```
pacman::p_load(caret, data.table, MASS, ggplot2, gains, pROC, base)
options(digits = 3)
knitr::opts_chunk$set(echo = FALSE, fig.width=12, fig.height=6, fig.path = 'Figs/')
theme_set(theme_classic())
```

Answer 1:

```
##                                    1
## capital_run_length_total   309.148
## capital_run_length_longest  86.179
## capital_run_length_average   7.142
## word_freq_george             1.264
## word_freq_you                0.994
## word_freq_your               0.942
## word_freq_hp                 0.878
## word_freq_free               0.445
## word_freq_hpl                0.423
## char_freq_!                  0.404
```

#Removing numerical Spam column and replacing with character column due to normalization

Using only the predictors from answer 1

Split the data into training (80%) and validation/test set (20%)

Normalize the data

Estimate preprocessing parameters

Transform the data using the estimated parameters

Answer 2,3,4 &5:

Performing LDA

#prior probabilities : These probabilities are the ones that already exist in training data

#Coefficients: They help to create boundary of separation between the two different classes Spam and no Spam.

We can see that the coefficient of word_freq_your has greater value, suggesting that it has greater influence on Spams than the other variables.

Answer 5:

The discriminant function is a linear combination of 10 variables and uses statistical distance:

**(0.3875*word_freq_free) + (0.2466*word_freq_you) + (0.5715*word_freq_yo*
*- (0.2354*word_freq_hp) - (0.1506*word_freq_hpl) - (0.2104*word_freq_geor
+ (0.3268*char_freq_!) + (0.0527*capital_run_length_average) +
(0.1297*capital_run_length_longest) + (0.3725*capital_run_length_total)**

The estimated score maximises the between-class separation and minimises the within-class separation.

Using the estimated scores, the posterior probabilities are generated to provide the probability of an observation belonging to a class.

```
## Call:
## lda(Spam_no_spam ~ ., data = spam.train.norm)
##
## Prior probabilities of groups:
## NotSpam    Spam
##   0.606   0.394
##
## Group means:
##         word_freq_free word_freq_you word_freq_your word_freq_hp
## NotSpam         -0.203        -0.231         -0.318        0.206
## Spam             0.312         0.355          0.489       -0.317
##         word_freq_hpl word_freq_george `char_freq_!`
## NotSpam         0.185            0.146        -0.201
## Spam           -0.284           -0.224         0.309
##         capital_run_length_average capital_run_length_longest
## NotSpam                    -0.0853                     -0.169
## Spam                        0.1311                      0.259
##         capital_run_length_total
## NotSpam                   -0.200
## Spam                       0.308
##
## Coefficients of linear discriminants:
##                                  LD1
## word_freq_free                0.3875
```

```
## word_freq_you               0.2466
## word_freq_your              0.5715
## word_freq_hp               -0.2354
## word_freq_hpl              -0.1506
## word_freq_george           -0.2104
## `char_freq_!`               0.3268
## capital_run_length_average  0.0527
## capital_run_length_longest  0.1297
## capital_run_length_total    0.3725
```

```
## NotSpam    Spam
##   0.606   0.394
```

```
##                                LD1
## word_freq_free              0.3875
## word_freq_you               0.2466
## word_freq_your              0.5715
## word_freq_hp               -0.2354
## word_freq_hpl              -0.1506
## word_freq_george           -0.2104
## `char_freq_!`               0.3268
## capital_run_length_average  0.0527
## capital_run_length_longest  0.1297
## capital_run_length_total    0.3725
```

```
##         LD1
## 1  -0.84633
## 2   3.68654
## 3   0.83065
## 4  -0.56349
## 5   0.00457
## 6   1.40882
## 7   0.77136
## 8   0.71558
## 9   1.01091
## 10  0.15680
## 11  1.94065
## 12 -0.14989
## 13 -0.84338
## 14  0.72253
## 15 -0.18500
## 16  1.96024
## 17  0.30550
## 18 -0.20094
## 19  2.70037
## 20  1.39788
## 21  0.10767
## 22  2.24426
## 23  1.31168
## 24  0.65789
## 25  1.00153
## 26  0.54663
## 27  0.54663
```

```
## 28  0.14994
## 29 -0.40011
## 30  0.50460
## 31  1.29104
## 32  0.30076
## 33  1.61577
## 34 -0.79387
## 35  0.51663
## 36  0.45497
## 37 -0.87658
## 38  0.63095
## 39 -0.23802
## 40  1.69006
## 41  0.82620
## 42  0.01148
## 43  0.49981
## 44  1.41565
## 45 -0.07470
## 46 -0.25776
## 47  0.72212
## 48 -0.54954
## 49  0.62342
## 50  1.40217
```

Answer 6:

There is only 1 linear discriminant (LD1) in the model because there are only 2 classes - Spam and Non Spam. Number of Linear Discriminants = Number of classes - 1. 100% separation between the 2 classes Spam and Non Spam is achieved by this single discriminant. Linear Discriminant Estimated Scores ($x) are predicted for each observation in the training/validation dataset using the LD1 coefficients

Answer 7:

Generate LDA plot

plot training

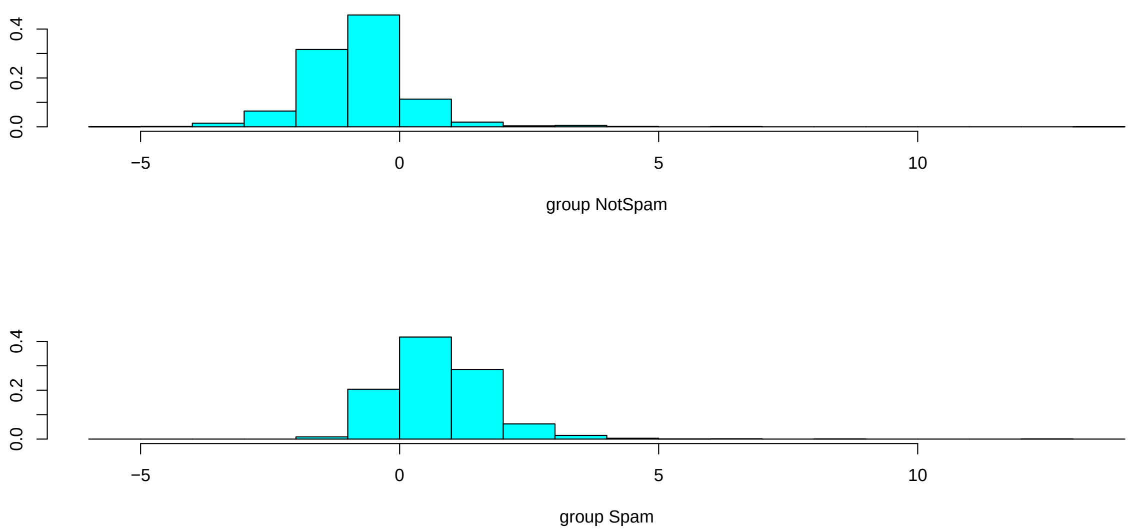
We find almost similar shaped curves for both training and valdation.

Answer 8:

Confusion matrix, check model accuracy

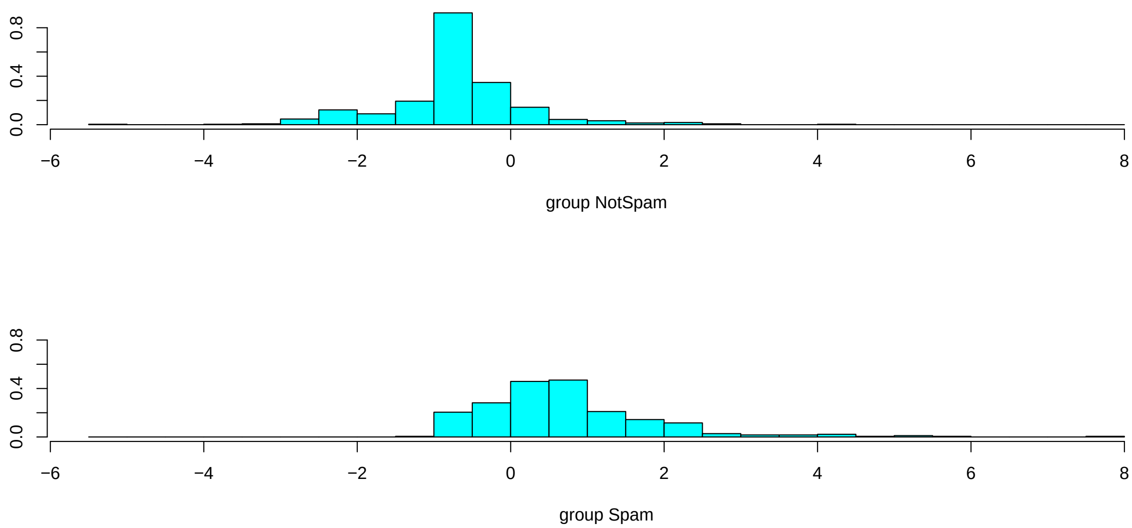
```
## Call:
## lda(Spam_no_spam ~ ., data = spam.train.norm)
##
## Prior probabilities of groups:
## NotSpam    Spam
##   0.606   0.394
##
## Group means:
##         word_freq_free word_freq_you word_freq_your word_freq_hp
## NotSpam         -0.203        -0.231         -0.318        0.206
## Spam             0.312         0.355          0.489       -0.317
##         word_freq_hpl word_freq_george `char_freq_!`
## NotSpam         0.185            0.146        -0.201
## Spam           -0.284           -0.224         0.309
##         capital_run_length_average capital_run_length_longest
## NotSpam                    -0.0853                     -0.169
## Spam                        0.1311                      0.259
##         capital_run_length_total
## NotSpam                   -0.200
## Spam                       0.308
##
## Coefficients of linear discriminants:
```

```
##                                LD1
## word_freq_free               0.3875
## word_freq_you                0.2466
## word_freq_your               0.5715
## word_freq_hp                -0.2354
## word_freq_hpl               -0.1506
## word_freq_george            -0.2104
## `char_freq_!`                0.3268
## capital_run_length_average   0.0527
## capital_run_length_longest   0.1297
## capital_run_length_total     0.3725
```




```
## Call:
## lda(Spam_no_spam ~ ., data = spam.valid.norm)
##
## Prior probabilities of groups:
## NotSpam    Spam
##   0.606   0.394
##
## Group means:
##         word_freq_free word_freq_you word_freq_your word_freq_hp
## NotSpam         -0.213        -0.193         -0.269        0.161
## Spam             0.371         0.240          0.495       -0.320
##         word_freq_hpl word_freq_george `char_freq_!`
## NotSpam         0.195            0.216        -0.159
## Spam           -0.281           -0.224         0.319
##         capital_run_length_average capital_run_length_longest
## NotSpam                    -0.0908                     -0.181
## Spam                        0.0942                      0.156
##         capital_run_length_total
## NotSpam                   -0.229
## Spam                       0.205
##
```

```
## Coefficients of linear discriminants:
##                                  LD1
## word_freq_free               0.54536
## word_freq_you                0.12850
## word_freq_your               0.40938
## word_freq_hp                -0.26697
## word_freq_hpl               -0.24728
## word_freq_george            -0.20626
## `char_freq_!`                0.20537
## capital_run_length_average -0.00742
## capital_run_length_longest  0.57710
## capital_run_length_total    0.32996
```




```
## Confusion Matrix and Statistics
##
##
##           NotSpam Spam
##   NotSpam     502  118
##   Spam         55  244
##
##                Accuracy : 0.812
##                  95% CI : (0.785, 0.837)
##     No Information Rate : 0.606
##     P-Value [Acc > NIR] : < 0.0000000000000002
##
##                   Kappa : 0.593
##
##  Mcnemar's Test P-Value : 0.00000243
##
##             Sensitivity : 0.901
##             Specificity : 0.674
##          Pos Pred Value : 0.810
##          Neg Pred Value : 0.816
```

```
##             Prevalence : 0.606
##         Detection Rate : 0.546
##   Detection Prevalence : 0.675
##      Balanced Accuracy : 0.788
##
##       'Positive' Class : NotSpam
##
```

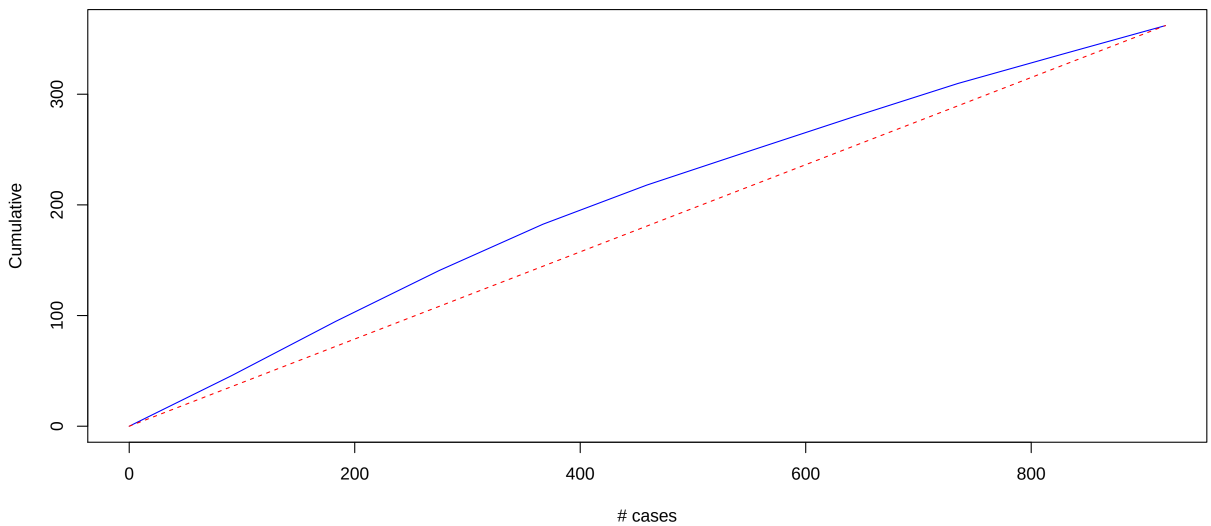
sensitivity = 0.901 = 505/(502+55)

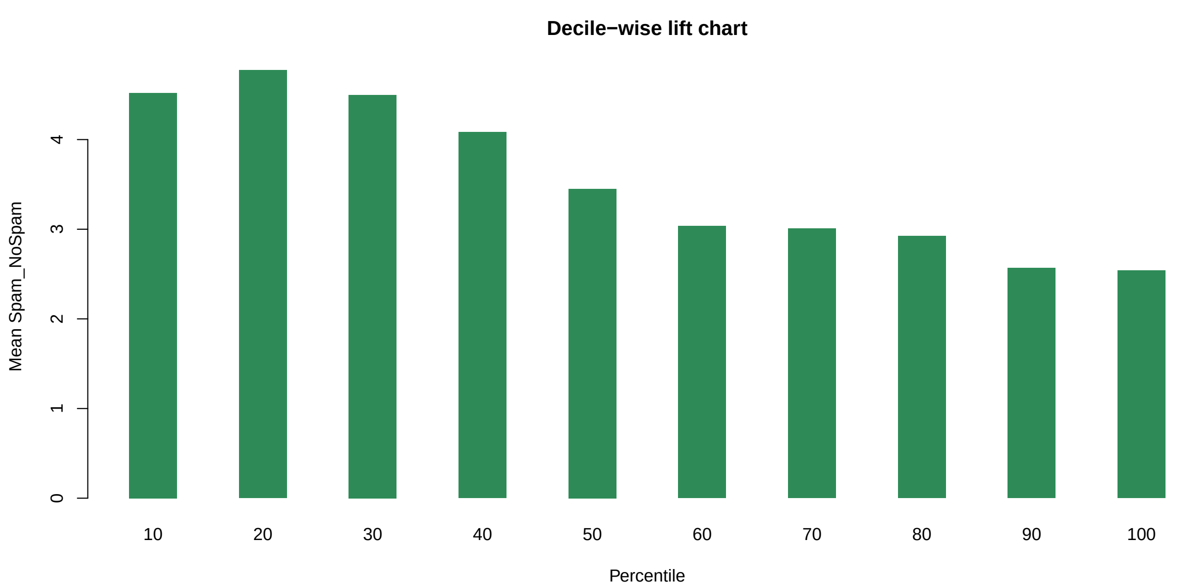
specificity = 0.674 = 242/(244+118)

Answer 9:

Lift Chart

Computing gains relative to spam





#The lift chart generated from the model has higher slope and hence the model is more effective.

Decile Lift Charts

The majority of classified observations fall in the first 3 or 4 deciles. Hencethe generated model is effective.

Answer 10:

#0.5 is the default threshold

#considering probability threshold of 0.2

```
## [1] 299
```

```
## [1] 529
```

#accuracy of model changes if we use a different probability threshold. When threshold decreases, the accuracy of the model decreases