BUAN6356_Homewrok3_Group11

Jeffrey Chang,Naishal Kanubhai Thakkar,Bhavana Chowdary Kandimalla,Navya Bulusu,Rucha Nichkawade

10/28/2019

```
if(!require("pacman")) install.packages("pacman")
pacman::p_load(caret, data.table, MASS, ggplot2)
```

Question 1:

##Examine how each predictor differs between the spam and non-spam e-mails by comparing the spam-class(1) average and non-spam-class(0) average. Identify 10 predictors for which the difference between the spam-class average and non-spam class average is highest.

```
sbase = read.csv(file = "spambase.data",header=FALSE)
cnames = read.csv("spambase.names", comment.char="|" , sep=":", header=FALSE)
colnames(sbase)[58] = "spam"
cnames1 <-as.matrix(cnames[-1,-2])
colnames(sbase)[1:57]=c(cnames1)

spambase_0 <- sbase[sbase$spam==0,]
spambase_1 <-sbase[sbase$spam==1,]

Mean_spambase_0 = colMeans(spambase_0[-58])
Mean_spambase_1 = colMeans(spambase_1[-58])

Difference = abs(Mean_spambase_0 - Mean_spambase_1)
column_list = sort.list(Difference,decreasing = TRUE)
head(column_list,10)
```

```
##  [1] 57 56 55 27 19 21 25 16 26 52
```

```
sbase_num <- sbase[,c(57,56,55,27,19,21,25,16,26,52,58)]
colnames(sbase_num)
```

```
##  [1] "capital_run_length_total"   "capital_run_length_longest"
##  [3] "capital_run_length_average" "word_freq_george"
##  [5] "word_freq_you"              "word_freq_your"
##  [7] "word_freq_hp"               "word_freq_free"
##  [9] "word_freq_hpl"              "char_freq_!"
## [11] "spam"
```

```
sbase_new = sbase[,c(57,56,55,27,19,21,25,16,26,52,58)]
sbase_new1 = sbase[,c(57,56,55,27,19,21,25,16,26,52,58)]

sbase_new$spam <- factor(sbase_new$spam, levels = c(0,1),
                         labels = c("Non-spam", "Spam"))
set.seed(42)
library('caTools')
sample = sample.split(sbase_new$spam , SplitRatio = .80)
train.df= subset(sbase_new, sample == TRUE)
valid.df= subset(sbase_new, sample == FALSE)
valid.df1= subset(sbase_new1, sample == FALSE)
```

```
# Normalize the data
    # Estimate preprocessing parameters
norm.values  <- preProcess(train.df, method = c("center", "scale"))
    # Transform the data using the estimated parameters
train.df.norm <- predict(norm.values, train.df)
valid.df.norm <- predict(norm.values, valid.df)
valid.df1.norm <- predict(norm.values, valid.df1)

# Predict - using Validation data
#pred1.valid <- predict(lda1, valid.df.norm)
#pred1.valid
```

As the result, We choose top 10 predictors are:

1.capital_run_length_total

2.capital_run_length_longest

3.capital_run_length_average

4.word_freq_george

5.word_freq_you

6.word_freq_your

7.word_freq_hp

8.word_freq_free

9.word_freq_hpl

10.char_freq_!

Question 2:

Perform a linear discriminant analysis using the training dataset. Include only 10 predictors identified in the question above in the model.

```
lda1 <- lda(spam~., data = train.df.norm)
lda1
```

```
## Call:
## lda(spam ~ ., data = train.df.norm)
##
## Prior probabilities of groups:
##  Non-spam      Spam
## 0.6059783 0.3940217
##
## Group means:
##          capital_run_length_total capital_run_length_longest
## Non-spam               -0.1984063                 -0.1653298
## Spam                    0.3051352                  0.2542659
##          capital_run_length_average word_freq_george word_freq_you
## Non-spam                -0.08871288        0.1465656    -0.2271454
## Spam                     0.13643428       -0.2254077     0.3493340
##          word_freq_your word_freq_hp word_freq_free word_freq_hpl
## Non-spam     -0.3113688    0.2058265     -0.2046994     0.1881824
## Spam          0.4788637   -0.3165470      0.3148135    -0.2894115
##          `char_freq_!`
## Non-spam    -0.2644065
## Spam         0.4066390
##
## Coefficients of linear discriminants:
##                                    LD1
## capital_run_length_total    0.36793159
## capital_run_length_longest  0.09877380
## capital_run_length_average  0.06155527
## word_freq_george           -0.20467412
## word_freq_you               0.20944930
## word_freq_your              0.53645180
## word_freq_hp               -0.22152480
## word_freq_free              0.37102910
## word_freq_hpl              -0.16366205
## `char_freq_!`               0.44035130
```

Question 3:

What are the prior probabilities?

```
lda1$prior
```

```
##  Non-spam      Spam
## 0.6059783 0.3940217
```

Prior probabilities is the probability before buliding the model. (Usually assumed prior probability is equal or 50 represenations of one data point and -50 representations of the other) The prior probability for Non-spam group is 0.606 while the prior probability for Spam group is 0.394. This means that most of the records 61% are believed to be from Non-Spam class and 39% are in the spam class.

Question 4:

What are the coefficients of linear discriminants? Explain.

```
lda1$scaling
```

```
##                                  LD1
## capital_run_length_total    0.36793159
## capital_run_length_longest  0.09877380
## capital_run_length_average  0.06155527
## word_freq_george           -0.20467412
## word_freq_you               0.20944930
## word_freq_your              0.53645180
## word_freq_hp               -0.22152480
## word_freq_free              0.37102910
## word_freq_hpl              -0.16366205
## `char_freq_!`               0.44035130
```

Coefficients of the linear discriminants are estimates for each predictor variable. These coefficients are used to generate linear combinations of the predictor variables which are used to create decision boundaries.

Question 5:

Generate linear discriminants using your analysis. How are they used in classifying spams and non-spams?

```
pred2.valid <- predict(lda1, valid.df.norm)
head(pred2.valid$x,10)
```

```
##             LD1
## 1   0.670147649
## 2   1.238063060
## 4   0.007471833
## 13  0.730390418
## 16  1.038815300
## 17  1.013829954
## 21 -0.349687180
## 23  1.008809282
## 24  1.073122133
## 28 -0.187455665
```

The generated lda1 is a linear combination of all the 10 predictors and it is given by (0.3679*word_freq_free)* + *(0.0988*capital_run_length_longest) + (0.0616*capital_run_length_average)* - *(0.2047*word_freq_george) - (0.2094*word_freq_you)* + *(0.5365*word_freq_your) - (0.2215*world_freq_hp)* + *(0.3710*word_freq_free) - (0.1637*word_freq_hpl)* + *(0.4404* `char_freq_!`)

Using predict function the posterior probabilities are generated which helps us in classifying spam and non-spam.

Question 6:

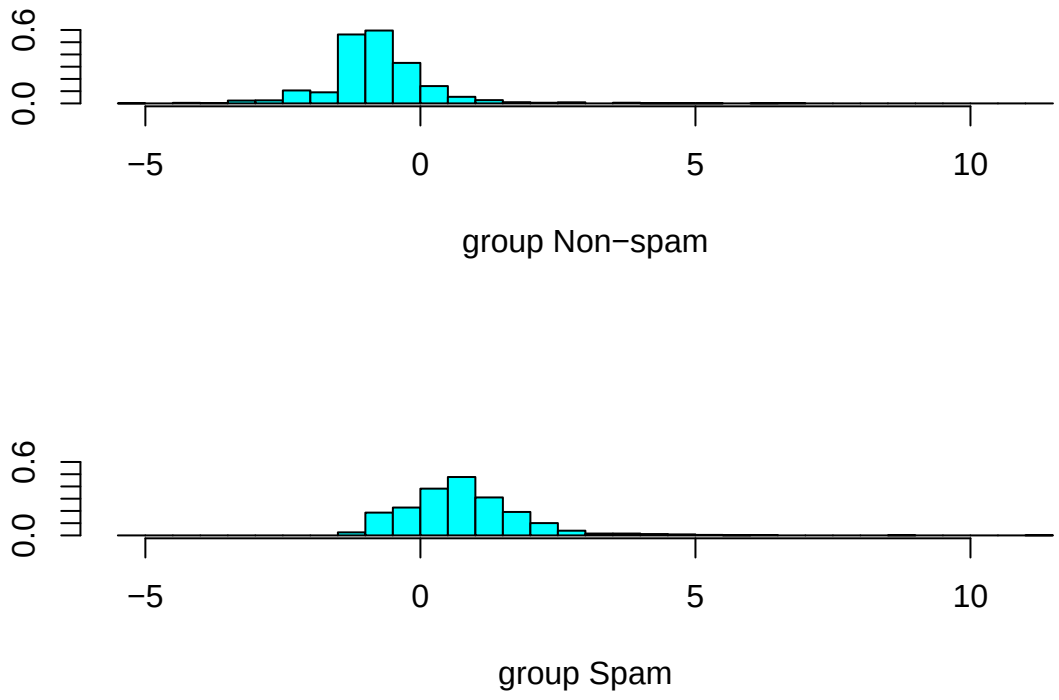
How many linear discriminants are in the model? Why?

There is only 1 discriminants in the model,LD1.As there are only two categories in the target variable, the no of discriminants in the model is 2-1=1.

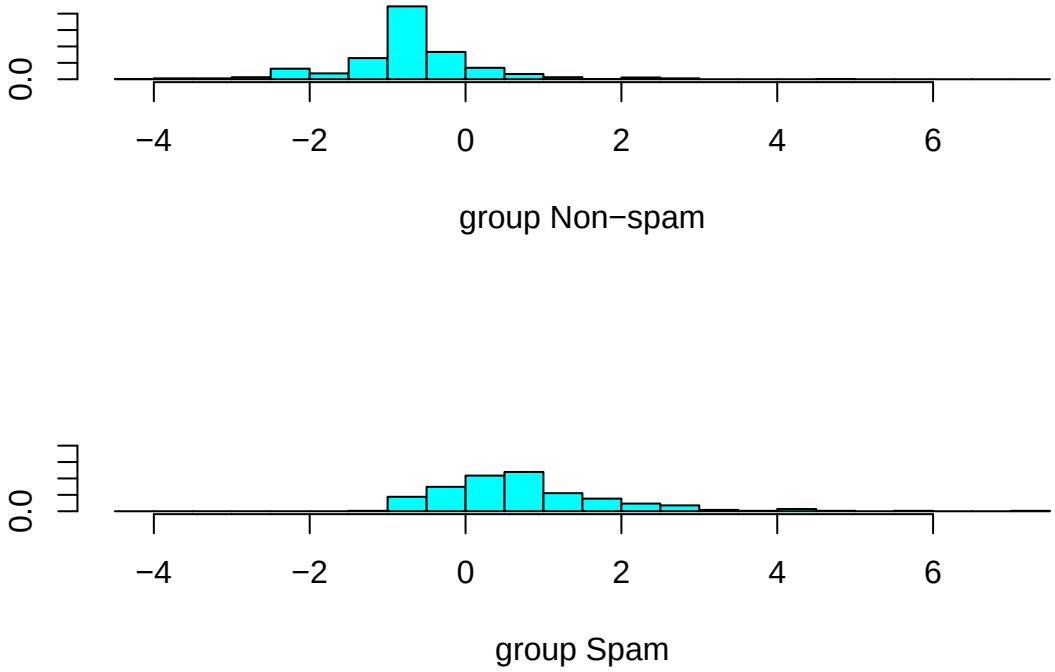
Question 7:

Generate LDA plot using the training and validation data. What information is presented in these plots? How are they different?

```
plot(lda1)
```




```
lda2<-lda(spam~.,data=valid.df.norm)
plot(lda2)
```



When we plot a graph of using train data set, we can see before 0 the most of the data is in NON SPAM class. In the graph below it we can see that most of the data is in SPAM class.

We can see the same trend when we plot a graph using validation data set, This indicates the model is good for class separation.

Question 8:

Generate the relevant confusion matrix. What are the sensitivity and specificity?

```
pred1.valid <- predict(lda1, valid.df.norm)

# Confusion matrix
acc1 <- table( pred1.valid$class, valid.df.norm$spam)  # pred v actual
confusionMatrix(acc1)
```

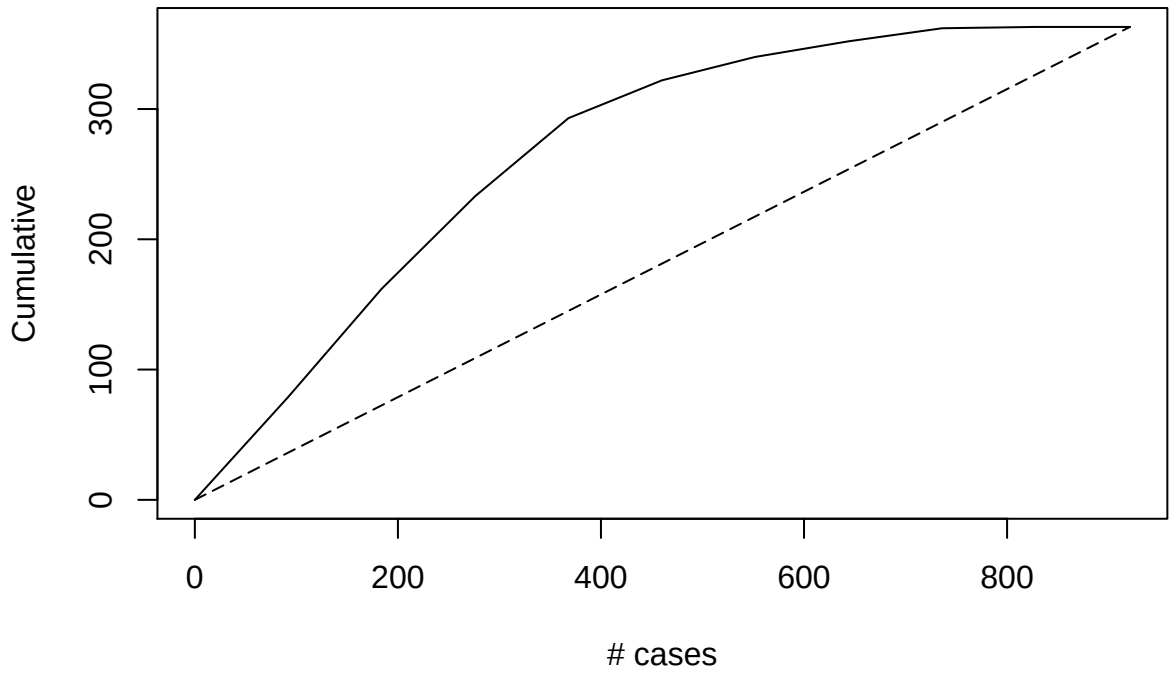
```
## Confusion Matrix and Statistics
##
##
##            Non-spam Spam
##   Non-spam      508  109
##   Spam           50  254
##
##                Accuracy : 0.8274
##                  95% CI : (0.8014, 0.8512)
##     No Information Rate : 0.6059
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.628
##
##  Mcnemar's Test P-Value : 4.231e-06
##
##             Sensitivity : 0.9104
##             Specificity : 0.6997
##          Pos Pred Value : 0.8233
##          Neg Pred Value : 0.8355
##              Prevalence : 0.6059
##          Detection Rate : 0.5516
##    Detection Prevalence : 0.6699
##       Balanced Accuracy : 0.8051
##
##        'Positive' Class : Non-spam
##
```

The Sensitivity of the model is 0.910 and Specificity is 0.700.

Question 9:

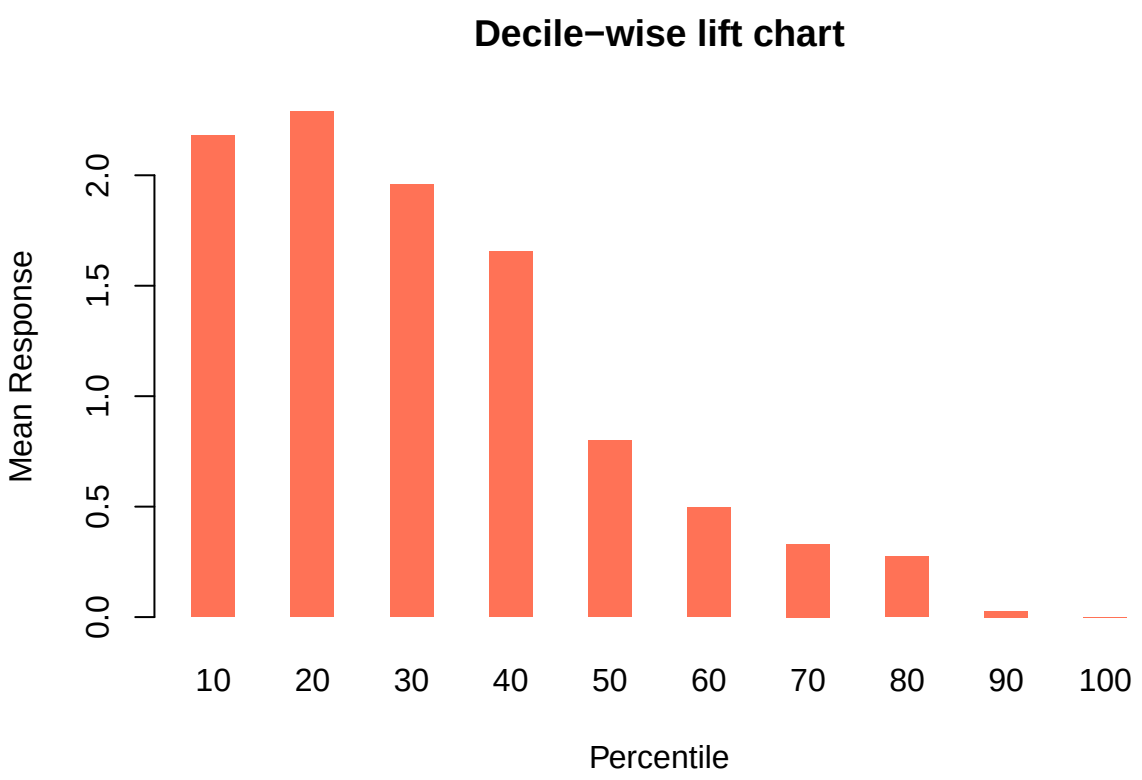
Generate lift and decile charts for the validation dataset and evaluate the effectiveness of the model in identifying spams.

```
library('gains')
gain <- gains(valid.df1.norm$spam, pred1.valid$posterior[,2], groups = 10)
  ### Plot Lift Chart
plot(c(0,gain$cume.pct.of.total*sum(valid.df1.norm$spam))~c(0,gain$cume.obs)
     ,xlab = "# cases",
     ylab = "Cumulative", main = "", type = "l")
lines(c(0,sum(valid.df1.norm$spam))~c(0, dim(valid.df1.norm)[1]), lty = 5)
```




```
###Plot Decile Chart

barplot(gain$mean.resp/mean(valid.df1.norm$spam), names.arg = gain$depth, space = 1.3,
        xlab = "Percentile", ylab = "Mean Response", main = "Decile-wise lift chart",
        col = "coral1", border = NA)
```



The ideal lift chart would be the one where we are able to predict all 304 spams in the initial 304 cases. In our model it takes almost around 600 cases to predict all 304 spams.We can predict spams using almost 60% of validation data. Hence our model is decent in predicting spams.

In the decile chart, the mean response increases and then starts to decrease. The ideal decile chart should be the one where mean response is highest in the first decile and gradually decrease.

Question 10:

Does accuracy of model changes if you use a probability threshold of 0.2. Explain your answer.

```
sum(pred2.valid$posterior[, 2] >=.5)
```

```
## [1] 304
```

```
sum(pred2.valid$posterior[, 2] >=.2)
```

```
## [1] 545
```

```
pred_5 <- factor( ifelse(pred2.valid$posterior[,2] >= 0.5, 1, 0) )
pred_2 <- factor( ifelse(pred2.valid$posterior[,2] >= 0.2, 1, 0) )

confusionMatrix(table(pred_5,valid.df1.norm$spam))
```

```
## Confusion Matrix and Statistics
##
##
## pred_5   0   1
##      0 508 109
##      1  50 254
```

```
##
##                Accuracy : 0.8274
##                  95% CI : (0.8014, 0.8512)
##     No Information Rate : 0.6059
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.628
##
##  Mcnemar's Test P-Value : 4.231e-06
##
##             Sensitivity : 0.9104
##             Specificity : 0.6997
##          Pos Pred Value : 0.8233
##          Neg Pred Value : 0.8355
##              Prevalence : 0.6059
##          Detection Rate : 0.5516
##    Detection Prevalence : 0.6699
##       Balanced Accuracy : 0.8051
##
##        'Positive' Class : 0
##
```

```
confusionMatrix(table(pred_2,valid.df1.norm$spam))
```

```
## Confusion Matrix and Statistics
##
##
## pred_2   0   1
##      0 352  24
##      1 206 339
##
##                Accuracy : 0.7503
##                  95% CI : (0.721, 0.7779)
##     No Information Rate : 0.6059
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.5192
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.6308
##             Specificity : 0.9339
##          Pos Pred Value : 0.9362
##          Neg Pred Value : 0.6220
##              Prevalence : 0.6059
##          Detection Rate : 0.3822
##    Detection Prevalence : 0.4083
##       Balanced Accuracy : 0.7824
##
##        'Positive' Class : 0
##
```

Yes, the accruracy changes, it decreases from 0.827 to 0.75 because more number of observations are classified as spam.