Predictive Modelling - Chapter3_Exercises

Chathura J Gunasekara

Tuesday, September 23, 2014

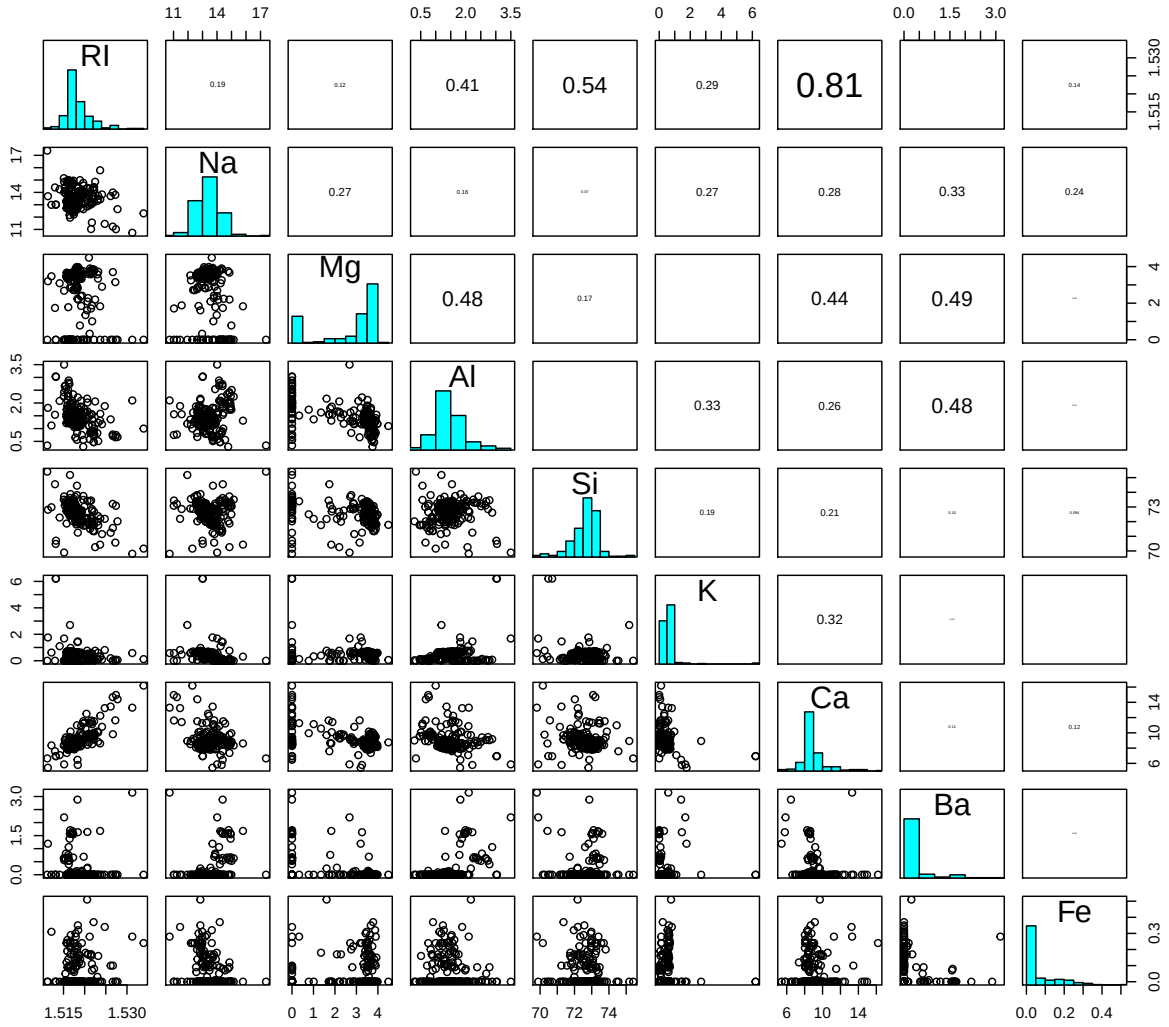
3.1) Glass Identification Data set There are nine predictors, including the refractive index and percentages of eight elements: Na, Mg, Al, Si, K, Ca, Ba, and Fe.

(a)

```
rm(list=ls())
#install.packages("mlbench")
source("functions.R")
library(mlbench)
data(Glass)

summary(Glass)
```

```
##        RI              Na              Mg              Al
##  Min.   :1.51   Min.   :10.7   Min.   :0.00   Min.   :0.29
##  1st Qu.:1.52   1st Qu.:12.9   1st Qu.:2.12   1st Qu.:1.19
##  Median :1.52   Median :13.3   Median :3.48   Median :1.36
##  Mean   :1.52   Mean   :13.4   Mean   :2.68   Mean   :1.45
##  3rd Qu.:1.52   3rd Qu.:13.8   3rd Qu.:3.60   3rd Qu.:1.63
##  Max.   :1.53   Max.   :17.4   Max.   :4.49   Max.   :3.50
##        Si              K               Ca              Ba
##  Min.   :69.8   Min.   :0.000   Min.   : 5.43   Min.   :0.000
##  1st Qu.:72.3   1st Qu.:0.122   1st Qu.: 8.24   1st Qu.:0.000
##  Median :72.8   Median :0.555   Median : 8.60   Median :0.000
##  Mean   :72.7   Mean   :0.497   Mean   : 8.96   Mean   :0.175
##  3rd Qu.:73.1   3rd Qu.:0.610   3rd Qu.: 9.17   3rd Qu.:0.000
##  Max.   :75.4   Max.   :6.210   Max.   :16.19   Max.   :3.150
##        Fe         Type
##  Min.   :0.000   1:70
##  1st Qu.:0.000   2:76
##  Median :0.000   3:17
##  Mean   :0.057   5:13
##  3rd Qu.:0.100   6: 9
##  Max.   :0.510   7:29
```

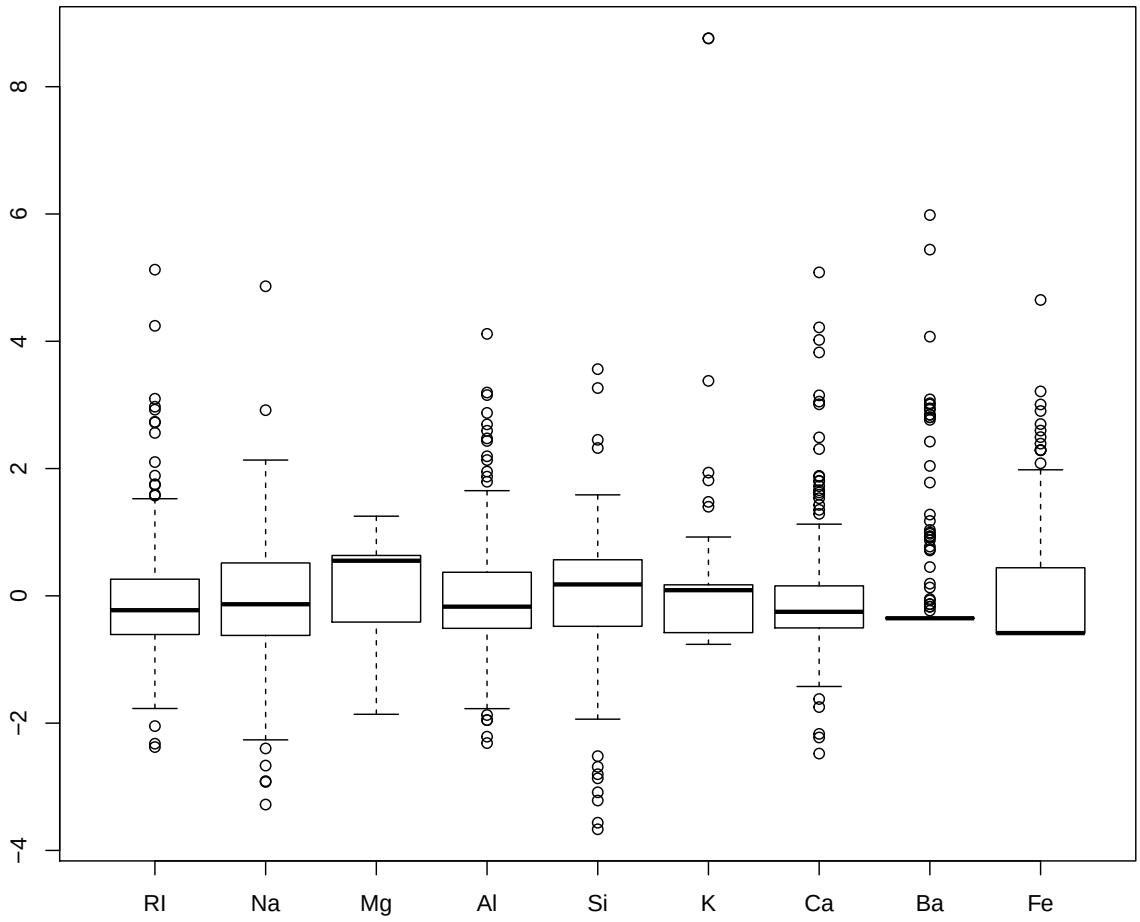
```
glass_red <- Glass[,1:9]

pairs(glass_red,upper.panel=panel.cor,diag.panel=panel.hist)
```



The figure shows distribution and relationship between predictors. The scatter plot,histogram and correlation between predictors are plotted in one figure. The size of the font shows how large the correlation is. for example correlation between RI and Ca is high.So we may possibly use one predictor instead of both after analysis with other variables. Distribution and scatter plots show the skewness and outliars.

(b) Yes , there appear to be outliars. Boxplots are used to visualize outliars. The following figure is plotted after centering and scaling the data set.

```
boxplot(scale(glass_red, center = TRUE, scale = TRUE))
```



Skeness is found using the follwoing function

```
library(e1071)
skewValues <- apply(glass_red, 2, skewness)
skewValues  ##skewValues
```

```
##      RI      Na      Mg      Al      Si       K      Ca      Ba      Fe
##  1.6027  0.4478 -1.1365  0.8946 -0.7202  6.4601  2.0184  3.3687  1.7298
```

A rule of thumb for skeness is -1 to 1 so RI, Mg, K,Ca,Ba,Fe are skewed.

(c) any relevant transformations of one or more predictors that might improve the classification model are found by using Box and Cox transformation.

```
library(caret)
```

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

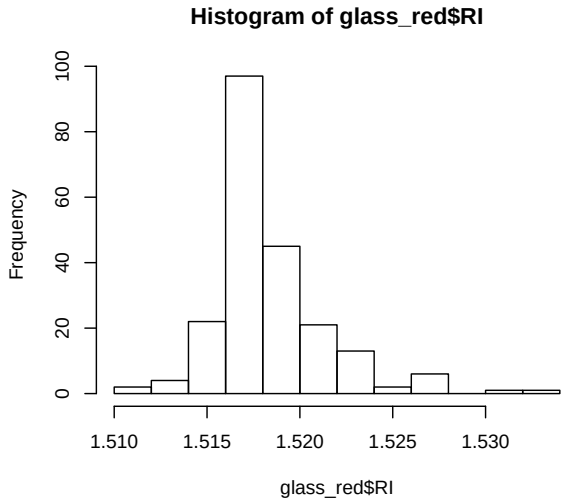
```
RITrans <- BoxCoxTrans(glass_red$RI)
RITrans
```

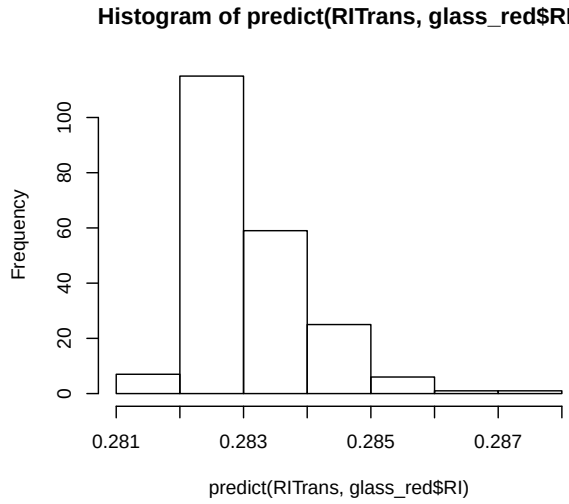
```
## Box-Cox Transformation
##
## 214 data points used to estimate Lambda
##
## Input data summary:
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1.51    1.52    1.52    1.52    1.52    1.53
##
## Largest/Smallest: 1.02
## Sample Skewness: 1.6
##
## Estimated Lambda: -2
```

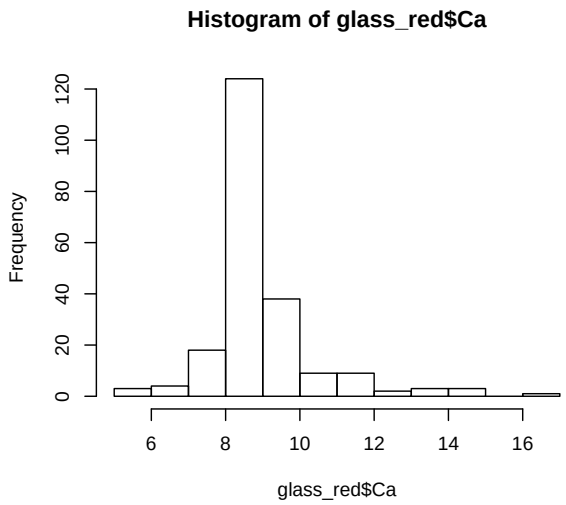
```
par(mfrow=c(2,2), pch=16)

hist(glass_red$RI)
hist(predict(RITrans,glass_red$RI))
CaTrans <- BoxCoxTrans(glass_red$Ca)
CaTrans
```

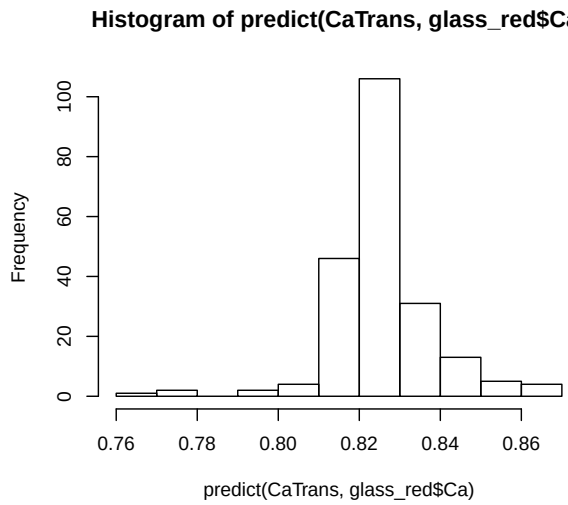
```
## Box-Cox Transformation
##
## 214 data points used to estimate Lambda
##
## Input data summary:
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    5.43    8.24    8.60    8.96    9.17   16.20
##
## Largest/Smallest: 2.98
## Sample Skewness: 2.02
##
## Estimated Lambda: -1.1
```

```
hist(glass_red$Ca)
hist(predict(CaTrans,glass_red$Ca))
```





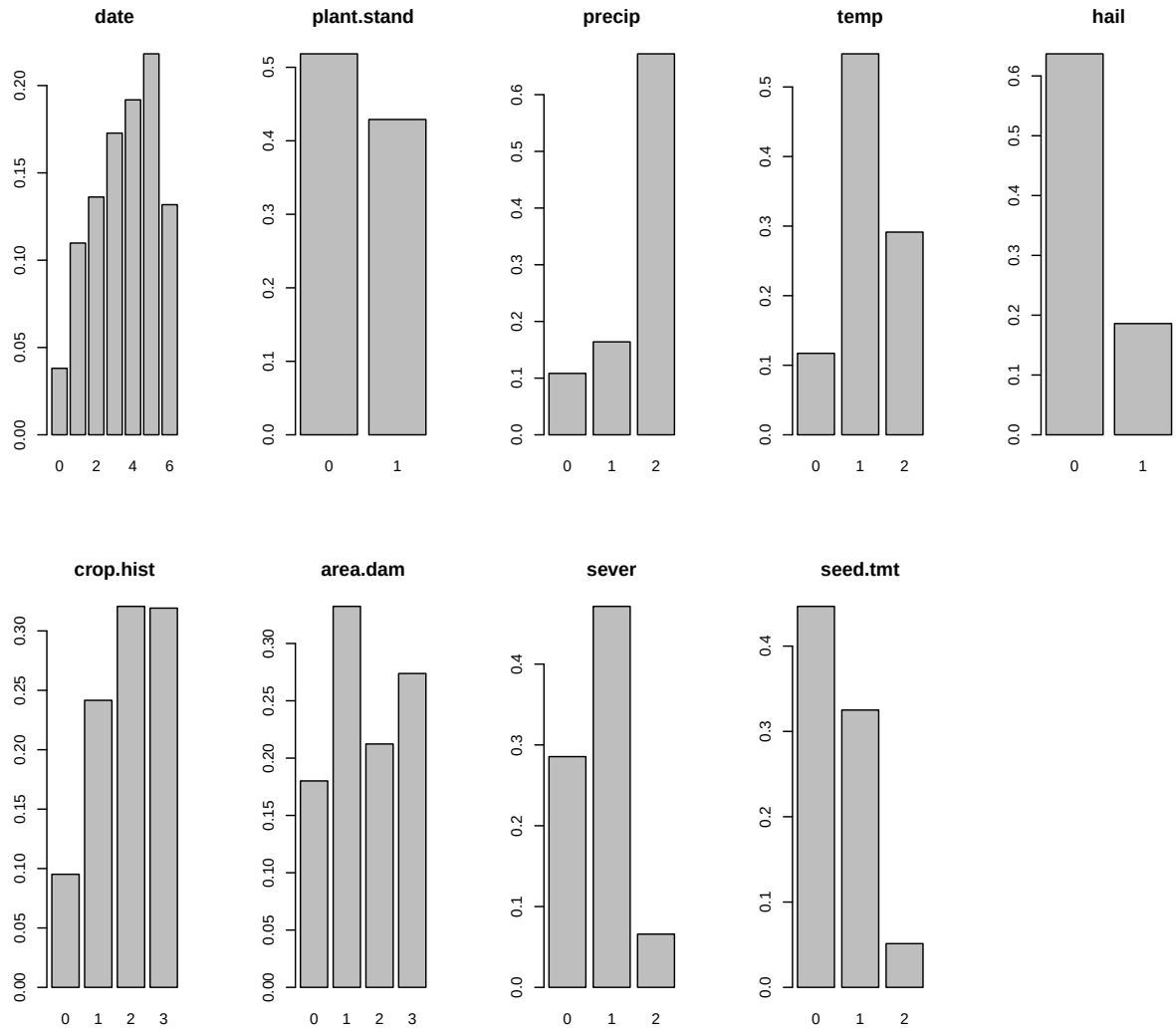




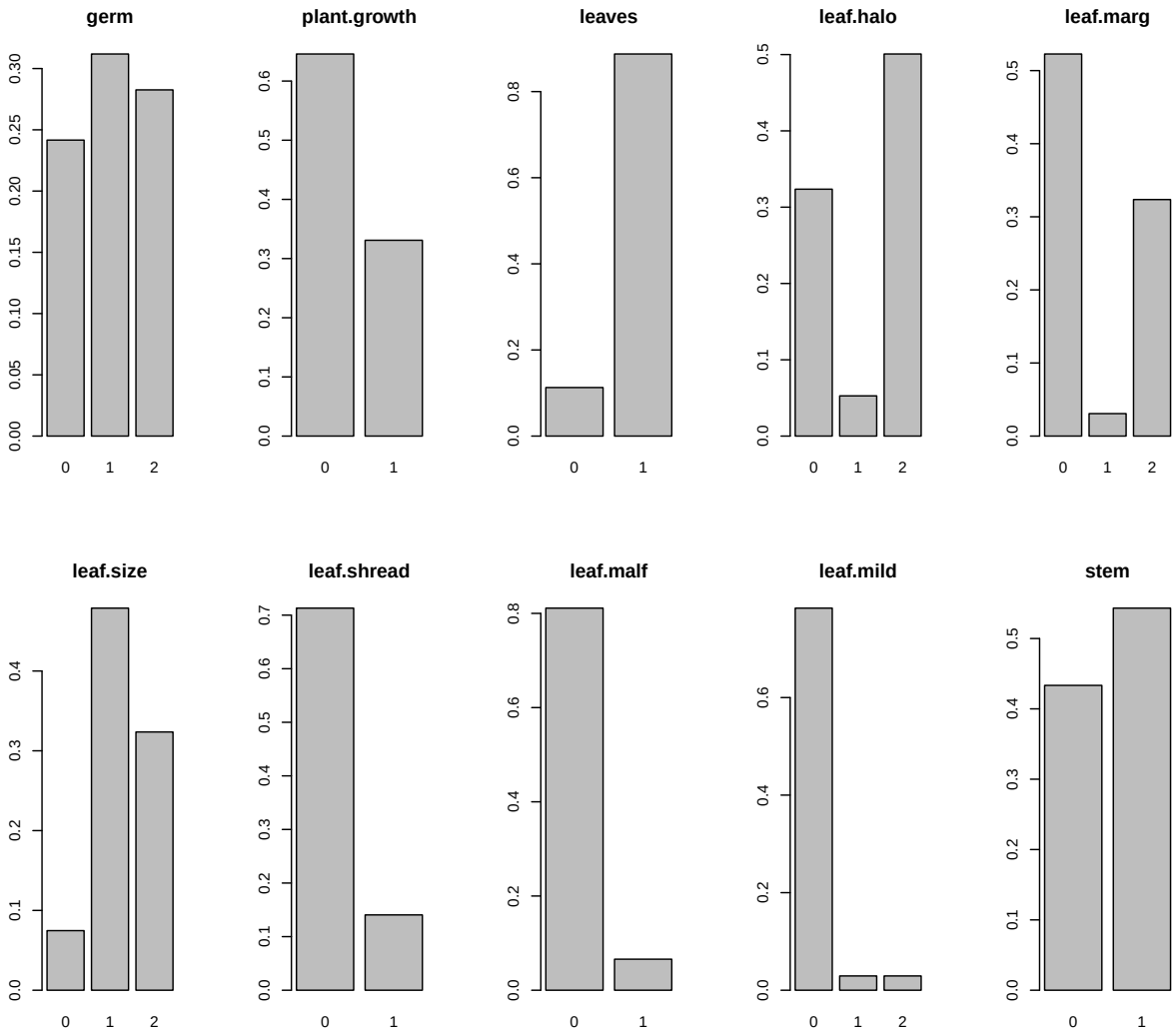

```
MgTrans <- BoxCoxTrans(glass_red$Mg)
KTrans <- BoxCoxTrans(glass_red$K)
BaTrans <- BoxCoxTrans(glass_red$Ba)
FeTrans <- BoxCoxTrans(glass_red$Fe)

#For Mg,K,B,Fe lambda could could not be estimated.
```

(3.2) (a) Frequency distribution of the data can be represented in histograms as below.

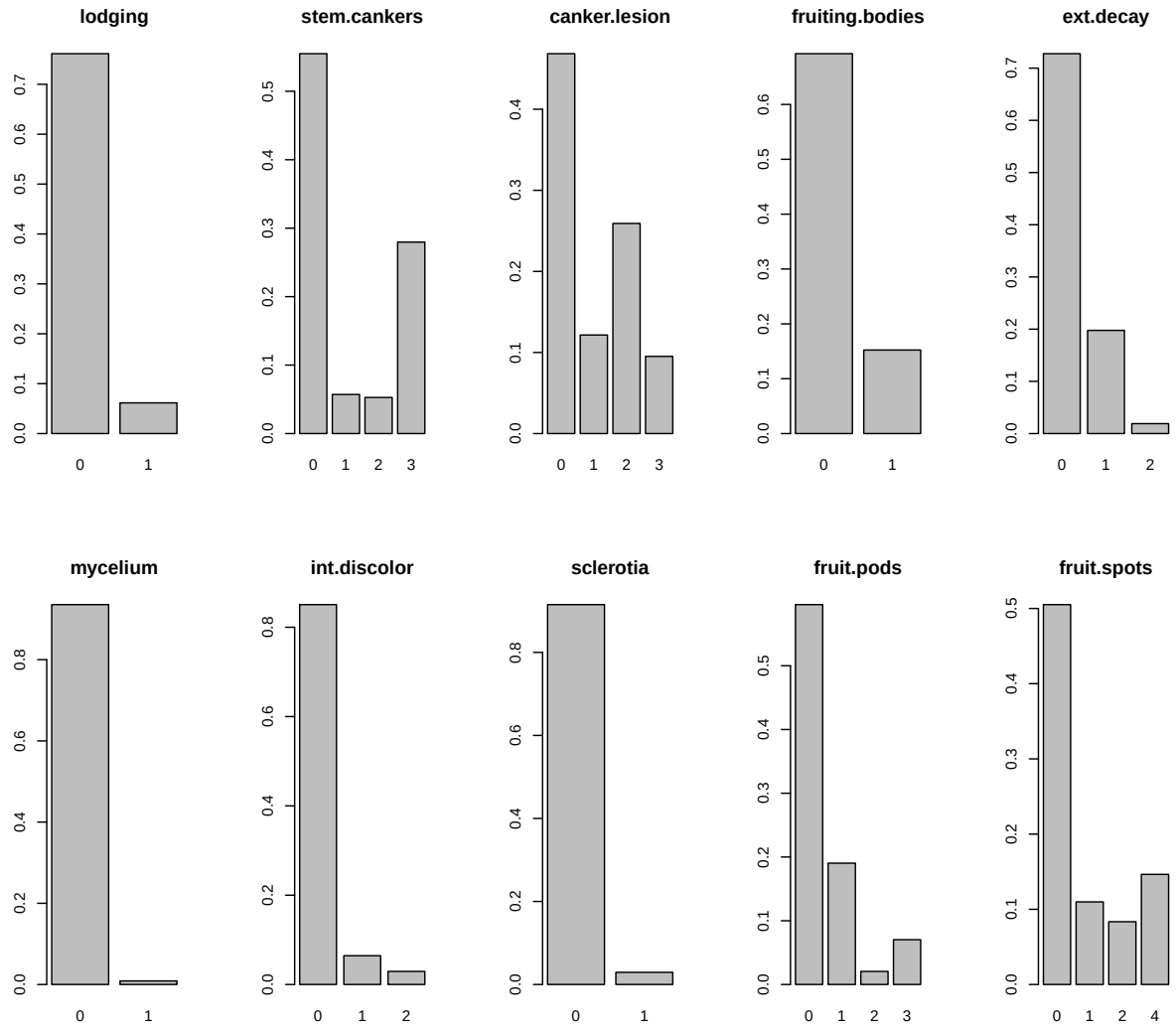
```
rm(list=ls())
library(mlbench)
data(Soybean)
par(mfrow=c(2,5), pch=16)
for (i in 2:10 ) {
barplot(table(Soybean[i])/683,main=names(Soybean)[i])
}
par(mfrow=c(2,5), pch=16)
```



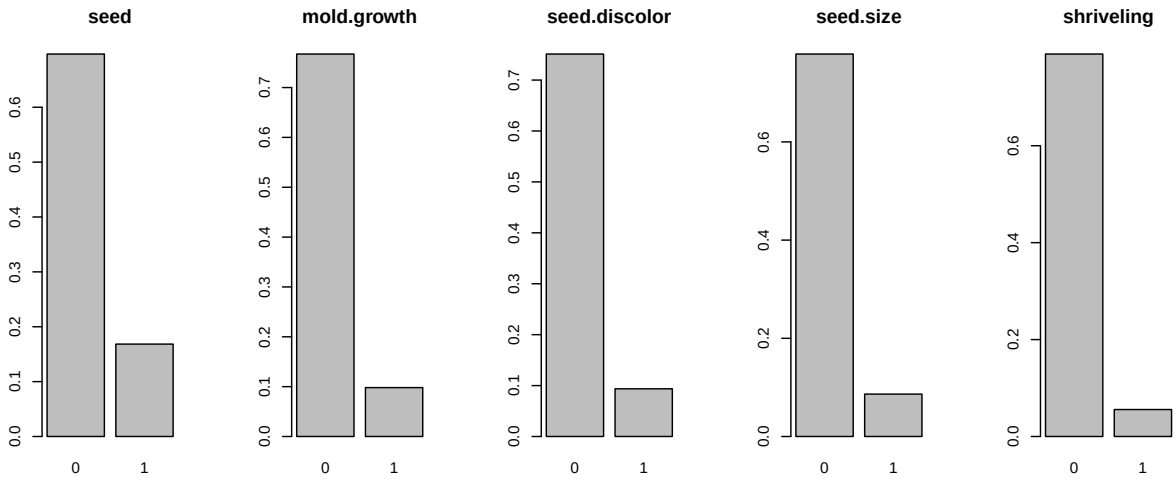

```
for (i in 11:20 ) {
barplot(table(Soybean[i])/683,main=names(Soybean)[i])
}
```

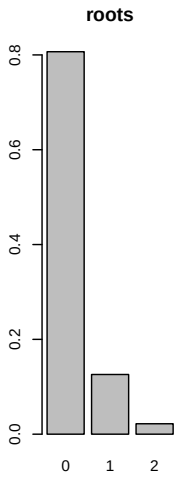



```
par(mfrow=c(2,5), pch=16)
for (i in 21:30 ) {
barplot(table(Soybean[i])/683,main=names(Soybean)[i])
}
```




```
par(mfrow=c(2,5), pch=16)
for (i in 31:36 ) {
barplot(table(Soybean[i])/683,main=names(Soybean)[i])
}
```





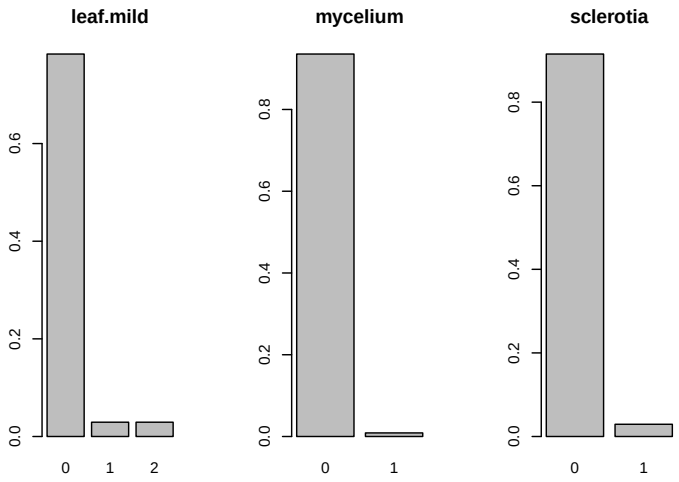
"nearZerovar" function can be used to determine the degenaracy of the predictors. After applying the function , following three predictors were found to be degenerate.

```
library(caret)
removeColumns <-nearZeroVar(Soybean)
names(Soybean)[removeColumns]
```

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

From following frequency distributions it is clear that number of one catagory is large compared to other.

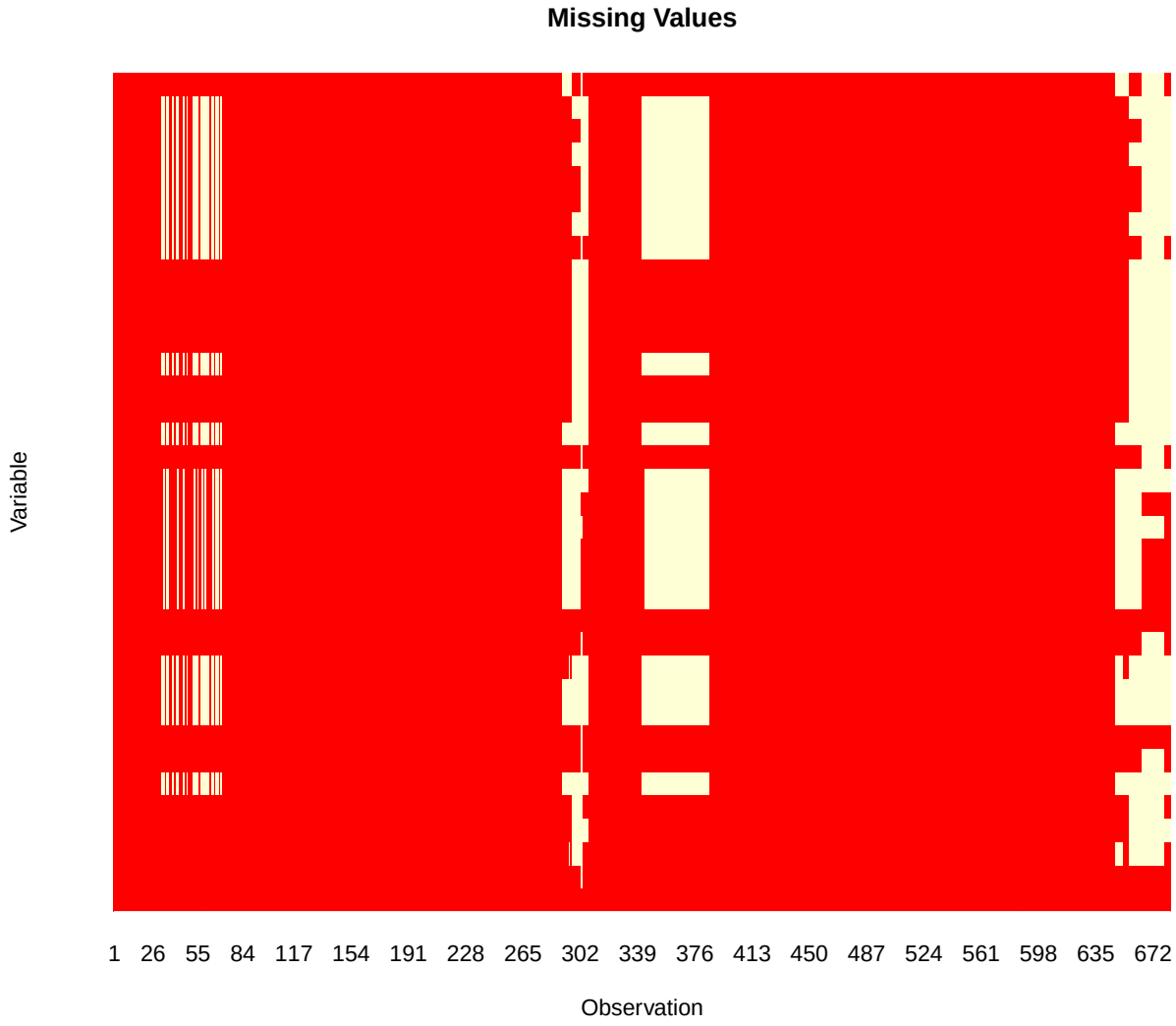
```
par(mfrow=c(2,5), pch=16)
barplot(table(Soybean$leaf.mild)/683,main="leaf.mild")
barplot(table(Soybean$mycelium)/683,main="mycelium")
barplot(table(Soybean$sclerotia)/683,main="sclerotia")
```



“leaf.mild” “mycelium” “sclerotia” these predictors have near zero varience so its possible to remove these and it may be advantageous to remove the variable from the model.

(b)

```
image(is.na(Soybean), main = "Missing Values", xlab = "Observation", ylab = "Variable",
    xaxt = "n", yaxt = "n", bty = "n")
axis(1, seq(0, 1, length.out = nrow(Soybean)), 1:nrow(Soybean), col = "white")
```



yes there are some predictors which is more likely to be missing. Data related to leaf are not avaialble most of the time. Maybe because that specifice class of data is rare so it not feasible to do find on them.

yes. There seems to be pattern in missing data. From observing the data set, some classes have many not available data points.

(c) 1 . *To deal with missing data the method of elemination of all the observations which has missing values. the omit function will eliminate all data points with missing values.*

```
#na.omit() function is used to delete all the data points with missing values.
Soybean_del <- na.omit(Soybean)
summary(Soybean_del)
```

```
##                  Class     date    plant.stand precip  temp    hail
##  brown-spot         : 92   0: 19   0:347       0: 74   0: 72   0:435
##  alternarialeaf-spot: 91   1: 51   1:215       1: 84   1:334   1:127
##  frog-eye-leaf-spot : 91   2: 66               2:404   2:156
##  anthracnose        : 44   3: 86
##  brown-stem-rot     : 44   4:118
```

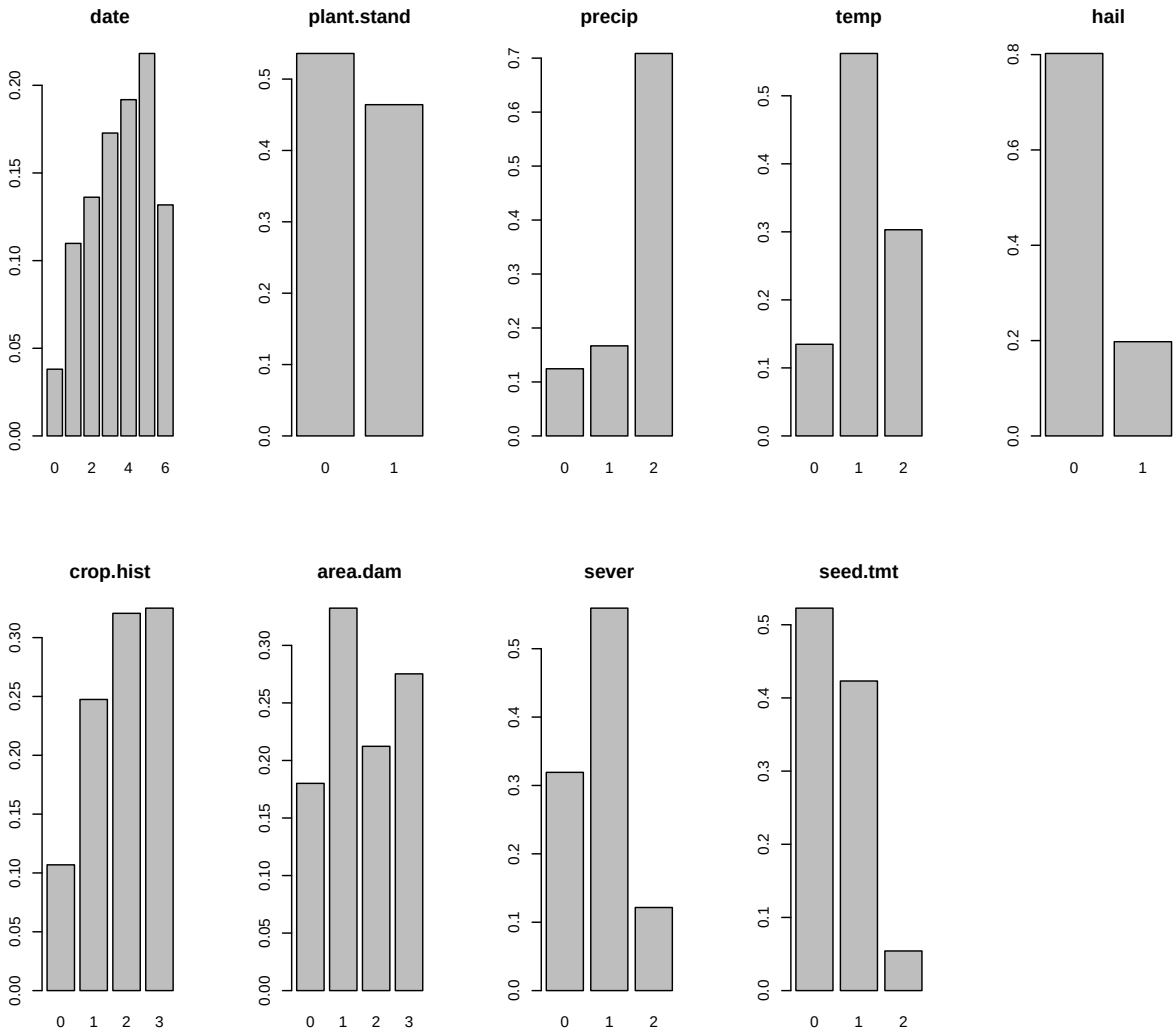
```
##  bacterial-blight   : 20   5:140
##  (Other)            :180   6: 82
##  crop.hist area.dam sever   seed.tmt germ    plant.growth leaves
##  0: 55     0:113    0:195   0:305    0:160   0:426        0: 62
##  1:142     1:147    1:322   1:222    1:211   1:136        1:500
##  2:182     2:135    2: 45   2: 35    2:191
##  3:183     3:167
##
##
##
##  leaf.halo leaf.marg leaf.size leaf.shread leaf.malf leaf.mild stem
##  0:188     0:357     0: 51     0:466       0:541     0:522     0:282
##  1: 36     1: 17     1:323     1: 96       1: 21     1: 20     1:280
##  2:338     2:188     2:188                           2: 20
##
##
##
##
##  lodging stem.cankers canker.lesion fruiting.bodies ext.decay mycelium
##  0:520   0:358        0:305         0:473           0:427     0:556
##  1: 42   1: 30        1: 83         1: 89           1:135     1:  6
##          2: 16        2:109                         2:  0
##          3:158        3: 65
##
##
##
##  int.discolor sclerotia fruit.pods fruit.spots seed    mold.growth
##  0:498        0:542     0:407      0:345       0:473   0:510
##  1: 44        1: 20     1:115      1: 75       1: 89   1: 52
##  2: 20                  2:  0      2: 42
##                         3: 40      4:100
##
##
##
##  seed.discolor seed.size shriveling roots
##  0:513         0:532     0:539      0:551
##  1: 49         1: 30     1: 23      1: 10
##                                     2:  1
##
##
##
##
```

2. *Missing Value Imputation with kNN*

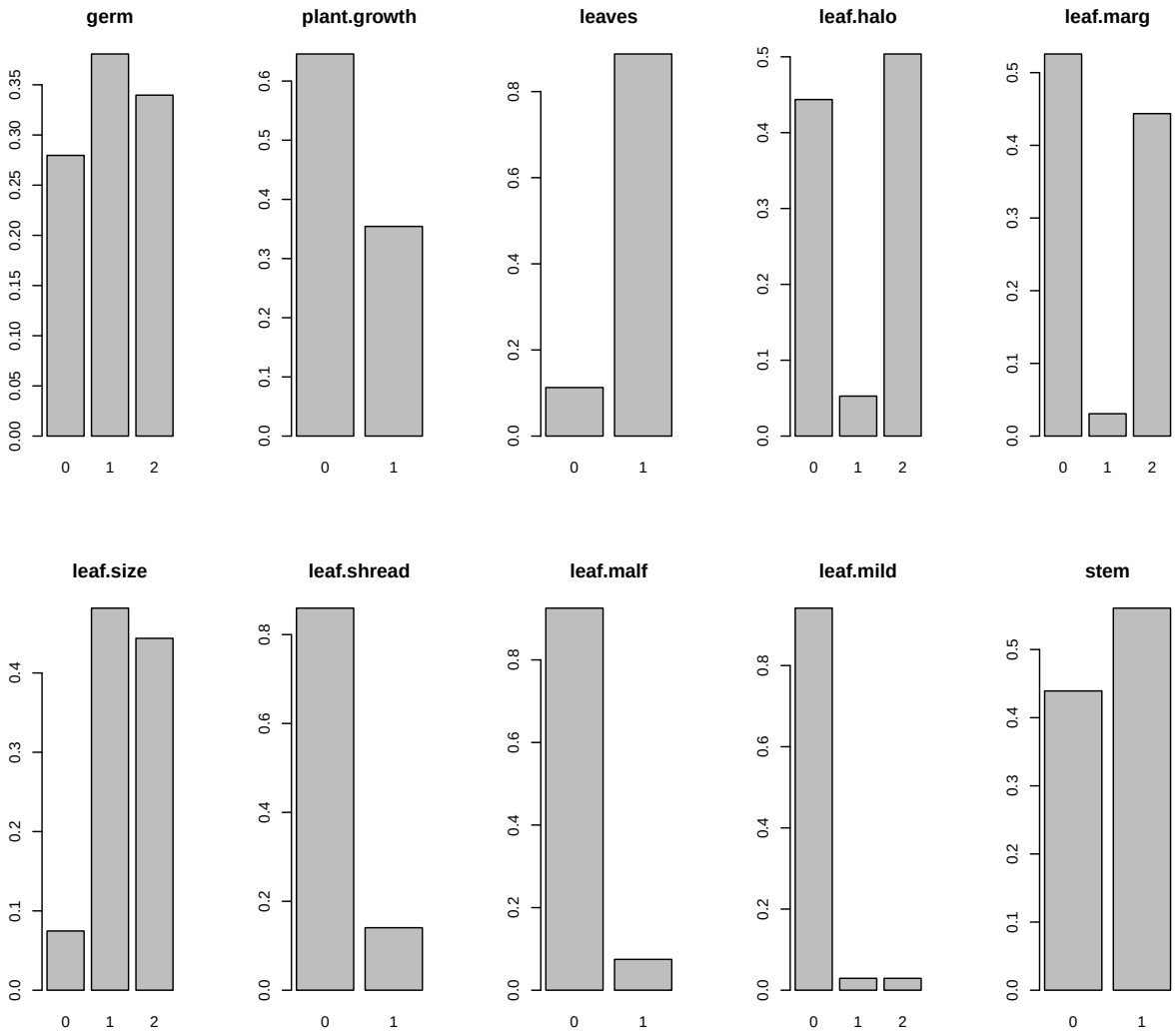
```
library(scrime)
#Imputes missing values in a matrix composed of categorical variables using k Nearest Neighbors. "scrim
Soybean_mat <- data.matrix(Soybean[3:36])#convert to data frame to matrix (only columns 3 to 36 )

Soybean_imp <- knncatimpute(Soybean_mat,nn = 5, dist = "cohen") #imputation function
Soybean_imp <- Soybean_imp-1 # the imputation function replace 0 with 1 so after imputation 1 is reduce
Soybean_imp <- as.data.frame(Soybean_imp)#convert back to a data frame
Soybean_imp <-cbind(Soybean[1:2],Soybean_imp[1:34]) # combined with class names.
```

Visualize data after imputation

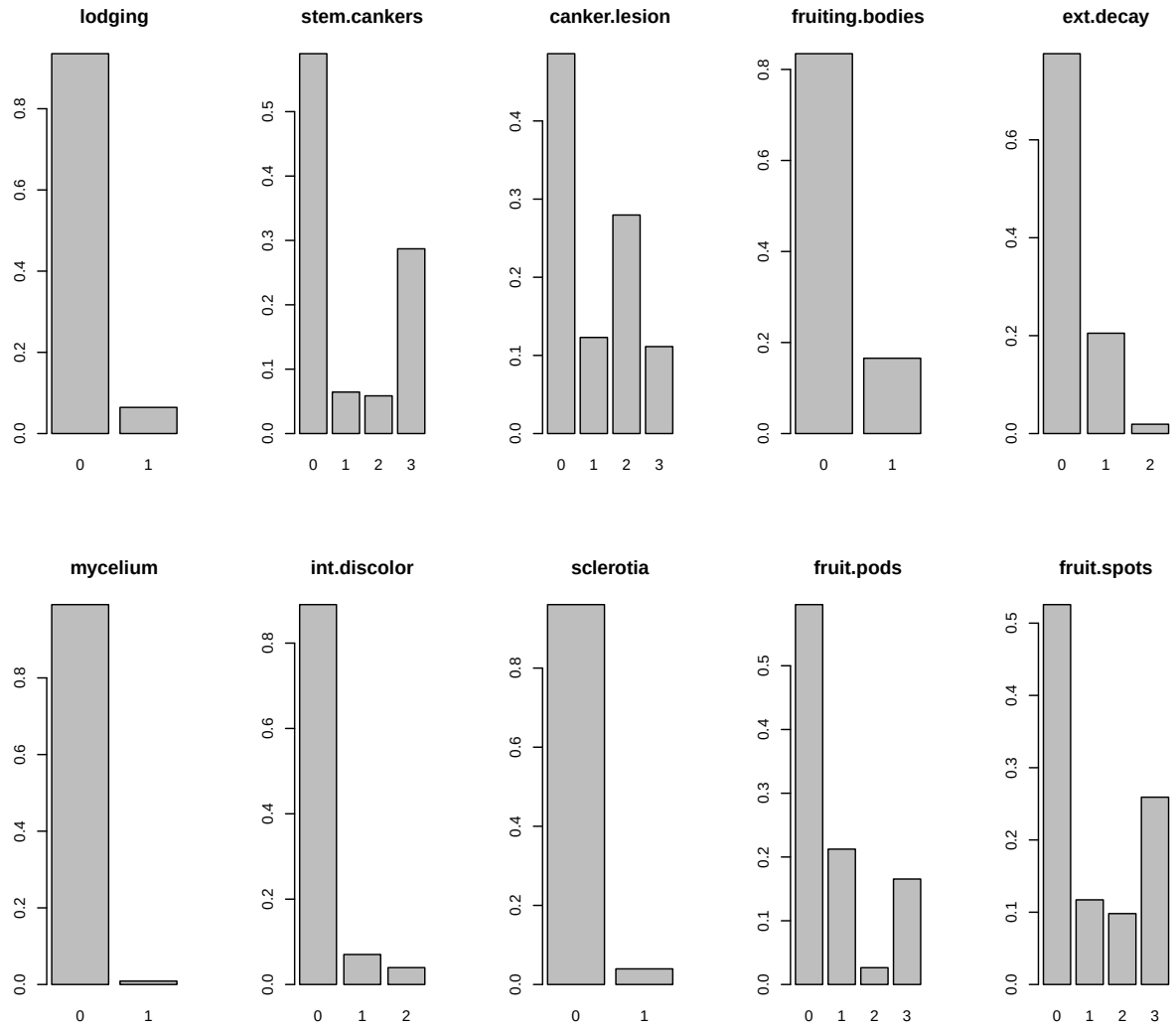
```
par(mfrow=c(2,5), pch=16)
for (i in 2:10 ) {
barplot(table(Soybean_imp[i])/683,main=names(Soybean_imp)[i])
}
par(mfrow=c(2,5), pch=16)
```



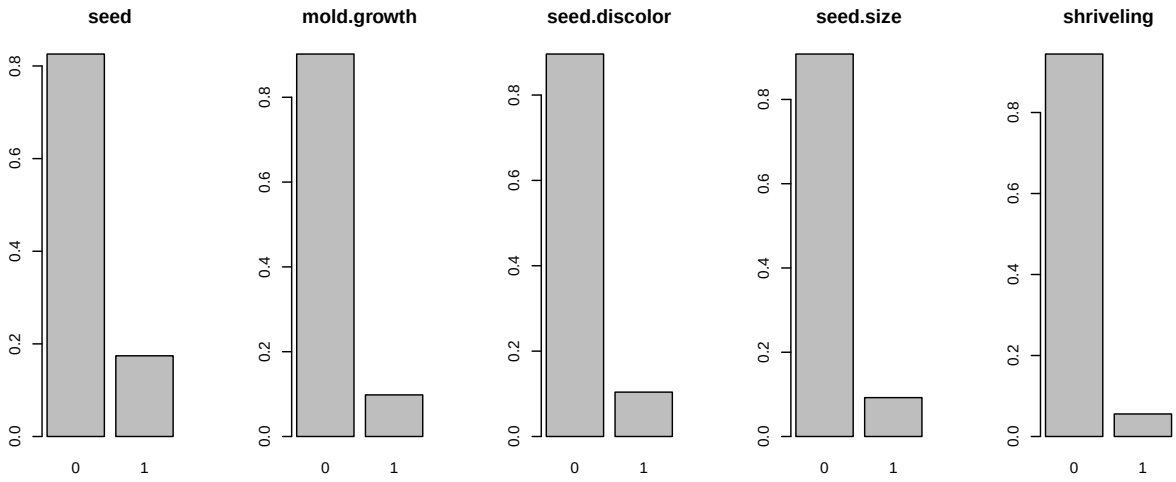

```
for (i in 11:20 ) {
barplot(table(Soybean_imp[i])/683,main=names(Soybean_imp)[i])
}
```

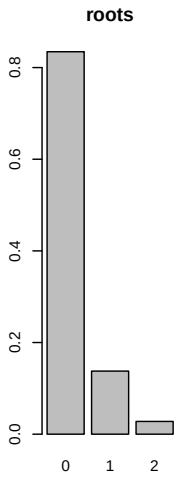



```
par(mfrow=c(2,5), pch=16)
for (i in 21:30 ) {
barplot(table(Soybean_imp[i])/683,main=names(Soybean_imp)[i])
}
```




```
par(mfrow=c(2,5), pch=16)
for (i in 31:36 ) {
barplot(table(Soybean_imp[i])/683,main=names(Soybean_imp)[i])
}
```





3.3) a.

```
library(caret)
data(BloodBrain)
```

b. A degenerate distribution is the probability distribution of a discrete random variable whose support consists of only one value. Examples include a two-headed coin and rolling a die whose sides all show the same number. While this distribution does not appear random in the everyday sense of the word, it does satisfy the definition of random variable. variance should be zero or very close to zero

"nearZerovar" function can be used to determine the degenaracy of the predictors. After applying the function , following three predictors can be treated as degenerate.

```
library(caret)

removeColumns <-nearZeroVar(bbbDescr)
names(bbbDescr)[removeColumns]
```

```
## [1] "negative"     "peoe_vsa.2.1" "peoe_vsa.3.1" "a_acid"
## [5] "vsa_acid"     "frac.anion7." "alert"
```

c.

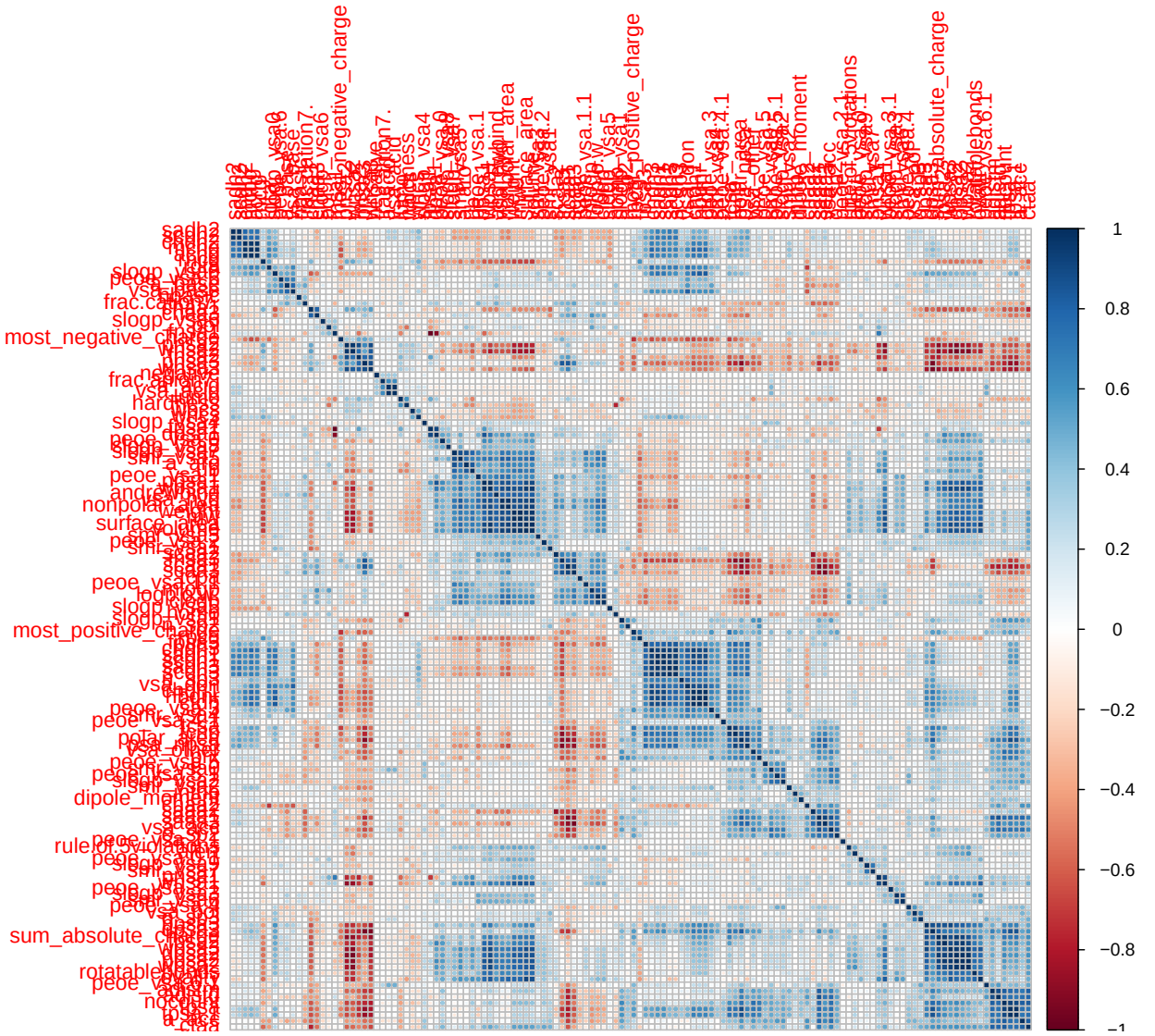
```
corrrelation <- cor(bbbDescr)
dim(corrrelation)
```

```
## [1] 134 134
```

```
corrrelation[1:4, 1:4]
```

```
##              tpsa   nbasic negative  vsa_hyd
## tpsa      1.00000 -0.03825  0.03464 -0.09954
## nbasic   -0.03825  1.00000  0.09160  0.13445
## negative  0.03464  0.09160  1.00000 -0.03898
## vsa_hyd  -0.09954  0.13445 -0.03898  1.00000
```

```
library(corrplot)
corrplot(corrrelation, order = "hclust")
```



Yes , Theare are high correlations between some predictors based on the correlation plot shows a correlation matrix of the training set. Each pairwise correlation is computed from the training data and colored according to its magnitude. This visualization is symmetric: the top and bottom diagonals show identical information. Dark blue colors indicate strong positive correlations, dark red is used for strong negative correlations, and white implies no empirical relationship between the predictors. In this figure, the predictor variables have been grouped using a clustering technique (Everitt et al. 2011) so that collinear groups of predictors are adjacent to one another. Looking along the diagonal, there are blocks of strong positive correlations that indicate "clusters" of collinearity. Near the center of the diagonal is a large block of predictors from the first channel. This function searches through a correlation matrix and returns a vector of integers corresponding to columns to remove to reduce pair-wise correlations.

```
highCorr <- findCorrelation(corrrelation, cutoff = .75,verbose=FALSE)

length(highCorr)
```

```
## [1] 66
```

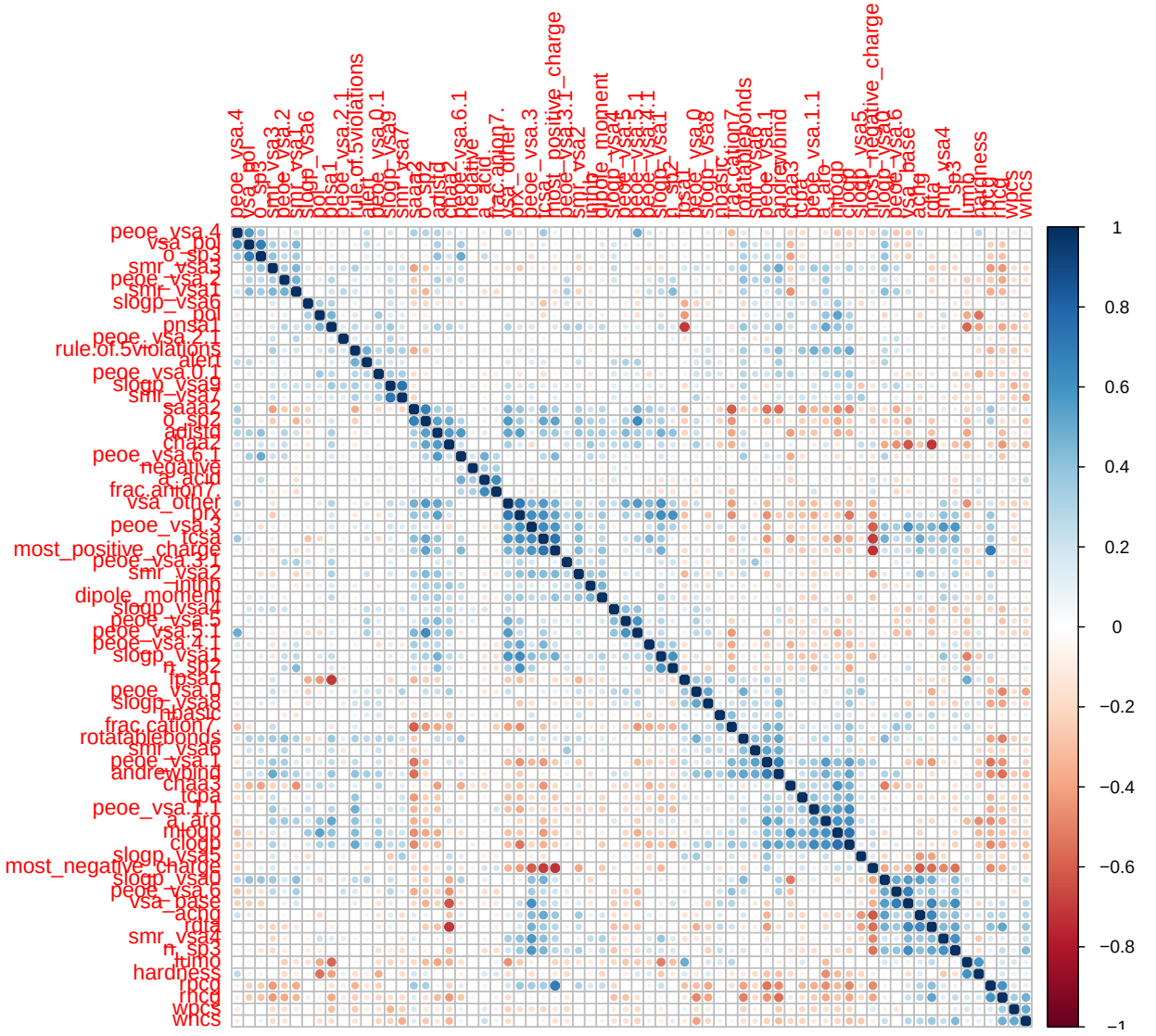
```
head(highCorr)
```

```
## [1]   1  48  67 100 103  78
```

```
#removed these predictoros from data set
filteredbbbDescr <- bbbDescr[, -highCorr]
dim(filteredbbbDescr)
```

```
## [1] 208  68
```

```
#after removing redo the correlation plot
corrrelation <- cor(filteredbbbDescr)
corrplot(corrrelation, order = "hclust")
```



Yes, it had a dramatic effect on number of predictors as if removed about half of the original predictors.

principal components is another technique for mitigating the effect of strong correlations between predictors. However, these techniques make the connection between the predictors and the outcome more complex.