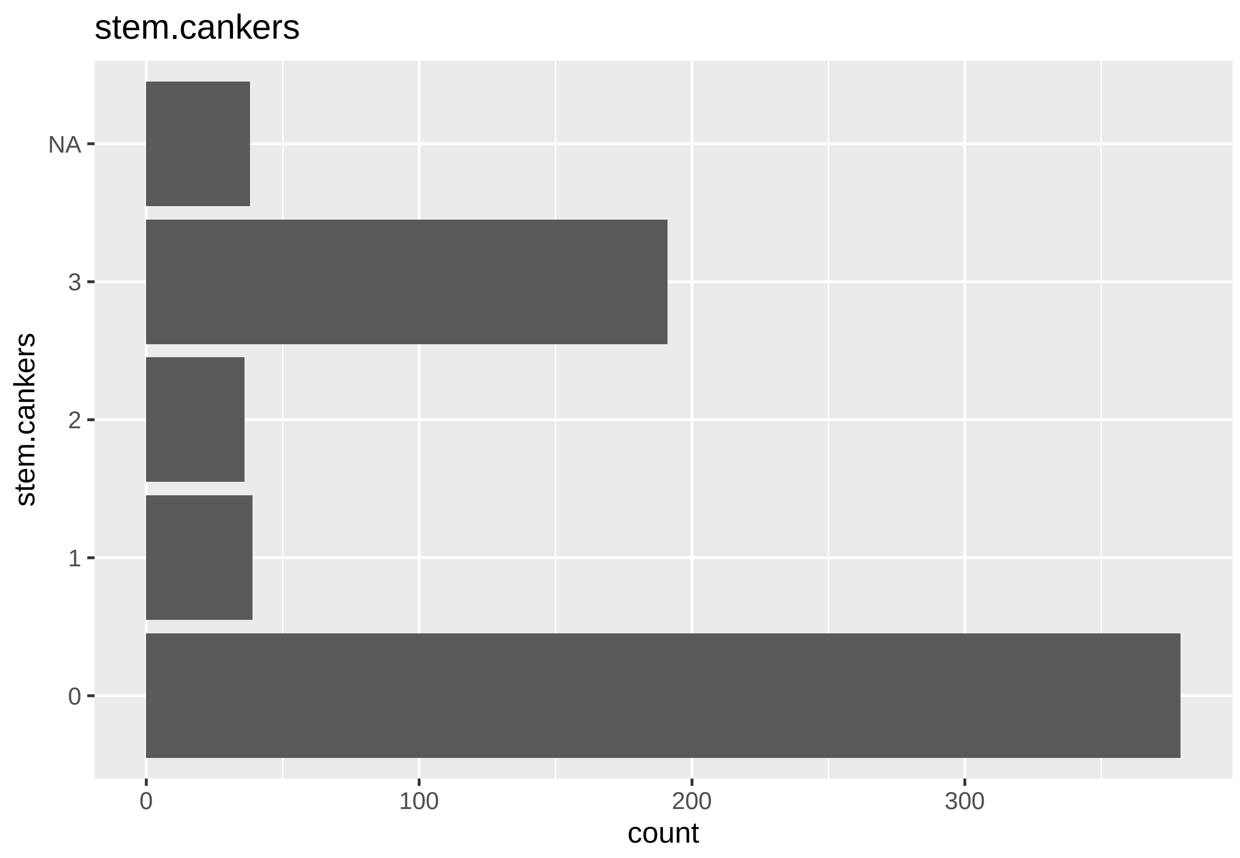
```
##
## [[20]]
```



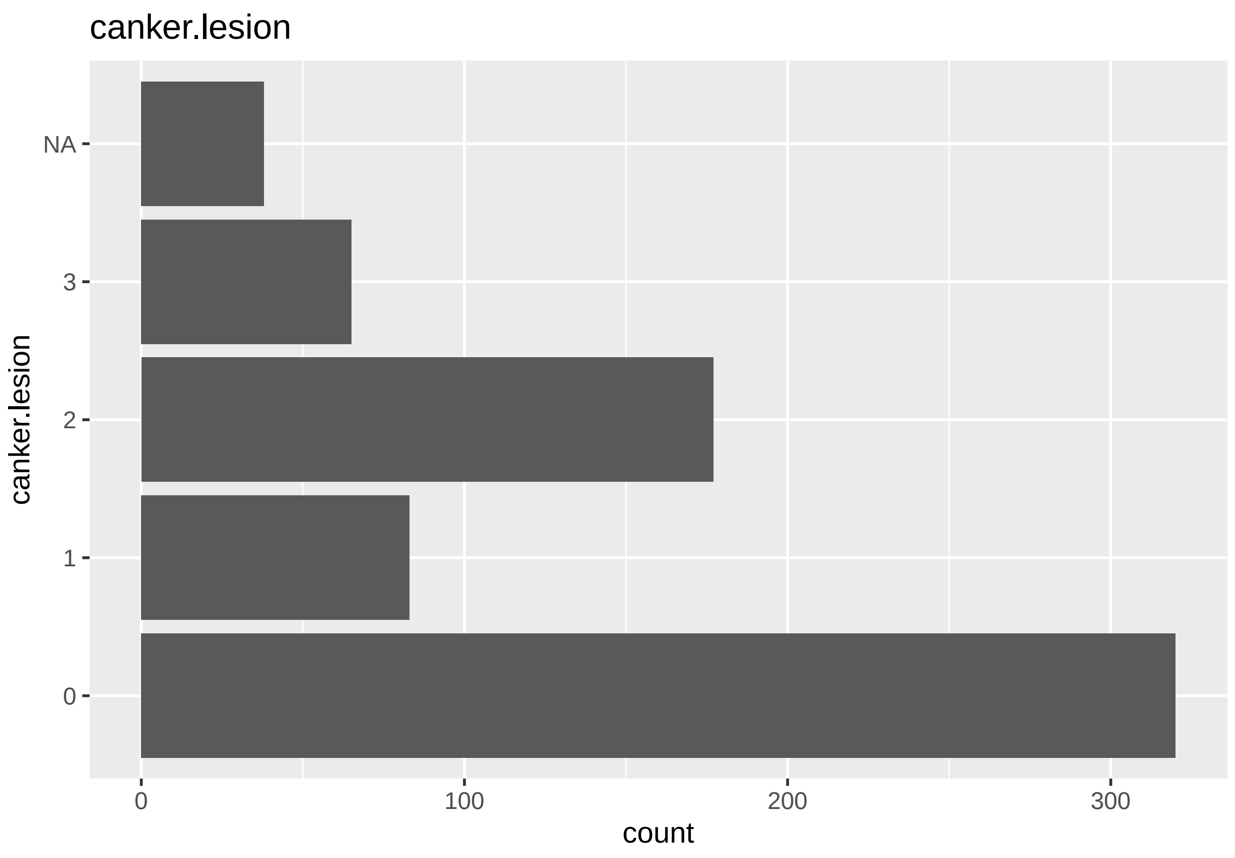
```
##
## [[21]]
```



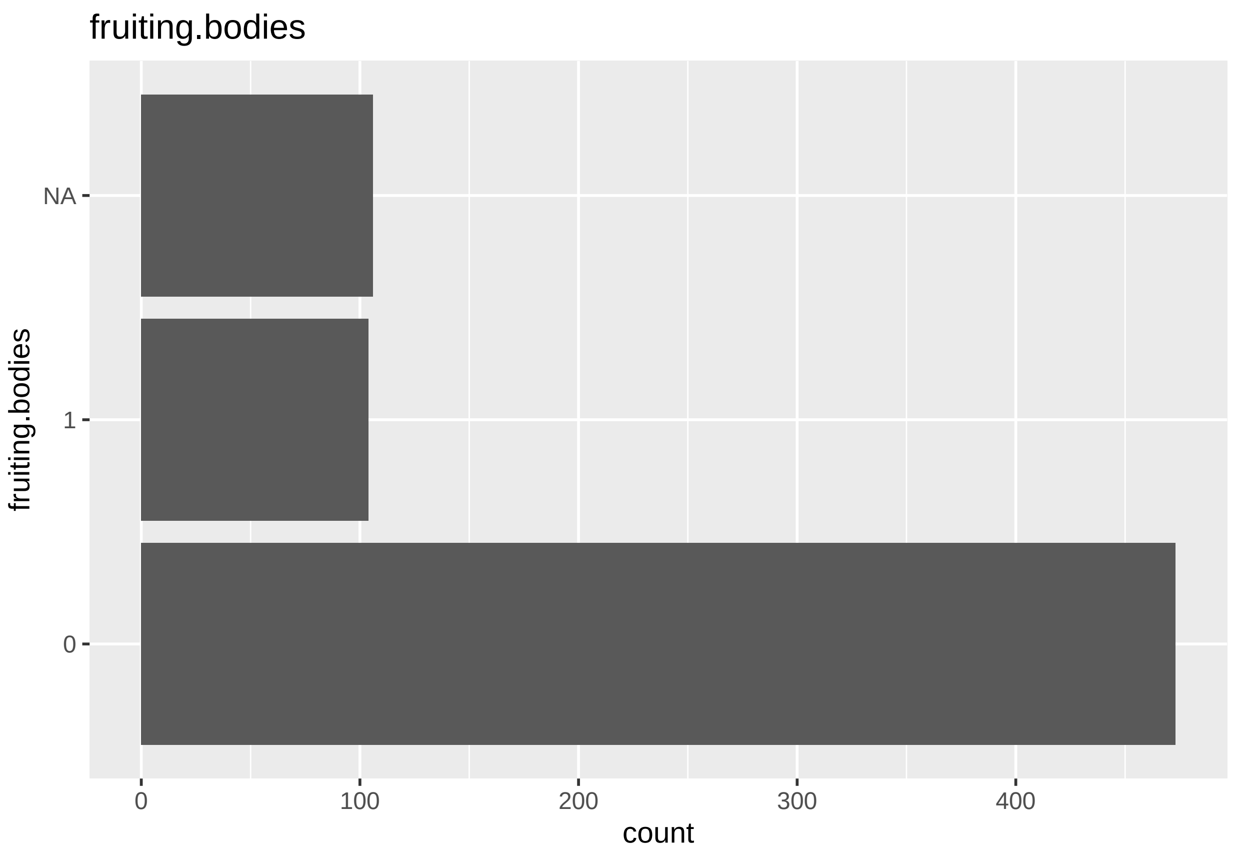
```
##
## [[22]]
```



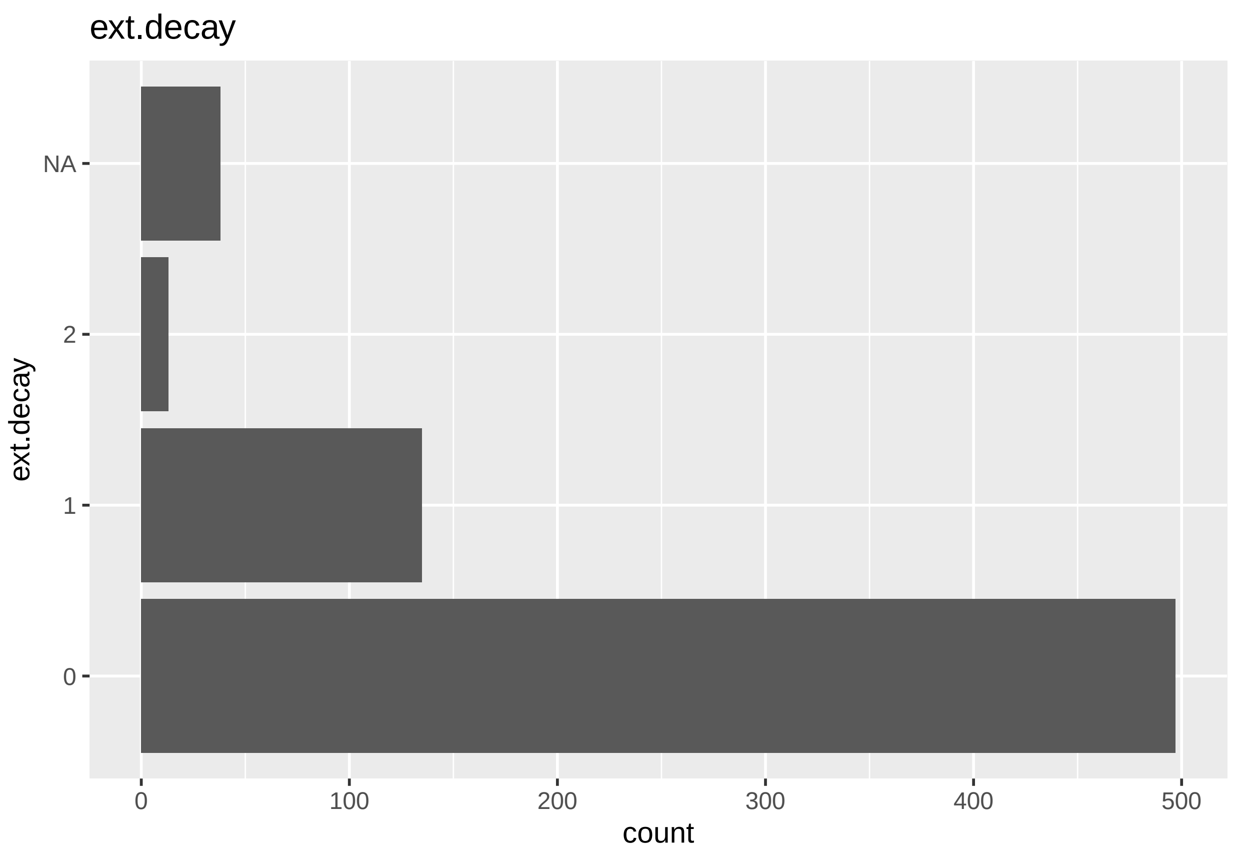
```
##
## [[23]]
```



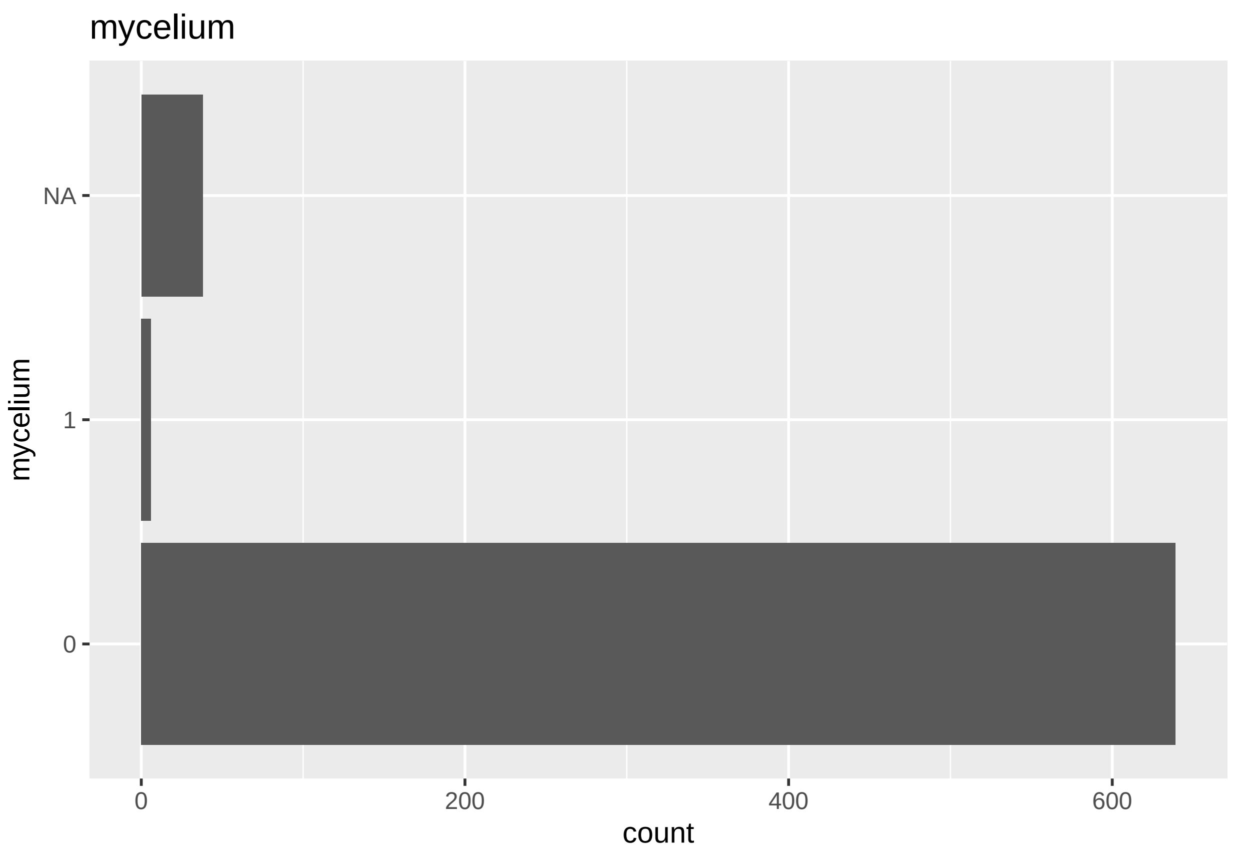
```
##
## [[24]]
```


fruiting.bodies
NA
1
0
fruiting.bodies
0
100
200
300
400
count

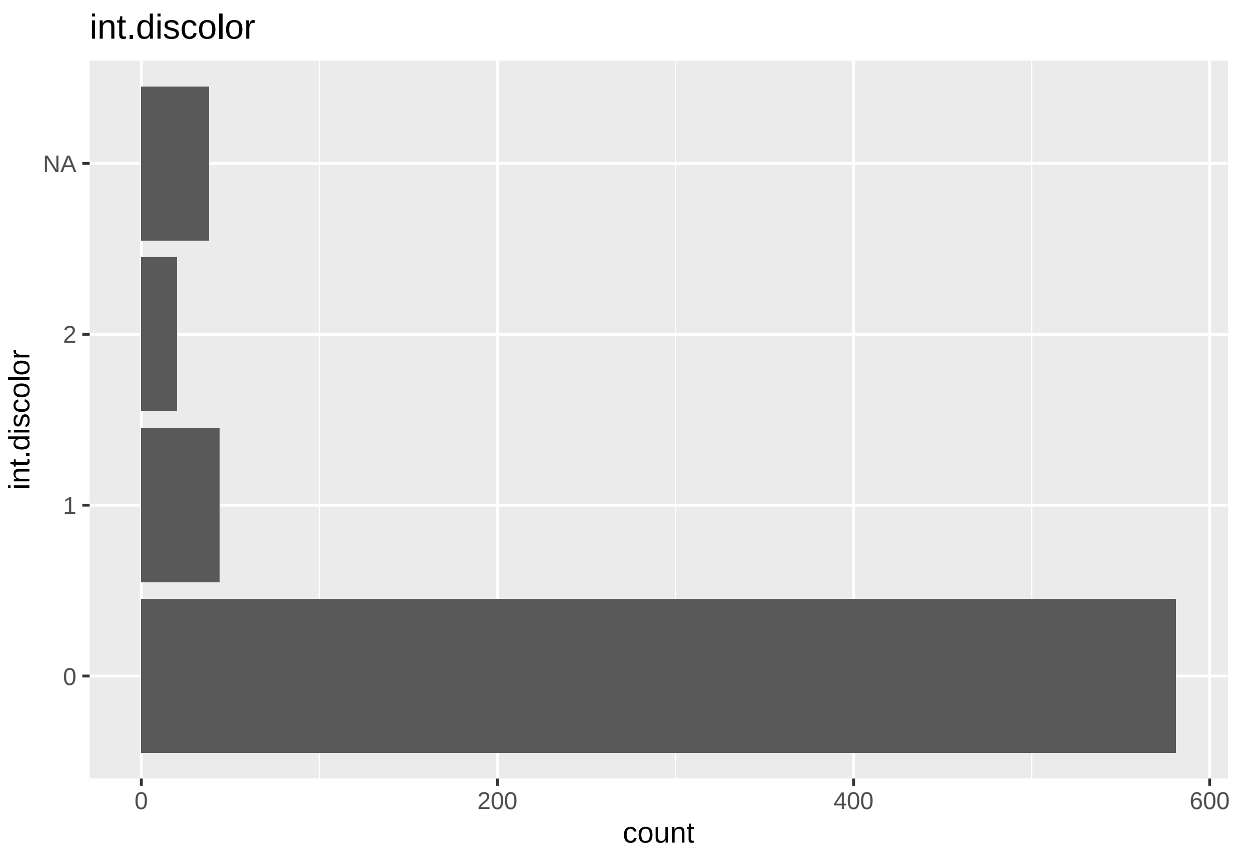
```
##
## [[25]]
```



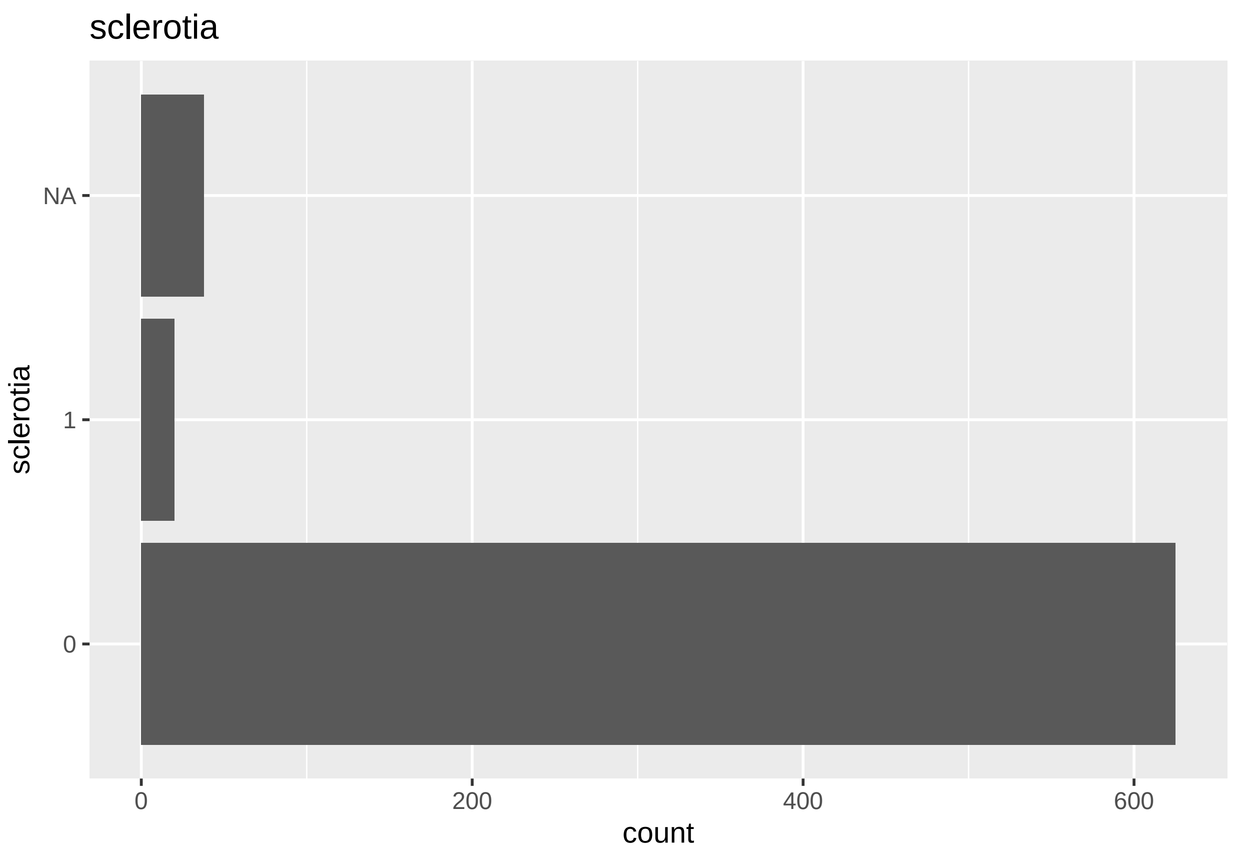
```
##
## [[26]]
```



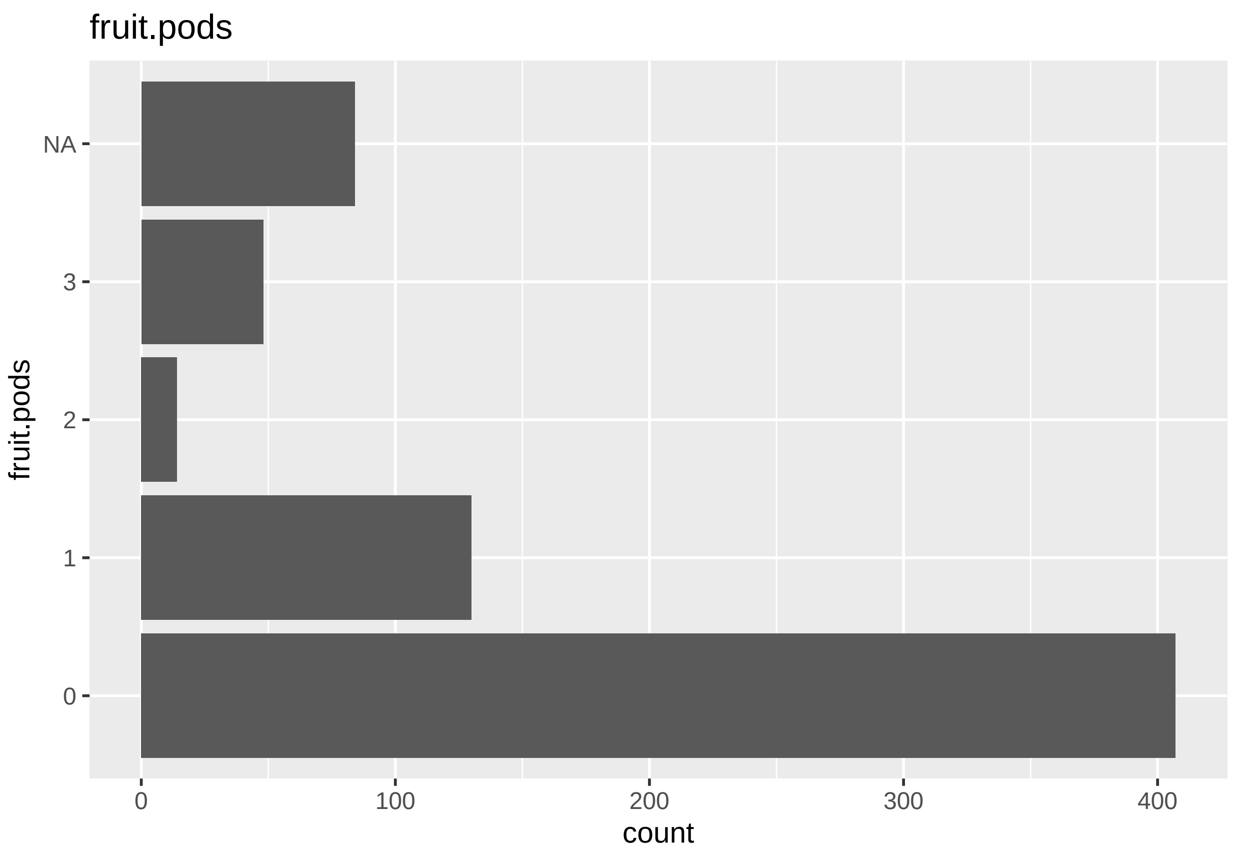
```
##
## [[27]]
```



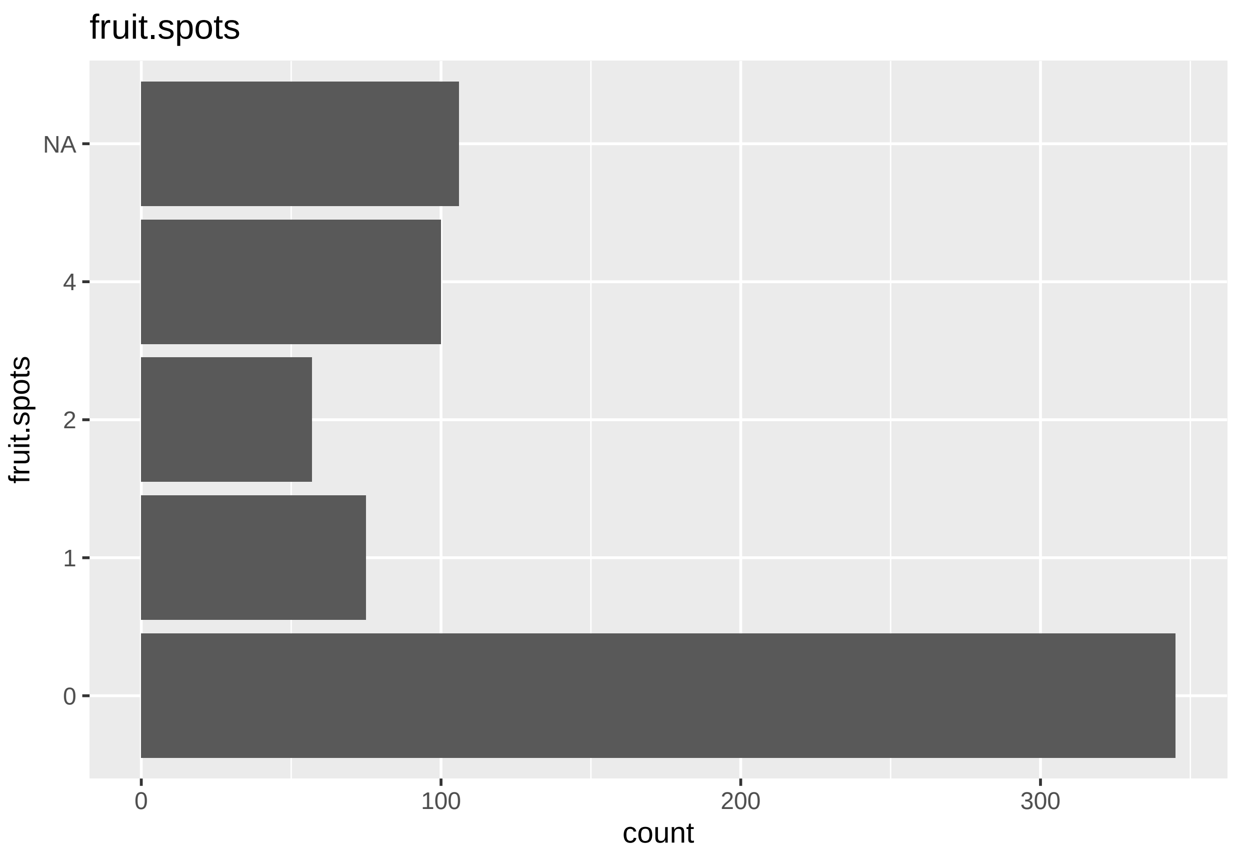
```
##
## [[28]]
```



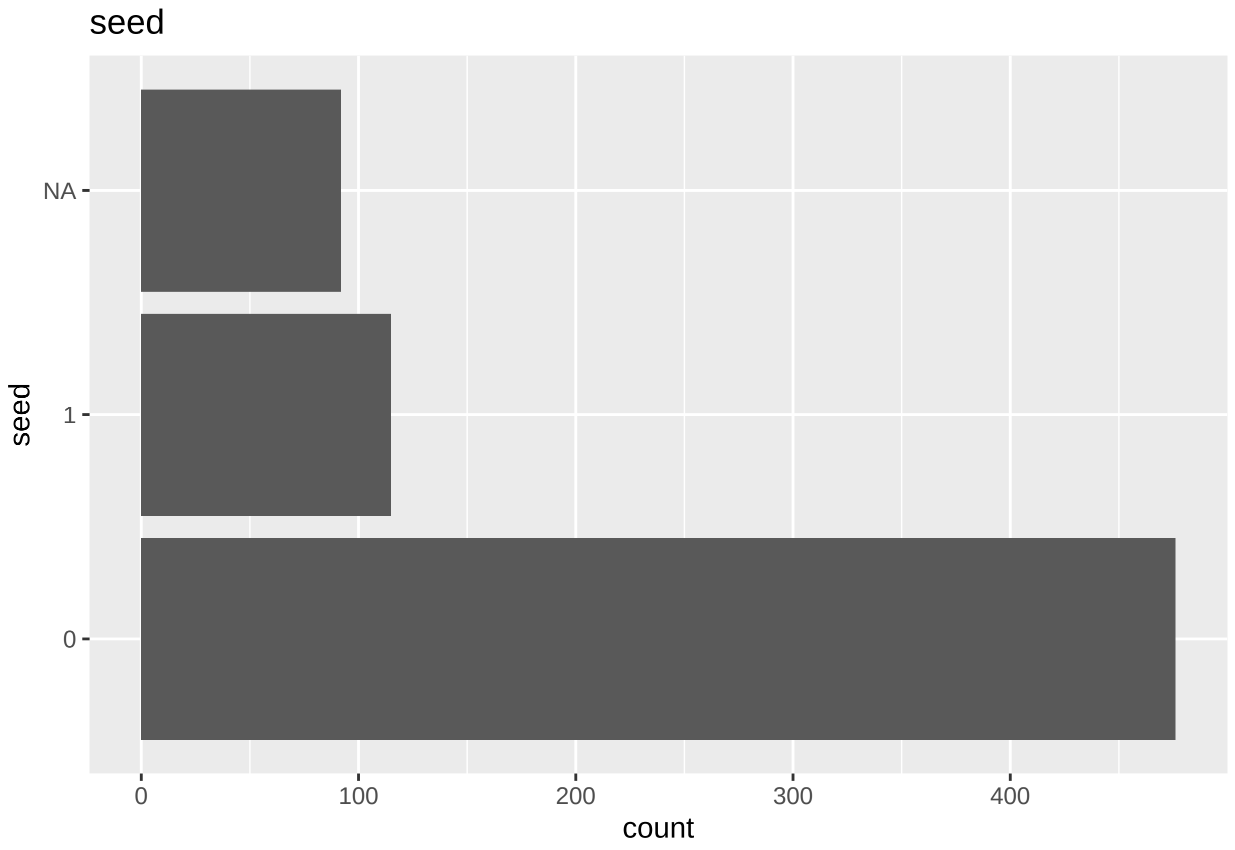
```
##
## [[29]]
```



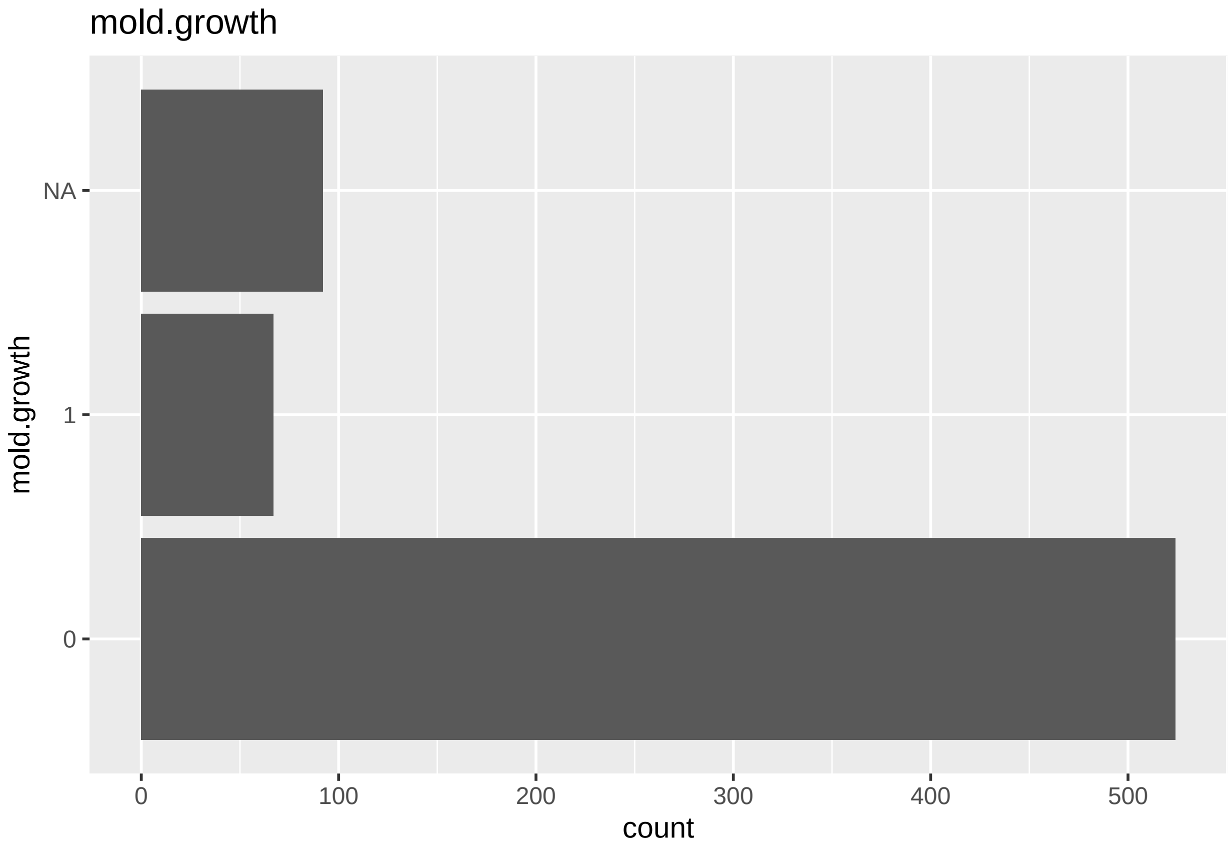
```
##
## [[30]]
```



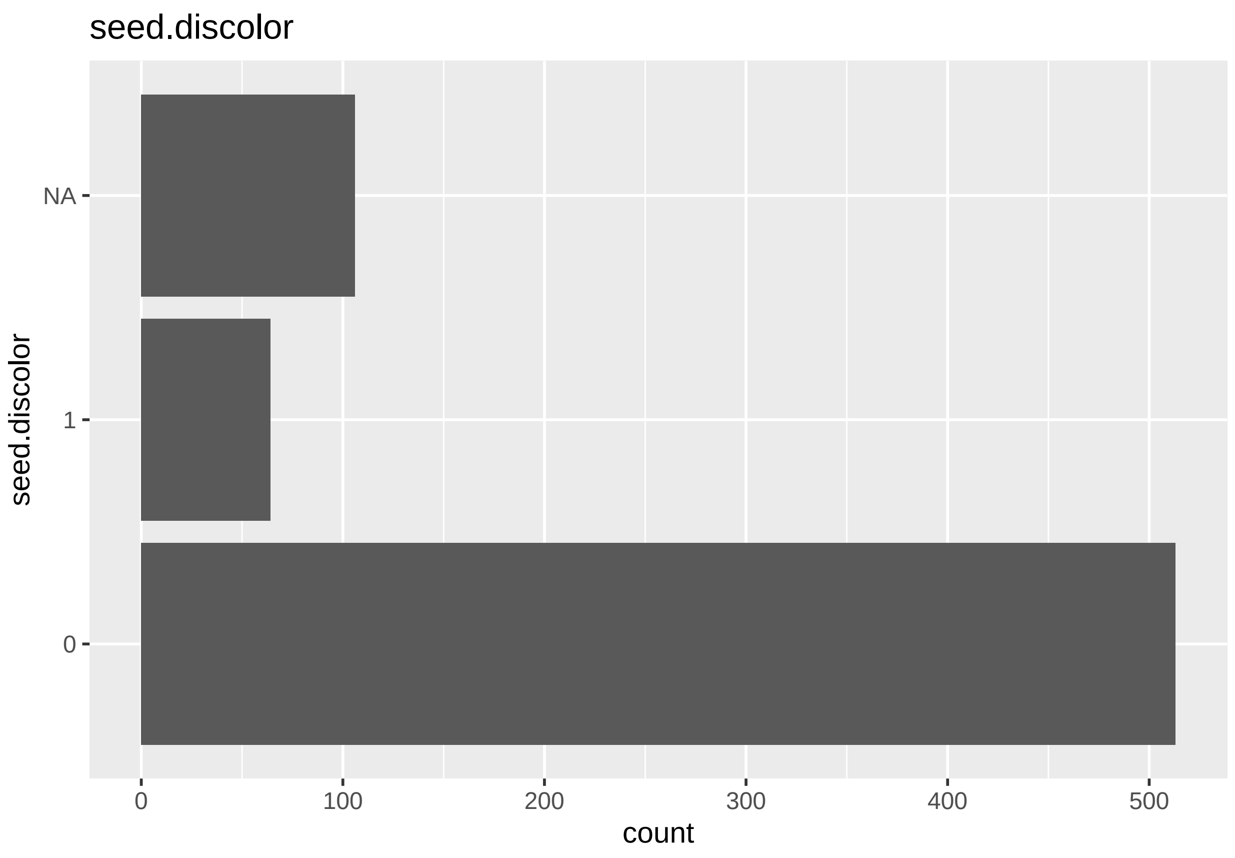
```
##
## [[31]]
```



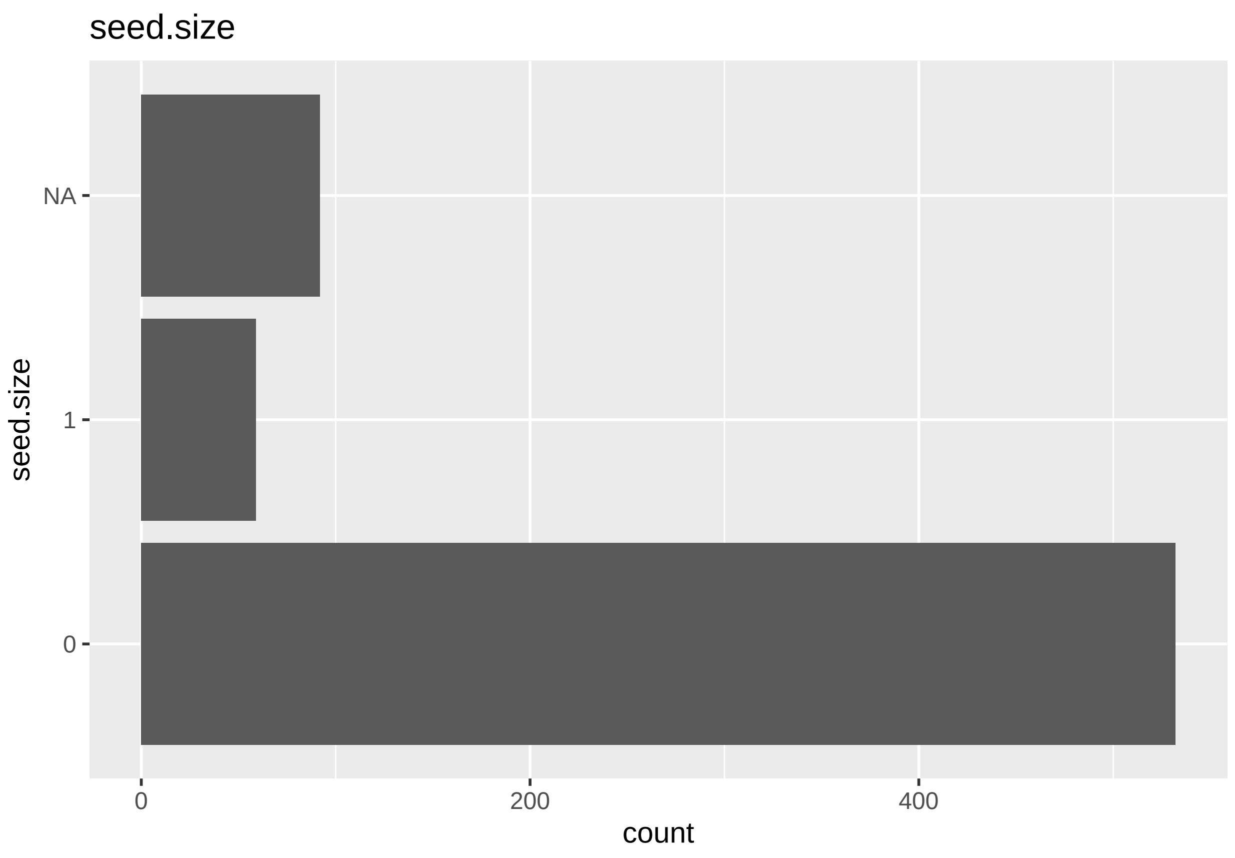
```
##
## [[32]]
```



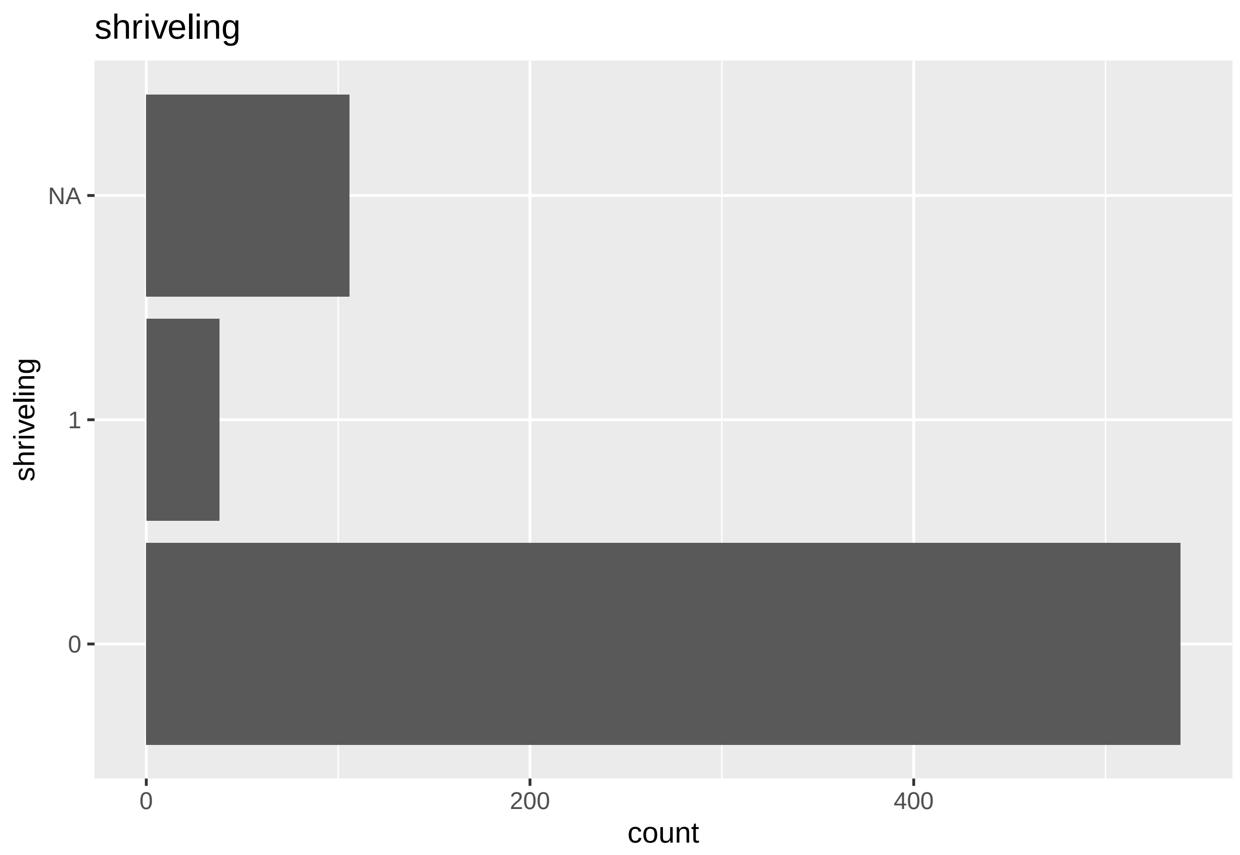
```
##
## [[33]]
```



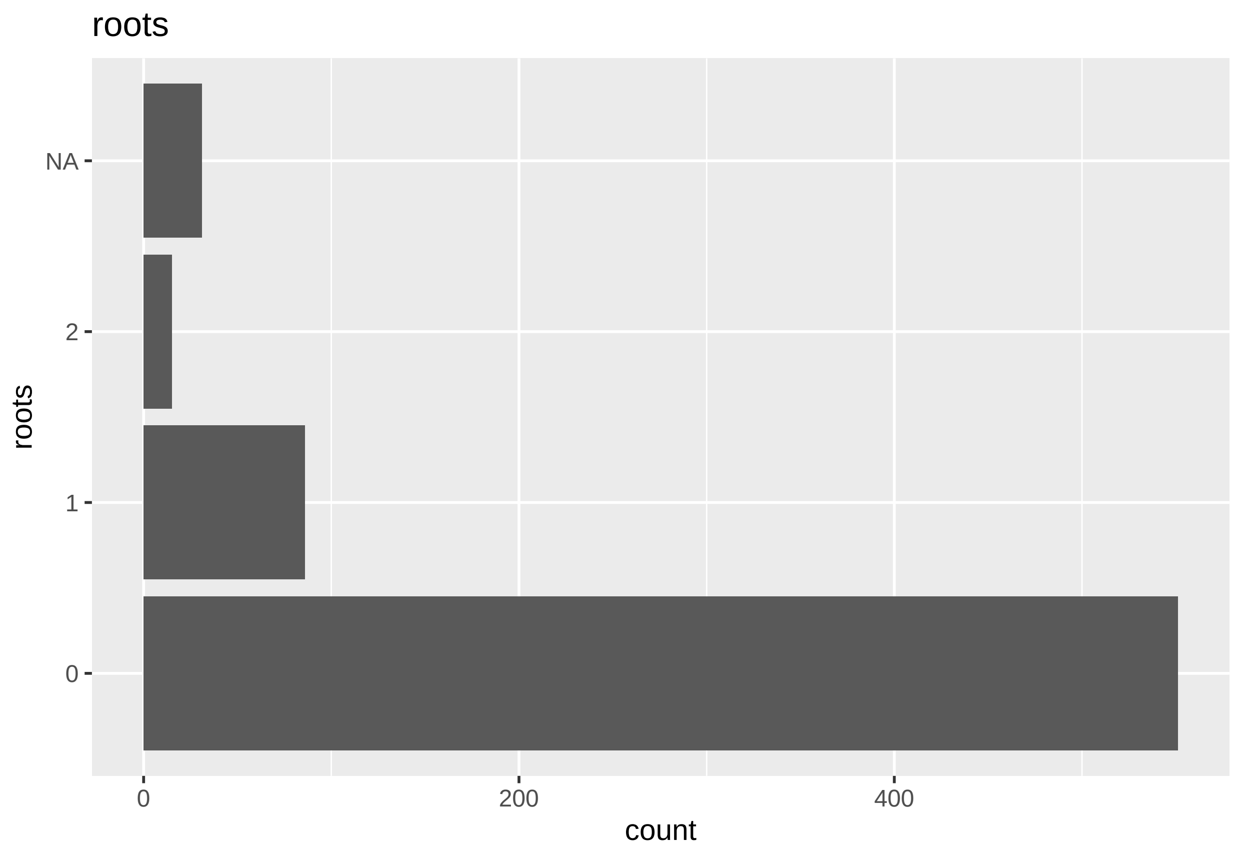
```
##
## [[34]]
```



```
##
## [[35]]
```



```
##
## [[36]]
```

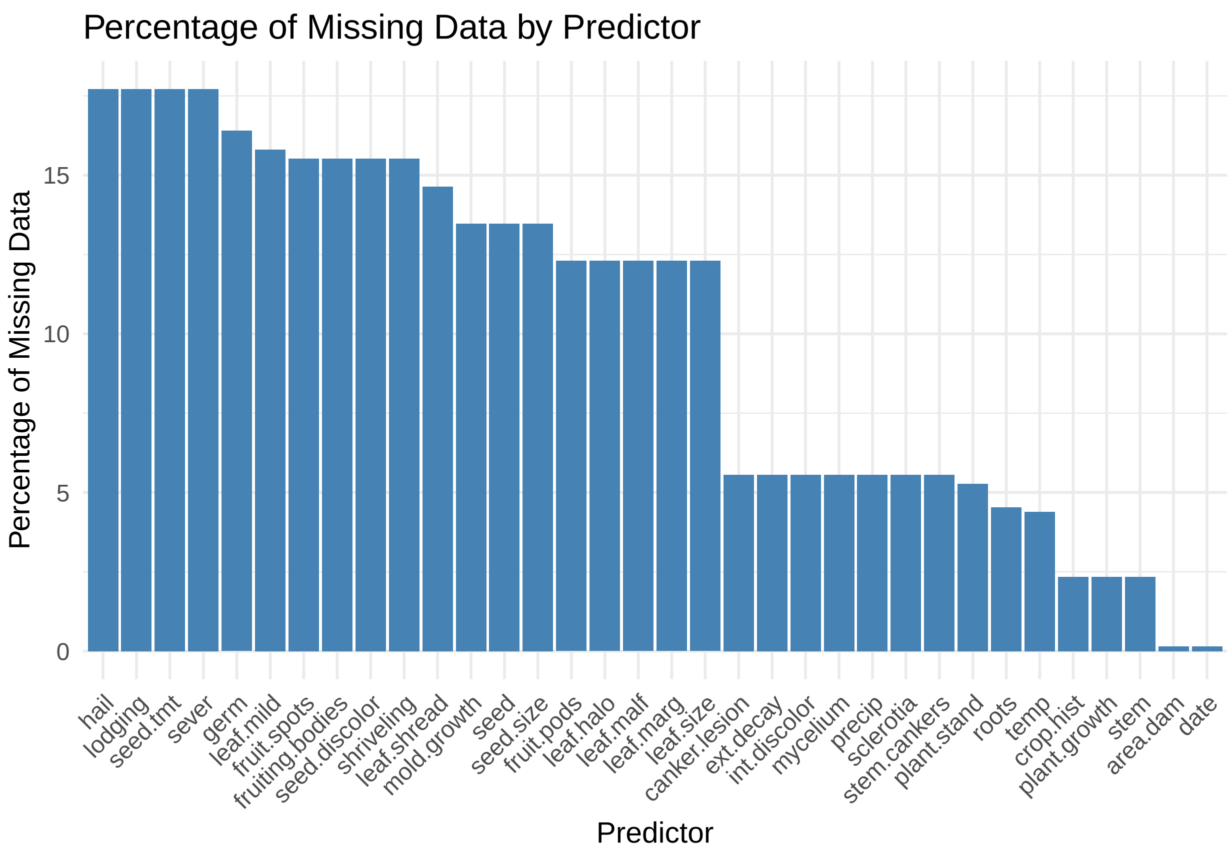


(b) Roughly 18% of the data are missing. Are there particular predictors that are more likely to be missing? Is the pattern of missing data related to the classes?

```r
# Calculate the percentage of missing data for each predictor
missing_data <- colSums(is.na(Soybean)) / nrow(Soybean) * 100

# Convert to a data frame for easier manipulation
missing_predictors <- data.frame(Predictor = names(missing_data),
                                 MissingPercent = as.numeric(missing_data))

# Check which predictors have missing data
print(missing_predictors)
```

```
##          Predictor MissingPercent
## 1            Class      0.0000000
## 2             date      0.1464129
## 3      plant.stand      5.2708638
## 4           precip      5.5636896
## 5             temp      4.3923865
## 6             hail     17.7159590
## 7        crop.hist      2.3426061
## 8         area.dam      0.1464129
## 9            sever     17.7159590
## 10        seed.tmt     17.7159590
## 11            germ     16.3982430
## 12    plant.growth      2.3426061
## 13          leaves      0.0000000
## 14       leaf.halo     12.2986823
## 15       leaf.marg     12.2986823
## 16       leaf.size     12.2986823
## 17     leaf.shread     14.6412884
## 18       leaf.malf     12.2986823
## 19       leaf.mild     15.8125915
## 20            stem      2.3426061
## 21         lodging     17.7159590
## 22    stem.cankers      5.5636896
## 23   canker.lesion      5.5636896
## 24 fruiting.bodies     15.5197657
## 25       ext.decay      5.5636896
## 26        mycelium      5.5636896
## 27    int.discolor      5.5636896
## 28       sclerotia      5.5636896
## 29      fruit.pods     12.2986823
## 30     fruit.spots     15.5197657
## 31            seed     13.4699854
## 32     mold.growth     13.4699854
## 33   seed.discolor     15.5197657
## 34       seed.size     13.4699854
## 35      shriveling     15.5197657
## 36           roots      4.5387994
```

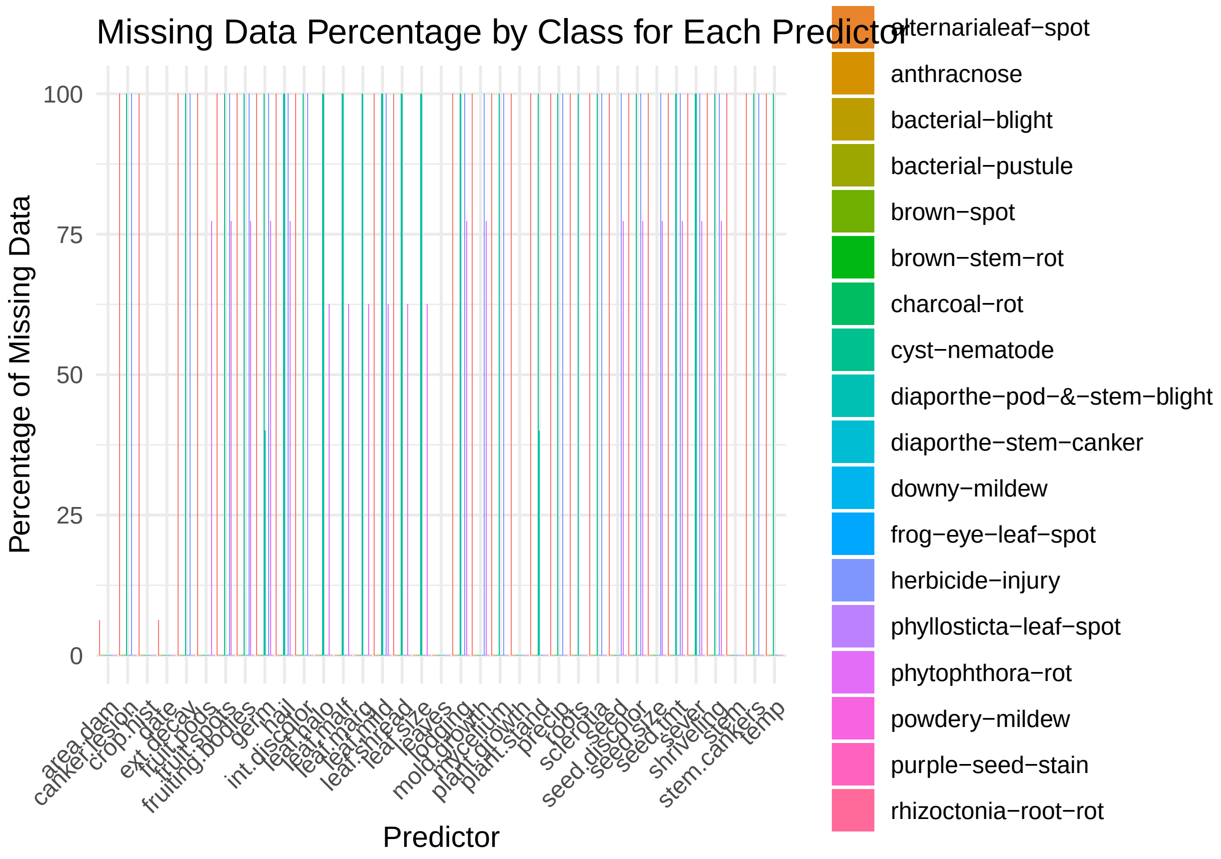
```r
# Filter for predictors with more than 0% missing data for visualization
missing_predictors <- missing_predictors %>%
  filter(MissingPercent > 0)

# Plotting missing predictors
ggplot(missing_predictors, aes(x = reorder(Predictor, -MissingPercent), y = MissingPercent)) +
  geom_bar(stat = "identity", fill = "steelblue") +
  labs(title = "Percentage of Missing Data by Predictor",
       x = "Predictor",
       y = "Percentage of Missing Data") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



```r
# Calculate missing data by class for each predictor
missing_by_class <- Soybean %>%
  group_by(Class) %>%
  summarize(across(everything(), ~ sum(is.na(.)) / n() * 100))  # Percentage of missing data by class

# View the resulting data frame
print(missing_by_class)
```

```
## # A tibble: 19 x 36
##    Class   date plant.stand precip  temp  hail crop.hist area.dam sever seed.tmt
##    <fct>  <dbl>       <dbl>  <dbl> <dbl> <dbl>     <dbl>    <dbl> <dbl>    <dbl>
##  1 2-4-d~  6.25         100    100   100  100        100     6.25  100      100
##  2 alter~  0              0      0     0    0          0     0       0        0
##  3 anthr~  0              0      0     0    0          0     0       0        0
##  4 bacte~  0              0      0     0    0          0     0       0        0
##  5 bacte~  0              0      0     0    0          0     0       0        0
##  6 brown~  0              0      0     0    0          0     0       0        0
##  7 brown~  0              0      0     0    0          0     0       0        0
##  8 charc~  0              0      0     0    0          0     0       0        0
##  9 cyst-~  0            100    100   100  100          0     0     100      100
## 10 diapo~  0             40      0     0  100          0     0     100      100
## 11 diapo~  0              0      0     0    0          0     0       0        0
## 12 downy~  0              0      0     0    0          0     0       0        0
## 13 frog-~  0              0      0     0    0          0     0       0        0
## 14 herbi~  0              0    100     0  100          0     0     100      100
## 15 phyll~  0              0      0     0    0          0     0       0        0
## 16 phyto~  0              0      0     0   77.3        0     0      77.3     77.3
## 17 powde~  0              0      0     0   0           0     0      0        0
## 18 purpl~  0              0      0     0   0           0     0      0        0
## 19 rhizo~  0              0      0     0   0           0     0      0        0
## # i 26 more variables: germ <dbl>, plant.growth <dbl>, leaves <dbl>,
## #   leaf.halo <dbl>, leaf.marg <dbl>, leaf.size <dbl>, leaf.shread <dbl>,
## #   leaf.malf <dbl>, leaf.mild <dbl>, stem <dbl>, lodging <dbl>,
## #   stem.cankers <dbl>, canker.lesion <dbl>, fruiting.bodies <dbl>,
## #   ext.decay <dbl>, mycelium <dbl>, int.discolor <dbl>, sclerotia <dbl>,
## #   fruit.pods <dbl>, fruit.spots <dbl>, seed <dbl>, mold.growth <dbl>,
## #   seed.discolor <dbl>, seed.size <dbl>, shriveling <dbl>, roots <dbl>
```

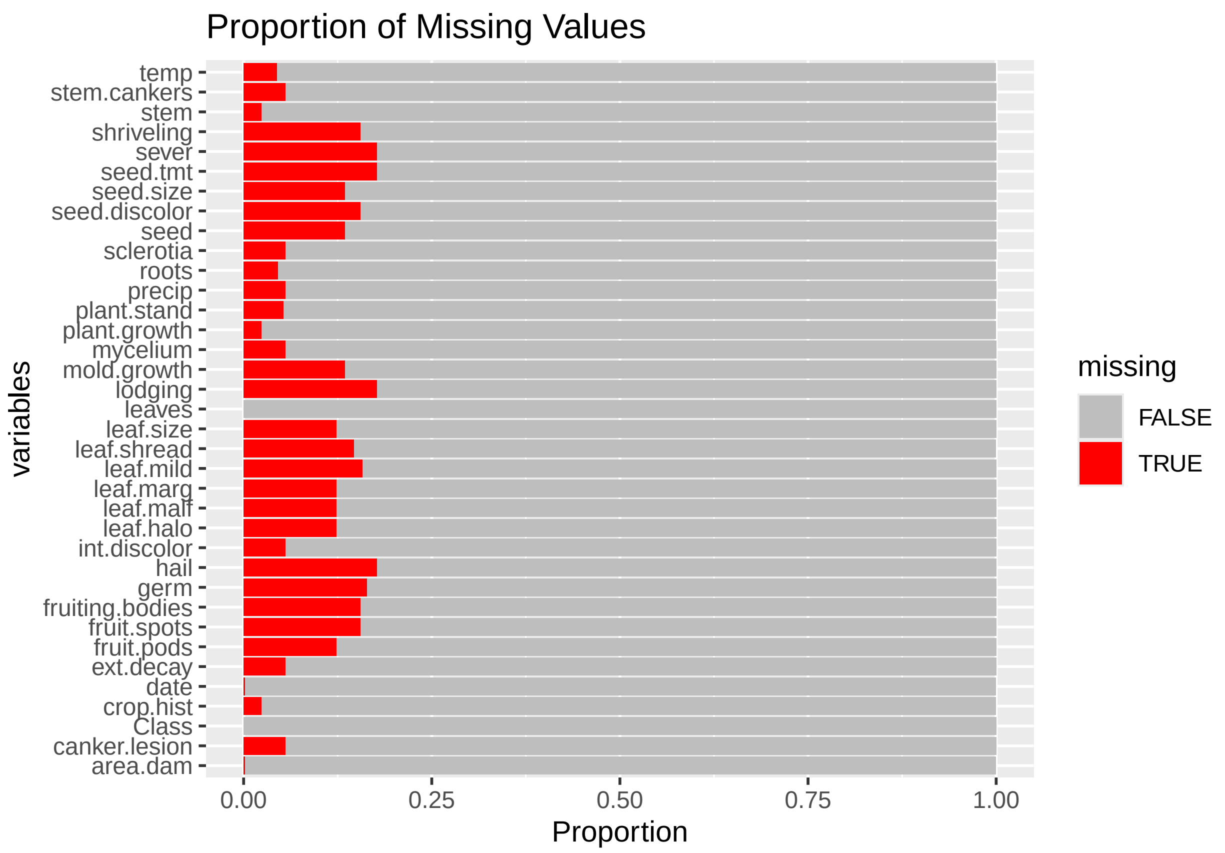
```r
# Reshape data for visualization
missing_by_class_long <- missing_by_class %>%
  pivot_longer(cols = -Class, names_to = "Predictor", values_to = "MissingPercent")

# Plotting missing data by class
ggplot(missing_by_class_long, aes(x = Predictor, y = MissingPercent, fill = Class)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(title = "Missing Data Percentage by Class for Each Predictor",
       x = "Predictor",
       y = "Percentage of Missing Data") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



ANSWER: Yes, a look at the bar plot shows that rhizoctonia-root, purple sees, powdery mildew, phytophthora rot andphylosticta leaf spot are the predictors that are more likely to be missing . Is the pattern of missing data related to the classes? YES.

```r
Soybean %>%
  summarise_all(list(~is.na(.)))%>%
  pivot_longer(everything(), names_to = "variables", values_to="missing") %>%
  count(variables, missing) %>%
  ggplot(aes(y = variables, x=n, fill = missing))+
  geom_col(position = "fill") +
  labs(title = "Proportion of Missing Values",
       x = "Proportion") +
  scale_fill_manual(values=c("grey","red"))
```

```
## Warning: Returning more (or less) than 1 row per `summarise()` group was deprecated in
## dplyr 1.1.0.
## i Please use `reframe()` instead.
## i When switching from `summarise()` to `reframe()`, remember that `reframe()`
##   always returns an ungrouped data frame and adjust accordingly.
## i The deprecated feature was likely used in the dplyr package.
##   Please report the issue at <https://github.com/tidyverse/dplyr/issues>.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



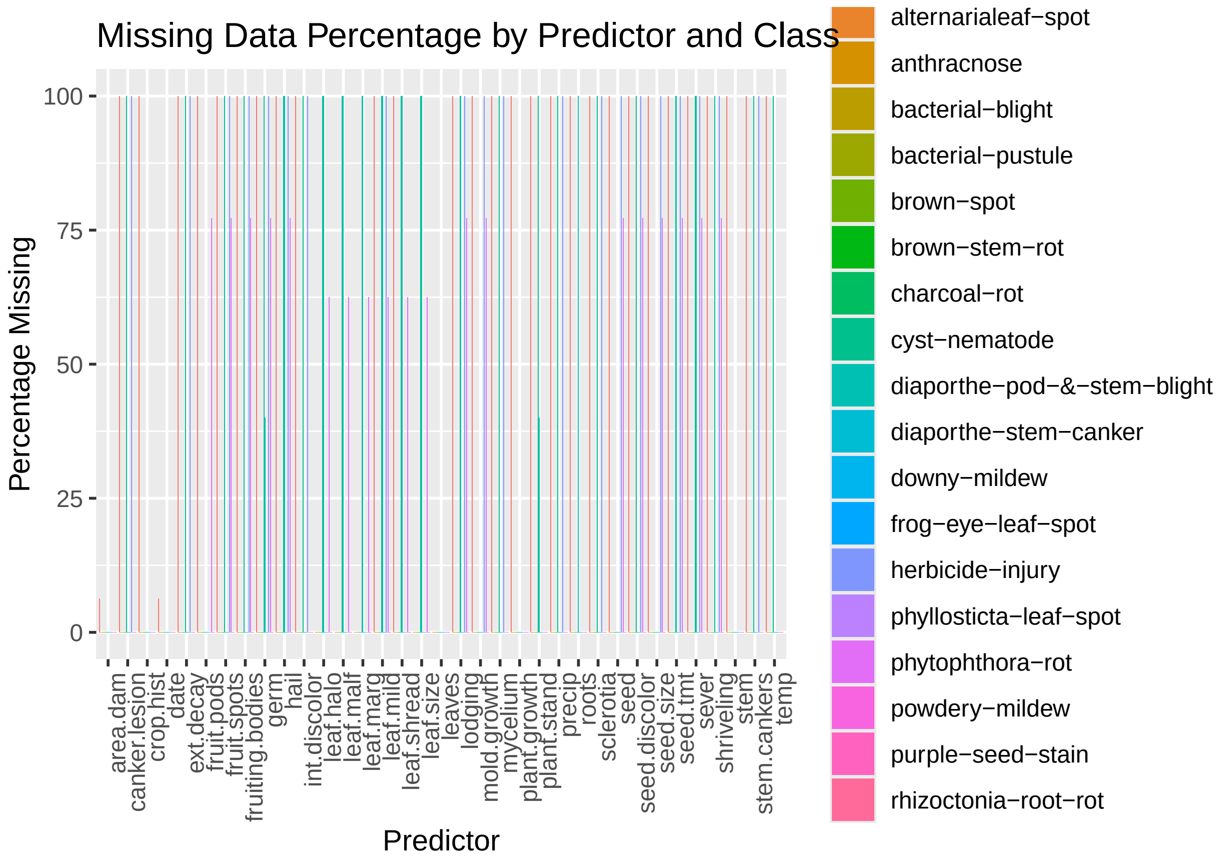
```r
# Step 2: Investigate if missing data is related to the classes
# Assume that the first column is the class label
class_column <- Soybean[, 1]
missing_by_class <- Soybean %>%
  mutate(Class = class_column) %>%
  group_by(Class) %>%
  summarise(across(everything(), ~ sum(is.na(.)) / n() * 100))

# Display the percentage of missing data per class
print("Missing data percentage by class:")
```

```
## [1] "Missing data percentage by class:"
```

```
print(missing_by_class)
```

```
## # A tibble: 19 x 36
##    Class   date plant.stand precip  temp  hail crop.hist area.dam sever seed.tmt
##    <fct>  <dbl>       <dbl>  <dbl> <dbl> <dbl>     <dbl>    <dbl> <dbl>    <dbl>
##  1 2-4-d~  6.25         100    100   100 100         100     6.25 100      100
##  2 alter~  0              0      0     0   0           0     0      0        0
##  3 anthr~  0              0      0     0   0           0     0      0        0
##  4 bacte~  0              0      0     0   0           0     0      0        0
##  5 bacte~  0              0      0     0   0           0     0      0        0
##  6 brown~  0              0      0     0   0           0     0      0        0
##  7 brown~  0              0      0     0   0           0     0      0        0
##  8 charc~  0              0      0     0   0           0     0      0        0
##  9 cyst-~  0            100    100   100 100           0     0    100      100
## 10 diapo~  0             40      0     0 100           0     0    100      100
## 11 diapo~  0              0      0     0   0           0     0      0        0
## 12 downy~  0              0      0     0   0           0     0      0        0
## 13 frog-~  0              0      0     0   0           0     0      0        0
## 14 herbi~  0              0    100     0 100           0     0    100      100
## 15 phyll~  0              0      0     0   0           0     0      0        0
## 16 phyto~  0              0      0     0  77.3         0     0     77.3     77.3
## 17 powde~  0              0      0     0   0           0     0      0        0
## 18 purpl~  0              0      0     0   0           0     0      0        0
## 19 rhizo~  0              0      0     0   0           0     0      0        0
## # i 26 more variables: germ <dbl>, plant.growth <dbl>, leaves <dbl>,
## #   leaf.halo <dbl>, leaf.marg <dbl>, leaf.size <dbl>, leaf.shread <dbl>,
## #   leaf.malf <dbl>, leaf.mild <dbl>, stem <dbl>, lodging <dbl>,
## #   stem.cankers <dbl>, canker.lesion <dbl>, fruiting.bodies <dbl>,
## #   ext.decay <dbl>, mycelium <dbl>, int.discolor <dbl>, sclerotia <dbl>,
## #   fruit.pods <dbl>, fruit.spots <dbl>, seed <dbl>, mold.growth <dbl>,
## #   seed.discolor <dbl>, seed.size <dbl>, shriveling <dbl>, roots <dbl>
```

```
# Step 5: Visualize the pattern of missing data by class
# Reshape data for plotting
missing_by_class_long <- missing_by_class %>%
  gather(key = "Predictor", value = "MissingPercent", -Class)

# Plot missing data by class and predictor
ggplot(missing_by_class_long, aes(x = Predictor, y = MissingPercent, fill = Class)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(title = "Missing Data Percentage by Predictor and Class", x = "Predictor", y = "Percentage Missi
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```



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

```
# Step 1: Calculate the percentage of missing data for each predictor
missing_data <- Soybean %>%
  summarise(across(everything(), ~ sum(is.na(.)) / n() * 100))

# Display the missing data percentage for each predictor
print("Percentage of missing data per predictor:")
```

```
## [1] "Percentage of missing data per predictor:"
```

```
print(missing_data)
```

```
##   Class      date plant.stand  precip     temp     hail crop.hist  area.dam
## 1     0 0.1464129    5.270864 5.56369 4.392387 17.71596  2.342606 0.1464129
##      sever seed.tmt     germ plant.growth leaves leaf.halo leaf.marg leaf.size
## 1 17.71596 17.71596 16.39824     2.342606      0  12.29868  12.29868  12.29868
##   leaf.shread leaf.malf leaf.mild     stem  lodging stem.cankers canker.lesion
## 1    14.64129  12.29868  15.81259 2.342606 17.71596      5.56369       5.56369
##   fruiting.bodies ext.decay mycelium int.discolor sclerotia fruit.pods
## 1        15.51977   5.56369  5.56369      5.56369   5.56369   12.29868
##   fruit.spots     seed mold.growth seed.discolor seed.size shriveling    roots
## 1    15.51977 13.46999    13.46999      15.51977  13.46999   15.51977 4.538799
```