Project-01: SEMMA with Regularized Logistic Regression

Md Al Masum Bhuiyan

September 24, 2018

Contents

1 Problem-1: Data Collection **2**

2 Problem-2: Exploratory Data Analysis **2**

2.1 Removing column ID 3

2.2 Inspecting the distinct values of each variable 3

2.3 Visualizing Missing values 4

2.4 Frequency Distribution of the Target variable Class 7

2.5 Inspecting the outlying records 7

2.6 Association using Chi^2 and Fisher test 8

2.6.1 Measure the assoicaiton between Class and other variables 9

3 Problem-3: Data Partition **11**

4 Problem 4: **11**

4.1 Problem-4(a): Builiding a Logistic Rgression Model 11

4.2 Problem-4(b): Selecting the best tuning parameter 11

4.3 Problem-4(c): Final 'best' model by pooling D1 and D2. 13

5 Problem-5 **13**

5.1 Final logistic model to the test data D3 13

5.2 ROC curve and AUC 14

1 Problem-1: Data Collection

We obtain the data directly in R using mlbench package and save the data in a CSV file. The code is as follows:

```
#install.packages("mlbench")
data(BreastCancer, package="mlbench")
data <-BreastCancer
write.csv(data, file="BreastCancer.csv", row.names =FALSE)
```

2 Problem-2: Exploratory Data Analysis

To perfrom exploratory data analysis, we first inspect the data types, quantity and percentage of zeros, infinte numbers, missing, and unique values of data using *df_status* function:

```
#install.packages("funModeling")
library("funModeling")
dim(data)
```

```
## [1] 699  11
```

```
head(data)
```

```
##        Id Cl.thickness Cell.size Cell.shape Marg.adhesion Epith.c.size
## 1 1000025            5         1          1             1            2
## 2 1002945            5         4          4             5            7
## 3 1015425            3         1          1             1            2
## 4 1016277            6         8          8             1            3
## 5 1017023            4         1          1             3            2
## 6 1017122            8        10         10             8            7
##   Bare.nuclei Bl.cromatin Normal.nucleoli Mitoses     Class
## 1           1           3               1       1    benign
## 2          10           3               2       1    benign
## 3           2           3               1       1    benign
## 4           4           3               7       1    benign
## 5           1           3               1       1    benign
## 6          10           9               7       1 malignant
```

```
#str(data)
df_status(data) # Getting the metrics about data types, zeros, infinite numbers, and missing v
```

```
##          variable q_zeros p_zeros q_na p_na q_inf p_inf            type
## 1              Id       0       0    0 0.00     0     0       character
## 2    Cl.thickness       0       0    0 0.00     0     0 ordered-factor
## 3       Cell.size       0       0    0 0.00     0     0 ordered-factor
## 4      Cell.shape       0       0    0 0.00     0     0 ordered-factor
## 5   Marg.adhesion       0       0    0 0.00     0     0 ordered-factor
## 6    Epith.c.size       0       0    0 0.00     0     0 ordered-factor
## 7     Bare.nuclei       0       0   16 2.29     0     0          factor
## 8      Bl.cromatin       0       0    0 0.00     0     0          factor
## 9  Normal.nucleoli       0       0    0 0.00     0     0          factor
## 10         Mitoses       0       0    0 0.00     0     0          factor
## 11           Class       0       0    0 0.00     0     0          factor
##    unique
## 1     645
## 2      10
## 3      10
## 4      10
## 5      10
## 6      10
## 7      10
## 8      10
## 9      10
## 10      9
## 11      2
```

2.1 Removing column ID

We prepare the data by removing column ID, Since it will not affect in our analysis.

```r
dat <- data[, -1]
dim(dat)
```

```
## [1] 699  10
```

2.2 Inspecting the distinct values of each variable

We inspect the distinct values of each variable of data. The following code shows that the traget Class has two categorical levels: 458 benign and 241 malignant. The other variable is also shown 10 levels with number of values.

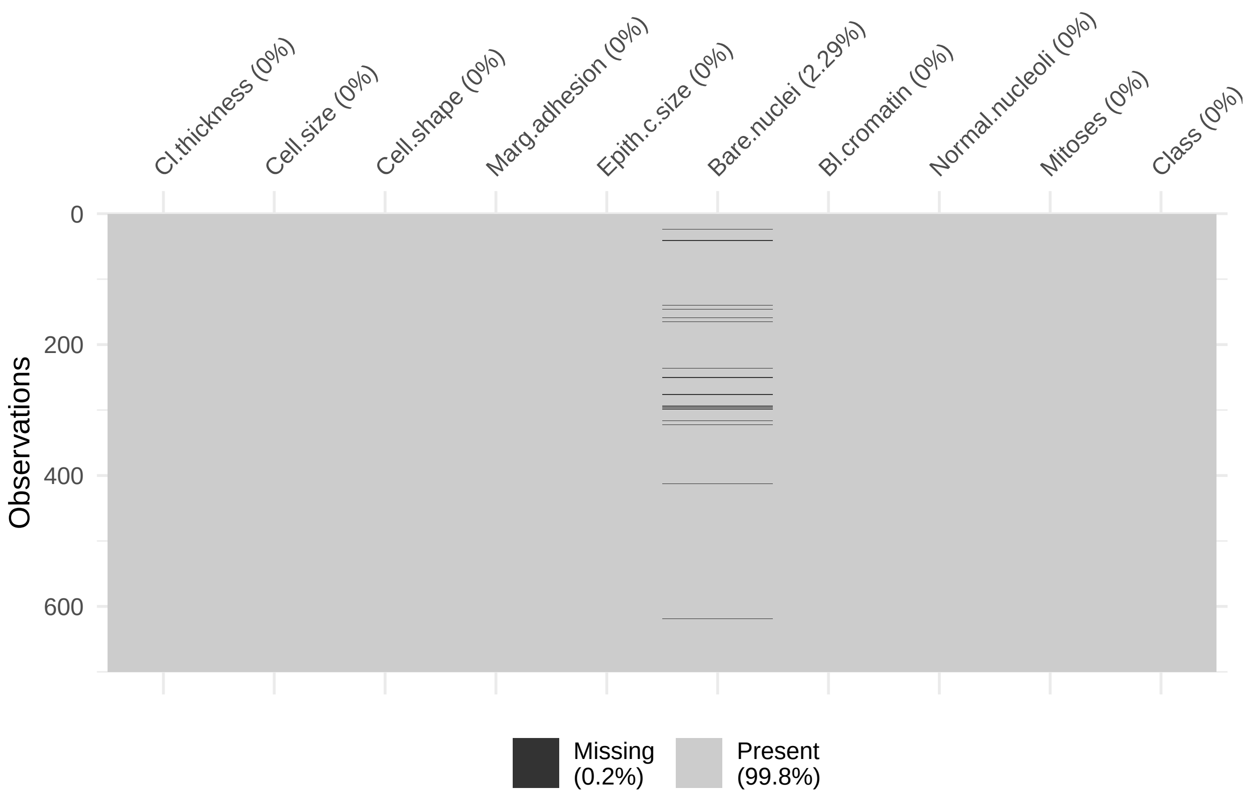
```r
for (j in 1:NCOL(dat)){
  print(colnames(dat)[j])
  print(table(dat[,j], useNA="ifany"))
}
```

```
## [1] "Cl.thickness"
##
##   1   2   3   4   5   6   7   8   9  10
## 145  50 108  80 130  34  23  46  14  69
## [1] "Cell.size"
##
##   1   2   3   4   5   6   7   8   9  10
## 384  45  52  40  30  27  19  29   6  67
## [1] "Cell.shape"
##
##   1   2   3   4   5   6   7   8   9  10
## 353  59  56  44  34  30  30  28   7  58
## [1] "Marg.adhesion"
## 
##   1   2   3   4   5   6   7   8   9  10 
## 407  58  58  33  23  22  13  25   5  55 
## [1] "Epith.c.size"
## 
##   1   2   3   4   5   6   7   8   9  10 
##  47 386  72  48  39  41  12  21   2  31 
## [1] "Bare.nuclei"
## 
##    1    2    3    4    5    6    7    8    9   10 <NA> 
##  402   30   28   19   30    4    8   21    9  132   16 
## [1] "Bl.cromatin"
## 
##   1   2   3   4   5   6   7   8   9  10 
## 152 166 165  40  34  10  73  28  11  20 
## [1] "Normal.nucleoli"
## 
##   1   2   3   4   5   6   7   8   9  10 
## 443  36  44  18  19  22  16  24  16  61 
## [1] "Mitoses"
## 
##   1   2   3   4   5   6   7   8  10 
## 579  35  33  12   6   3   9   8  14 
## [1] "Class"
## 
##    benign malignant 
##       458       241
```

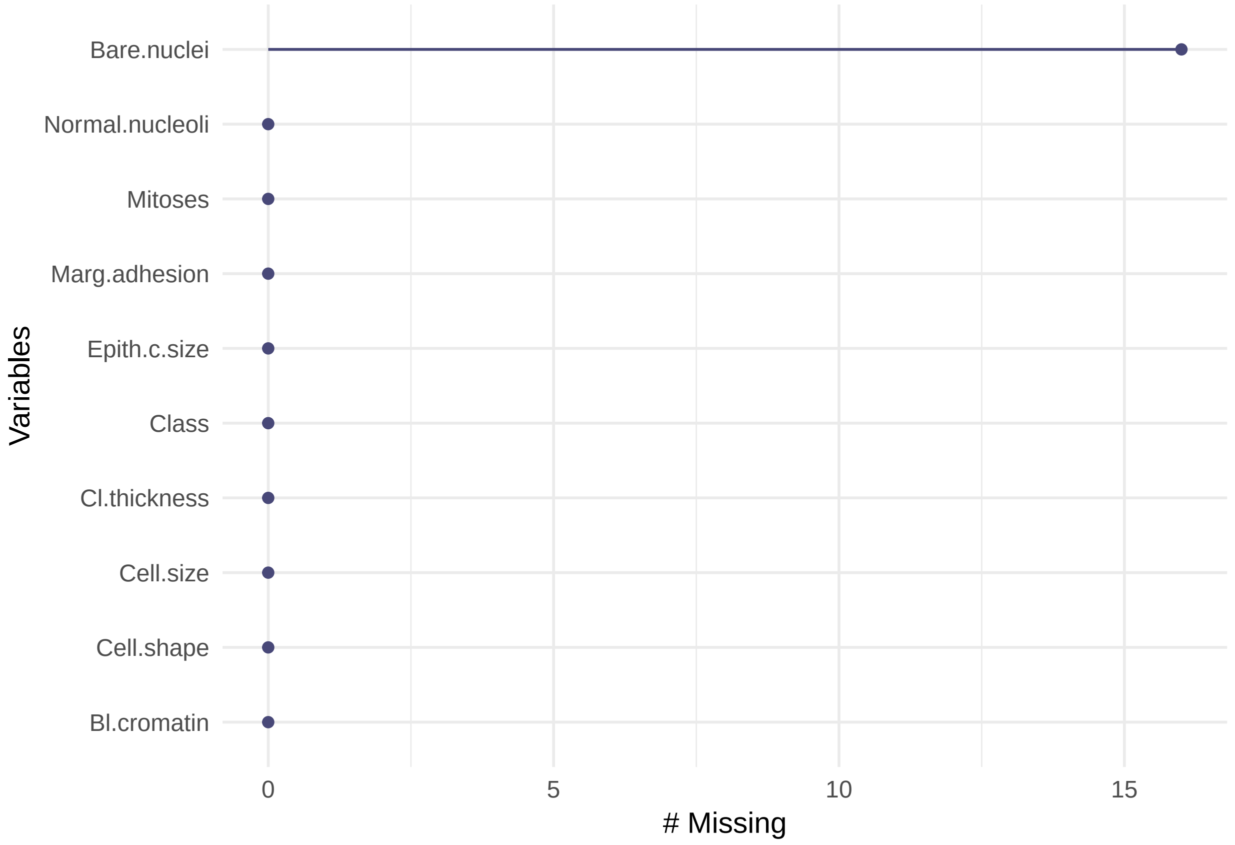
2.3 Visualizing Missing values

We now visualize the quantity, percentage of missing values for each variables. We see that only the Bare.nuclei variable has 16 NA values, which is 2.29 percent of the data. We also visualize the the missing values for categorical variables and then we omit the NA data.

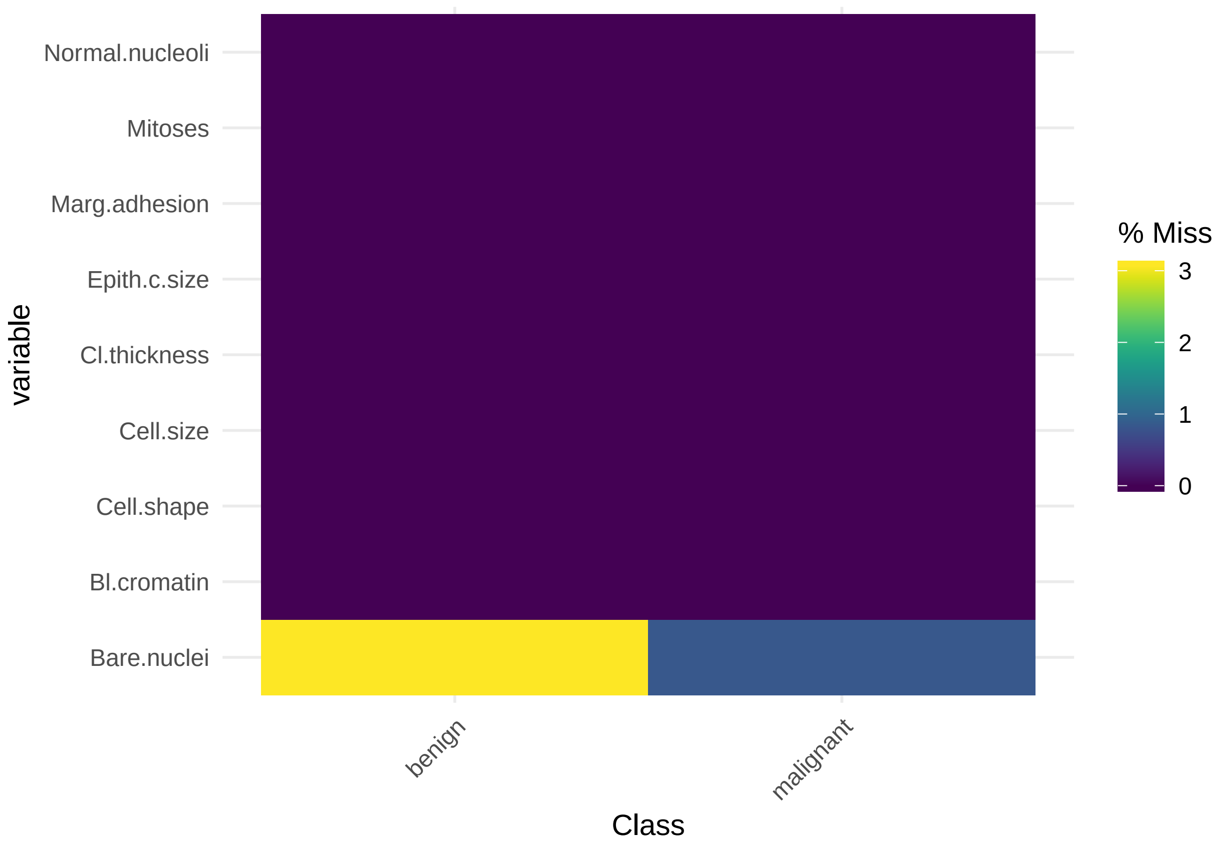
```r
#install.packages("naniar")
library(naniar)
vis_miss(dat)
```



```
gg_miss_var(dat)
```



```
##Visulaizing the missing values for target Class of each variables
gg_miss_fct(x = dat, fct = Class)
```



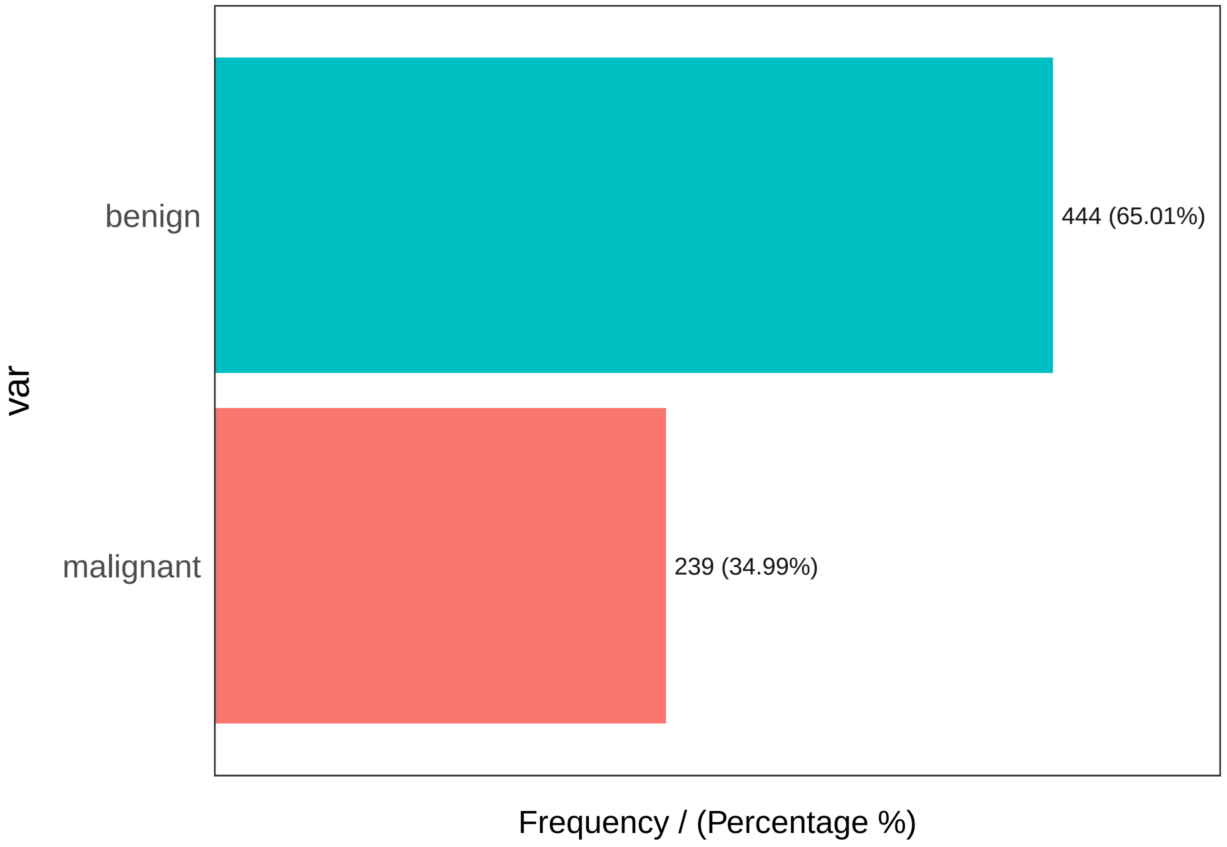
```
# remove rows containing missing values
dat <- na.omit(dat)
dim(dat)
```

```
## [1] 683  10
```

2.4 Frequency Distribution of the Target variable Class

The following code shows the frequency, percentage, cumulative percentage of the target variable Class.

```
freq(dat$Class)
```



```
##         var frequency percentage cumulative_perc
## 1    benign       444      65.01           65.01
## 2 malignant       239      34.99          100.00
```

To analyze the data, we make numeric values 1 and 0 for categorical variable Class.

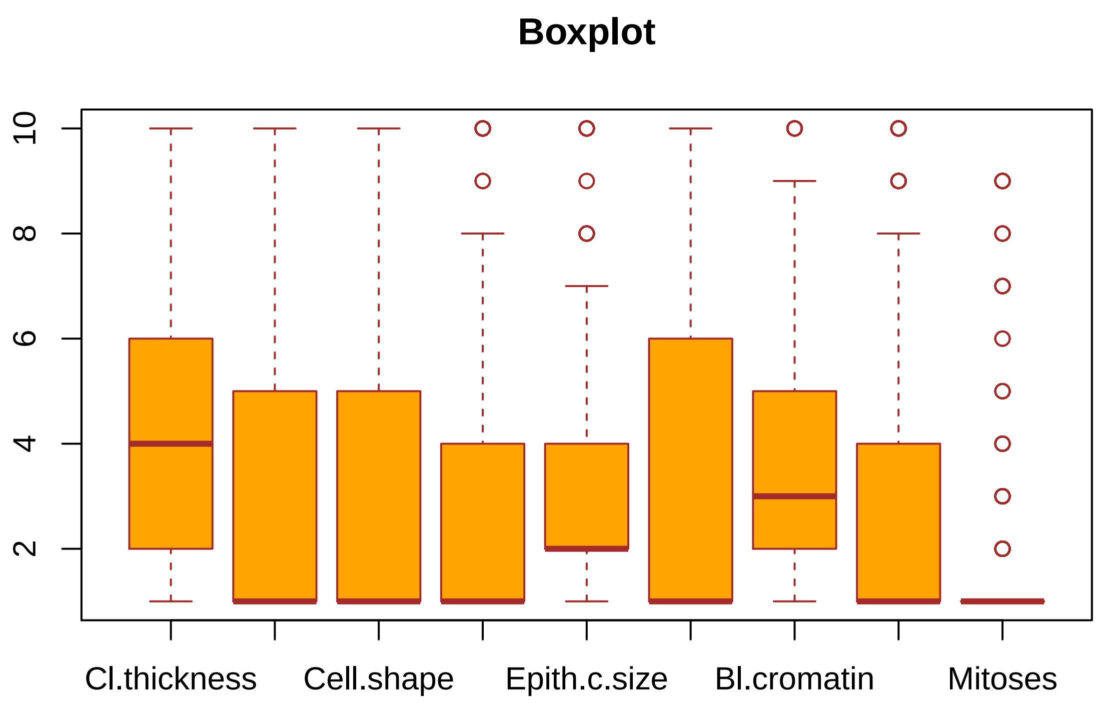
```
dat$y <- ifelse(dat$Class=="benign", 1, 0)
dim(dat)
```

```
## [1] 683  11
```

2.5 Inspecting the outlying records

We now plot a Box plot to inspect the outlying records for each variable. As we see in the graph, *Marg_adhesion*, *Epith.c.size*, *Bl_cromatin*, *Normal_Nucleoli* has few outlying value. *Mitoses* variable hase most of the outlying values.

```
data_box <- dat[, -11:-10] # Removing Class variable (Bcz of many variables)
boxplot(data_box, main = "Boxplot", horizontal = F, col="orange",
        border="brown")
```



To analyze the data, we change the character variable of into numeric variable.

```
dat <- dat[, c(1:9, 11)]
dat <- apply(dat, 2, FUN=function(x) {as.numeric(as.character(x))})
dat <-na.omit(dat)
dat <- as.data.frame(dat)
```

2.6 Association using Chi^2 and Fisher test

To check the assoication between Class and other attributes, we use Chi^2 and fisher test. Our hypotheses are as follows:

H_0: The two variables are independent,

H_1: The two variables are dependent.

since the p-value is less than the significance level (0.05) for all cases (between class and other attributes), we reject the null hypothesis and conclude that the Class and other attribute are dependent to each other. The code and result are as follows:

```
#============================================================
# Chi-sq and Fisfter test for Class and Other varibales
#============================================================
dat1 <- dat[, -11]
chitest <- matrix (0, 9, 4)
ftest <- matrix (0, 9, 2)
```

```r
for (j in 1: (ncol(dat1)-1)){
  testor <- table(as.vector(dat1 [, ncol(dat1)]), as.numeric(dat1[, j]))
  chi2 <- chisq.test(testor, correct=FALSE)
  chitest[j, ] <- c(colnames(dat1)[j], round(chi2$statistic, digits = 2), chi2$p.value,chi2$pa
  s = fisher.test(testor, simulate.p.value = TRUE, B=1e5)
  ftest[j, ] <- c(colnames(dat1)[j], s$p.value)
}

colnames(chitest) <- c("Names", "Statistics", "p-values", "D.Freedom")
colnames(ftest) <- c("Names", "p-values")
names(dimnames(chitest)) <- list("", "Association among Class and other predictors Using Chi te
names(dimnames(ftest)) <- list("", "Association among Class and other predictors Using Fisher
chitest
```

```
##        Association among Class and other predictors Using Chi test
##
##         Names              Statistics p-values                D.Freedom
##    [1,] "Cl.thickness"    "378.08"   "6.47143951518931e-76"  "9"
##    [2,] "Cell.size"       "539.79"   "1.71638971098766e-110" "9"
##    [3,] "Cell.shape"      "523.07"   "6.57844762931427e-107" "9"
##    [4,] "Marg.adhesion"   "390.06"   "1.80795747026517e-78"  "9"
##    [5,] "Epith.c.size"    "447.86"   "8.21759531792845e-91"  "9"
##    [6,] "Bare.nuclei"     "489.01"   "1.2957665166586e-99"   "9"
##    [7,] "Bl.cromatin"     "453.21"   "5.90593696241214e-92"  "9"
##    [8,] "Normal.nucleoli" "416.63"   "3.86380729078817e-84"  "9"
##    [9,] "Mitoses"         "191.97"   "3.13852341562463e-37"  "8"
```

```r
ftest
```

```
##        Association among Class and other predictors Using Fisher test
##
##         Names              p-values
##    [1,] "Cl.thickness"    "9.99990000099999e-06"
##    [2,] "Cell.size"       "9.99990000099999e-06"
##    [3,] "Cell.shape"      "9.99990000099999e-06"
##    [4,] "Marg.adhesion"   "9.99990000099999e-06"
##    [5,] "Epith.c.size"    "9.99990000099999e-06"
##    [6,] "Bare.nuclei"     "9.99990000099999e-06"
##    [7,] "Bl.cromatin"     "9.99990000099999e-06"
##    [8,] "Normal.nucleoli" "9.99990000099999e-06"
##    [9,] "Mitoses"         "9.99990000099999e-06"
```

2.6.1 Measure the assoicaiton between Class and other variables

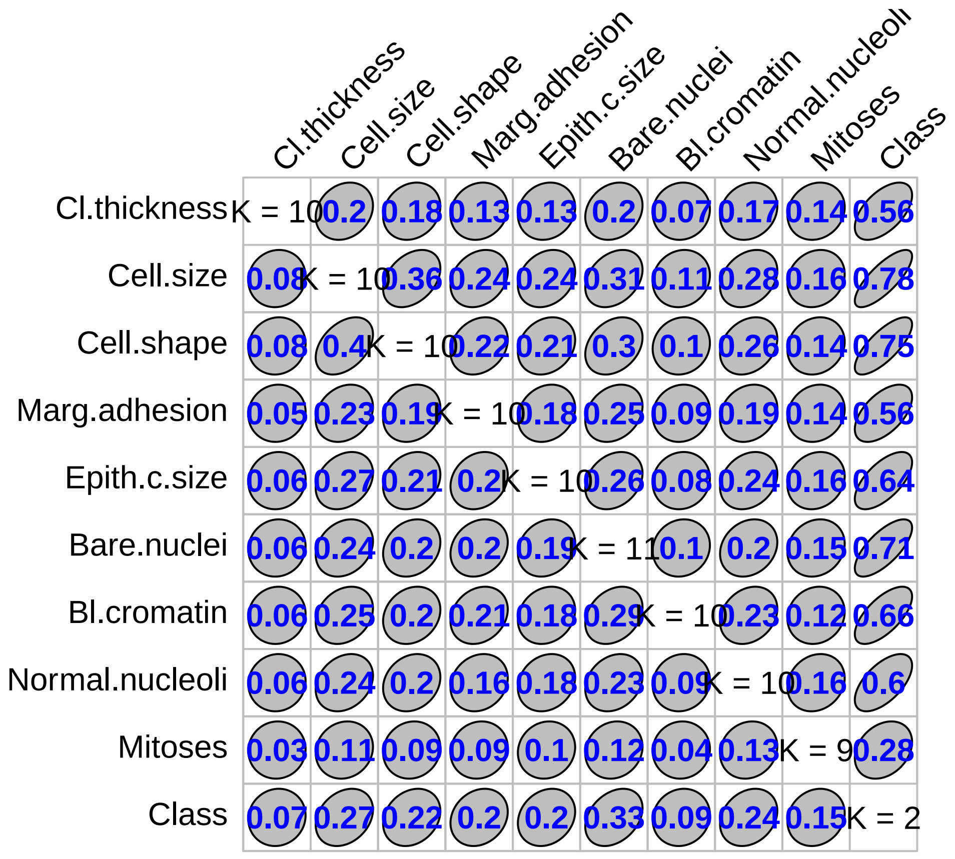
In this subsection, we study the assoiciation plot (given below) where the diagonal element K refers to number of unique levels for each variable. This measure of association indicates the strength of the relationship, whether, weak or strong. The off-diagonal elements contain the forward and backward τ measures for each variable pair. Specifically, the numerical values appearing in each row represent the association measure $\tau(x, y)\tau(x, y)$ from the variable xx indicated in the row name to the variable yy indicated in the column name.

For example, the variable Cell.size is almost perfectly predictable (i.e. $\tau(x, y) = 0.78$) from Class and this forward association is quite strong. The forward association suggest that x=Cell.size is highly predictive of y=Class. It indicates that if we know a Cell.size, then we can easily predict its Class.

On the contrary, the reverse association y=class and x= Cell.size (i.e. $\tau(y, x) = 0.27$); is a strong association and indicates that if we know the Class then its easy to predict its Cell.size.

From chi-squared and Fisher significance test, we have found Normal.Neocleoli and Cell.thickness are dependent to each other. But forward and reverse association plot suggest that x=Normal.neocleuli shape is weakly associated to y= Cell.thickness (i.e.$\tau(x, y) = 0.17$) and (i.e.$\tau(y, x) = 0.060$). So we conclude that although these two variables are significant but their association is weak; i.e. it will be difficult to predict one from another.

```
library(GoodmanKruskal)
varset1<- c("Cl.thickness", "Cell.size", "Cell.shape",  "Marg.adhesion",  "Epith.c.size", "Bar
associate1<- subset(data, select = varset1)
GKmatrix1<- GKtauDataframe(associate1)
plot(GKmatrix1, corrColors = "blue")
```



3 Problem-3: Data Partition

In this section, I partition the data into three parts, the training data D1, the validation data D2, and the test data D3, with a ratio of 2 : 1 : 1.

```r
set.seed(123)
n <- nrow(dat)
id.split <- sample(x=1:3, size = n, replace =TRUE, prob=c(0.5, 0.25, 0.25))
dat.train <- dat[id.split ==1, ]
dat.valid <- dat[id.split ==2, ]
dat.test <- dat[id.split == 3, ]
```

4 Problem 4:

4.1 Problem-4(a): Builiding a Logistic Rgression Model

Here, I fit the regularized logistic regression using the training data D1 with Lasso model. In glmnet function, the family argument specify that we want a "binomial" model which tells glmnet() to fit a logistic function to the data.

```r
#install.packages("glmnet")
library(glmnet)
formula0 <- y~Cl.thickness + Cell.size + Cell.shape + Marg.adhesion + Epith.c.size + Bare.nucl
X <- model.matrix (as.formula(formula0), data = dat.train)
y <- dat.train$y
```

4.2 Problem-4(b): Selecting the best tuning parameter

Next I would like to see how the model is doing when predicting Class(y) on data. I used *pred* function in the form of P(y=1|X) using parameter $type =' response'$ which tells predict to return probabilities. The decision boundary will be 0.5. If P(y=1|X) > 0.5 then y = 1 (benign) otherwise y=0 (malignant). Therefore I used misclassification rate and the mean square error (mse) for the predicted probabilities.

I select the best tuning parameter using the validation data D2 and choosing the minimum mean squared error(mse).

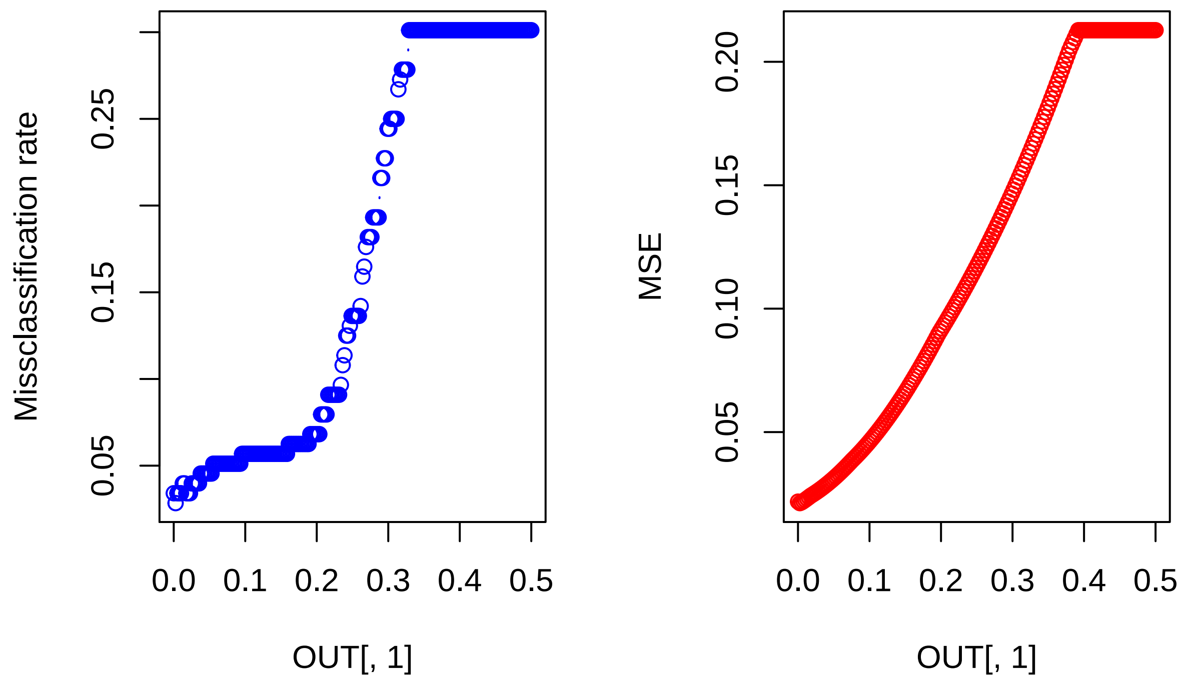
```r
X.valid <- model.matrix (as.formula(formula0), data = dat.valid)
y.valid <- dat.valid$y

Lambda <- seq(0.0001, 0.5, length.out = 200)
L <- length(Lambda)
OUT <- matrix (0, L, 3)
for (i in 1:L){
  fit <- glmnet(x=X, y=y, family ="binomial", alpha =1, #lasso
                lambda=Lambda[i], standardize=T, thresh = 1e-07, maxit=1000)
  pred <- predict(fit, newx=X.valid, s=Lambda[i], type="response")
  miss.rate <- mean(y.valid != (pred > 0.5))
  mse <- mean((y.valid - pred)^2)
  OUT[i, ] <- c(Lambda[i], miss.rate, mse)

}
head(OUT)
```

```
##             [,1]       [,2]       [,3]
## [1,] 0.000100000 0.03409091 0.02181988
## [2,] 0.002612060 0.02840909 0.02123402
## [3,] 0.005124121 0.03409091 0.02156797
## [4,] 0.007636181 0.03409091 0.02204101
## [5,] 0.010148241 0.03409091 0.02254983
## [6,] 0.012660302 0.03977273 0.02308470
```

```
par(mfrow = c(1,2))
plot(OUT[, 1], OUT[,2], type = "b", col = "blue", ylab = "Missclassification rate")
plot(OUT[, 1], OUT[,3], type = "b", col = "red", ylab = "MSE")
```



```
(lambda.best <- OUT[which.min(OUT[, 3]), 1])
```

```
## [1] 0.00261206
```

```
(miss.rate_Lambda <- OUT[which.min(OUT[, 3]), 2])
```

```
## [1] 0.02840909
```

The best fitted lambda is 0.00261206, where the corresponding misclassification rate is is 0.02840909. So I conclude that the accuracy on this model is good.

4.3 Problem-4(c): Final 'best' model by pooling D1 and D2.

I then present the final 'best' model fit by pooling D1 and D2 together. Using the beta from fit.best model, I see the coefficients of the model and select the important predictors.

```
X.12 = rbind(X, X.valid)
y.12 = c(y, y.valid)
fit.best <- glmnet (x=X.12, y=y.12, family ="binomial", alpha=1,  #LASSO
        lambda = lambda.best, standardize = T, thresh = 1e-07, maxit=1000)
names(fit.best)
```

```
##  [1] "a0"         "beta"       "df"         "dim"        "lambda"
##  [6] "dev.ratio"  "nulldev"    "npasses"    "jerr"       "offset"
## [11] "classnames" "call"       "nobs"
```

```
fit.best$beta # Finding important variables.
```

```
## 10 x 1 sparse Matrix of class "dgCMatrix"
##                        s0
## (Intercept)      .
## Cl.thickness    -0.4891238
## Cell.size        .
## Cell.shape      -0.2656879
## Marg.adhesion   -0.3596817
## Epith.c.size    -0.2128013
## Bare.nuclei     -0.2988293
## Bl.cromatin     -0.3582619
## Normal.nucleoli -0.1435005
## Mitoses         -0.3063777
```

Since I do not get any coefficients for Cell.size, it is not an important predictor for the predective model. The important predictors are Cl.thickness, Cell.shape, Marg.adhesion, Epith.c.size ,Bare.nuclei, Bl.cromatin, Normal.nucleoli, and Mitoses.

5 Problem-5

5.1 Final logistic model to the test data D3

Using test data D3, I apply the final logistic model.

```
X.test <- model.matrix(object=~Cl.thickness + Cell.size +Cell.shape + Marg.adhesion + Epith.c.s
pred <- predict(fit.best, newx = X.test, s =lambda.best, type="response")
dim(pred)
```

```
## [1] 158   1
```

5.2 ROC curve and AUC

ROC suggests the accuracy of a classification model at a threshold value. It determines the model's accuracy using Area Under Curve (AUC). The AUC also referred to as index of accuracy (A) or concordant index (ci), represents the performance of the ROC curve. The idea is that higher the area, better the model.

```
library(cvAUC)
yobs <- dat.test$y
AUC <- ci.cvAUC(predictions = pred, labels =yobs, folds=1:NROW(dat.test), confidence = 0.95)
AUC
```

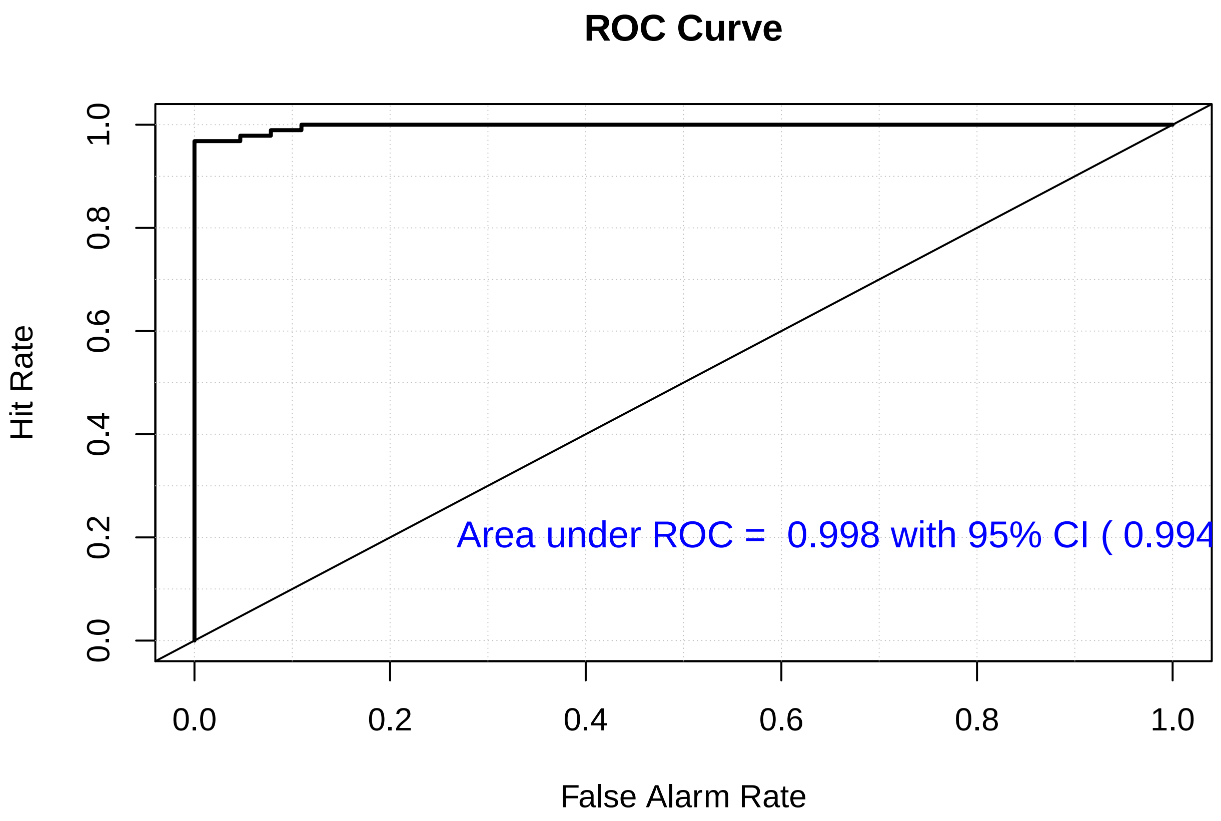
```
## $cvAUC
## [1] 0.9975066
##
## $se
## [1] 0.001775487
##
## $ci
## [1] 0.9940268 1.0000000
##
## $confidence
## [1] 0.95
```

```
auc.ci <- round(AUC$ci, digits = 3)

library(verification)
mod.glm <- verify(obs = yobs, pred = pred)
```

```
## If baseline is not included, baseline values  will be calculated from the  sample obs.
```

```
roc.plot(mod.glm, plot.thres=NULL)
text(x=0.7, y=0.2, paste("Area under ROC = ", round(AUC$cvAUC, digits = 3), "with 95% CI (",
                         auc.ci[1], ",", auc.ci[2], ").", sep = " "), col="blue", cex =1.2)
```



ROC is plotted between True Positive Rate (Y axis) and False Positive Rate (X axis). In this plot, our aim is to push the curve (shown below) toward 1 (left corner) and maximize the area under curve. The diagonal line represents the ROC curve at 0.5 threshold. Since AUC is 0.99 (closer to 1) and the curve almost appraoches to 1, we can say that the model have good predictive ability.