OR 568

Assignment 1 – Basics of R, Descriptive Statistics and Data Preprocessing

Part I: (30pts) Review of R Basics.

Q1: Use R as a calculator to compute the following values. After you do so, cut and paste your input and output from R to Word. Add numbering in Word to identify each part of each problem. (Do this for every problem from now on.)
(a) 27*(38-17)
(b) $\boldsymbol{ln}(\mathbf{147})$
(c) $\sqrt{\mathbf{436/12}}$

Answer:

(a)
```
> 27*(38-17)

[1] 567
```

(b)
```
> log(147)
[1] 4.990433
```

(c)
```
> sqrt(436/12)
[1] 6.027714
```

Q2: Create the following vectors in R.
a = (5, 10, 15, 20, ..., 160)
b = (87, 86, 85, ..., 56)
Use vector arithmetic to multiply these vectors and call the result d. Select subsets of d to id entify the following.
(a) What are the 19th, 20th, and 21st elements of d?
(b) What are all of the elements of d which are less than 2000?
(c) How many elements of d are greater than 6000?

Answer:

```
> a<-seq(5,160,by=5)
> a
 [1]   5  10  15  20  25  30  35  40  45  50  55  60  65  70  75  80  85  90  95 100 105 110 115 1
20 125 130 135 140
[29] 145 150 155 160

b<-c(87:56)
> b
```

```
 [1] 87 86 85 84 83 82 81 80 79 78 77 76 75 74 73 72 71 70 69 68 67 66 65 64 63 62 61 60
59 58 57 56

> d<-a*b
> d
 [1]  435  860 1275 1680 2075 2460 2835 3200 3555 3900 4235 4560 4875 5180 5475 57
60 6035 6300 6555 6800 7035 7260
[23] 7475 7680 7875 8060 8235 8400 8555 8700 8835 8960
```

(a)
```
> d[19:21]
[1] 6555 6800 7035
```

(b)
```
> d[d<2000]
[1]  435  860 1275 1680
```

(c)
```
> length(d[d>6000])
[1] 16
```

Q3: Using d from problem Q2, use R to compute the following statistics of d:
(a) sum
(b) mean and median
(c) standard deviation

Answers:

(a)
```
> sum(d)
[1] 175120
```

(b)
```
> mean(d)
[1] 5472.5
> median(d)
[1] 5897.5
```

(c)
```
> sd(d)
[1] 2608.563
```

Q4: Read the dataset named "cars" and check the first several rows of the dataset.
```
> data(cars)
> head(cars)
```
(a) Plot a histogram of distance using the hist function
(b) Generate a boxplot of speed.
(c) Use the plot(,) function (e.g. plot(variableX, variableY) to create a scatterplot of dist agai
nst speed.

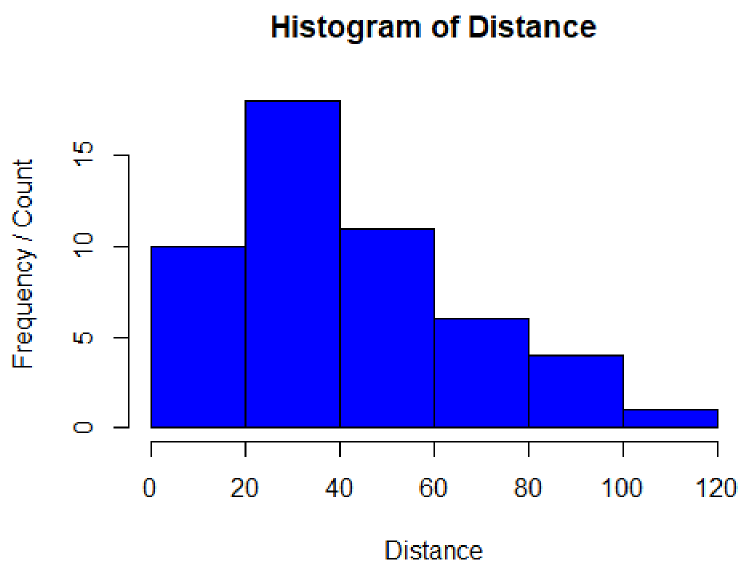
[Note: You can also create the graphs in parts (a) to (c) using ggplot2 package].

Answers:

```
> data(cars)
> head(cars)
  speed dist
1     4    2
2     4   10
3     7    4
4     7   22
5     8   16
6     9   10
```

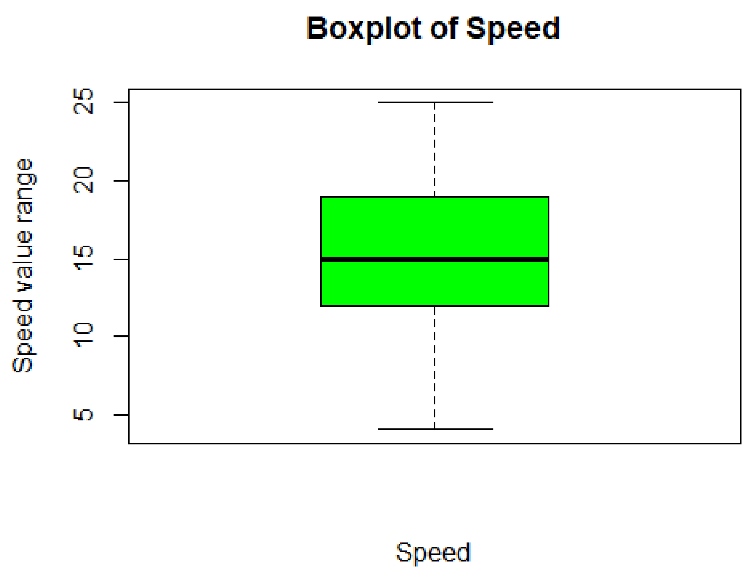
(a)Plot a histogram of distance using the hist function

```
> hist(cars$dist,main = "Histogram of Distance", xlab = "Distance", ylab = "Frequency / Cou
nt", col ="blue")
```



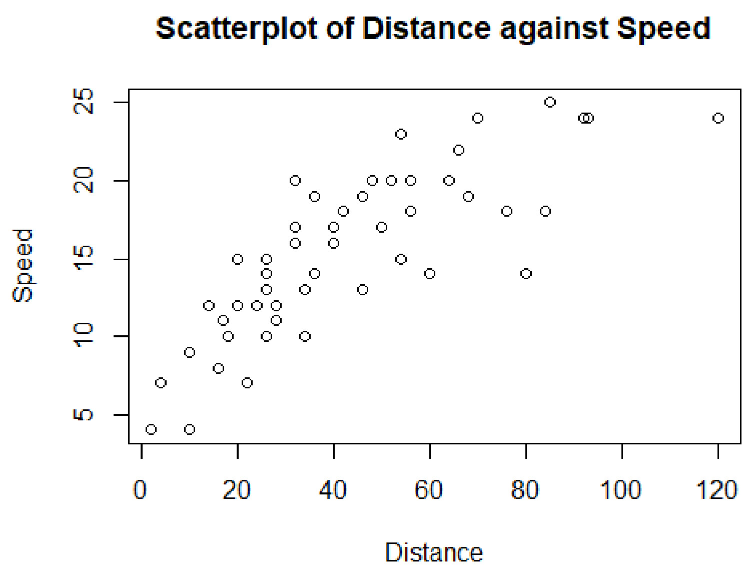
(b) Generate a boxplot of speed.

```
> boxplot(cars$speed, col = "green", main="Boxplot of Speed", xlab = "Speed", ylab = "Speed
value range")
```



(c) Use the plot(,) function (e.g. plot(variableX, variableY) to create a scatterplot of dist against speed.

```
> plot(x = cars$dist, y = cars$speed, main = "Scatterplot of Distance against Speed", xlab = "
Distance", ylab = "Speed")
```



Part II: (35pts) Data Preprocessing (Exercise 3.1 of APM Book).

(a) Using visualizations, explore the predictor variables to understand their distributions as well as the relationships between predictors. Provide the pairwise scatter plots and investigate the correlation matrix.

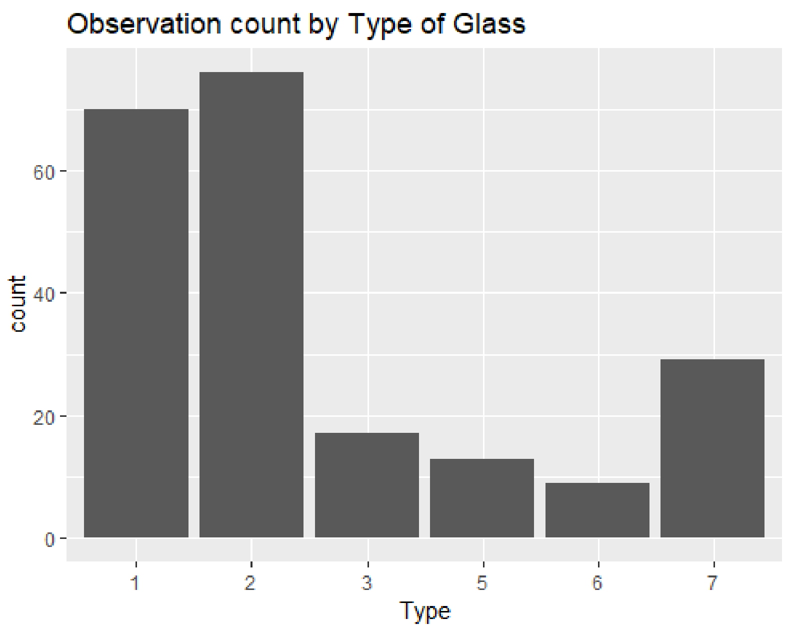
```
> library(mlbench)
> data(Glass)
> str(Glass)
'data.frame':	214 obs. of  10 variables:
 $ RI  : num  1.52 1.52 1.52 1.52 1.52 ...
 $ Na  : num  13.6 13.9 13.5 13.2 13.3 ...
 $ Mg  : num  4.49 3.6 3.55 3.69 3.62 3.61 3.6 3.61 3.58 3.6 ...
 $ Al  : num  1.1 1.36 1.54 1.29 1.24 1.62 1.14 1.05 1.37 1.36 ...
 $ 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 ...
```

To explore the Predictor Variables, we can plot Histogram, Density Plots and Boxplots. Structure of the Glass Dataset above can let us know about the classes of each dataset.

I have showed histogram of the target variable for each of the glass types.

Histogram of the target variable i.e. Type

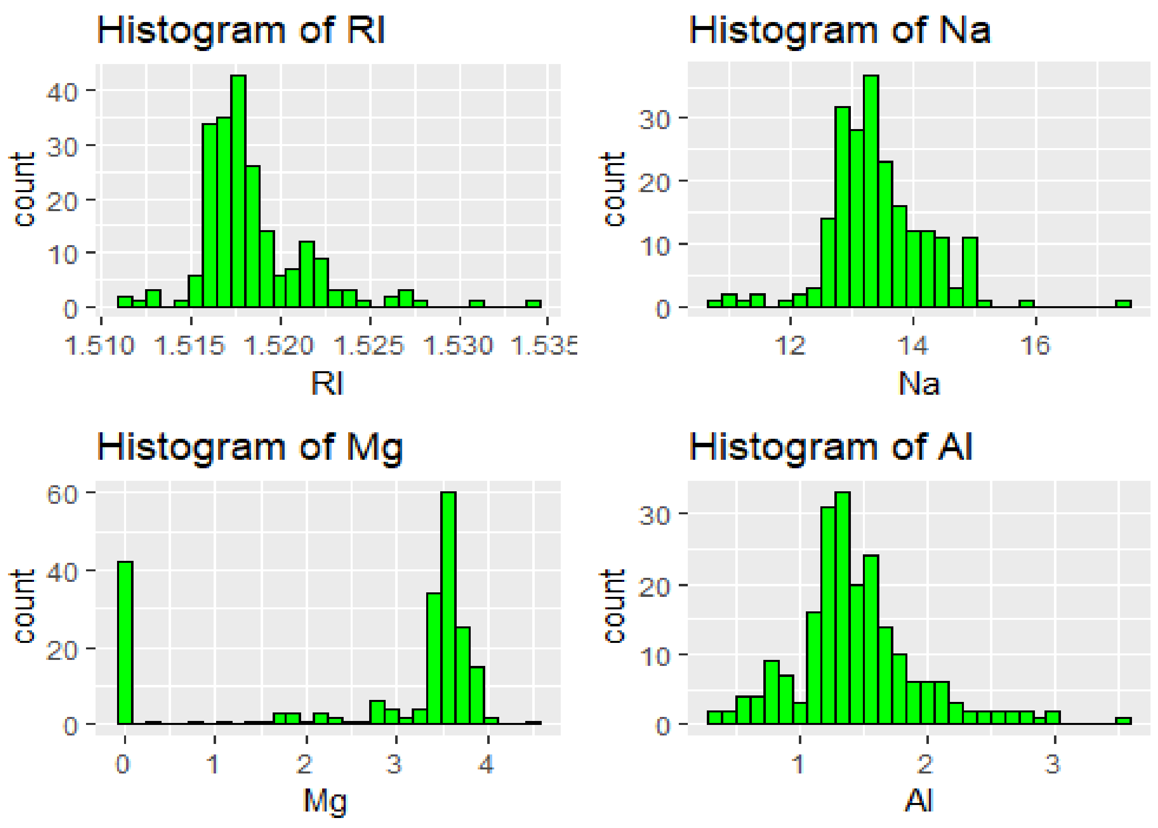
```
> ggplot(Glass,aes(x=Type))+geom_bar()+ggtitle("Observation count by Type of Glass")
```

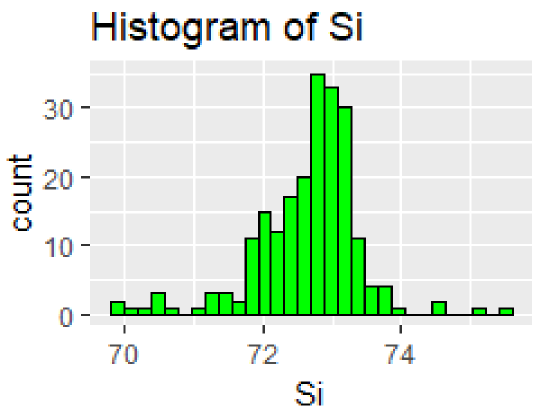


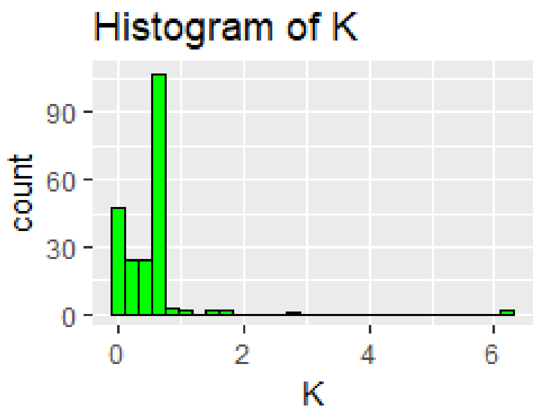
The above command tells me that in my Glass dataset most of the glass types are of type 1 or of type 2.

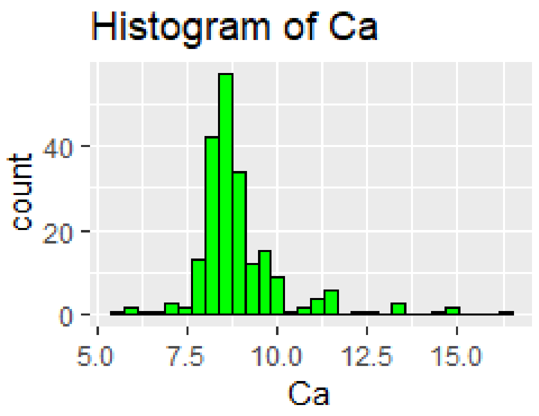
Histogram of all the predictor variables:

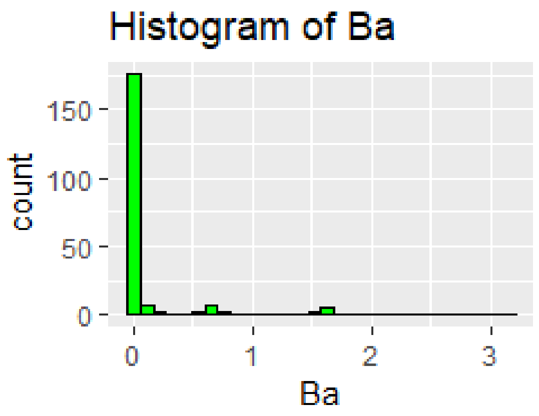
```
> GlassRI<-ggplot(Glass, aes(x=RI)) +
+   geom_histogram(color="black", fill="green")+ggtitle("Histogram of RI")
> GlassNa<-ggplot(Glass, aes(x=Na)) +
+   geom_histogram(color="black", fill="green")+ggtitle("Histogram of Na")
> GlassMg<-ggplot(Glass, aes(x=Mg)) +
+   geom_histogram(color="black", fill="green")+ggtitle("Histogram of Mg")
> GlassAl<-ggplot(Glass, aes(x=Al)) +
+   geom_histogram(color="black", fill="green")+ggtitle("Histogram of Al")

> grid.arrange(GlassRI, GlassNa,GlassMg,GlassAl, ncol=2)

> GlassSi<-ggplot(Glass, aes(x=Si)) +
+   geom_histogram(color="black", fill="green")+ggtitle("Histogram of Si")
> GlassK<-ggplot(Glass, aes(x=K)) +
+   geom_histogram(color="black", fill="green")+ggtitle("Histogram of K")
> GlassCa<-ggplot(Glass, aes(x=Ca)) +
+   geom_histogram(color="black", fill="green")+ggtitle("Histogram of Ca")
> GlassBa<-ggplot(Glass, aes(x=Ba)) +
+   geom_histogram(color="black", fill="green")+ggtitle("Histogram of Ba")
>
> grid.arrange(GlassSi, GlassK, GlassCa, GlassBa, ncol=2)

> GlassFe<-ggplot(Glass, aes(x=Fe)) +
+   geom_histogram(color="black", fill="green")+ggtitle("Histogram of Fe")
>
> GlassFe
```

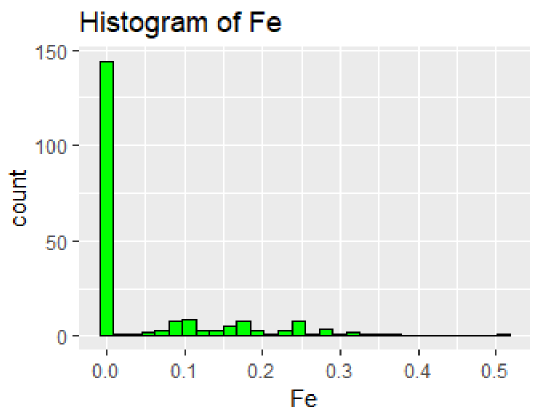








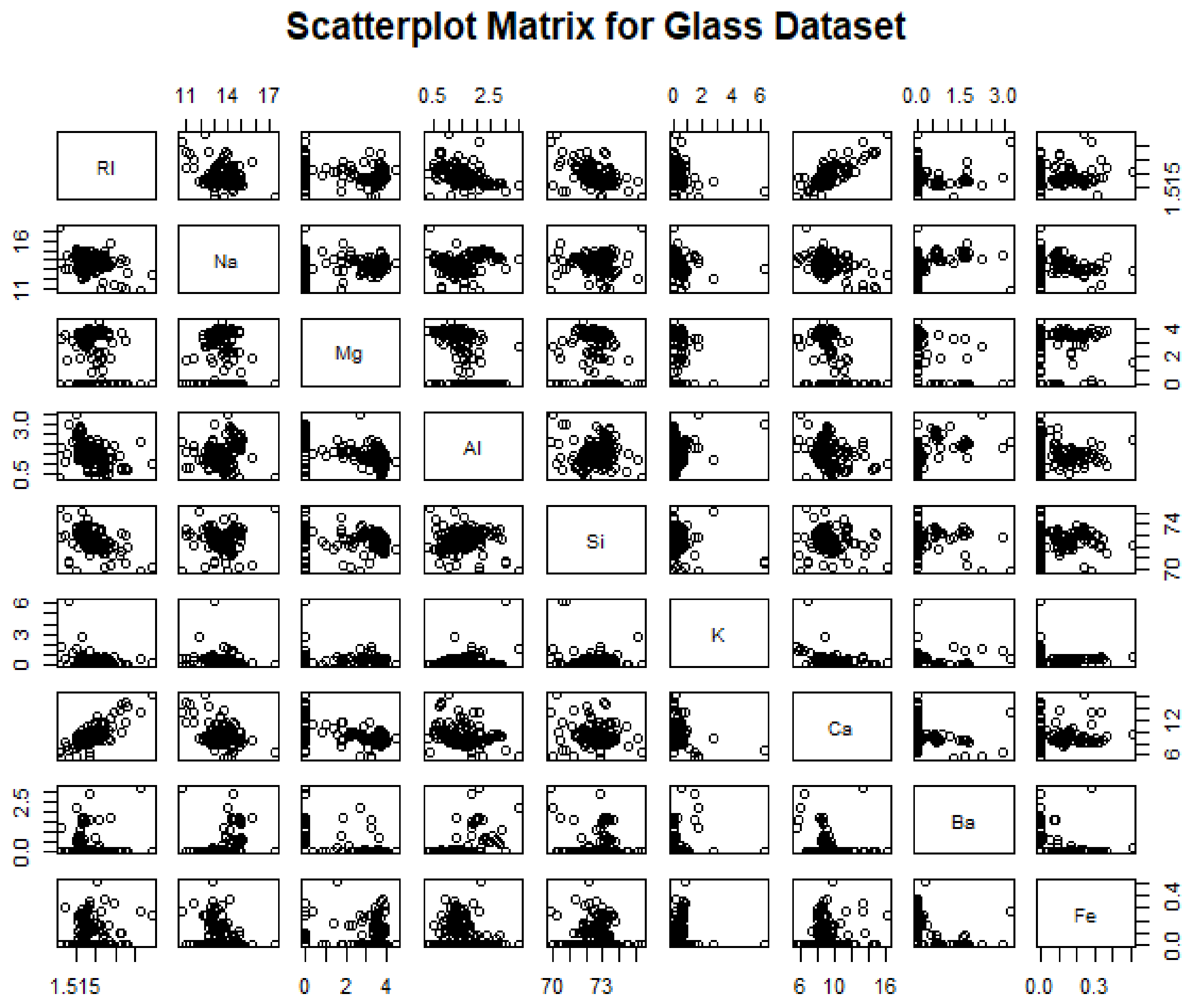




From the plots, we can see that Mg appears to have a Bimodal distribution. Also, it can be seen that few predictors such as Ba and Fe are rightly skewed.

Pairwise Scatterplots of all the predictors:

```
> pairs(Glass[,-10],main="Scatterplot Matrix for Glass Dataset")
```



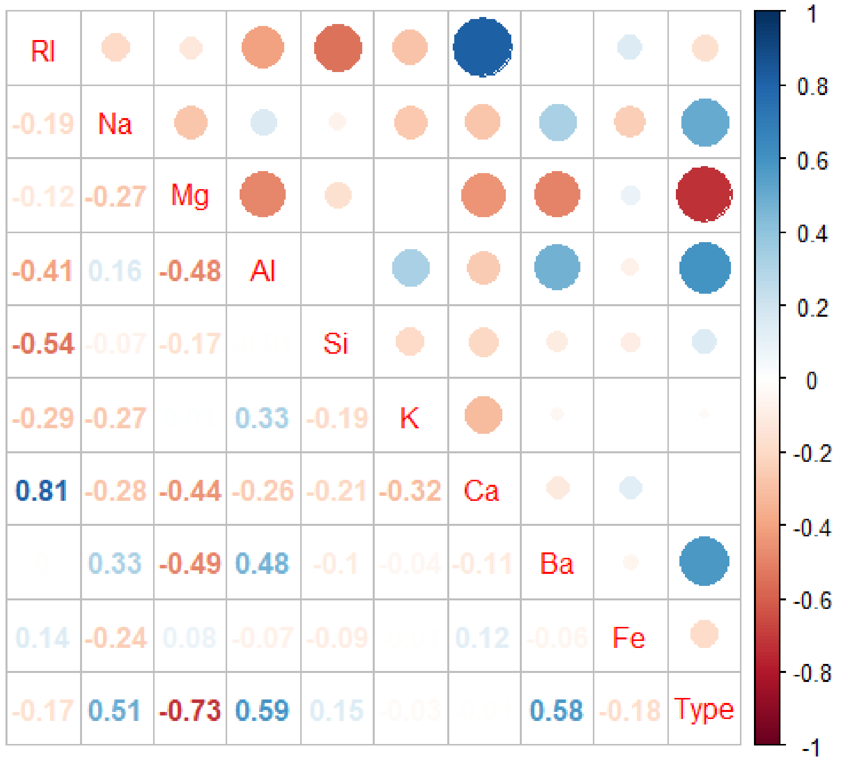
Scatterplot matrix above helps me to visualize the correlations between variables of Glass Dataset.

Correlation Value of each predictor w.r.t. Type:

```
> cor(Glass[,-10],as.numeric(Glass[,10]))
RI -0.168739357
Na  0.506424080
Mg -0.728159518
Al  0.591197598
Si  0.149690687
K  -0.025834560
Ca -0.008997841
Ba  0.577676375
Fe -0.183206747
```

Correlation Matrix:

```
> library(corrplot)
> Glasscorr <- Glass[,1:length(Glass)]
> Glasscorr<-data.matrix(Glasscorr)
> round(cor(Glasscorr),2)
        RI    Na    Mg    Al    Si     K    Ca    Ba    Fe  Type
RI    1.00 -0.19 -0.12 -0.41 -0.54 -0.29  0.81  0.00  0.14 -0.17
Na   -0.19  1.00 -0.27  0.16 -0.07 -0.27 -0.28  0.33 -0.24  0.51
Mg   -0.12 -0.27  1.00 -0.48 -0.17  0.01 -0.44 -0.49  0.08 -0.73
Al   -0.41  0.16 -0.48  1.00 -0.01  0.33 -0.26  0.48 -0.07  0.59
Si   -0.54 -0.07 -0.17 -0.01  1.00 -0.19 -0.21 -0.10 -0.09  0.15
K    -0.29 -0.27  0.01  0.33 -0.19  1.00 -0.32 -0.04 -0.01 -0.03
Ca    0.81 -0.28 -0.44 -0.26 -0.21 -0.32  1.00 -0.11  0.12 -0.01
Ba    0.00  0.33 -0.49  0.48 -0.10 -0.04 -0.11  1.00 -0.06  0.58
Fe    0.14 -0.24  0.08 -0.07 -0.09 -0.01  0.12 -0.06  1.00 -0.18
Type -0.17  0.51 -0.73  0.59  0.15 -0.03 -0.01  0.58 -0.18  1.00
> corrplot.mixed(cor(Glasscorr),lower = "number",upper = "circle")
```



It can be clearly seen that few of predictors are correlated such as Ca and RI with correlation value of 0.81 whereas most of predictors are uncorrelated.

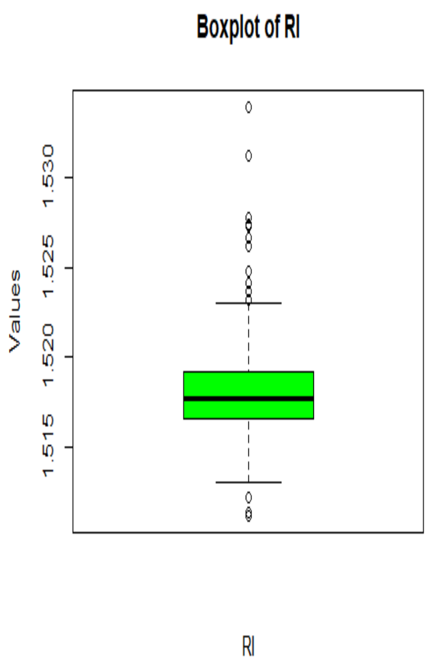
(b) Do there appear to be any outliers in the data? Are any predictors skewed?

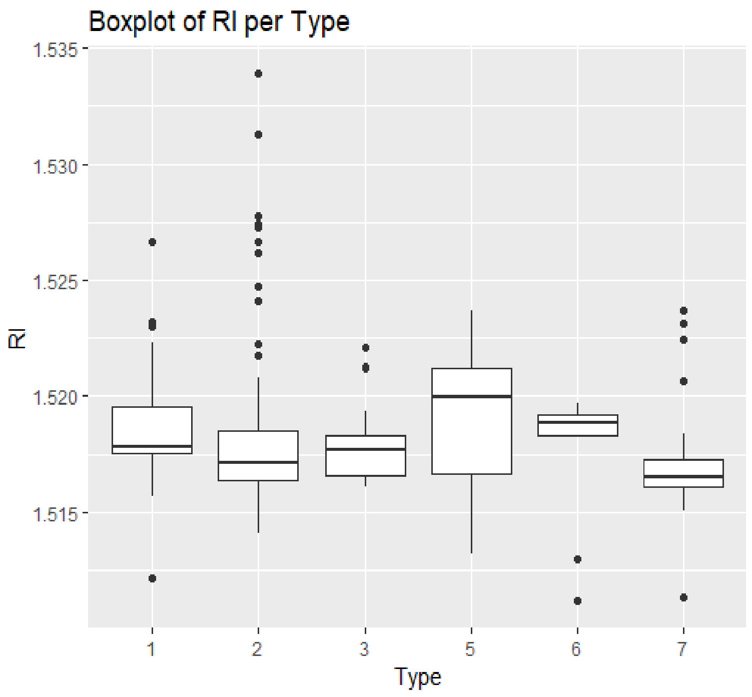
To check if there is any outlier in our data, we plot the Boxplot of each of the predictor variable and also of each variable w.r.t. Type:

Boxplot:

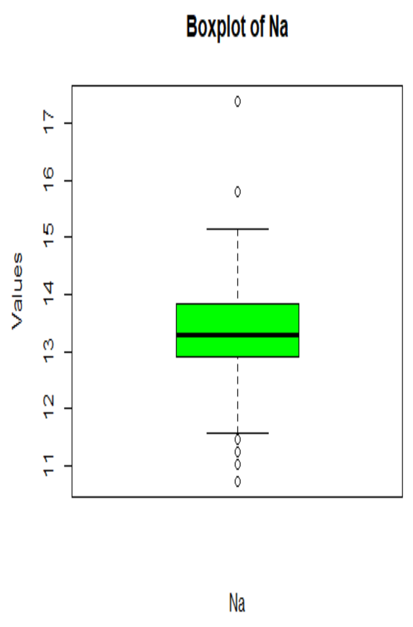
```
> boxplot(Glass$RI, col = "green", main="Boxplot of RI", xlab = "RI", ylab =
"Values")
```

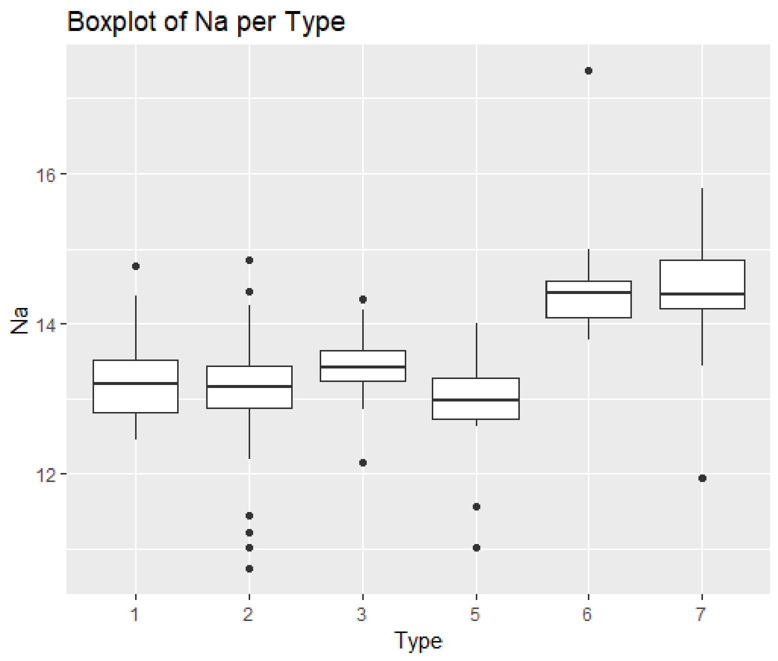
```
> plot.BoxRI <- ggplot(Glass, aes(x = Type, y = RI)) +geom_boxplot()
+ ggtitle("Boxplot of RI per Type")
> plot.BoxRI
```



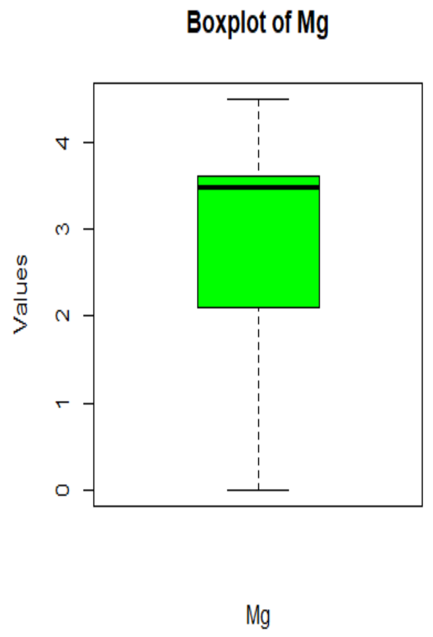


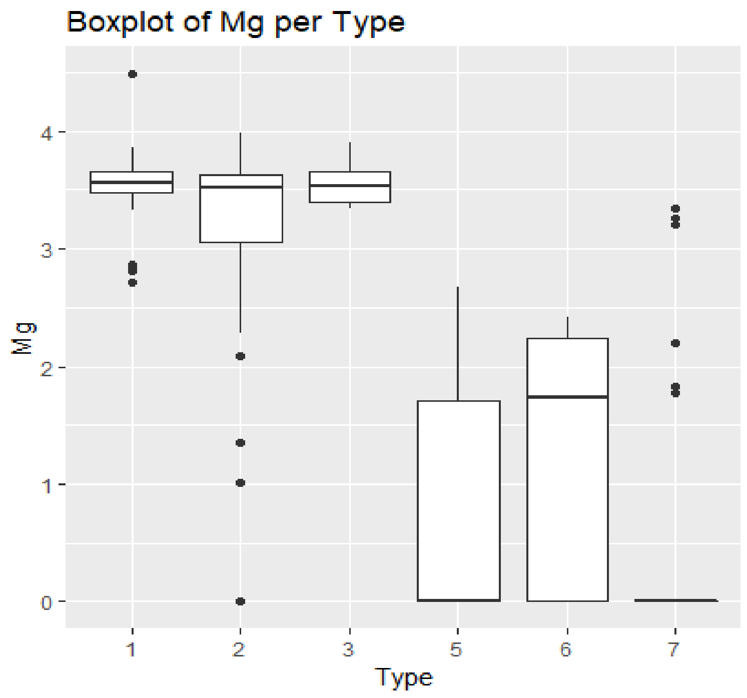
```
> boxplot(Glass$Na, col = "green", main="Boxplot of Na", xlab = "Na", ylab =
"Values")

> plot.BoxNa <- ggplot(Glass, aes(x = Type, y = Na)) +geom_boxplot()
+ ggtitle("Boxplot of Na per Type")
> plot.BoxNa
```





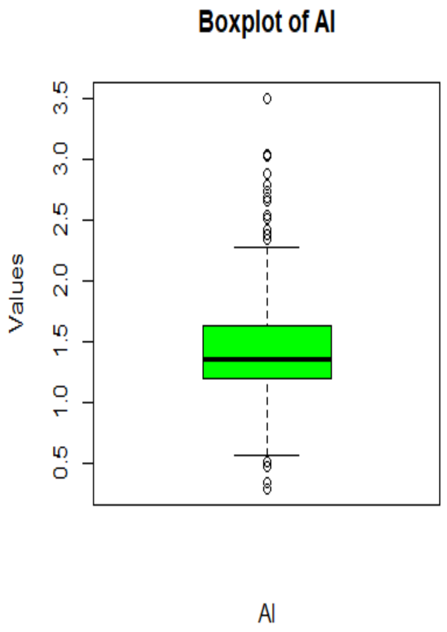
```
> boxplot(Glass$Mg, col = "green", main="Boxplot of Mg", xlab = "Mg", ylab =
"Values")

> plot.BoxMg <- ggplot(Glass, aes(x = Type, y = Mg)) +geom_boxplot()
+ ggtitle("Boxplot of Mg per Type")
> plot.BoxMg
```

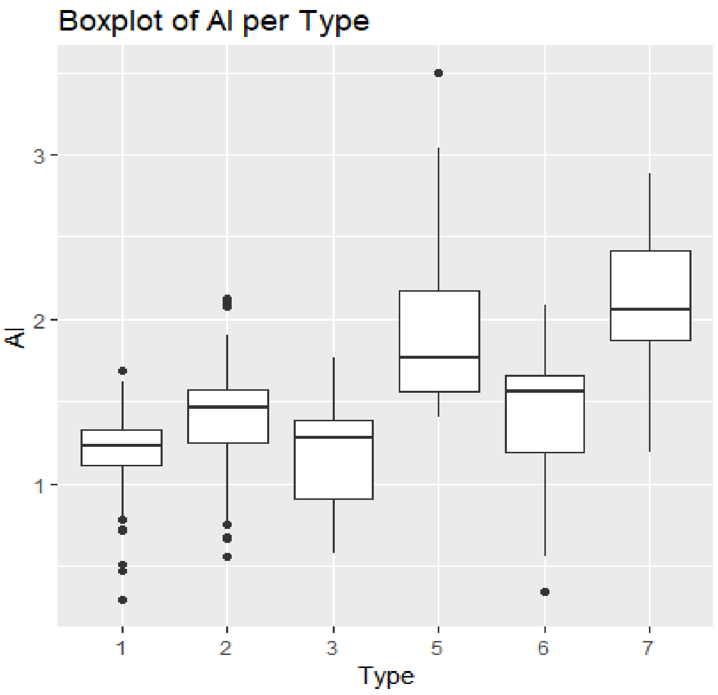




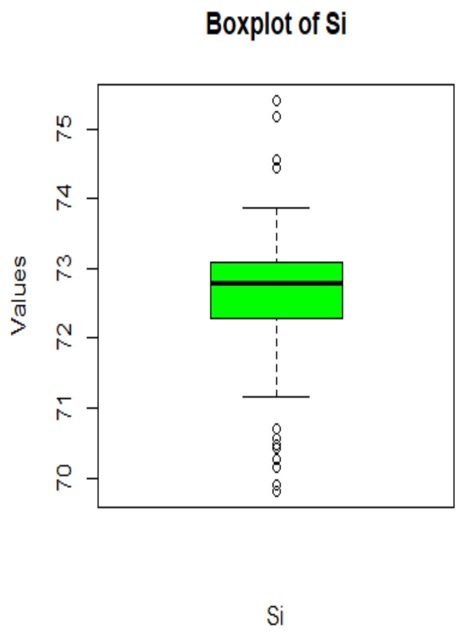
```
> boxplot(Glass$Al, col = "green", main="Boxplot of Al", xlab = "Al", ylab =
"Values")

> plot.BoxAl <- ggplot(Glass, aes(x = Type, y = Al)) +geom_boxplot()
```

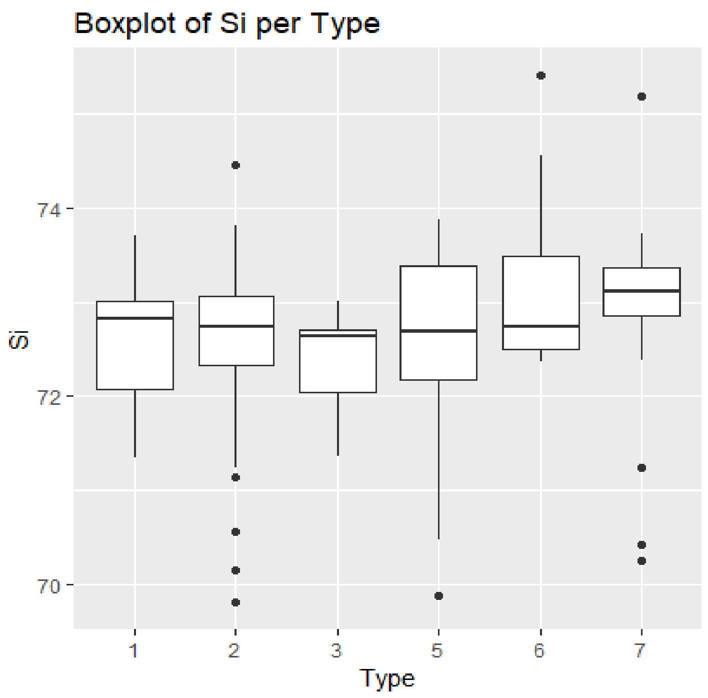
```
+ ggtitle("Boxplot of Al per Type")
> plot.BoxAl
```



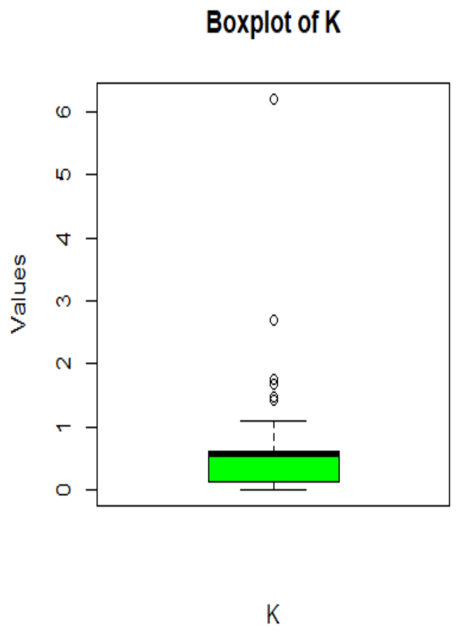


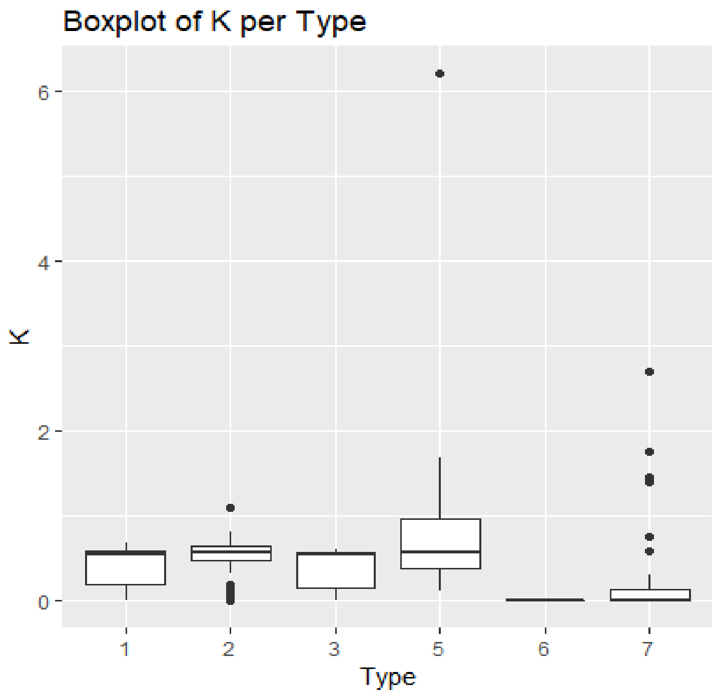
```
> boxplot(Glass$Si, col = "green", main="Boxplot of Si", xlab = "Si", ylab =
"Values")

> plot.BoxSi <- ggplot(Glass, aes(x = Type, y = Si)) +geom_boxplot()
+ ggtitle("Boxplot of Si per Type")
> plot.BoxSi
```





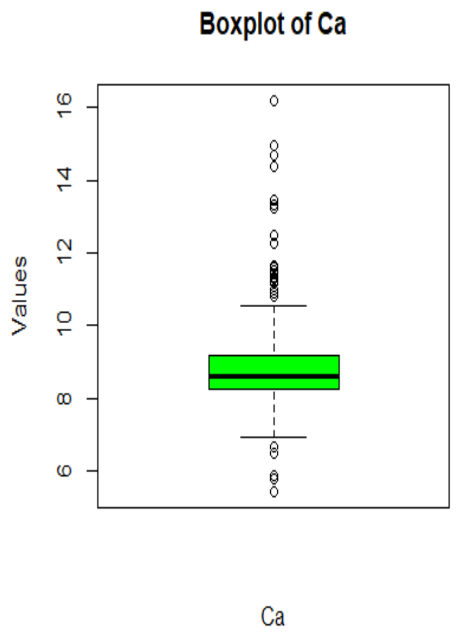
```
> boxplot(Glass$K, col = "green", main="Boxplot of K", xlab = "K", ylab = "Va
lues")

> plot.BoxK <- ggplot(Glass, aes(x = Type, y = K)) +geom_boxplot()
+ ggtitle("Boxplot of K per Type")
> plot.BoxK
```

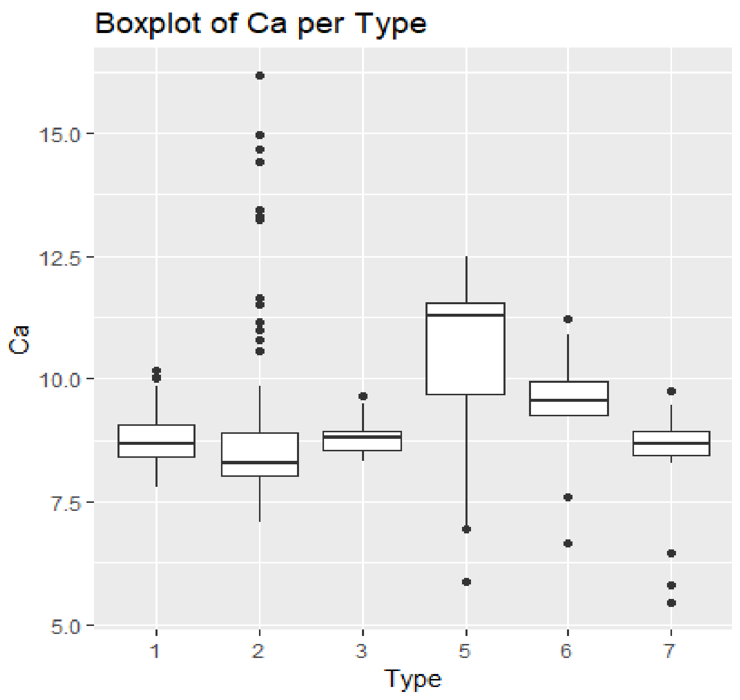




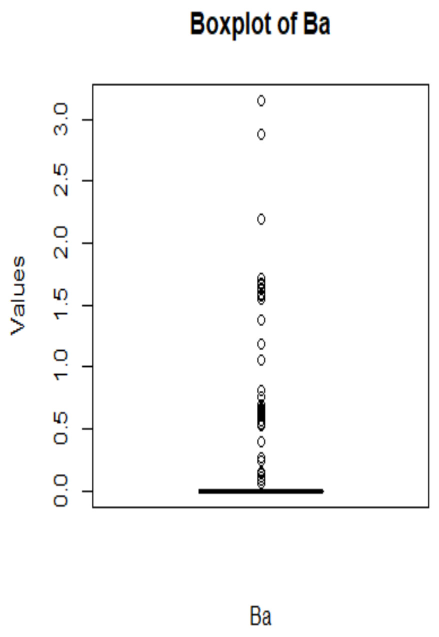
```
> boxplot(Glass$Ca, col = "green", main="Boxplot of Ca", xlab = "Ca", ylab =
"Values")
```

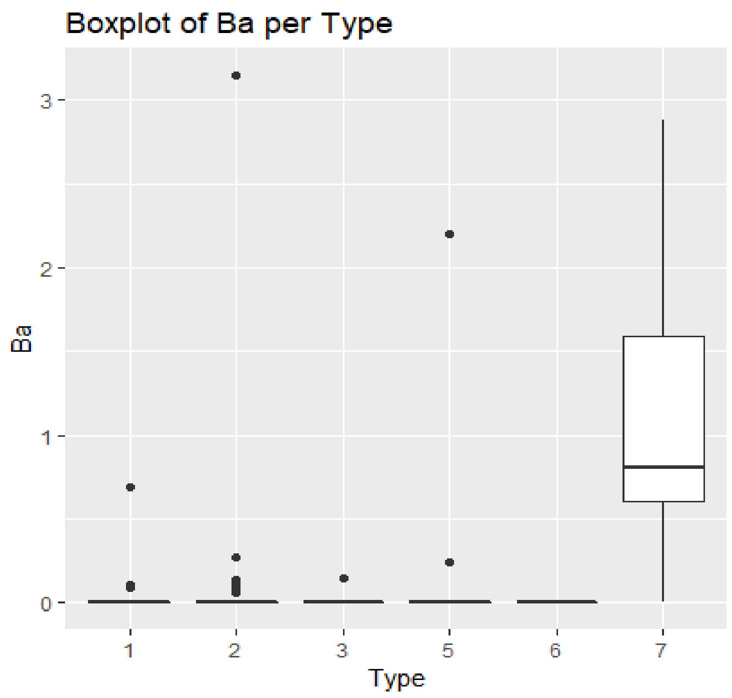
```
> plot.BoxCa <- ggplot(Glass, aes(x = Type, y = Ca)) +geom_boxplot()
+ ggtitle("Boxplot of Ca per Type")
> plot.BoxCa
```



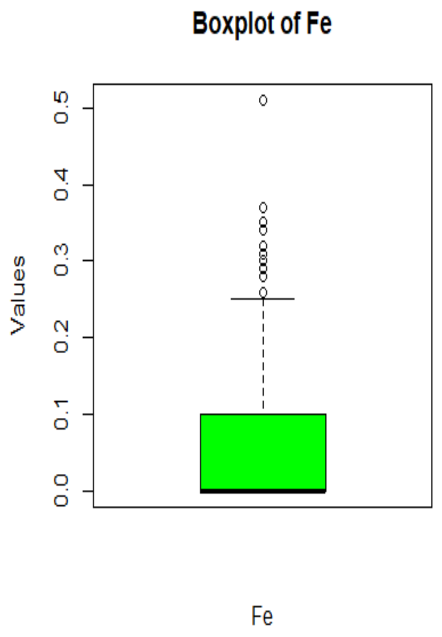


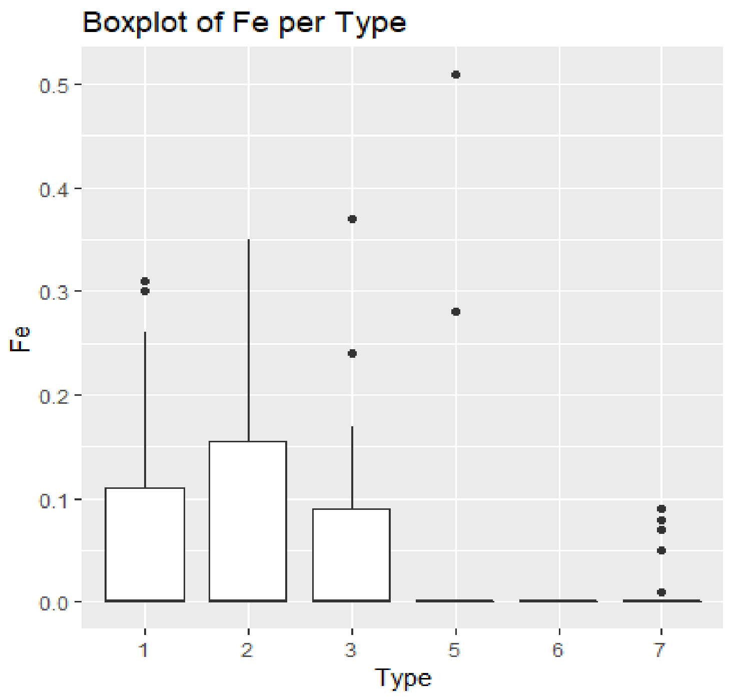
```
> boxplot(Glass$Ba, col = "green", main="Boxplot of Ba", xlab = "Ba", ylab =
"Values")

> plot.BoxBa <- ggplot(Glass, aes(x = Type, y = Ba)) +geom_boxplot()
+ ggtitle("Boxplot of Ba per Type")
> plot.BoxBa
```





```
> boxplot(Glass$Fe, col = "green", main="Boxplot of Fe", xlab = "Fe", ylab =
"Values")

> plot.BoxFe <- ggplot(Glass, aes(x = Type, y = Fe)) +geom_boxplot()
+ ggtitle("Boxplot of Fe per Type")
> plot.BoxFe
```





Here, we can see that all the predictors have outliers except Mg. However, when I tried to do boxplots for each predictor w.r.t. Type it can be seen that for few predictors such as Mg there are outliers present when we try to plot it wr.t. each Type. Also, Ba has outliers but interestingly it is only because of Type 7.

Computing Skewness:

Skewness value for each of the predictor variable:

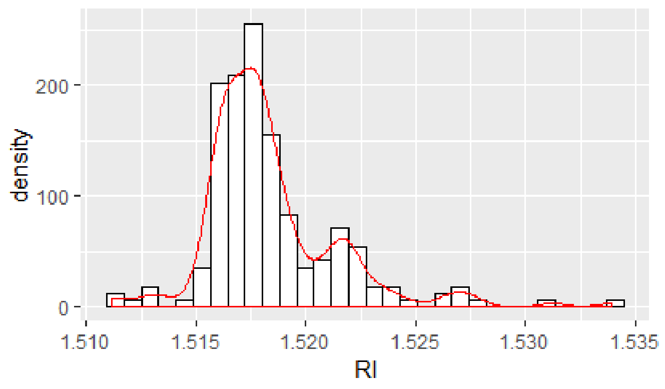
```
> library(e1071)
> skewness(Glass$RI)
[1] 1.602715
> skewness(Glass$Na)
[1] 0.4478343
> skewness(Glass$Mg)
[1] -1.136452
> skewness(Glass$Al)
[1] 0.8946104
> skewness(Glass$Si)
[1] -0.7202392
> skewness(Glass$K)
[1] 6.460089
> skewness(Glass$Ca)
[1] 2.018446
> skewness(Glass$Ba)
[1] 3.36868
> skewness(Glass$Fe)
[1] 1.729811
```

```
> skewValues<- apply(GlassData,2,skewness)
> skewValues
        RI         Na         Mg         Al         Si          K         Ca 
 1.6027151  0.4478343 -1.1364523  0.8946104 -0.7202392  6.4600889  2.0184463 
        Ba         Fe 
 3.3686800  1.7298107 
```

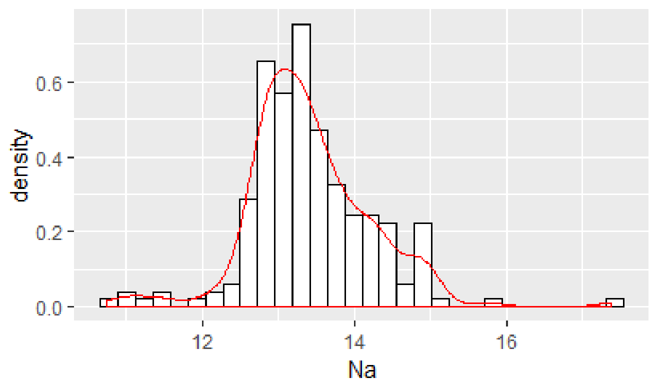
Histogram with Density Plot for checking skewness:

Here, we also look at the Density Plots for Skewness in Data graphically:

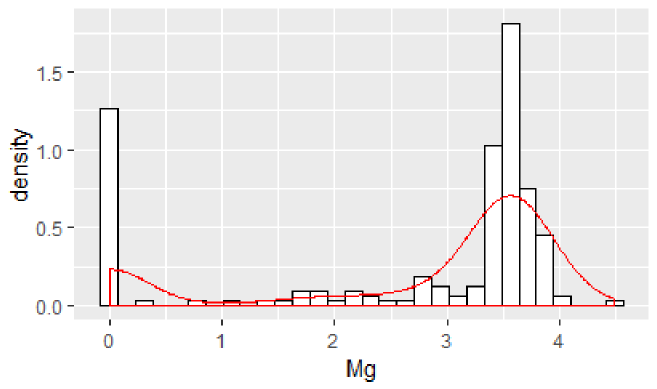
```
> ggplot(Glass, aes(x=RI)) + geom_histogram(aes(y=..density..),colour="black"
, fill="white") +
+   geom_density(alpha=.5, colour="red")
```



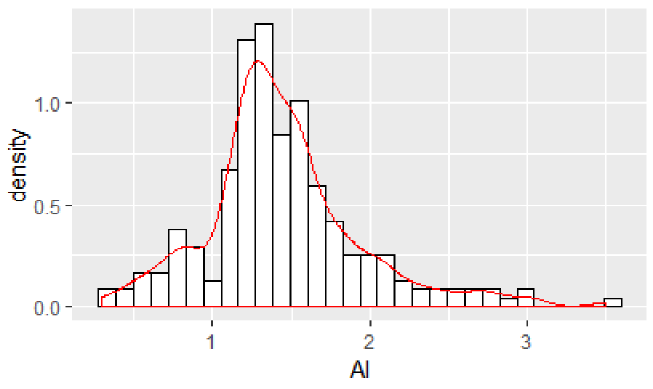
```
> ggplot(Glass, aes(x=Na)) + geom_histogram(aes(y=..density..),colour="black"
, fill="white") +
+   geom_density(alpha=.5, colour="red")
```



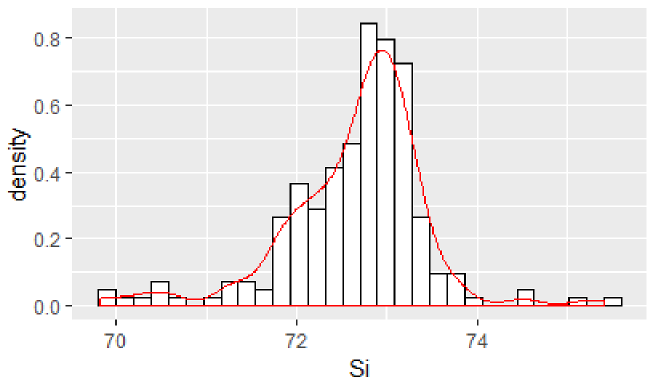
```
> ggplot(Glass, aes(x=Mg)) + geom_histogram(aes(y=..density..),colour="black"
, fill="white") +
+   geom_density(alpha=.5, colour="red")
```



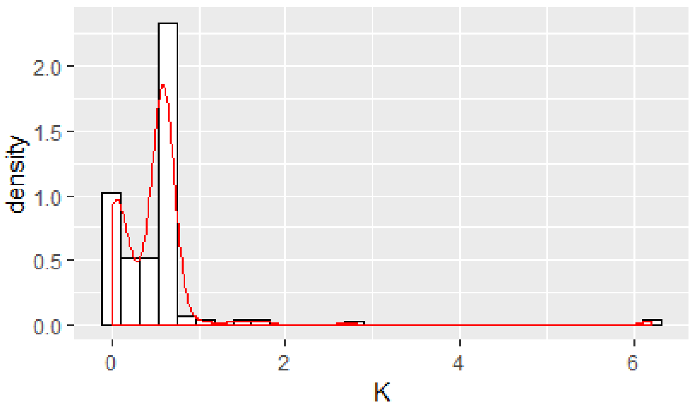
```
> ggplot(Glass, aes(x=Al)) + geom_histogram(aes(y=..density..),colour="black"
, fill="white") +
+   geom_density(alpha=.5, colour="red")
```



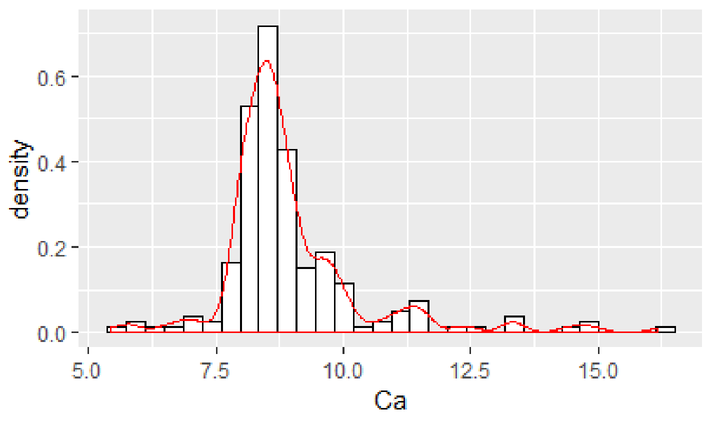
```
> ggplot(Glass, aes(x=Si)) + geom_histogram(aes(y=..density..),colour="black"
, fill="white") +
+   geom_density(alpha=.5, colour="red")
```



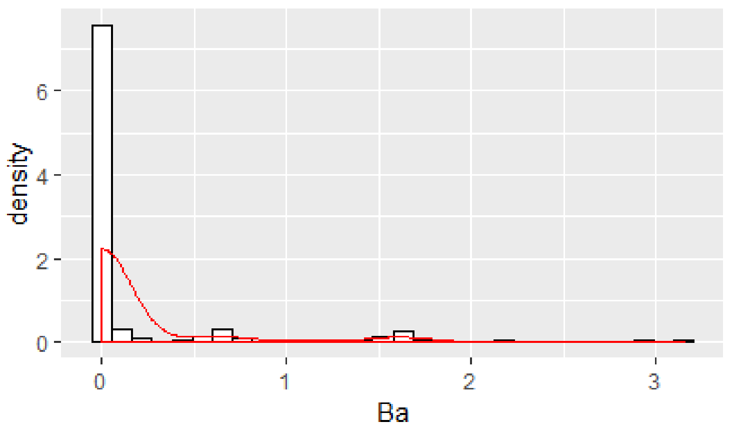
```
> ggplot(Glass, aes(x=K)) + geom_histogram(aes(y=..density..),colour="black",
fill="white") +
+   geom_density(alpha=.5, colour="red")
```



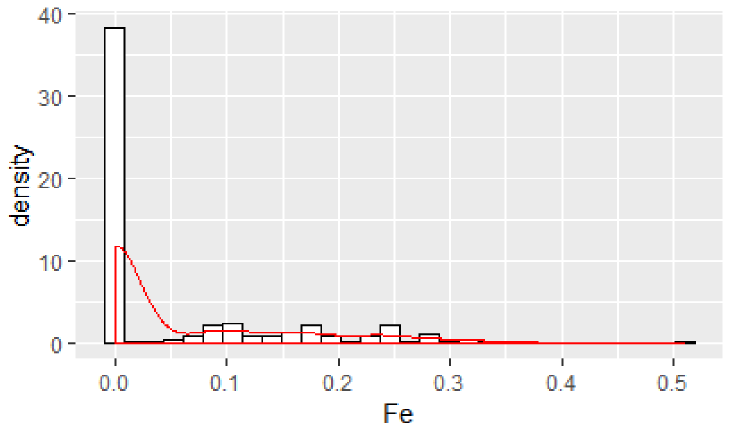
```
> ggplot(Glass, aes(x=Ca)) + geom_histogram(aes(y=..density..),colour="black"
, fill="white") +
+   geom_density(alpha=.5, colour="red")
```



```
> ggplot(Glass, aes(x=Ba)) + geom_histogram(aes(y=..density..),colour="black"
, fill="white") +
+   geom_density(alpha=.5, colour="red")
```



```
> ggplot(Glass, aes(x=Fe)) + geom_histogram(aes(y=..density..),colour="black"
, fill="white") +
+   geom_density(alpha=.5, colour="red")
```



After plotting, we can see that predictors Mg, K, Ba and Fe has some skewness in its data.

(c) Are there any relevant transformations of one or more predictors that might improve the classification model? (Hint: You could transform the predictors using the BoxCox Transformation. This can be done using mathematical formulation, or using the "preprocess" function in the AppliedPredictiveModeling package).

We do know that most predictive methods work when the predictors do not show significant shewness. However, since we have skewed data and outliers in our dataset we apply the BoxCox Transformation. But we do know that BoxCox can only be applied to possible skewness.

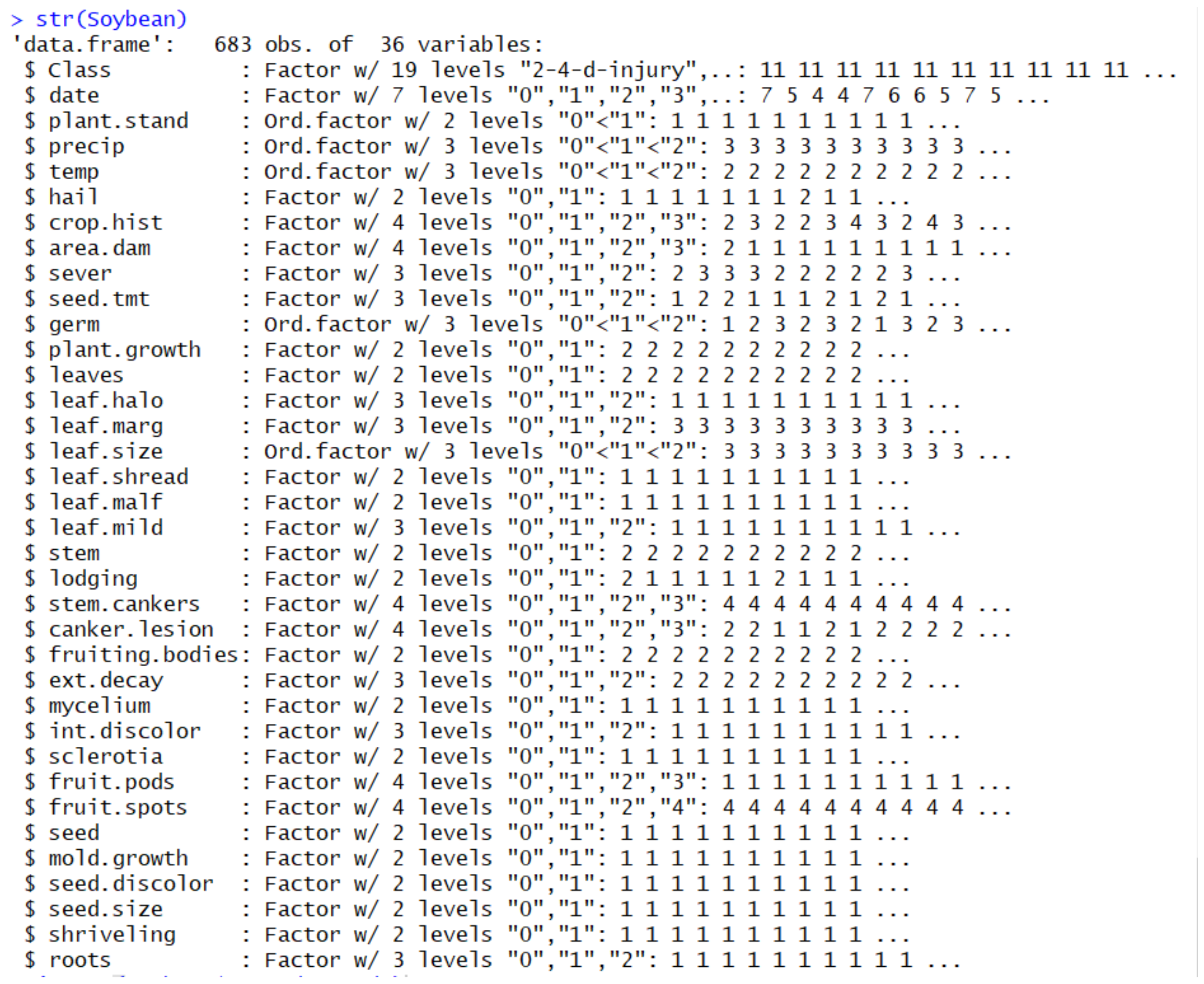
```
> library(caret)
>
> Glasstrans<-preProcess(Glass[,-10], method = "BoxCox")
> transformed<-predict(Glasstrans, Glass[,-10])
> skewValuesAfterTrans<-apply(transformed[,-10],2,skewness)
>
> skewValues
        RI         Na         Mg         Al         Si          K         Ca 
Ba         Fe 
 1.6027151  0.4478343 -1.1364523  0.8946104 -0.7202392  6.4600889  2.0184463 
3.3686800  1.7298107 
> skewValuesAfterTrans
         RI          Na          Mg          Al          Si           K 
Ca          Ba          Fe 
 1.56566039  0.03384644 -1.13645228  0.09105899 -0.65090568  6.46008890 -0.19
395573  3.36867997  1.72981071
```

After applying BoxCox Transformation, it can be seen that there is no relevant transformation improvement for any of the predictors.

Part III: (35pts) Data Preprocessing (Exercise 3.2 of APM Book).

(a) Investigate the frequency distributions for the categorical predictors. Are there any extremely unbalanced categorical predictors? In the extreme case, if there is only one value for the predictor, it is called a degenerate case. Such degenerate predictors should be removed from subsequent analysis.

```
> data(Soybean)
> str(Soybean)
```



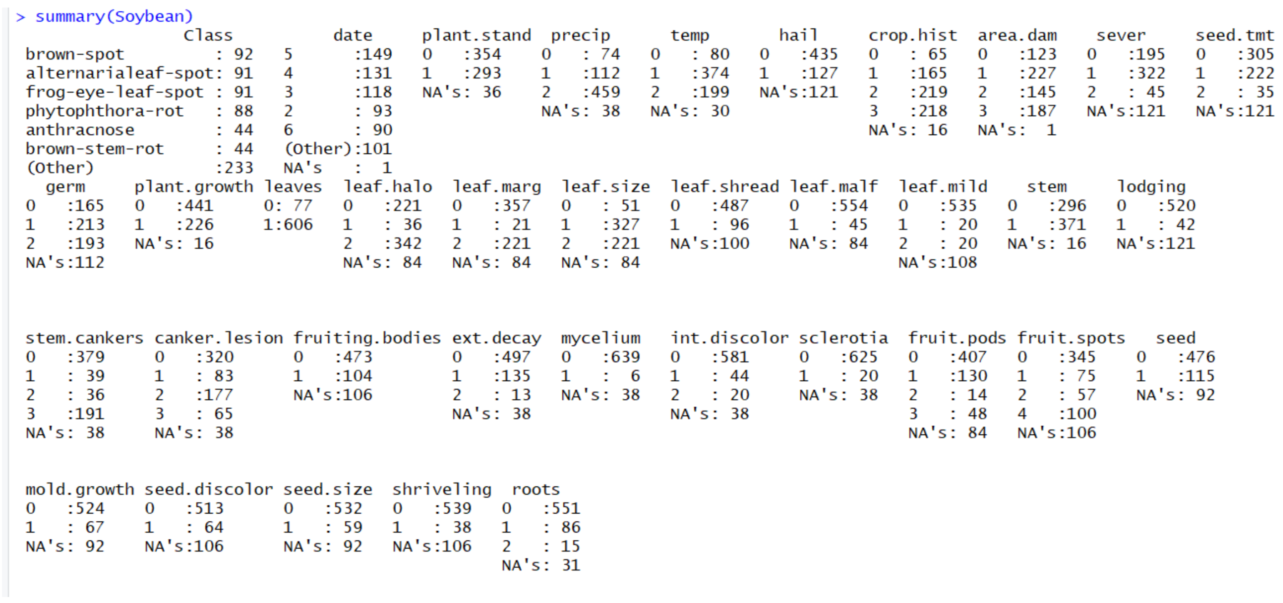
```
> str(Soybean)
'data.frame':   683 obs. of  36 variables:
 $ Class          : Factor w/ 19 levels "2-4-d-injury",..: 11 11 11 11 11 11 11 11 11 11 ...
 $ date           : Factor w/ 7 levels "0","1","2","3",..: 7 5 4 4 7 6 6 5 7 5 ...
 $ plant.stand    : Ord.factor w/ 2 levels "0"<"1": 1 1 1 1 1 1 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 ...
```

After looking at the structure of the Soybean Dataset, we can see that few predictors have ordered factors. So, we change the ordered factors to factors. Also, we can see that the target variable has 19 different classes.

Frequency Distributions of Predictor Variables for Soybean Dataset:

Summary lets us know about the frequency distributions of the Predictor variables. Also, we can use the table function for getting the frequency distributions for each predictor variables and plot its histogram.

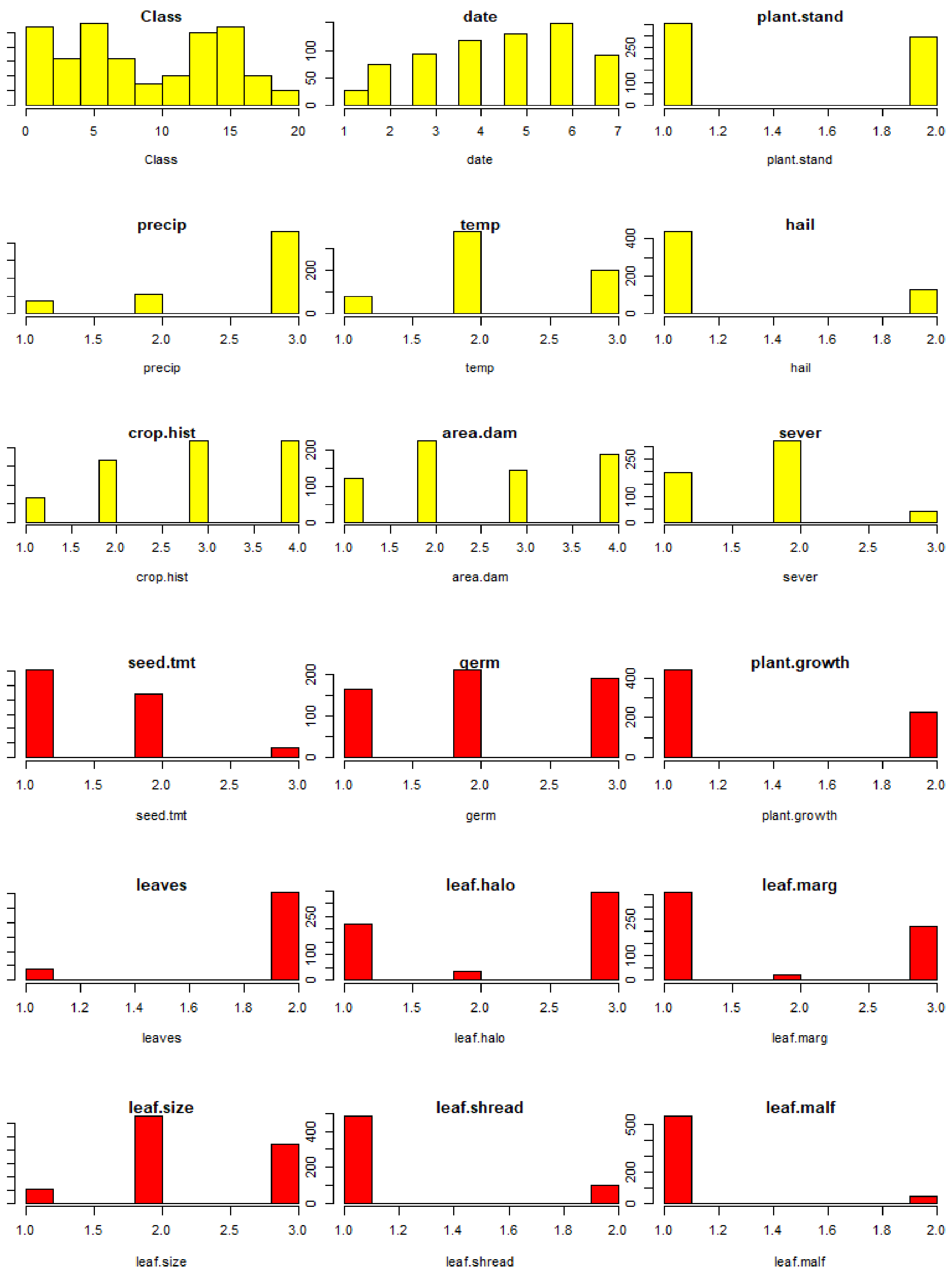
```
> summary(Soybean)
```

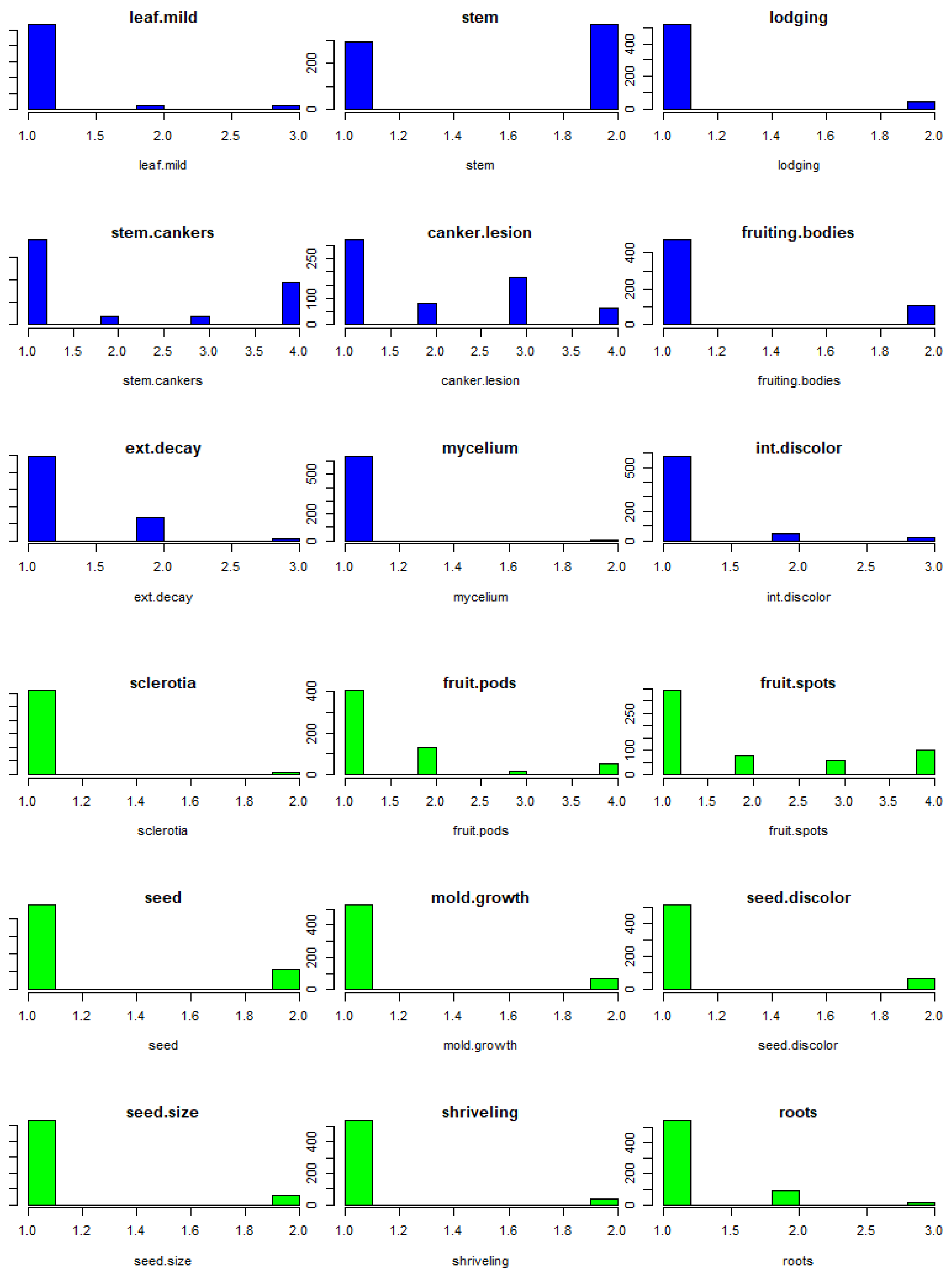


```
> summary(Soybean)
                 Class          date      plant.stand  precip      temp       hail      crop.hist  area.dam     sever      seed.tmt
 brown-spot            : 92   5      :149   0   :354   0   : 74   0   : 80   0   :435   0   : 65   0   :123   0   :195   0   :305
 alternarialeaf-spot: 91   4      :131   1   :293   1   :112   1   :374   1   :127   1   :165   1   :227   1   :322   1   :222
 frog-eye-leaf-spot : 91   3      :118   NA's: 36   2   :459   2   :199   NA's:121   2   :219   2   :145   2   : 45   2   : 35
 phytophthora-rot   : 88   2      : 93              NA's: 38   NA's: 30              3   :218   3   :187   NA's:121   NA's:121
 anthracnose        : 44   6      : 90                                               NA's: 16   NA's:  1
 brown-stem-rot     : 44   (Other):101
 (Other)            :233   NA's   :  1
   germ     plant.growth leaves  leaf.halo  leaf.marg  leaf.size  leaf.shread leaf.malf  leaf.mild    stem     lodging
 0   :165   0   :441     0: 77   0   :221   0   :357   0   : 51   0   :487    0   :554   0   :535   0   :296   0   :520
 1   :213   1   :226     1:606   1   : 36   1   : 21   1   :327   1   : 96    1   : 45   1   : 20   1   :371   1   : 42
 2   :193   NA's: 16             2   :342   2   :221   2   :221   NA's:100    NA's: 84   2   : 20   NA's: 16   NA's:121
 NA's:112                        NA's: 84   NA's: 84   NA's: 84                          NA's:108



 stem.cankers canker.lesion fruiting.bodies ext.decay  mycelium   int.discolor sclerotia  fruit.pods fruit.spots   seed
 0   :379     0   :320      0   :473        0   :497   0   :639   0   :581     0   :625   0   :407   0   :345   0   :476
 1   : 39     1   : 83      1   :104        1   :135   1   :  6   1   : 44     1   : 20   1   :130   1   : 75   1   :115
 2   : 36     2   :177      NA's:106        2   : 13   NA's: 38   2   : 20     NA's: 38   2   : 14   2   : 57   NA's: 92
 3   :191     3   : 65                      NA's: 38              NA's: 38                3   : 48   4   :100
 NA's: 38     NA's: 38                                                                    NA's: 84   NA's:106

 mold.growth seed.discolor seed.size  shriveling  roots
 0   :524    0   :513      0   :532   0   :539   0   :551
 1   : 67    1   : 64      1   : 59   1   : 38   1   : 86
 NA's: 92    NA's:106      NA's: 92   NA's:106   2   : 15
                                                 NA's: 31
```

We can use Histograms to plot the Frequency distributions of each of the predictors graphically.

```
> par(mfrow=c(3,3), mai = c(1, 0.1, 0.1, 0.1))
> for(i in 1:9)
+ {  hist(as.numeric(Soybean[,i]), main = colnames(Soybean[i]), col = "yellow
", xlab = colnames(Soybean[i]))}


> par(mfrow=c(3,3), mai = c(1, 0.1, 0.1, 0.1))
> for(i in 10:18)
+ {  hist(as.numeric(Soybean[,i]), main = colnames(Soybean[i]), col = "red",
xlab = colnames(Soybean[i]))}


> par(mfrow=c(3,3), mai = c(1, 0.1, 0.1, 0.1))
> for(i in 19:27)
+ {  hist(as.numeric(Soybean[,i]), main = colnames(Soybean[i]), col = "blue",
xlab = colnames(Soybean[i]))}


> par(mfrow=c(3,3), mai = c(1, 0.1, 0.1, 0.1))
> for(i in 28:36)
+ {  hist(as.numeric(Soybean[,i]), main = colnames(Soybean[i]), col = "green"
, xlab = colnames(Soybean[i]))}
```


Class
date
plant.stand
precip
temp
hail
crop.hist
area.dam
sever
seed.tmt
germ
plant.growth
leaves
leaf.halo
leaf.marg
leaf.size
leaf.shread
leaf.malf


leaf.mild
stem
lodging
stem.cankers
canker.lesion
fruiting.bodies
ext.decay
mycelium
int.discolor
sclerotia
fruit.pods
fruit.spots
seed
mold.growth
seed.discolor
seed.size
shriveling
roots

After plotting the histograms, we can see that we do have cases of extremely unbalanced categorical predictors. We also do have cases of degenerate and near degenerate case where most of the values of a predictor is of only one value. So, we remove these near zero variance predictors from consequent analysis as they do not have much variations in its data.

```
> library(mlbench)
>library( caret )
> zerocol = nearZeroVar(Soybean)
> colnames( Soybean )[zerocol]
[1] "leaf.mild" "mycelium"  "sclerotia"
> Soybean = Soybean[,-zerocol]
```

(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?

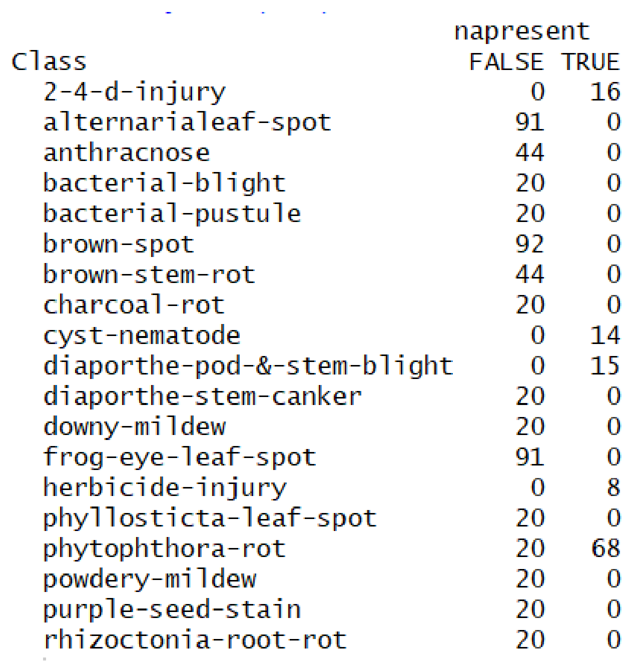
Since, 18% of the data is missing. We explore for the count of missing data for each of the predictor value.

```
> apply(Soybean[,2:33],2,function(x){sum(is.na(x))})
          Class            date      plant.stand          precip            temp
              0               1               36              38              30
           hail       crop.hist         area.dam           sever        seed.tmt
            121              16                1             121             121
           germ    plant.growth           leaves       leaf.halo       leaf.marg
            112              16                0              84              84
      leaf.size     leaf.shread        leaf.malf            stem         lodging
             84             100               84              16             121
   stem.cankers   canker.lesion  fruiting.bodies       ext.decay    int.discolor
             38              38              106              38              38
     fruit.pods     fruit.spots             seed     mold.growth    seed.discolor
             84             106               92              92             106
      seed.size      shriveling            roots
             92             106               31
```

From the above predictor count we can see that for few predictors we have a large number of missing values. E.g. Hail, sever, etc. This can be because the entries for No Hail being zero would have been left blank.

To check which particular predictor are more likely to be missing than others:

```
> Soybean$napresent = apply(Soybean[,2:33],1,function(x){sum(is.na(x))>0})
> table(Soybean[,c(1,34)])
```



```
                                napresent
Class                            FALSE TRUE
  2-4-d-injury                       0   16
  alternarialeaf-spot               91    0
  anthracnose                       44    0
  bacterial-blight                  20    0
  bacterial-pustule                 20    0
  brown-spot                        92    0
  brown-stem-rot                    44    0
  charcoal-rot                      20    0
  cyst-nematode                      0   14
  diaporthe-pod-&-stem-blight        0   15
  diaporthe-stem-canker             20    0
  downy-mildew                      20    0
  frog-eye-leaf-spot                91    0
  herbicide-injury                   0    8
  phyllosticta-leaf-spot            20    0
  phytophthora-rot                  20   68
  powdery-mildew                    20    0
  purple-seed-stain                 20    0
  rhizoctonia-root-rot              20    0
```

From the above, we can see that for most of the class types we have no missing values whereas for few of the classes we have all the values missing. So, we can clearly say that the pattern of missing data is clearly related to the classes.

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

Handling of missing data can be performed in two ways – Either by eliminating or imputing. But it’s always a good practice to delete NA’s with less missing values and impute when we have a large number of missing values.

Since, in our data we have a lot of missing values we perform imputation by replacing NA’s with the Mode value.

```
> sum(is.na(Soybean))
[1] 2337
>
> Soybeannew = sapply(Soybean,function(x){impute(x,mode)})
> sum(is.na(Soybeannew))
[1] 0
```

Appendix

R Code:

HW1.R

```
n<-15
n
a = 12
a
24->z
z
N<-26.42
N
n
ls()
rm(n)
ls()
?ls
help(rm)
apropos("help") # "help" in name

help.search("help") # "help" in name or summary; note quotes!
help.start() # also remember the R Commands web page (link on # class page)

name<-"Mike"
name

q1<-TRUE
q1

q2<-F
q2
ls()

a <- 12+14
a

3*5

(20-4)/2

7^2

exp(2) # e^2
log(10)
```

```
log10(10)
log2(64)
pi
cos(pi)
sqrt(100)

#HW Q1

27*(38-17) #567
log(147) #4.990433
sqrt(436/12) #6.027714


a <- c(1,7, 32, 16) #Array of vectors
a

b<-1:10
b

c<-20:15
c

d <- seq(1, 5, by=0.5)
d

e<- seq(0,10, length=5)
e
f<-rep(0,5)
f

g<-rep(1:3,4)
g

h<-rep(4:6,1:3)
h

x <- rnorm(5) #rnorm(n, mean = , sd = )  Standard normal random variables
x

y <- rnorm(7, 10, 3) # Normal r.v.s with mu = 10, sigma = 3
y

z <- runif(10) # Uniform(0, 1) random variables
z

c(1, 2, 3) + c(4, 5, 6)
```

```
c(1, 2, 3, 4) + c(10, 20)

c(1, 2, 3) + c(10, 20)

sqrt(c(100, 225, 400))

d
d[3]
d[5:7]

d>2.8
d[d>2.8]

length(d)

length(d[d > 2.8])

#a = (5, 10, 15, 20, ..., 160)
a<-seq(5,160,by=5)
a
#b = (87, 86, 85, ..., 56)
b<-c(87:56)
b

#Use vector arithmetic to multiply these vectors and call the result d.
#Select subsets of d to identify the following.
d<-a*b
d
#(a) What are the 19th, 20th, and 21st elements of d?
d[19:21]
#  (b) What are all of the elements of d which are less than 2000?
d[d<2000]
#  (c) How many elements of d are greater than 6000?
length(d[d>6000])


1:4

sum(1:4)

prod(1:4)

max(1:10)

min(1:10)
```

```
range(1:10)

X <- rnorm(10)
X

mean(X)

sort(X)

median(X)

var(X)

sd(X)

#q3 answers
sum(d)
mean(d)
median(d)
sd(d)

data(iris)
head(iris)
hist(iris$Petal.Length)
hist(iris[,4])# alternative specification

boxplot(iris$Petal.Length)
boxplot(Petal.Length~Species, data=iris) # Formula description,
# side-by-side boxplots

#Q4

data(cars)
head(cars)
#(a) Plot a histogram of distance using the hist function
hist(cars$dist,main = "Histogram of Distance", xlab = "Distance", ylab = "Frequency /
Count", col ="blue")

ggplot(data = cars,aes(cars$dist))+ geom_histogram(breaks=seq(0, 120, by = 20))

#(b) Generate a boxplot of speed.
boxplot(cars$speed, col = "green", main="Boxplot of Speed", xlab = "Speed", ylab = "Speed
value range")

#(c) Use the plot(,) function (e.g. plot(variableX, variableY) to create a
#scatterplot of dist against speed.
plot(x = cars$dist, y = cars$speed, main = "Scatterplot of Distance against Speed", xlab =
"Distance", ylab = "Speed")

ggplot(data = cars, aes(dist, speed))+ geom_point()

#[Note: You can also create the graphs in parts (a) to (c) using ggplot2 package].

#Part II

library(mlbench)
data(Glass)
str(Glass)
?Glass

library(ggplot2)
#(a) Using visualizations, explore the predictor variables to understand their
#distributions as well as the relationships between predictors.
#Provide the pairwise scatter plots and investigate the correlation matrix.

#To explore predictor variables we can use either histograms or density plots. I have
showed histogram
#an density in one plot for each of the glass types.

#histogram of the target variable i.e. Y
ggplot(Glass,aes(x=Type))+geom_bar()+ggtitle("Observation count by Type of Glass")
#The above command tells me that in my Glass dataset most of the glass types
#are of type 1 or of type 2.

#Histogram of all the predictor variables
GlassRI<-ggplot(Glass, aes(x=RI)) +
  geom_histogram(color="black", fill="green")+ggtitle("Histogram of RI")
GlassNa<-ggplot(Glass, aes(x=Na)) +
  geom_histogram(color="black", fill="green")+ggtitle("Histogram of Na")
GlassMg<-ggplot(Glass, aes(x=Mg)) +
  geom_histogram(color="black", fill="green")+ggtitle("Histogram of Mg")
GlassAl<-ggplot(Glass, aes(x=Al)) +
  geom_histogram(color="black", fill="green")+ggtitle("Histogram of Al")

grid.arrange(GlassRI, GlassNa,GlassMg,GlassAl, ncol=2)

GlassSi<-ggplot(Glass, aes(x=Si)) +
  geom_histogram(color="black", fill="green")+ggtitle("Histogram of Si")
GlassK<-ggplot(Glass, aes(x=K)) +
  geom_histogram(color="black", fill="green")+ggtitle("Histogram of K")
GlassCa<-ggplot(Glass, aes(x=Ca)) +
  geom_histogram(color="black", fill="green")+ggtitle("Histogram of Ca")
GlassBa<-ggplot(Glass, aes(x=Ba)) +
  geom_histogram(color="black", fill="green")+ggtitle("Histogram of Ba")

grid.arrange(GlassSi, GlassK, GlassCa, GlassBa, ncol=2)

GlassFe<-ggplot(Glass, aes(x=Fe)) +
  geom_histogram(color="black", fill="green")+ggtitle("Histogram of Fe")
GlassFe


ggplot(data = cars, aes(dist, speed))+ geom_point()

##Correlation Matrix
library(corrplot)
Glasscorr <- Glass[,1:length(Glass)]
Glasscorr<-data.matrix(Glasscorr)
round(cor(Glasscorr),2)
corrplot.mixed(cor(Glasscorr),lower = "number",upper = "circle")
#corrplot(cor(Glasscorr), method = "circle")

#Correlation value of each predictor w.r.t. Type
cor(Glass[,-10],as.numeric(Glass[,10]))

#pairwise scatterplots of all the attributes
pairs(Type~.,data=Glass, main="Simple Scatterplot Matrix")
pairs(Glass[,-10],main="Scatterplot Matrix for Glass Dataset")



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

#Exploring the predictor variables w.r.t. outliers(boxplot) and distributions(histogram)
#Boxplots ---- for Outlier Analysis
boxplot(Glass)

BoxRI<-boxplot(Glass$RI, col = "green", main="Boxplot of RI", xlab = "RI", ylab = "Values")
BoxNa<-boxplot(Glass$Na, col = "green", main="Boxplot of Na", xlab = "Na", ylab =
"Values")
BoxMg<-boxplot(Glass$Mg, col = "green", main="Boxplot of Mg", xlab = "Mg", ylab =
"Values")
BoxAl<-boxplot(Glass$Al, col = "green", main="Boxplot of Al", xlab = "Al", ylab = "Values")
boxplot(Glass[,1:3], col = "green")

grid()
grid.arrange(BoxRI, BoxNa, BoxMg, BoxAl, ncol=2)
```

```
BoxSi<-boxplot(Glass$Si, col = "green", main="Boxplot of Si", xlab = "Si", ylab = "Values")
BoxK<-boxplot(Glass$K, col = "green", main="Boxplot of K", xlab = "K", ylab = "Values")
BoxCa<-boxplot(Glass$Ca, col = "green", main="Boxplot of Ca", xlab = "Ca", ylab = "Values")
BoxBa<-boxplot(Glass$Ba, col = "green", main="Boxplot of Ba", xlab = "Ba", ylab = "Values")
BoxFe<-boxplot(Glass$Fe, col = "green", main="Boxplot of Fe", xlab = "Fe", ylab = "Values")


grid.arrange(BoxSi, BoxK, BoxCa, BoxBa, ncol=2)

plot.BoxRI <- ggplot(Glass, aes(x = Type, y = RI)) +geom_boxplot() + ggtitle("Boxplot of RI
per Type")
plot.BoxRI

plot.BoxNa <- ggplot(Glass, aes(x = Type, y = Na)) +geom_boxplot() + ggtitle("Boxplot of Na
per Type")
plot.BoxNa

plot.BoxMg <- ggplot(Glass, aes(x = Type, y = Mg)) +geom_boxplot() + ggtitle("Boxplot of
Mg per Type")
plot.BoxMg

plot.BoxAl <- ggplot(Glass, aes(x = Type, y = Al)) +geom_boxplot() + ggtitle("Boxplot of Al
per Type")
plot.BoxAl

plot.BoxSi <- ggplot(Glass, aes(x = Type, y = Si)) +geom_boxplot() + ggtitle("Boxplot of Si
per Type")
plot.BoxSi

plot.BoxK <- ggplot(Glass, aes(x = Type, y = K)) +geom_boxplot() + ggtitle("Boxplot of K per
Type")
plot.BoxK

plot.BoxCa <- ggplot(Glass, aes(x = Type, y = Ca)) +geom_boxplot() + ggtitle("Boxplot of Ca
per Type")
plot.BoxCa

plot.BoxBa <- ggplot(Glass, aes(x = Type, y = Ba)) +geom_boxplot() + ggtitle("Boxplot of Ba
per Type")
plot.BoxBa

plot.BoxFe <- ggplot(Glass, aes(x = Type, y = Fe)) +geom_boxplot() + ggtitle("Boxplot of Fe
per Type")
plot.BoxFe
```

```
#Compute skewness

library(e1071)
skewness(Glass$RI)
skewness(Glass$Na)
skewness(Glass$Mg)
skewness(Glass$Al)
skewness(Glass$Si)
skewness(Glass$K)
skewness(Glass$Ca)
skewness(Glass$Ba)
skewness(Glass$Fe)

GlassData<-Glass[,-10]

skewValues<- apply(GlassData,2,skewness)
skewValues

#Histograms with density plot for checking skewness

ggplot(Glass, aes(x=RI)) + geom_histogram(aes(y=..density..),colour="black", fill="white") +
  geom_density(alpha=.5, colour="red")

ggplot(Glass, aes(x=Na)) + geom_histogram(aes(y=..density..),colour="black", fill="white")
+
  geom_density(alpha=.5, colour="red")

ggplot(Glass, aes(x=Mg)) + geom_histogram(aes(y=..density..),colour="black", fill="white")
+
  geom_density(alpha=.5, colour="red")

ggplot(Glass, aes(x=Al)) + geom_histogram(aes(y=..density..),colour="black", fill="white") +
  geom_density(alpha=.5, colour="red")

ggplot(Glass, aes(x=Si)) + geom_histogram(aes(y=..density..),colour="black", fill="white") +
  geom_density(alpha=.5, colour="red")

ggplot(Glass, aes(x=K)) + geom_histogram(aes(y=..density..),colour="black", fill="white") +
  geom_density(alpha=.5, colour="red")

ggplot(Glass, aes(x=Ca)) + geom_histogram(aes(y=..density..),colour="black", fill="white")
+
  geom_density(alpha=.5, colour="red")

ggplot(Glass, aes(x=Ba)) + geom_histogram(aes(y=..density..),colour="black", fill="white")
+
  geom_density(alpha=.5, colour="red")

ggplot(Glass, aes(x=Fe)) + geom_histogram(aes(y=..density..),colour="black", fill="white")
+
  geom_density(alpha=.5, colour="red")
#Observations:  1. K and Mg has second modes near zero
#               2. Ca, Ba, Fe, RI has skewness


#(c) Are there any relevant transformations of one or more predictors that might
#improve the classification model? (Hint: You could transform the predictors using
#the BoxCox Transformation. This can be done using mathematical formulation, or
#using the “preprocess” function in the AppliedPredictiveModeling package).
library(MASS)
library(forecast)

lambdaMg = BoxCox.lambda(Glass$Mg)
lambdaMg




library(caret)

Glasstrans<-preProcess(Glass[,-10], method = "BoxCox")
transformed<-predict(Glasstrans, Glass[,-10])
skewValuesAfterTrans<-apply(transformed[,-10],2,skewness)

skewValues
skewValuesAfterTrans


##Ba, Fe, Mg, K

####
CoxRI<-BoxCoxTrans(Glass$K)
CoxRI

#part III

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

```
#(a) Investigate the frequency distributions for the categorical predictors.
#Are there any extremely unbalanced categorical predictors? In the extreme case,
#if there is only one value for the predictor, it is called a degenerate case.
#Such degenerate predictors should be removed from subsequent analysis.

library(dplyr)
summary(Soybean)

#Frequency Distributions for the categorical predictors

par(mfrow=c(3,3), mai = c(1, 0.1, 0.1, 0.1))
for(i in 1:9)
{  hist(as.numeric(Soybean[,i]), main = colnames(Soybean[i]), col = "yellow", xlab =
colnames(Soybean[i]))}

par(mfrow=c(3,3), mai = c(1, 0.1, 0.1, 0.1))
for(i in 10:18)
{  hist(as.numeric(Soybean[,i]), main = colnames(Soybean[i]), col = "red", xlab =
colnames(Soybean[i]))}

par(mfrow=c(3,3), mai = c(1, 0.1, 0.1, 0.1))
for(i in 19:27)
{  hist(as.numeric(Soybean[,i]), main = colnames(Soybean[i]), col = "blue", xlab =
colnames(Soybean[i]))}

par(mfrow=c(3,3), mai = c(1, 0.1, 0.1, 0.1))
for(i in 28:36)
{  hist(as.numeric(Soybean[,i]), main = colnames(Soybean[i]), col = "green", xlab =
colnames(Soybean[i]))}


summary(Soybean)

table(Soybean$Class)
barplot(table(Soybean$Class))

table(Soybean$date)
barplot(table(Soybean$date))

table(Soybean$plant.stand)
barplot(table(Soybean$plant.stand))

table(Soybean$precip)
barplot(table(Soybean$precip))
```

```
table(Soybean$temp)
barplot(table(Soybean$temp))

table(Soybean$hail)
barplot(table(Soybean$hail))

table(Soybean$crop.hist)
barplot(table(Soybean$crop.hist))

table(Soybean$area.dam)
barplot(table(Soybean$area.dam))

table(Soybean$sever)
barplot(table(Soybean$sever))

table(Soybean$seed.tmt)
barplot(table(Soybean$seed.tmt))

table(Soybean$germ)
barplot(table(Soybean$germ))

table(Soybean$plant.growth)
barplot(table(Soybean$plant.growth))

table(Soybean$leaves)
barplot(table(Soybean$leaves))

table(Soybean$leaf.halo)
barplot(table(Soybean$leaf.halo))

table(Soybean$leaf.marg)
barplot(table(Soybean$leaf.marg))

table(Soybean$leaf.size)
barplot(table(Soybean$leaf.size))

table(Soybean$leaf.shread)
barplot(table(Soybean$leaf.shread))

table(Soybean$leaf.malf)
barplot(table(Soybean$leaf.malf))

table(Soybean$leaf.mild)
barplot(table(Soybean$leaf.mild))

table(Soybean$stem)
```

```
barplot(table(Soybean$stem))

table(Soybean$lodging)
barplot(table(Soybean$lodging))

table(Soybean$stem.cankers)
barplot(table(Soybean$stem.cankers))

table(Soybean$canker.lesion)
barplot(table(Soybean$canker.lesion))

table(Soybean$fruiting.bodies)
barplot(table(Soybean$fruiting.bodies))

table(Soybean$ext.decay)
barplot(table(Soybean$ext.decay))

table(Soybean$mycelium)
barplot(table(Soybean$mycelium))

table(Soybean$int.discolor)
barplot(table(Soybean$int.discolor))

table(Soybean$sclerotia)
barplot(table(Soybean$sclerotia))

table(Soybean$fruit.pods)
barplot(table(Soybean$fruit.pods))

table(Soybean$fruit.spots)
barplot(table(Soybean$fruit.spots))

table(Soybean$seed)
barplot(table(Soybean$seed))

table(Soybean$mold.growth)
barplot(table(Soybean$mold.growth))

table(Soybean$seed.discolor)
barplot(table(Soybean$seed.discolor))

table(Soybean$seed.size)
barplot(table(Soybean$seed.size))

table(Soybean$shriveling)
barplot(table(Soybean$shriveling))
```

```
table(Soybean$roots)
barplot(table(Soybean$roots))

#To identify and remove NearZero Variance Predictors
library(caret)
zerocol = nearZeroVar(Soybean)
colnames( Soybean )[zerocol]
Soybean = Soybean[,-zerocol]

#(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?

#Counting NA's in each column:

apply(Soybean[,2:33],2,function(x){sum(is.na(x))})

# To check which particular predictor have more NA's then others:

Soybean$napresent = apply(Soybean[,2:33],1,function(x){sum(is.na(x))>0})
table(Soybean[,c(1,34)])

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

# For imputation of data for the NA's

library(caret)
#preProcess(Soybean[,2:33],method="knnImpute",na.remove=FALSE)

summary(Soybean)

#Imputation using MICE

library(mice)
#md.pattern(Soybean)

#Imputing the missing values

# imputed_Data <- mice(Soybean, m=34, maxit = 50, method = 'pmm', seed = 100)
# summary(imputed_Data)
#
# imputed_Data$imp$hail

# zerocolimp = nearZeroVar(imputed_Data)
# colnames( imputed_Data )[zerocolimp]
# imputed_Data = imputed_Data[,-zerocolimp]

###
#
# install.packages("Hmisc")
# library(Hmisc)
#
#
# Soybean$plant.stand=impute(as.factor(Soybean$plant.stand),mode)
#
# apply(Soybean[,2:33],2,function(x){sum(is.na(x))})

sum(is.na(Soybean))

Soybeannew = sapply(Soybean,function(x){impute(x,mode)})
sum(is.na(Soybeannew))
```