HW1 R Markdown

Santanu Mukherjee

02/20/2022

R Markdown

Chapter 3 - E-Book - Applied Predictive Modelling - Chapter 3 - Exercises page 58:

Q1

3.1. The UC Irvine Machine Learning Repository6 contains a data set related to glass identification. The data consist of 214 glass samples labeled as one of seven class categories. There are nine predictors, including the refractive index and percentages of eight elements: Na, Mg, Al, Si, K, Ca, Ba, and Fe. The data can be accessed via:

```r
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 ...
```

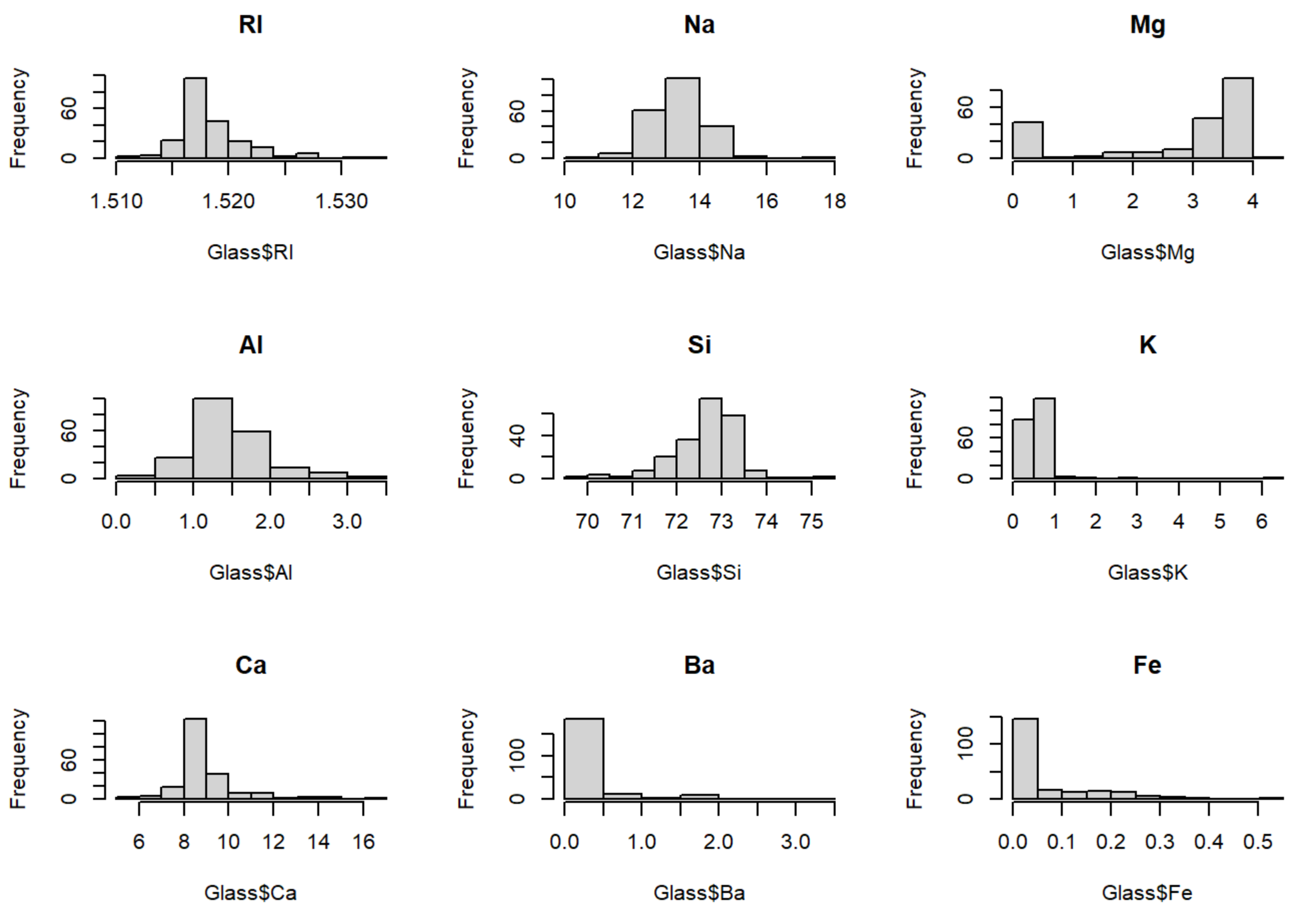
Q 3.1a

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

Answer 3.1 (a)

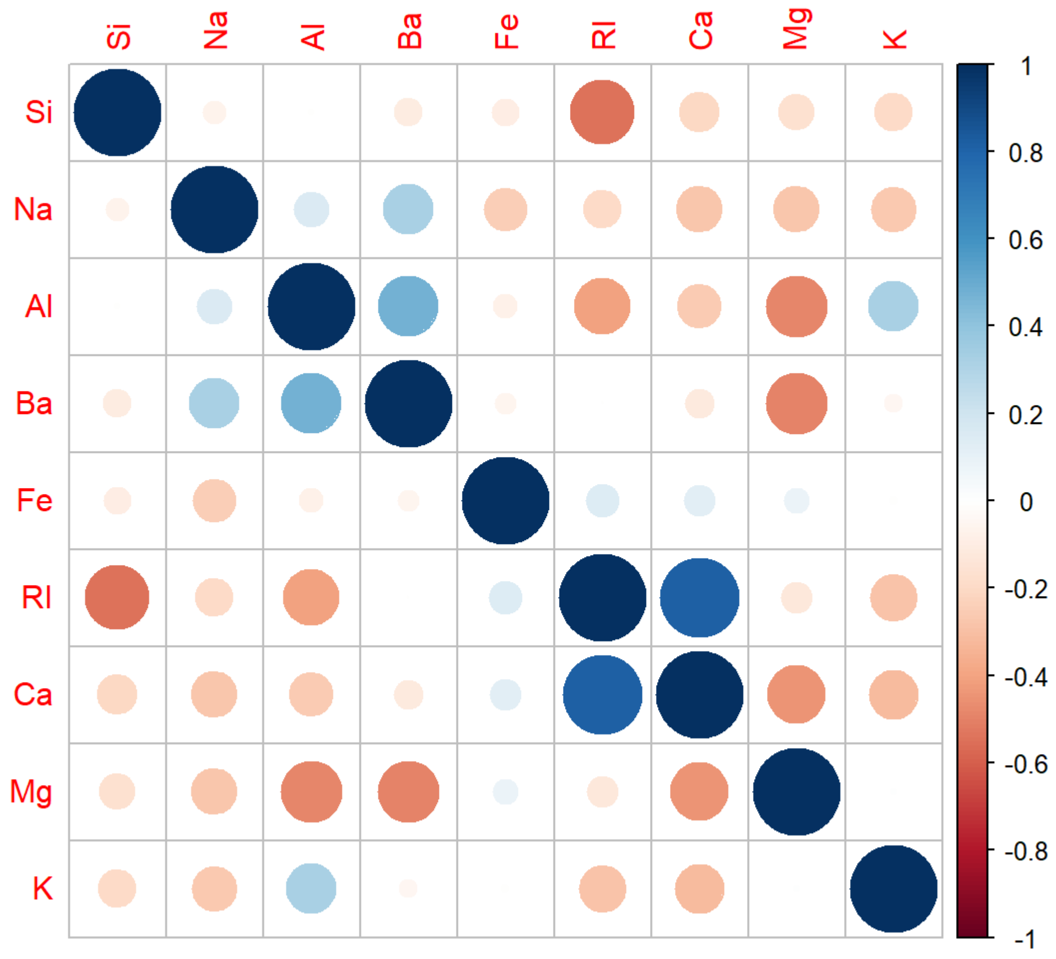
The histograms are used to visualize the distribution of each predictor and the correlation matrix plot to see the relationships between predictors.

```r
par(mfrow=c(3,3))
# Histograms
hist(Glass$RI, main="RI")
hist(Glass$Na, main="Na")
hist(Glass$Mg, main="Mg")
hist(Glass$Al, main="Al")
hist(Glass$Si, main="Si")
hist(Glass$K, main="K")
hist(Glass$Ca, main="Ca")
hist(Glass$Ba, main="Ba")
hist(Glass$Fe, main="Fe")
```

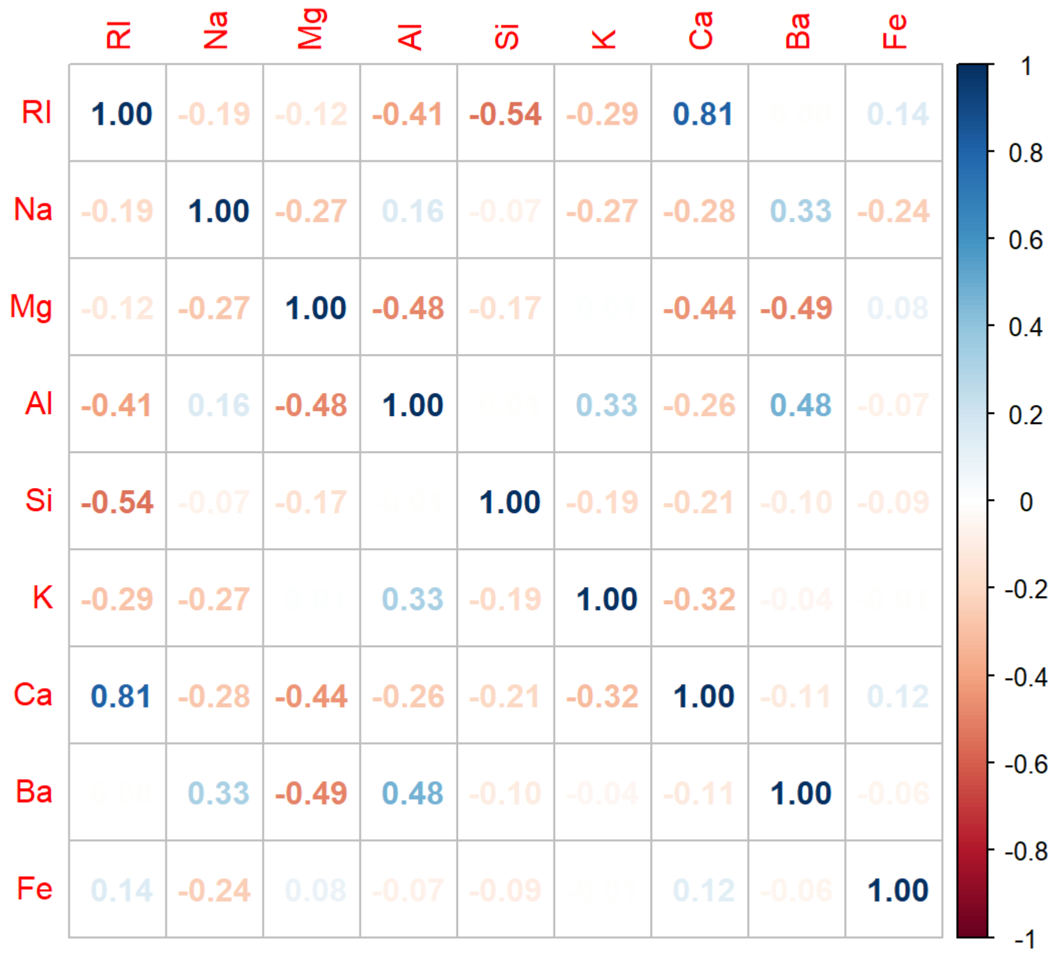


From the above histograms, we can see that some predictors are heavily skewed, such as Mg, K, Ba, and Fe. We will further confirm our visualizations about the skewness of each predictor in part (b). In addition, we use the correlation matrix to check the relationship between predictors shown below.

```r
#Correlation plot
Cor = round(cor(Glass[,1:9]), 4)
library(corrplot)
corrplot(Cor, order = "hclust")
```



```
corrplot(Cor, method="number")
```



```
highCorr <- findCorrelation(Cor, cutoff = .75)
head(highCorr)
```

```
## [1] 7
```

It can be observed that the predictors Ca and $R1$ are highly correlated with a correlation value of 0.81. By using $findCorrelation$ in R, we found out that the predictor Ca may be removed to reduce pair-wise correlation with a threshold of 0.75, as recommended by the textbook.

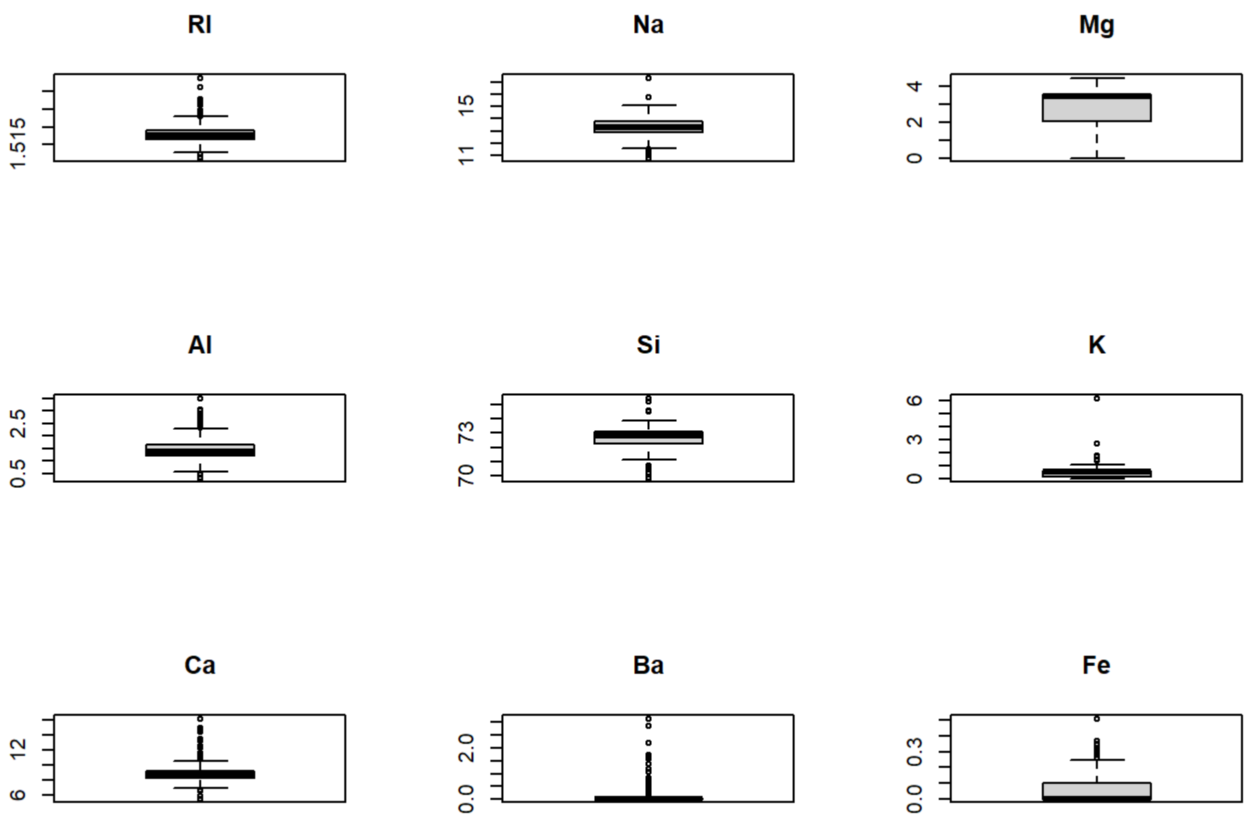
Q 3.1b

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

Answer 3.1 (b)

b. By drawing the boxplot of each predictor shown below, we can draw some conclusions as follows: (i) there are several predictors having a couple of potential outliers, such as RI, Ca, Ba (ii) we also observe that several predictors are heavily skewed, such as Mg, K, Ba, and Fe, which matches the findings from the histogram. So, it should be beneficial to employ the data transformation technique to resolve the outlier issues in the predictors.

```
# Boxplot
par(mfrow=c(3,3))
boxplot(Glass$RI, main="RI")
boxplot(Glass$Na, main="Na")
boxplot(Glass$Mg, main="Mg")
boxplot(Glass$Al, main="Al")
boxplot(Glass$Si, main="Si")
boxplot(Glass$K, main="K")
boxplot(Glass$Ca, main="Ca")
boxplot(Glass$Ba, main="Ba")
boxplot(Glass$Fe, main="Fe")
```



In addition, we calculated the skewness of each predictor in the following table. The value Skewness can be used as a guide to prioritize the transformation of the predictor. We here consider the predictor to be nearly symmetric if the value falls between -0.5 and 0.5, moderately skewed if the absolute value falls between 0.5 and 1, and heavily skewed, otherwise.

Q 3.1c

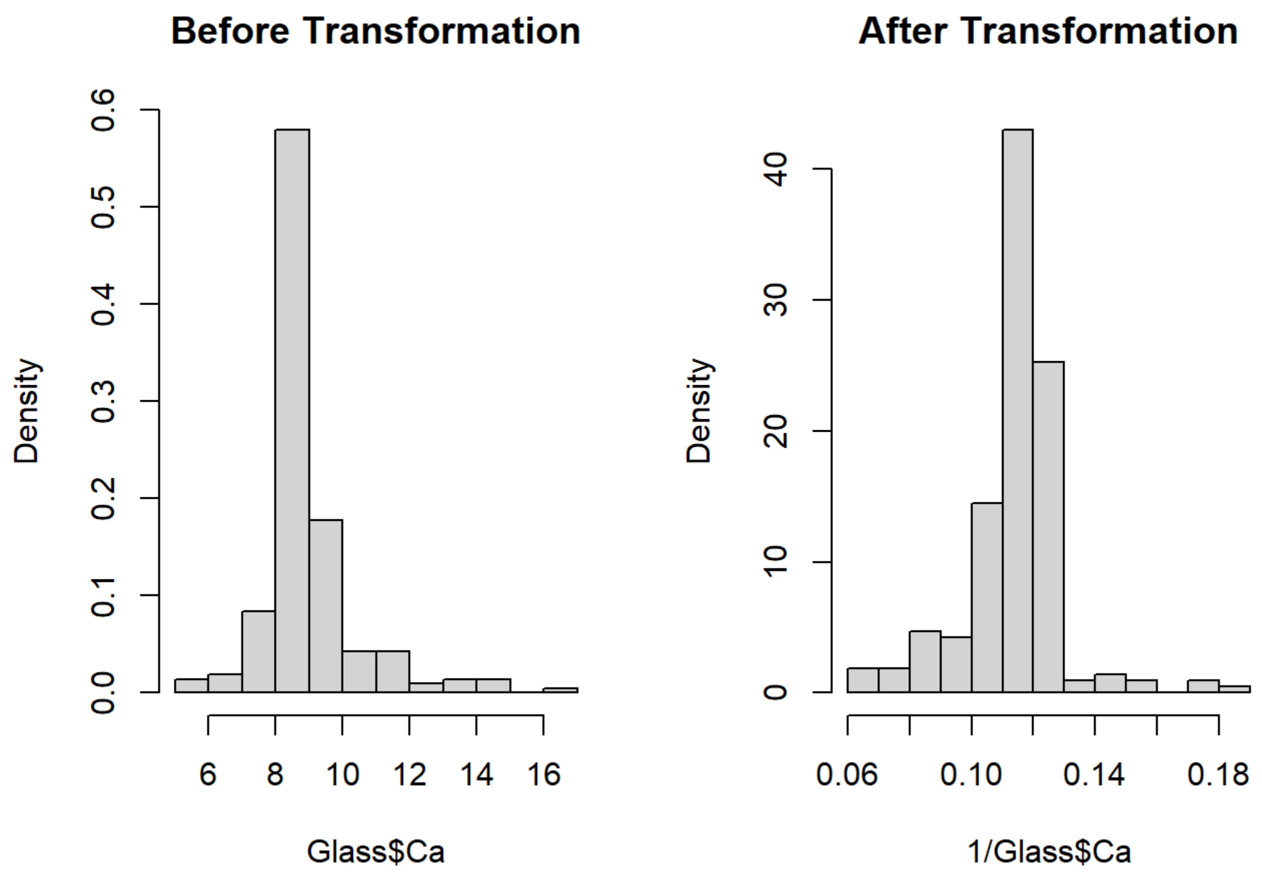
c. Are there any relevant transformations of one or more predictors that might improve the classification model?

Answer 3.1 (c)

```
# Skewness
skewValue = apply(Glass[,1:9], 2, skewness)
skewValue
```

```
##         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
```

```
par(mfrow=c(1,2))
hist(Glass$Ca, prob=T,main="Before Transformation")
hist(1/Glass$Ca, prob=T,main="After Transformation")
```



We observe from this table that the data transformation techniques, such as the Box-Cox transformation, may be required to resolve skewness.

Chapter 3 - E-Book - Applied Predictive Modelling - Chapter 3 - Exercises page 58-59:

Q 3.2

3.2. 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:

```
library(mlbench)
data(Soybean)
str(Soybean)
```

```
## 'data.frame':    683 obs. of  36 variables:
##  $ Class          : Factor w/ 19 levels "2-4-d-injury",..: 11 11 11 11 11 11 11 11 11 11 ...
##  $ date           : Factor w/ 7 levels "0","1","2","3",..: 7 5 4 4 7 6 6 5 7 5 ...
##  $ plant.stand    : Ord.factor w/ 2 levels "0"<"1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ precip         : Ord.factor w/ 3 levels "0"<"1"<"2": 3 3 3 3 3 3 3 3 3 3 ...
##  $ temp           : Ord.factor w/ 3 levels "0"<"1"<"2": 2 2 2 2 2 2 2 2 2 2 ...
##  $ hail           : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 2 1 1 ...
##  $ crop.hist      : Factor w/ 4 levels "0","1","2","3": 2 3 2 2 3 4 3 2 4 3 ...
##  $ area.dam       : Factor w/ 4 levels "0","1","2","3": 2 1 1 1 1 1 1 1 1 1 ...
##  $ sever          : Factor w/ 3 levels "0","1","2": 2 3 3 3 2 2 2 2 2 3 ...
##  $ seed.tmt       : Factor w/ 3 levels "0","1","2": 1 2 2 1 1 1 2 1 2 1 ...
##  $ germ           : Ord.factor w/ 3 levels "0"<"1"<"2": 1 2 3 2 3 2 1 3 2 3 ...
##  $ plant.growth   : Factor w/ 2 levels "0","1": 2 2 2 2 2 2 2 2 2 2 ...
##  $ leaves         : Factor w/ 2 levels "0","1": 2 2 2 2 2 2 2 2 2 2 ...
##  $ leaf.halo      : Factor w/ 3 levels "0","1","2": 1 1 1 1 1 1 1 1 1 1 ...
##  $ leaf.marg      : Factor w/ 3 levels "0","1","2": 3 3 3 3 3 3 3 3 3 3 ...
##  $ leaf.size      : Ord.factor w/ 3 levels "0"<"1"<"2": 3 3 3 3 3 3 3 3 3 3 ...
##  $ leaf.shread    : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ leaf.malf      : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ leaf.mild      : Factor w/ 3 levels "0","1","2": 1 1 1 1 1 1 1 1 1 1 ...
##  $ stem           : Factor w/ 2 levels "0","1": 2 2 2 2 2 2 2 2 2 2 ...
##  $ lodging        : Factor w/ 2 levels "0","1": 2 1 1 1 1 1 2 1 1 1 ...
##  $ stem.cankers   : Factor w/ 4 levels "0","1","2","3": 4 4 4 4 4 4 4 4 4 4 ...
##  $ canker.lesion  : Factor w/ 4 levels "0","1","2","3": 2 2 1 1 2 1 2 2 2 2 ...
##  $ fruiting.bodies: Factor w/ 2 levels "0","1": 2 2 2 2 2 2 2 2 2 2 ...
##  $ ext.decay      : Factor w/ 3 levels "0","1","2": 2 2 2 2 2 2 2 2 2 2 ...
##  $ mycelium       : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ int.discolor   : Factor w/ 3 levels "0","1","2": 1 1 1 1 1 1 1 1 1 1 ...
##  $ sclerotia      : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ fruit.pods     : Factor w/ 4 levels "0","1","2","3": 1 1 1 1 1 1 1 1 1 1 ...
##  $ fruit.spots    : Factor w/ 4 levels "0","1","2","4": 4 4 4 4 4 4 4 4 4 4 ...
##  $ seed           : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ mold.growth    : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ seed.discolor  : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ seed.size      : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ shriveling     : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ roots          : Factor w/ 3 levels "0","1","2": 1 1 1 1 1 1 1 1 1 1 ...
```

```
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
```

Q3.2 a

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

Answer 3.2 (a)

The variable “Class” is the outcome. The others are predictors. We can inspect degenerated distributions by using the function “Near Zero Variance”.

```
library(caret)

zero_cols = nearZeroVar( Soybean )
colnames( Soybean )[ zero_cols ]
```

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

```
Soybean1 = Soybean[,-zero_cols]
Soybean2 = Soybean[, zero_cols]
head(Soybean1)
```

```
##                  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 stem lodging stem.cankers canker.lesion fruiting.bodies
## 1           0         0    1       1            3             1               1
## 2           0         0    1       0            3             1               1
## 3           0         0    1       0            3             0               1
## 4           0         0    1       0            3             0               1
## 5           0         0    1       0            3             1               1
## 6           0         0    1       0            3             0               1
##   ext.decay int.discolor fruit.pods fruit.spots seed mold.growth seed.discolor
## 1         1            0          0           4    0           0             0
## 2         1            0          0           4    0           0             0
## 3         1            0          0           4    0           0             0
## 4         1            0          0           4    0           0             0
## 5         1            0          0           4    0           0             0
## 6         1            0          0           4    0           0             0
##   seed.size shriveling roots
## 1         0          0     0
## 2         0          0     0
## 3         0          0     0
## 4         0          0     0
## 5         0          0     0
## 6         0          0     0
```

```
head(Soybean2)
```

```
##   leaf.mild mycelium sclerotia
## 1         0        0         0
## 2         0        0         0
## 3         0        0         0
## 4         0        0         0
## 5         0        0         0
## 6         0        0         0
```

The “Near Zero Variance” shows that the presence of the 3 predictors $leaf.mild$, $mycelium$, $sclerotia$ actually degenerate distributions and removal of these 3 predictors would in fact improve model performance.

Q3.2 b

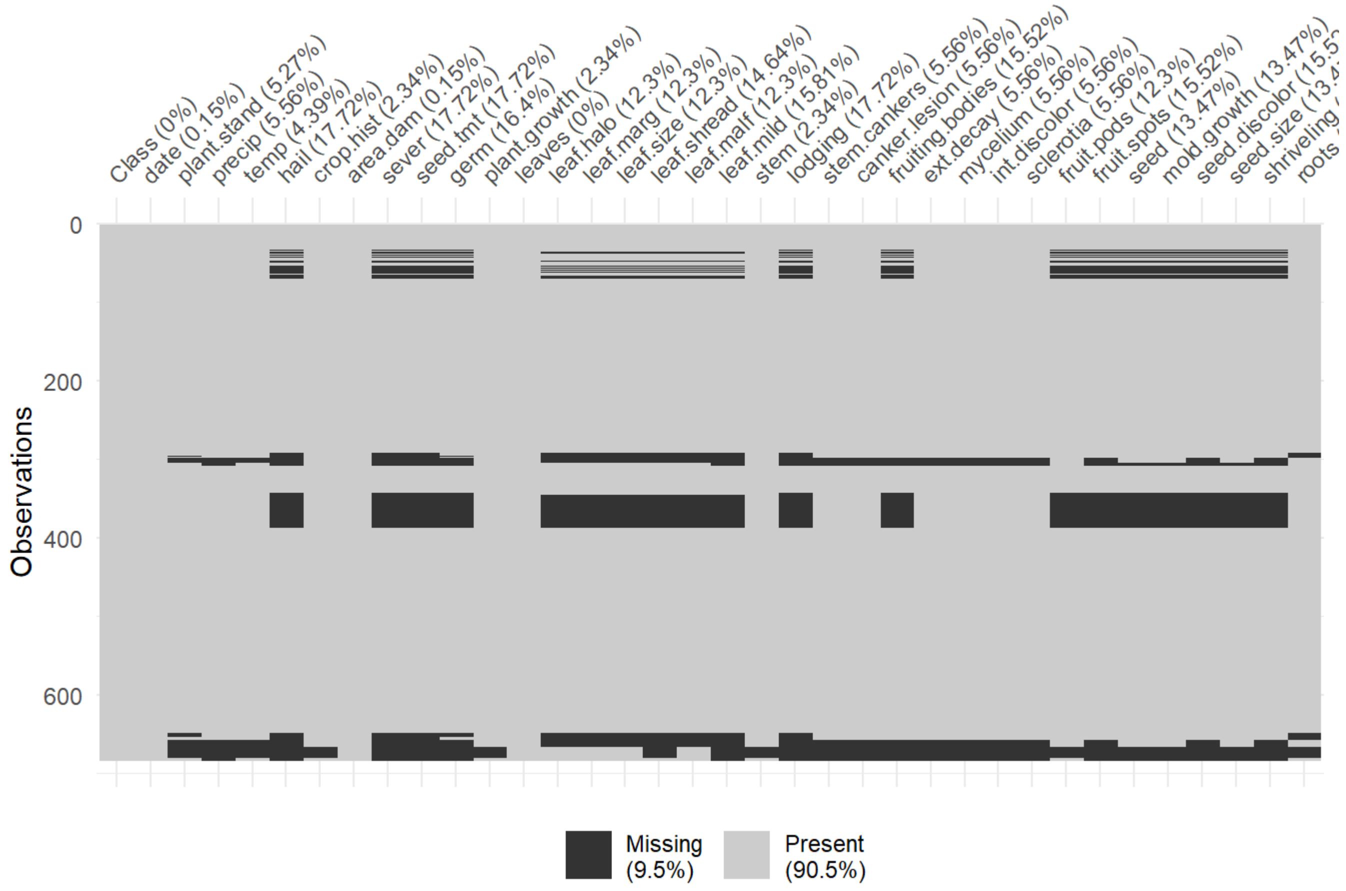
b. Roughly 18% of the data are missing. Are there particular predictors that are more likely to be missing? Is the pattern of missing data related to the classes?

Answer 3.2 (b)

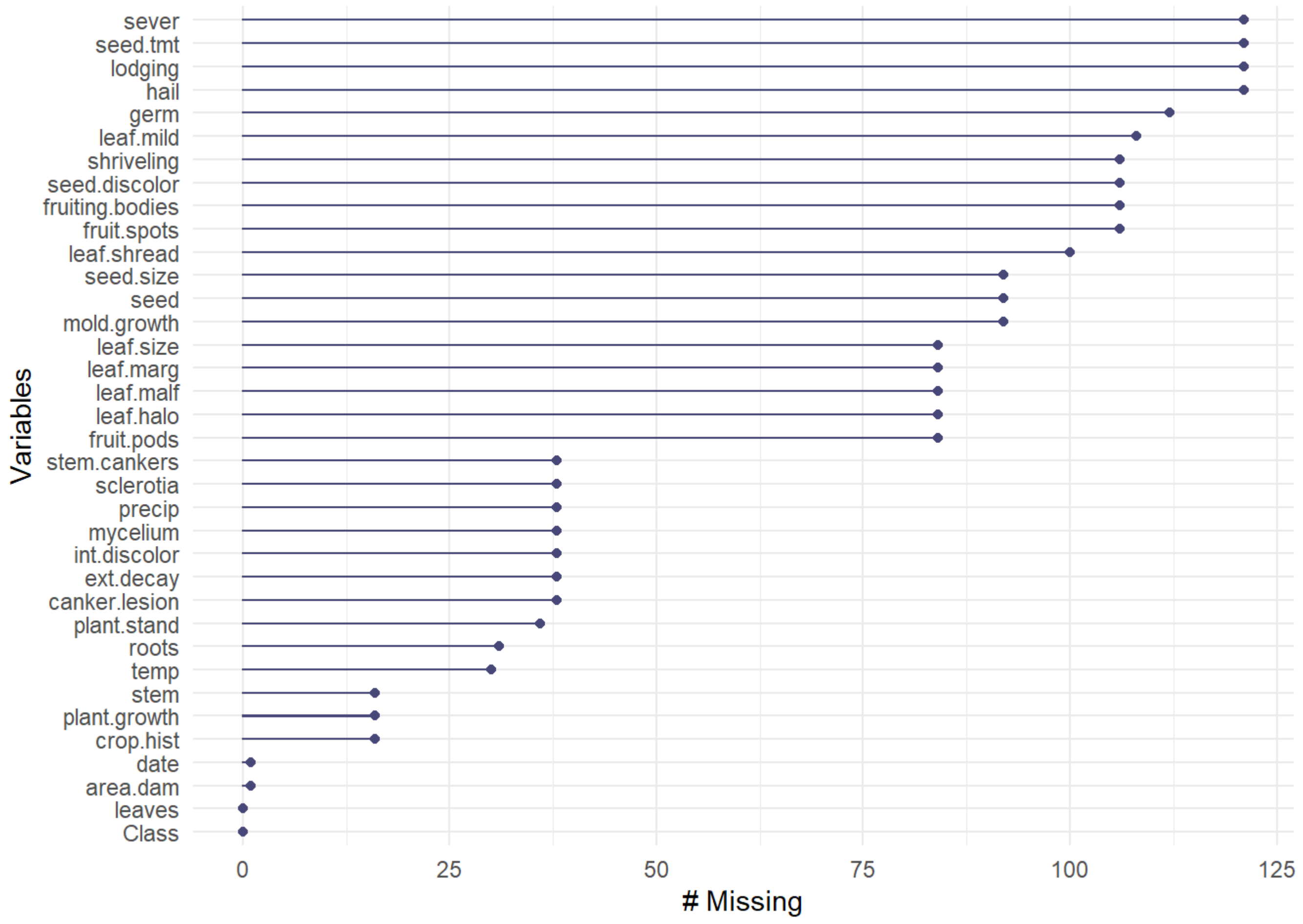
```
#Find missing values
library(naniar)
cbind(c("Number of Mising Values",
        "Number of Complete Values",
        "Proportion of Missing Values",
        "Proportion of Complete Values",
        "Percentage of Missing Values",
        "Percentage of complete Values"),
      rbind(n_miss(Soybean),
            n_complete(Soybean),
            round(prop_miss(Soybean),4),
            round(prop_complete(Soybean),4),
            round(pct_miss(Soybean),2),
            round(pct_complete(Soybean),2)))
```

```
##      [,1]                            [,2]
## [1,] "Number of Mising Values"       "2337"
## [2,] "Number of Complete Values"     "22251"
## [3,] "Proportion of Missing Values"  "0.095"
## [4,] "Proportion of Complete Values" "0.905"
## [5,] "Percentage of Missing Values"  "9.5"
## [6,] "Percentage of complete Values" "90.5"
```

```
vis_miss(Soybean)
```



```
gg_miss_var(Soybean)
```



Based on the visualization, we see that roughly 9.2% of the data are missing. Yes, there are some predictors that have more missing data than others, for example $sever$, $seed.tmt$ and $lodging$ has 121 missing values each.

Q3.2 c

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

Answer 3.2 (c)

```
# For imputation of data for the NA's
#

preProcess(Soybean[,-1], method=c("knnImpute"))
```

```
## Created from 562 samples and 35 variables
##
## Pre-processing:
##   - ignored (35)
```

```
#vis_miss(Soybean)
#gg_miss_var(Soybean)
```

Imputation

The process of estimating missing data points is called Imputation. This can be done in many different ways and it is always true that the best method depends on the problem.

We have seen sometimes people delete any indicator or unit that has missing values, although this is not the right way and in many cases this can be too restrictive. Ideally, one can obtain reasonable results despite small amounts of missing data, but if too much data is missing, the uncertainty can be too high to give a meaningful analysis. As usual, there should be a balance in the way missing data is handled.

Although there are multiple methods of imputation, the indicator method is commonly used.In this method, we can use data from other indicators to estimate the missing point. The core idea here is that if there is relation between indicators, it is very highly possible to guess the missing data points by using known values of other indicators. The form of this can be :

a. Simply substituting the *mean* or *median* of the normalized values of the other indicators.

b. Substituting the *mean or median of normalized values* of the other indicators *within the aggregation group*.

c. Using a more formal approach, based on regression or more generally on statistical modelling.

Chapter 4 - E-Book - Applied Predictive Modelling - Chapter 4 - Exercises page 90-92:

Q 4.4

4.4. Brodnjak-Vonina et al. (2005) develop a methodology for food laboratories to determine the type of oil from a sample. In their procedure, they used a gas chromatograph (an instrument that separate chemicals in a sample) to measure seven different fatty acids in an oil. These measurements would then be used to predict the type of oil in a food samples. To create their model, they used 96 samples2 of seven types of oils. These data can be found in the caret package using data(oil). The oil types are contained in a factor variable called oilType. The types are pumpkin (coded as A), sunflower (B), peanut (C), olive (D), soybean (E), rapeseed (F) and corn (G). In R,

```
data(oil)
str(oilType)
```

```
##  Factor w/ 7 levels "A","B","C","D",..: 1 1 1 1 1 1 1 1 1 1 ...
```

```
table(oilType)
```

```
## oilType
##  A  B  C  D  E  F  G
## 37 26  3  7 11 10  2
```

Q 4.4 a

a. Use the sample function in base R to create a completely random sample of 60 oils. How closely do the frequencies of the random sample match the original samples? Repeat this procedure several times of understand the variation in the sampling process.

Answer 4.4 (a)

```
# Population frequency of oil types:
#
print( table(oilType)/length(oilType) )
```

```
## oilType
##          A          B          C          D          E          F          G 
## 0.38541667 0.27083333 0.03125000 0.07291667 0.11458333 0.10416667 0.02083333
```

```
# Using the sample function in R to draw a completely random sample of 60 oils for 30 times:
#
set.seed(12345)
oilsamples <- vector(mode="list",length = 30)
for ( i in seq(along = oilsamples)){
  oilsamples[[i]] <- table(sample(oilType, size=60))
}
head (oilsamples,5)
```

```
## [[1]]
## 
##  A  B  C  D  E  F  G 
## 24 20  1  4  5  4  2 
## 
## [[2]]
## 
##  A  B  C  D  E  F  G 
## 23 17  1  4  7  8  0 
## 
## [[3]]
## 
##  A  B  C  D  E  F  G 
## 27 14  2  4  6  6  1 
## 
## [[4]]
## 
##  A  B  C  D  E  F  G 
## 25 17  0  4  6  7  1 
## 
## [[5]]
## 
##  A  B  C  D  E  F  G 
## 21 20  2  4  6  6  1
```

```
oilsamples <- do.call("rbind",oilsamples)
head (oilsamples,5)
```

```
##       A  B C D E F G
## [1,] 24 20 1 4 5 4 2
## [2,] 23 17 1 4 7 8 0
## [3,] 27 14 2 4 6 6 1
## [4,] 25 17 0 4 6 7 1
## [5,] 21 20 2 4 6 6 1
```

```
# How do its frequencies compare with that of the population:
#
summary(oilsamples/60)
```

```
##        A                B                C                  D
##  Min.   :0.3167   Min.   :0.2167   Min.   :0.00000   Min.   :0.03333
##  1st Qu.:0.3500   1st Qu.:0.2708   1st Qu.:0.03333   1st Qu.:0.06667
##  Median :0.3833   Median :0.2833   Median :0.03333   Median :0.06667
##  Mean   :0.3783   Mean   :0.2844   Mean   :0.03111   Mean   :0.07500
##  3rd Qu.:0.3958   3rd Qu.:0.3125   3rd Qu.:0.03333   3rd Qu.:0.08333
##  Max.   :0.4500   Max.   :0.3333   Max.   :0.05000   Max.   :0.11667
##        E                 F                 G
##  Min.   :0.06667   Min.   :0.03333   Min.   :0.00000
##  1st Qu.:0.10000   1st Qu.:0.08333   1st Qu.:0.01667
##  Median :0.10833   Median :0.10000   Median :0.01667
##  Mean   :0.11556   Mean   :0.09833   Mean   :0.01722
##  3rd Qu.:0.13333   3rd Qu.:0.11667   3rd Qu.:0.02917
##  Max.   :0.16667   Max.   :0.13333   Max.   :0.03333
```

Q 4.4 b

b. Use the caret package function createDataPartition to create a stratified random sample. How does this compare to the completely random samples?

Answer 4.4 (b)

```
set.seed(12345)
oilsamples2 <- createDataPartition(oilType, p = 0.60, times=20)
oilsamples2 <- lapply(oilsamples2, function(x, y) table(y[x]), y = oilType)
head (oilsamples2,5)
```

```
## $Resample01
##
##  A  B  C  D  E  F  G
## 23 16  2  5  7  6  2
##
## $Resample02
##
##  A  B  C  D  E  F  G
## 23 16  2  5  7  6  2
##
## $Resample03
##
##  A  B  C  D  E  F  G
## 23 16  2  5  7  6  2
##
## $Resample04
##
##  A  B  C  D  E  F  G
## 23 16  2  5  7  6  2
##
## $Resample05
##
##  A  B  C  D  E  F  G
## 23 16  2  5  7  6  2
```

```
oilsamples2 <- do.call("rbind",oilsamples2)
head (oilsamples2,5)
```

```
##             A  B C D E F G
## Resample01 23 16 2 5 7 6 2
## Resample02 23 16 2 5 7 6 2
## Resample03 23 16 2 5 7 6 2
## Resample04 23 16 2 5 7 6 2
## Resample05 23 16 2 5 7 6 2
```

```
# How do its frequencies compare with that of the population:
#
summary(oilsamples2/60)
```

```
##        A                 B                 C                  D
##  Min.   :0.3833   Min.   :0.2667   Min.   :0.03333   Min.   :0.08333
##  1st Qu.:0.3833   1st Qu.:0.2667   1st Qu.:0.03333   1st Qu.:0.08333
##  Median :0.3833   Median :0.2667   Median :0.03333   Median :0.08333
##  Mean   :0.3833   Mean   :0.2667   Mean   :0.03333   Mean   :0.08333
##  3rd Qu.:0.3833   3rd Qu.:0.2667   3rd Qu.:0.03333   3rd Qu.:0.08333
##  Max.   :0.3833   Max.   :0.2667   Max.   :0.03333   Max.   :0.08333
##        E                 F              G
##  Min.   :0.1167   Min.   :0.1   Min.   :0.03333
##  1st Qu.:0.1167   1st Qu.:0.1   1st Qu.:0.03333
##  Median :0.1167   Median :0.1   Median :0.03333
##  Mean   :0.1167   Mean   :0.1   Mean   :0.03333
##  3rd Qu.:0.1167   3rd Qu.:0.1   3rd Qu.:0.03333
##  Max.   :0.1167   Max.   :0.1   Max.   :0.03333
```

So, the sampling done with $createDataPartition$ has much less variability compared to using the $sample$ function in R.

Q 4.4 c

c. With such a small samples size, what are the options for determining performance of the model? Should a test set be used?

Answer 4.4 (c)

Ideally, choosing a data splitting strategy is always difficult. We could look at the possibility of $LOOCV$ only because, with the exception of class G, each class will be represented in each re-sample. We also know that some classification models require at least one sample from each class, so re-sampling these data may place a restriction one which models we can use.
I firmly believe that $LOOCV$ is very reliable when we want to measure model performance.

Now, the question regarding a $test\ set$, a $test\ set$ could be used if it only consisted of the classes with the most samples $(e.g. A, B and maybe E and F)$ although this method would only protect against overfitting.

Q 4.4 d

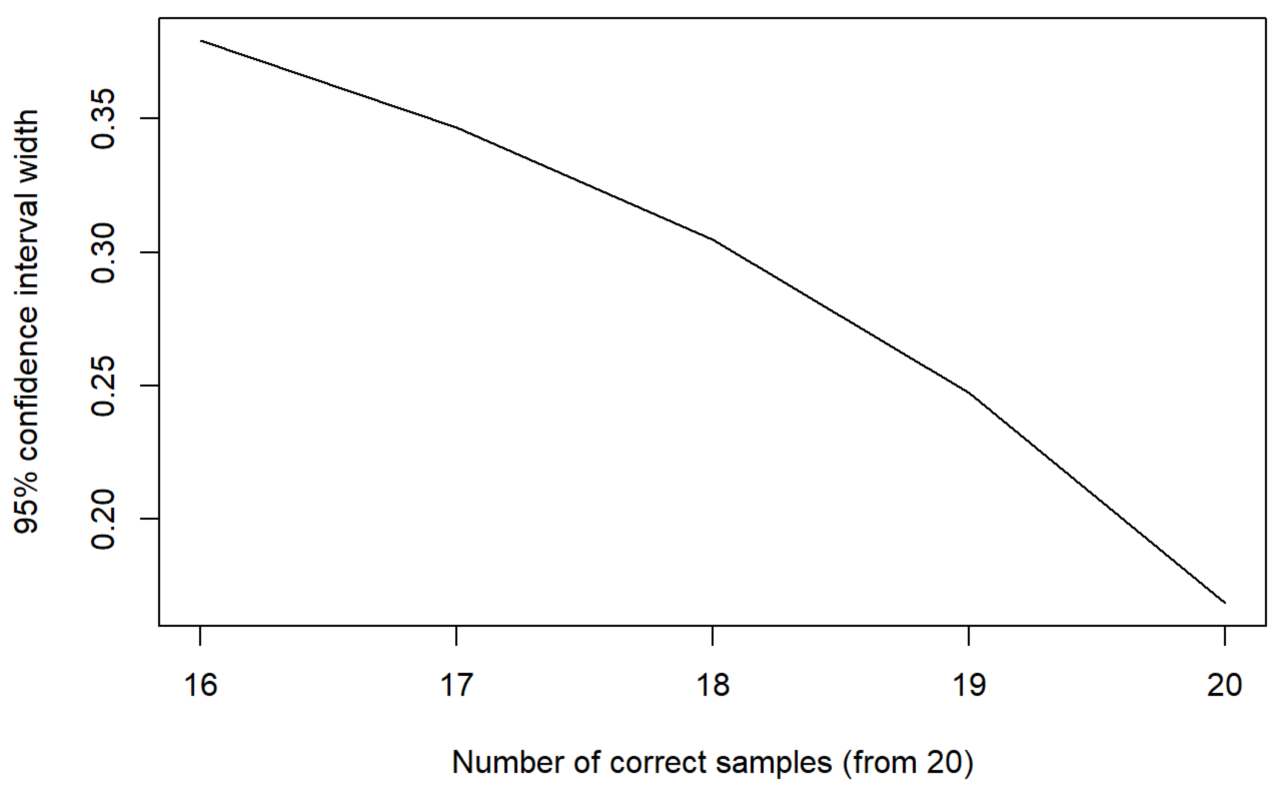
d. One method for understanding the uncertainty of a test set is to use a confidence interval. To obtain a confidence interval for the overall accuracy,the based R function binom.test can be used. It requires the user to input the number of samples and the number correctly classified to calculate the interval. For example, suppose a test set sample of 20 oil samples was set aside and 76 were used for model training. For this test set size and a model that is about 80% accurate (16 out of 20 correct), the confidence interval would be computed using

> binom.test(16, 20) Exact binomial test data: 16 and 20 number of successes = 16, number of trials = 20, p-value = 0.01182 alternative hypothesis: true probability of success is not equal to 0.5 95 percent confidence interval: 0.563386 0.942666 sample estimates: probability of success 0.8

In this case, the width of the 95% confidence interval is 37.9%. Try different samples sizes and accuracy rates to understand the trade-off between the uncertainty in the results, the model performance, and the test set size.

Answer 4.4 (d)

```
nbr_success = 16:20
ci_width = c()
for( nbrs in nbr_success ){
  bt_out = binom.test( nbrs, 20 )
  ci_width = c( ci_width, diff( bt_out$conf.int ) )
}
plot( nbr_success, ci_width, type='l', xlab='Number of correct samples (from 20)', ylab='95% con
fidence interval width' )
```



So, as we increase the sample size, the diagram shows that the width of the confidence interval is decreasing, and that is because it decreases the standard error. This means that as we increase the sample size in model building , we go towards model accuracy as there will be less variability.

Chapter 5 - Book - An Introduction to Statistical Learning - 5.4 - Exercises page 197:

$Q5.4.2$

We will now derive the probability that a given observation is part of a bootstrap sample. Suppose that we obtain a bootstrap sample from a set of n observations.

$Q5.4.2a$

What is the probability that the first bootstrap observation is not the jth observation from the original sample ? Justify your answer.

Answer 5.4.2 (a)

$1 - 1/n$

Q5.4.2b

What is the probability that the second bootstrap observation is not the jth observation from the original sample ?

Answer 5.4.2 (b)

$1 - 1/n$

Q5.4.2c

Argue that the probability that the jth observation is not in the bootstrap sample is $(1 - 1/n)^n$

Answer 5.4.2 (c)

In bootstrapping, we sample with replacement, and so the probability that the jth observation is **NOT** in the bootstrap sample is the product of the probabilities that each bootstrap observation is **NOT** the jth observation from the original sample, which means

$(1 - 1/n)$ * $(1 - 1/n)$ * .... * $(1 - 1/n)$ = $(1 - 1/n)^n$, as these probabilities are **independent**.

Q5.4.2d

When $n = 5$, what is the probability that the jth observation is in the bootstrap sample ?

Answer 5.4.2 (d)

So, $P(jth\ observation\ in\ bootstrap\ sample)$ = $(1 - 1/5)^5$ = 0.672

Q5.4.2e

When $n = 100$, what is the probability that the jth observation is in the bootstrap sample ?

Answer 5.4.2 (e)

So, $P(jth\ observation\ in\ bootstrap\ sample)$ = $(1 - 1/100)^{100}$ = 0.634

Q5.4.2f

When $n = 10,000$, what is the probability that the jth observation is in the bootstrap sample ?

Answer 5.4.2 (f)

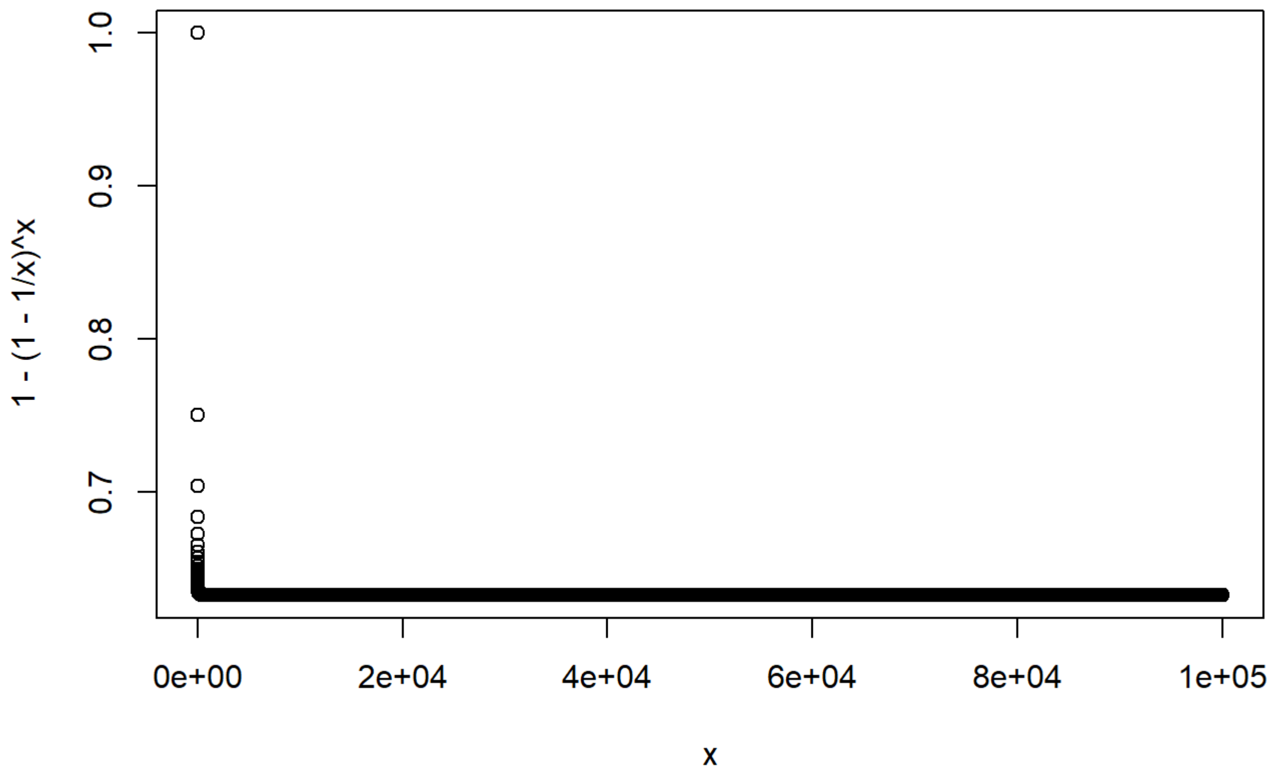
So, $P(jth\ observation\ in\ bootstrap\ sample)$ = $(1 - 1/10,000)^{10,000}$ = 0.632

Q5.4.2g

Create a plot that displays, for each integer value of n from $1\ to\ 100,000$, the probability that the jth observation is in the bootstrap sample. Comment on what you observe.

Answer 5.4.2 (g)

```
x <- 1:100000
plot(x, 1 - (1 - 1/x)^x)
```



It might be seen that the plot quickly reaches an asymptotic value of about 0.632.

Q5.4.2h

We will now investigate numerically the probability that a bootstrap sample of size $n = 100$ contains the jth observation. Here $j = 4$. We repeatedly create bootstrap samples, and each time we record whether or not the **fourth** observation is contained in the bootstrap sample.

```
store <- rep(NA, 10000)
for (i in 1:10000) {
    store[i] <- sum(sample(1:100, rep = TRUE) == 4) > 0
}
mean(store)
```

```
## [1] 0.6429
```

Comment on the results obtained.

Answer 5.4.2 (h)

From calculus, we know that $\lim_{x \to \infty}(1 + x/n)^n$ = e^x.
Now if we apply the above to our situation here, we get the probability that a bootstrap sample of size n contains the jth observation converges to $1 - 1/e$ = 0.632 as $n- > \infty$.

Q5.4.6

We continue to consider the use of a logistic regression model to predict the probability of **default** using **income** and **balance** on the **Default** data set. In particular, we will now compute estimates for the standard errors of the **income** and **balance** logistic regression coefficients in two different ways : (1) using the bootstrap, and (2) using the standard formula for computing the standard errors in the $glm()$ function. Do not forget to set a random seed before beginning your analysis.

Q5.4.6a

Using the $summary()$ and $glm()$ functions, determine the estimated standard errors for the coefficients associated with **income** and **balance** in a multiple logistic regression model that uses both predictors.

Answer 5.4.6 (a)

```
set.seed(1)
attach(Default)
fit.glm.6a <- glm(default ~ income + balance, data = Default, family = "binomial")
summary(fit.glm.6a)
```

```
##
## Call:
## glm(formula = default ~ income + balance, family = "binomial",
##     data = Default)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.4725  -0.1444  -0.0574  -0.0211   3.7245
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.154e+01  4.348e-01 -26.545  < 2e-16 ***
## income       2.081e-05  4.985e-06   4.174 2.99e-05 ***
## balance      5.647e-03  2.274e-04  24.836  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2920.6  on 9999  degrees of freedom
## Residual deviance: 1579.0  on 9997  degrees of freedom
## AIC: 1585
##
## Number of Fisher Scoring iterations: 8
```

The $glm()$ estimated standard errors for the coefficients associated with **income** and **balance** are $4.985e-06$ and $2.274e-04$ respectively. The intercept estimate is $4.348e-01$.

Q5.4.6b

Write a function, $boot.fn()$, that takes as input the **Default** data set as well as an index of the observations, and that outputs the coefficient estimates for **income** and **balance** in the multiple logistic regression model.

Answer 5.4.6 (b)

```
boot.fn <- function(df, trainid) {

return(coef(glm(default ~ income + balance, data=df, family=binomial, subset=trainid)))

}
```

Q5.4.6c

Use the $boot()$ function together with your $boot.fn()$ function to estimate the standard errors of the logistic regression coefficients for **income** and **balance**.

Answer 5.4.6 (c)

```
boot.fn(Default, 1:nrow(Default))
```

```
##   (Intercept)        income       balance 
## -1.154047e+01  2.080898e-05  5.647103e-03
```

```
boot(Default, boot.fn, R=100)
```

```
## 
## ORDINARY NONPARAMETRIC BOOTSTRAP
## 
## 
## Call:
## boot(data = Default, statistic = boot.fn, R = 100)
## 
## 
## Bootstrap Statistics :
##          original        bias     std. error
## t1* -1.154047e+01  8.556378e-03 4.122015e-01
## t2*  2.080898e-05 -3.993598e-07 4.186088e-06
## t3*  5.647103e-03 -4.116657e-06 2.226242e-04
```

Q5.4.6d

Comment on the estimated standard errors obtained using the $glm()$ function and using your bootstrap function.

Answer 5.4.6 (d)

Based on the output, the estimated standard errors obtained by the two methods are pretty close to each other.