CSCI 550 Project 1

Heather Koyuk, Muzhou Chen, Greg Martin

9/27/2022

Executive Summary

The first section EDA starts at analyzing the correlations of all predictors. As a result, the correlation between total.day.minutes and total.day.charge, total.eve.minutes and total.eve.charge, total.night.minutes and total.night.charge, total.intl.minutes and total.intl.charge are high. This information will be used in the future model selection to avoid multicollinearity. The visualization is shown as matrix of scatter plots from pairs(). Since the output from summary shows that the data contains 3333 observations, outliers tend to not be a problem. Based on our preliminary work we decided to use three potential predictors: number.vmail.messages, total.intl.calls, and customer.service.calls for fitting in the logistic regression.

In the second section LDA, we performed linear discriminant analysis. By using 10-fold cross validation we have a misclassification rate of 0.1428189 with a average AUC of 0.6382374. The process prints out every 2*2 ROC curve and iteration. The next section QDA applies quadratic discriminant analysis, providing an average misclassification rate of 0.1371168 and average AUC 0.6712765. The third section Naive Bayes makes the test with assuming features are independent in each class, which provides average misclassification rate 0.1352979. Every section shows a plot of ROC Curve-LDA: Iteration plot for each 10 fold iteration.

Finally, we retest LDA with the four most significant predictors (LDA with next most significant predictor), all potentially relevant predictors (LDA All potentially relevant predictors), and with the last 5 p-values (LDA All call metrics) (for fitting in logistic regression). Our first model with 3 predictors performs the best of the various LDA experiments, while the overall lowest average misclassification rate of 0.1352979 occurs in our Naive Bayes.

Since we are using a greedy method of forward selection for variable selection, potential future work could continue trying the different combinations of predictors to get the global optimal model for prediction.

EDA

```
num_c <- Customer_telecom %>%
  dplyr::select(
    account.length,
    number.vmail.messages,
    total.day.minutes,
    total.day.calls,
    total.day.charge,
    total.eve.minutes,
    total.eve.calls,
    total.eve.charge,
    total.night.minutes,
    total.night.calls,
    total.night.charge,
```

```
    total.intl.minutes,
    total.intl.calls,
    total.intl.charge,
    customer.service.calls
  )

cor(num_c)
```

```
##                         account.length number.vmail.messages total.day.minutes
## account.length            1.000000000         -0.0046278243      0.0062160205
## number.vmail.messages    -0.004627824          1.0000000000      0.0007782741
## total.day.minutes         0.006216021          0.0007782741      1.0000000000
## total.day.calls           0.038469882         -0.0095480677      0.0067504139
## total.day.charge          0.006214135          0.0007755235      0.9999999522
## total.eve.minutes        -0.006757142          0.0175620343      0.0070425110
## total.eve.calls           0.019259967         -0.0058643513      0.0157689932
## total.eve.charge         -0.006745302          0.0175777801      0.0070290353
## total.night.minutes      -0.008955192          0.0076811359      0.0043233666
## total.night.calls        -0.013176275          0.0071230629      0.0229724555
## total.night.charge       -0.008959535          0.0076632904      0.0043003570
## total.intl.minutes        0.009513902          0.0028561959     -0.0101545856
## total.intl.calls          0.020661428          0.0139573387      0.0080333570
## total.intl.charge         0.009545675          0.0028836579     -0.0100919742
## customer.service.calls   -0.003795939         -0.0132625831     -0.0134231864
##                         total.day.calls total.day.charge total.eve.minutes
## account.length               0.038469882     0.0062141347      -0.006757142
## number.vmail.messages       -0.009548068     0.0007755235       0.017562034
## total.day.minutes            0.006750414     0.9999999522       0.007042511
## total.day.calls              1.000000000     0.0067529620      -0.021451408
## total.day.charge             0.006752962     1.0000000000       0.007049607
## total.eve.minutes           -0.021451408     0.0070496072       1.000000000
## total.eve.calls              0.006462114     0.0157692822      -0.011430108
## total.eve.charge            -0.021449263     0.0070361315       0.999999776
## total.night.minutes          0.022937845     0.0043238794      -0.012583678
## total.night.calls           -0.019556965     0.0229724195       0.007585643
## total.night.charge           0.022926638     0.0043008608      -0.012592806
## total.intl.minutes           0.021564794    -0.0101568616      -0.011034714
## total.intl.calls             0.004574268     0.0080315720       0.002541292
## total.intl.charge            0.021666095    -0.0100942572      -0.011066621
## customer.service.calls      -0.018941930    -0.0134269694      -0.012984553
##                         total.eve.calls total.eve.charge total.night.minutes
## account.length               0.019259967     -0.006745302        -0.008955192
## number.vmail.messages       -0.005864351      0.017577780         0.007681136
## total.day.minutes            0.015768993      0.007029035         0.004323367
## total.day.calls              0.006462114     -0.021449263         0.022937845
## total.day.charge             0.015769282      0.007036131         0.004323879
## total.eve.minutes           -0.011430108      0.999999776        -0.012583678
## total.eve.calls              1.000000000     -0.011422894        -0.002092768
## total.eve.charge            -0.011422894      1.000000000        -0.012592020
## total.night.minutes         -0.002092768     -0.012592020         1.000000000
## total.night.calls            0.007709706      0.007595843         0.011203856
## total.night.charge          -0.002055984     -0.012601142         0.999999215
## total.intl.minutes           0.008702881     -0.011042582        -0.015207297
```

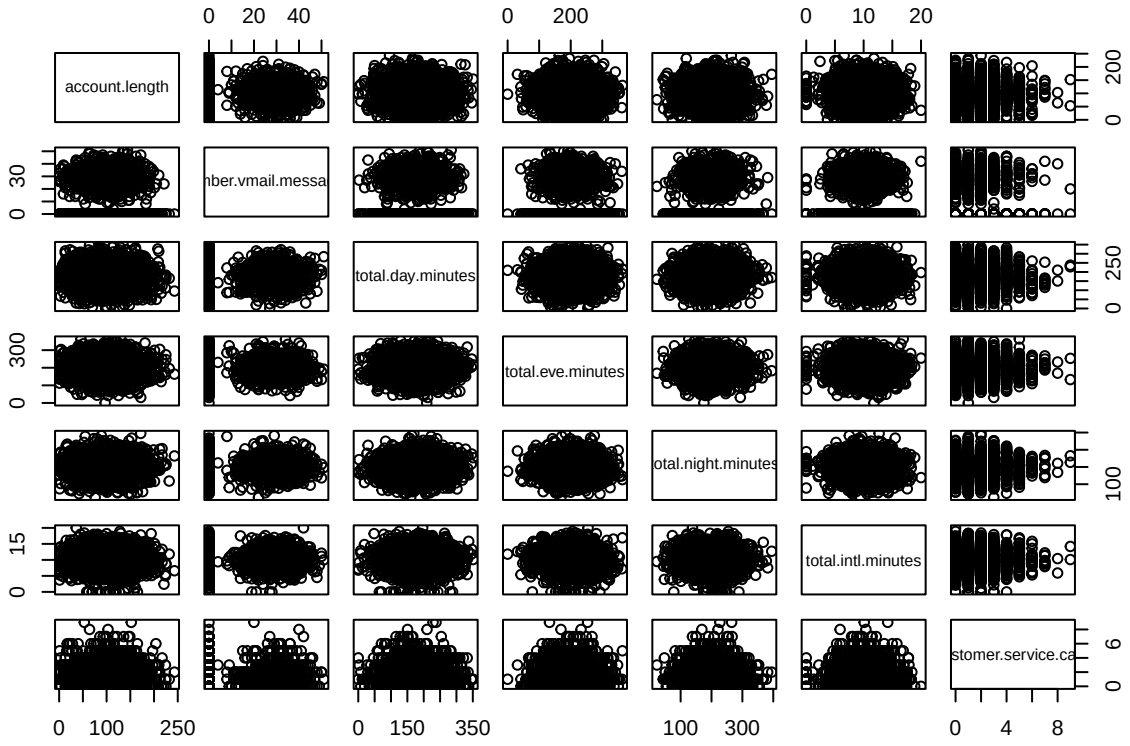
```
## total.intl.calls           0.017433692      0.002541458        -0.012353432
## total.intl.charge          0.008673858     -0.011074499        -0.015179849
## customer.service.calls     0.002422575     -0.012987407        -0.009287613
##                        total.night.calls total.night.charge total.intl.minutes
## account.length            -0.0131762751       -0.008959535        0.009513902
## number.vmail.messages      0.0071230629        0.007663290        0.002856196
## total.day.minutes          0.0229724555        0.004300357       -0.010154586
## total.day.calls           -0.0195569654        0.022926638        0.021564794
## total.day.charge           0.0229724195        0.004300861       -0.010156862
## total.eve.minutes          0.0075856431       -0.012592806       -0.011034714
## total.eve.calls            0.0077097055       -0.002055984        0.008702881
## total.eve.charge           0.0075958430       -0.012601142       -0.011042582
## total.night.minutes        0.0112038563        0.999999215       -0.015207297
## total.night.calls          1.0000000000        0.011187820       -0.013604996
## total.night.charge         0.0111878197        1.000000000       -0.015213526
## total.intl.minutes        -0.0136049964       -0.015213526        1.000000000
## total.intl.calls           0.0003045795       -0.012329215        0.032303884
## total.intl.charge         -0.0136301696       -0.015186139        0.999992742
## customer.service.calls    -0.0128019273       -0.009276954       -0.009639680
##                        total.intl.calls total.intl.charge
## account.length            0.0206614284       0.009545675
## number.vmail.messages     0.0139573387       0.002883658
## total.day.minutes         0.0080333570      -0.010091974
## total.day.calls           0.0045742682       0.021666095
## total.day.charge          0.0080315720      -0.010094257
## total.eve.minutes         0.0025412917      -0.011066621
## total.eve.calls           0.0174336921       0.008673858
## total.eve.charge          0.0025414580      -0.011074499
## total.night.minutes      -0.0123534324      -0.015179849
## total.night.calls         0.0003045795      -0.013630170
## total.night.charge       -0.0123292150      -0.015186139
## total.intl.minutes        0.0323038841       0.999992742
## total.intl.calls          1.0000000000       0.032372145
## total.intl.charge         0.0323721453       1.000000000
## customer.service.calls   -0.0175605992      -0.009674732
##                        customer.service.calls
## account.length                    -0.003795939
## number.vmail.messages             -0.013262583
## total.day.minutes                 -0.013423186
## total.day.calls                   -0.018941930
## total.day.charge                  -0.013426969
## total.eve.minutes                 -0.012984553
## total.eve.calls                    0.002422575
## total.eve.charge                  -0.012987407
## total.night.minutes               -0.009287613
## total.night.calls                 -0.012801927
## total.night.charge                -0.009276954
## total.intl.minutes                -0.009639680
## total.intl.calls                  -0.017560599
## total.intl.charge                 -0.009674732
## customer.service.calls             1.000000000
```

```
summary(Customer_telecom)
```

```
##     state           account.length   area.code         phone.number
##  Length:3333        Min.   :  1.0   Length:3333        Length:3333
##  Class :character   1st Qu.: 74.0   Class :character   Class :character
##  Mode  :character   Median :101.0   Mode  :character   Mode  :character
##                     Mean   :101.1
##                     3rd Qu.:127.0
##                     Max.   :243.0
##  international.plan voice.mail.plan    number.vmail.messages total.day.minutes
##  Length:3333        Length:3333        Min.   : 0.000        Min.   :  0.0
##  Class :character   Class :character   1st Qu.: 0.000        1st Qu.:143.7
##  Mode  :character   Mode  :character   Median : 0.000        Median :179.4
##                                        Mean   : 8.099        Mean   :179.8
##                                        3rd Qu.:20.000        3rd Qu.:216.4
##                                        Max.   :51.000        Max.   :350.8
##  total.day.calls total.day.charge total.eve.minutes total.eve.calls
##  Min.   :  0.0   Min.   : 0.00    Min.   :  0.0     Min.   :  0.0
##  1st Qu.: 87.0   1st Qu.:24.43    1st Qu.:166.6     1st Qu.: 87.0
##  Median :101.0   Median :30.50    Median :201.4     Median :100.0
##  Mean   :100.4   Mean   :30.56    Mean   :201.0     Mean   :100.1
##  3rd Qu.:114.0   3rd Qu.:36.79    3rd Qu.:235.3     3rd Qu.:114.0
##  Max.   :165.0   Max.   :59.64    Max.   :363.7     Max.   :170.0
##  total.eve.charge total.night.minutes total.night.calls total.night.charge
##  Min.   : 0.00    Min.   : 23.2       Min.   : 33.0     Min.   : 1.040
##  1st Qu.:14.16    1st Qu.:167.0       1st Qu.: 87.0     1st Qu.: 7.520
##  Median :17.12    Median :201.2       Median :100.0     Median : 9.050
##  Mean   :17.08    Mean   :200.9       Mean   :100.1     Mean   : 9.039
##  3rd Qu.:20.00    3rd Qu.:235.3       3rd Qu.:113.0     3rd Qu.:10.590
##  Max.   :30.91    Max.   :395.0       Max.   :175.0     Max.   :17.770
##  total.intl.minutes total.intl.calls total.intl.charge customer.service.calls
##  Min.   : 0.00      Min.   : 0.000   Min.   :0.000     Min.   :0.000
##  1st Qu.: 8.50      1st Qu.: 3.000   1st Qu.:2.300     1st Qu.:1.000
##  Median :10.30      Median : 4.000   Median :2.780     Median :1.000
##  Mean   :10.24      Mean   : 4.479   Mean   :2.765     Mean   :1.563
##  3rd Qu.:12.10      3rd Qu.: 6.000   3rd Qu.:3.270     3rd Qu.:2.000
##  Max.   :20.00      Max.   :20.000   Max.   :5.400     Max.   :9.000
##     churn
##  Length:3333
##  Class :character
##  Mode  :character
##
##
##
```

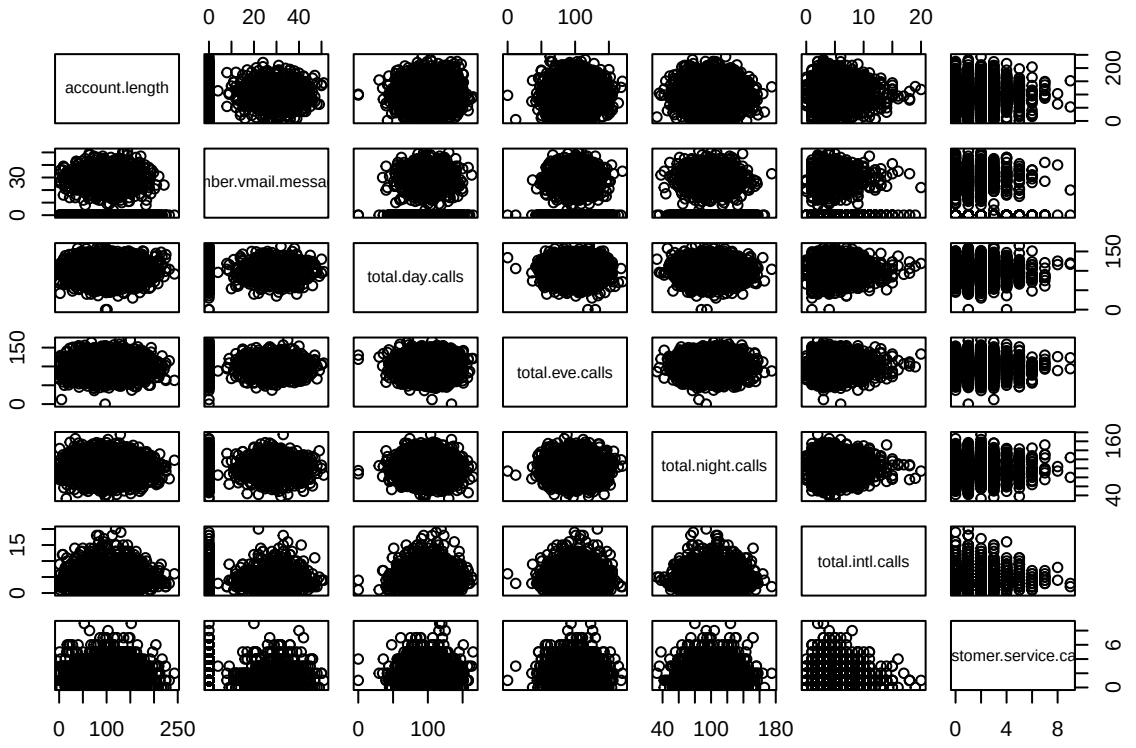
```
num_c <- num_c %>%
  dplyr::select(
    account.length,
    number.vmail.messages,
    total.day.minutes,
    total.day.calls,
    total.eve.minutes,
    total.eve.calls,
    total.night.minutes,
    total.night.calls,
    total.intl.minutes,
```

```
    total.intl.calls,
    customer.service.calls
  )

minutes_c <- num_c %>%
  dplyr::select(
    account.length,
    number.vmail.messages,
    total.day.minutes,
    total.eve.minutes,
    total.night.minutes,
    total.intl.minutes,
    customer.service.calls
  )

pairs(minutes_c)
```



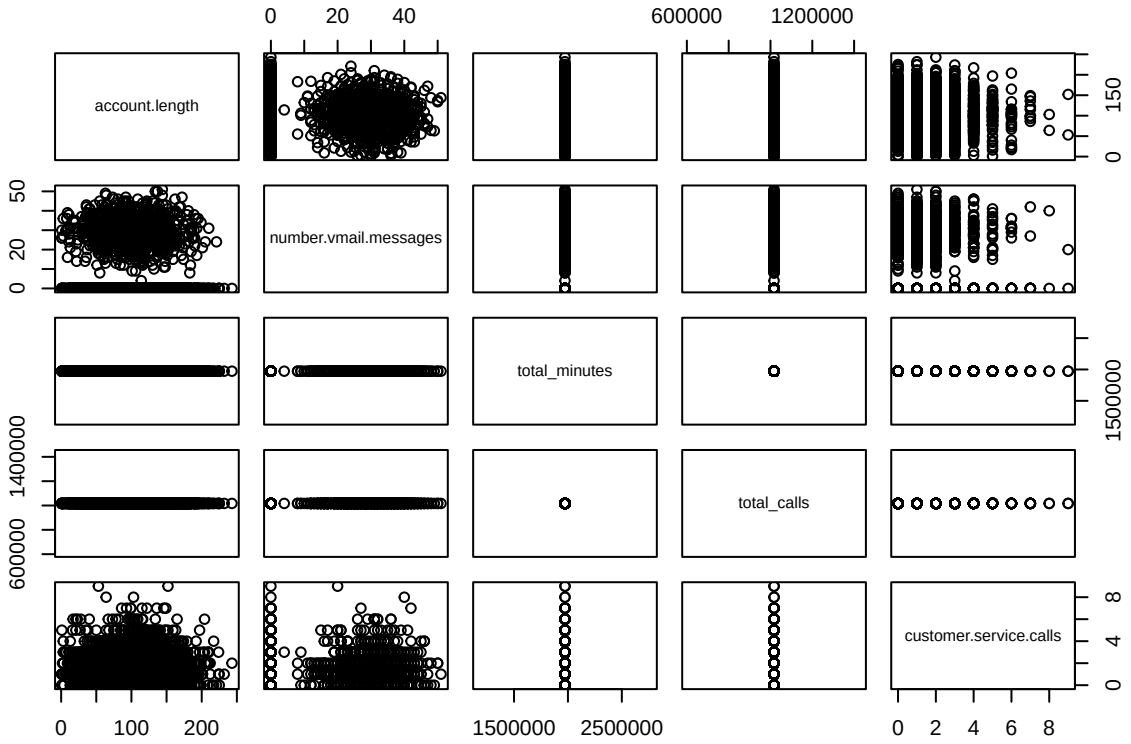

```
calls_c <-  Customer_telecom %>%
  dplyr::select(
    account.length,
    number.vmail.messages,
    total.day.calls,
    total.eve.calls,
    total.night.calls,
    total.intl.calls,
```

```
    customer.service.calls
  )
pairs(calls_c)
```




```
sums_c <- Customer_telecom %>%
  mutate(
    total_minutes = sum(
      total.day.minutes,
      total.eve.minutes,
      total.night.minutes,
      total.intl.minutes
    )
  ) %>%
  mutate(
    total_calls = sum(
      total.day.calls,
      total.eve.calls,
      total.night.calls,
      total.intl.calls
    )
  ) %>%
  dplyr::select(
    account.length,
    number.vmail.messages,
    total_minutes,
```

```
    total_calls,
    customer.service.calls
  )

pairs(sums_c)
```




```
Customer_telecom <- Customer_telecom %>%
  mutate(churn1 = if_else(churn == "False", 0, 1))

glm.fits <- glm(churn1 ~ total.day.minutes,
                data = Customer_telecom, family = binomial)
summary(glm.fits)
```

```
##
## Call:
## glm(formula = churn1 ~ total.day.minutes, family = binomial,
##     data = Customer_telecom)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.0241  -0.6001  -0.4902  -0.3738   2.8102
##
## Coefficients:
##                    Estimate Std. Error z value Pr(>|z|)
## (Intercept)       -3.929289   0.202823  -19.37   <2e-16 ***
```

```
## total.day.minutes  0.011272   0.000975   11.56   <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: 2758.3  on 3332  degrees of freedom
## Residual deviance: 2614.3  on 3331  degrees of freedom
## AIC: 2618.3
##
## Number of Fisher Scoring iterations: 5
```

```
glm.fits1 <- glm(
  churn1 ~ account.length +
    number.vmail.messages +
    total.day.minutes +
    total.day.calls +
    total.day.charge +
    total.eve.minutes +
    total.eve.calls +
    total.eve.charge +
    total.night.minutes +
    total.night.calls +
    total.night.charge +
    total.intl.minutes +
    total.intl.calls +
    total.intl.charge +
    customer.service.calls,
  data = Customer_telecom,
  family = binomial
)
summary(glm.fits1)
```

```
##
## Call:
## glm(formula = churn1 ~ account.length + number.vmail.messages +
##     total.day.minutes + total.day.calls + total.day.charge +
##     total.eve.minutes + total.eve.calls + total.eve.charge +
##     total.night.minutes + total.night.calls + total.night.charge +
##     total.intl.minutes + total.intl.calls + total.intl.charge +
##     customer.service.calls, family = binomial, data = Customer_telecom)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.7641  -0.5686  -0.4010  -0.2504   3.0143
##
## Coefficients:
##                          Estimate Std. Error z value Pr(>|z|)
## (Intercept)            -8.0194391  0.6829678 -11.742  < 2e-16 ***
## account.length          0.0014122  0.0013329   1.060  0.28936
## number.vmail.messages  -0.0246141  0.0045121  -5.455 4.89e-08 ***
## total.day.minutes      -0.6236691  3.1355848  -0.199  0.84234
## total.day.calls         0.0029075  0.0026459   1.099  0.27183
## total.day.charge        3.7430456 18.4447412   0.203  0.83919
```

```
## total.eve.minutes       0.4209563  1.5635207   0.269  0.78775
## total.eve.calls         0.0008723  0.0026313   0.332  0.74026
## total.eve.charge       -4.8742828 18.3943009  -0.265  0.79102
## total.night.minutes     0.0119276  0.8342215   0.014  0.98859
## total.night.calls       0.0009581  0.0027190   0.352  0.72457
## total.night.charge     -0.2044409 18.5377138  -0.011  0.99120
## total.intl.minutes     -1.4413457  5.0284283  -0.287  0.77439
## total.intl.calls       -0.0800674  0.0238065  -3.363  0.00077 ***
## total.intl.charge       5.6814577 18.6229945   0.305  0.76031
## customer.service.calls  0.4549897  0.0371831  12.236  < 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: 2758.3  on 3332  degrees of freedom
## Residual deviance: 2361.7  on 3317  degrees of freedom
## AIC: 2393.7
##
## Number of Fisher Scoring iterations: 5
```

Based on the result from our preprocess, the output from cor() shows all correlations between two variables. The two variables with high correlation will not be in the same model. The LRM glm.fits1 including all predictors provides a summary of z-Score and p-value. We will start fitting models with the predictors number.vmail.messages(p=4.89e-08), total.intl.calls(p=0.00077), and customer.service.calls(p<=2e-16).

Models

LDA

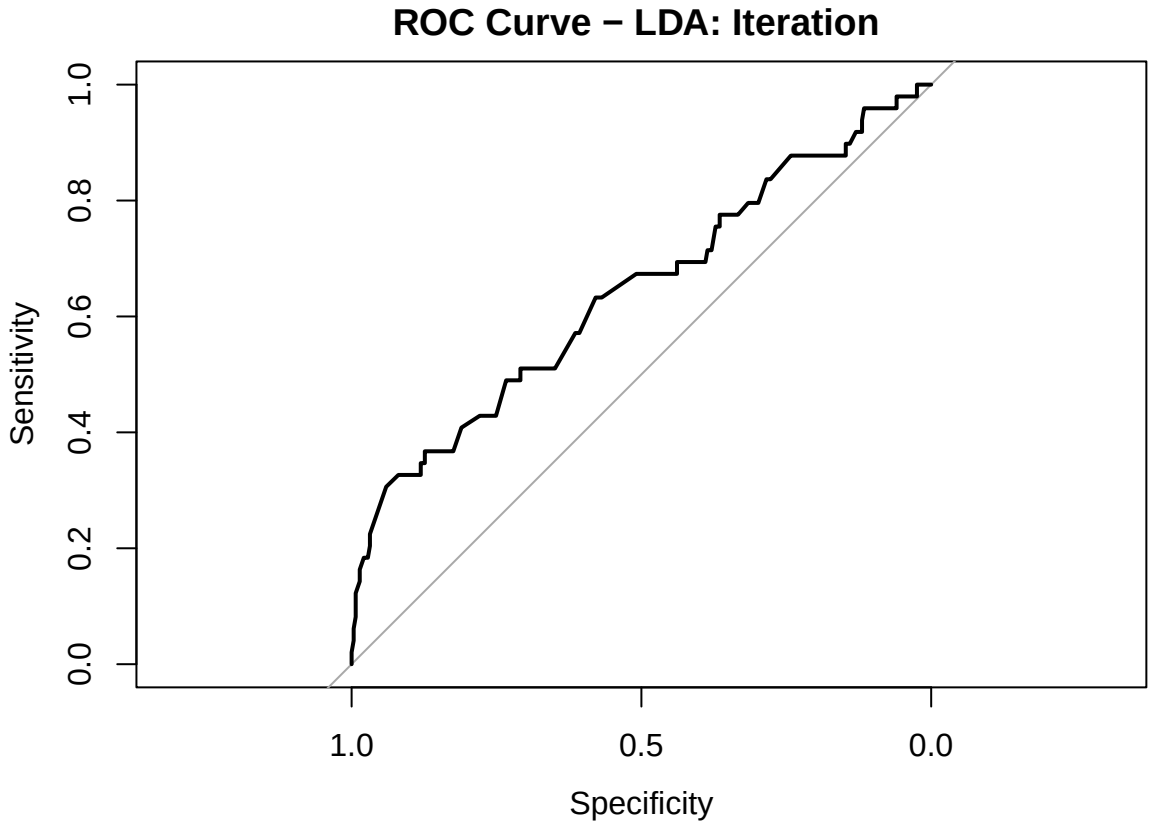
```
#Randomly shuffle the data
Customer_telecom <- Customer_telecom[sample(nrow(Customer_telecom)), ]

#Create 10 equally size folds
folds <- cut(seq(1, nrow(Customer_telecom)), breaks = 10, labels = FALSE)

totalMisclassificationRate = c()
runningTotal = c()

averageAUC = c()
auc = c()

#K-fold Cross Validation
for (i in 1:10) {
  testIndexes <- which(folds == i, arr.ind = TRUE)
  test_data <- Customer_telecom[testIndexes,]
  train_data <- Customer_telecom[-testIndexes,]
  m_lda <-
    lda(churn1 ~ number.vmail.messages + total.intl.calls + customer.service.calls,
        data = train_data)
  m_pred <- predict(m_lda, test_data)
  conf <-
    table(list(predicted = m_pred$class, observed = test_data$churn1))
  confusionMatrix(conf)
  print(conf)
  totalPreds = sum(conf)
  incorrectPreds <- conf[1, 2] + conf[2, 1]
  misclassificationRate <- incorrectPreds / totalPreds
  print(misclassificationRate)
  runningTotal <- c(runningTotal, misclassificationRate)

  roc_score = roc(response = test_data$churn1,
                  predictor = m_pred$posterior[, "1"])
  auc <- c(auc, roc_score$auc)
  print(roc_score$auc)
  plot(roc_score, main = "ROC Curve - LDA: Iteration ")
}
```

```
##          observed
## predicted   0   1
##         0 285  48
##         1   0   1
## [1] 0.1437126
```

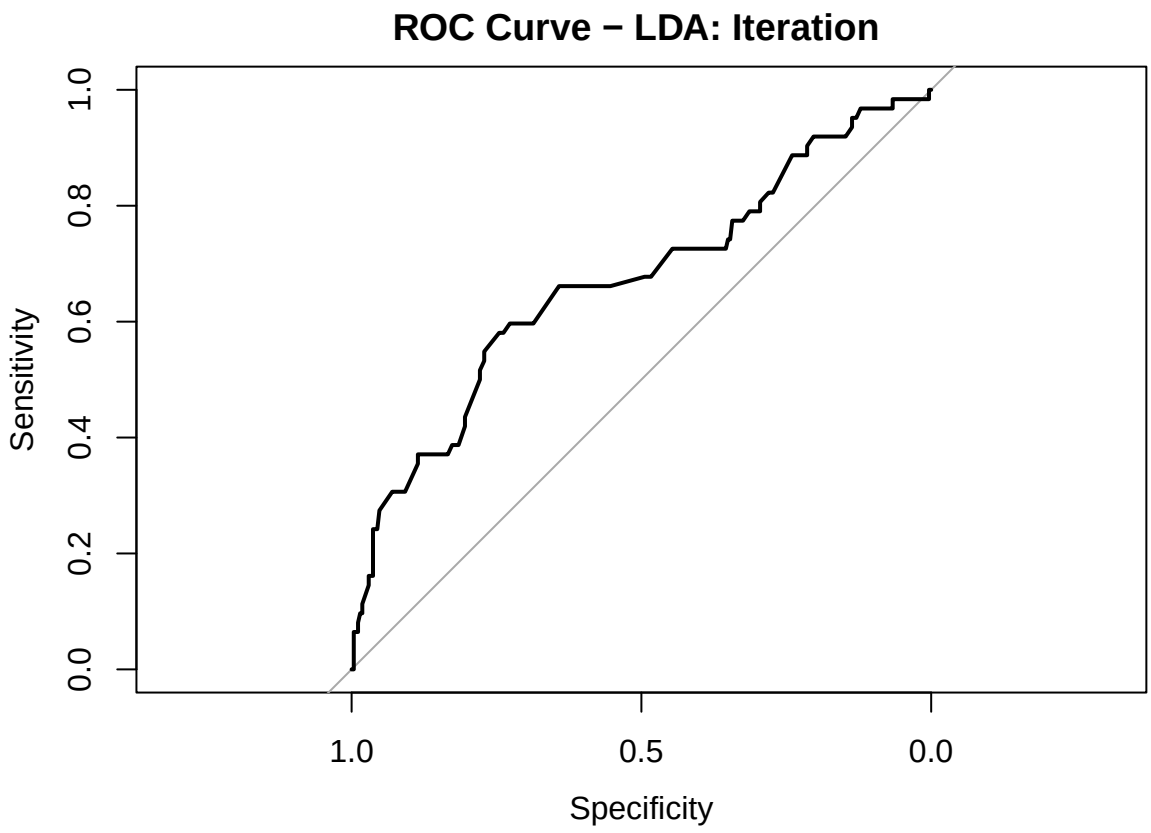
```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

```
## Area under the curve: 0.6421

##          observed
## predicted   0   1
##         0 270  62
##         1   1   0
## [1] 0.1891892

## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```



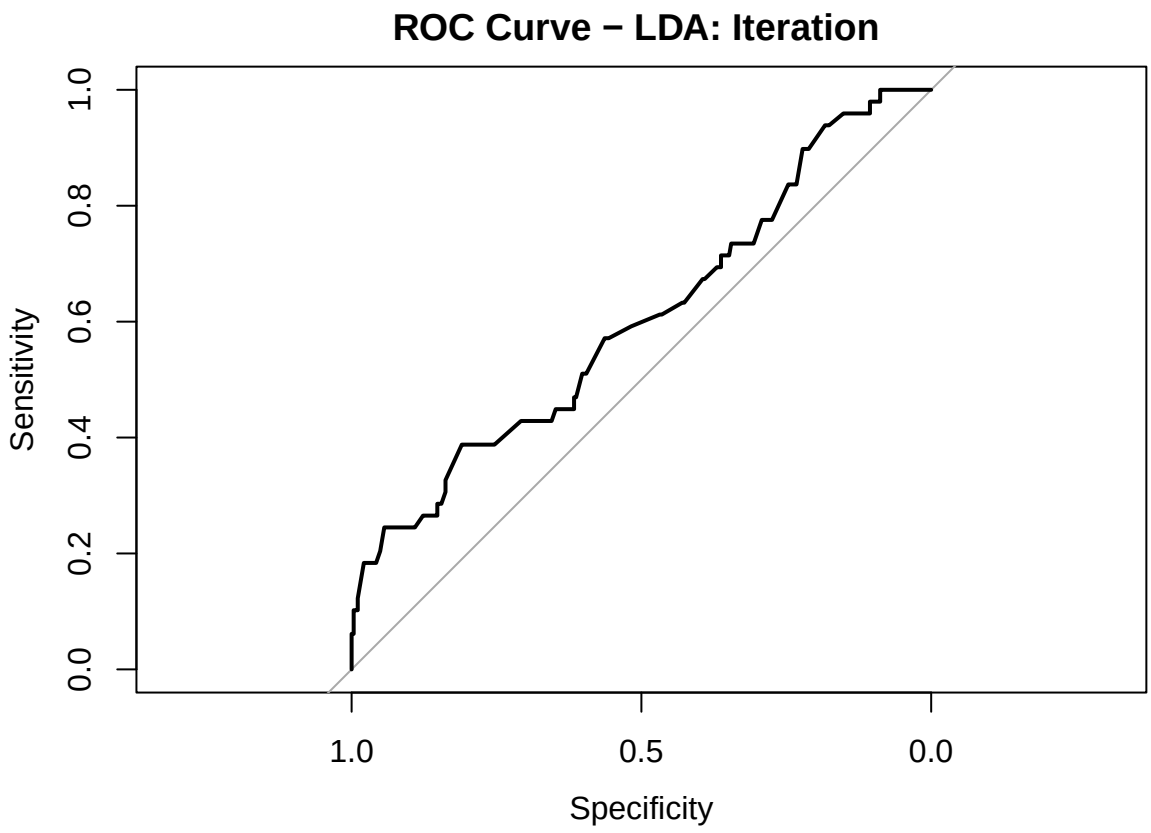

```
## Area under the curve: 0.6691

##          observed
## predicted   0   1
##         0 284  46
##         1   0   3
## [1] 0.1381381

## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```




```
## Area under the curve: 0.6031
```

```
##          observed
## predicted   0   1
##         0 281  42
##         1   5   5
## [1] 0.1411411
```

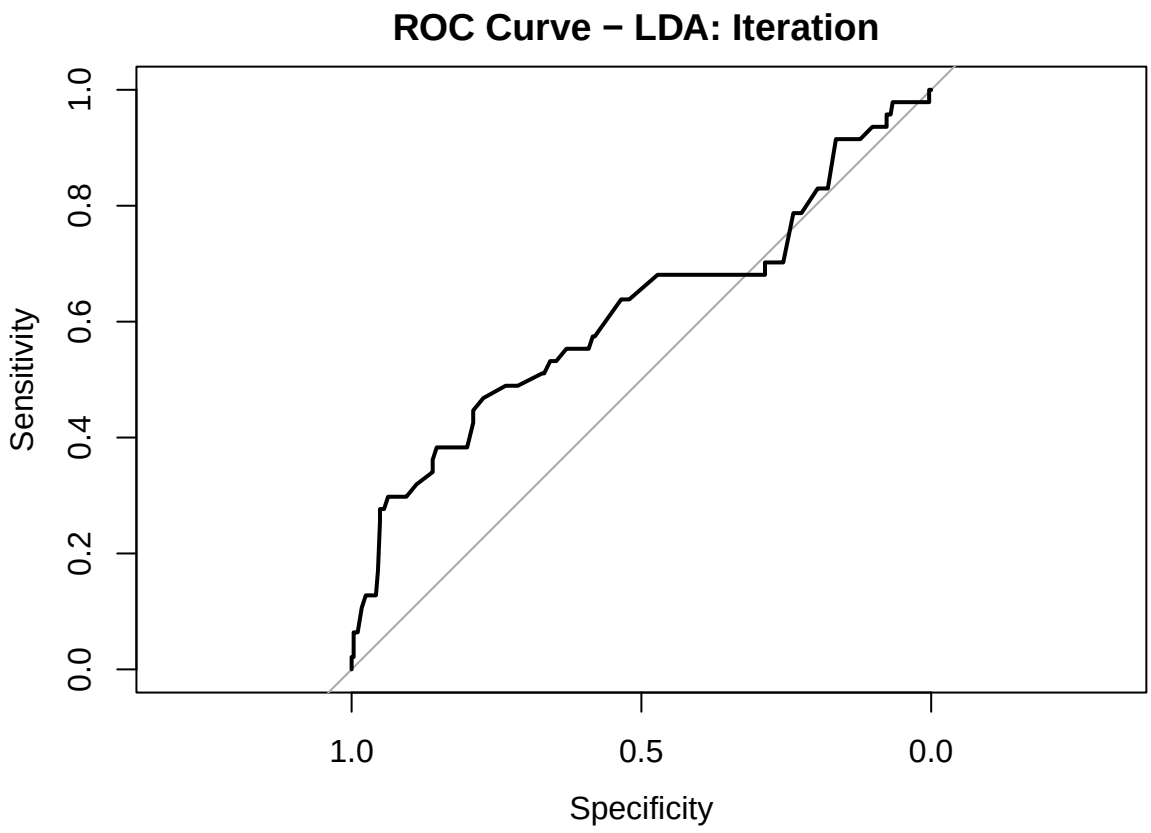
```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```




```
## Area under the curve: 0.6138
```

```
##          observed
## predicted   0   1
##         0 285  45
##         1   1   3
## [1] 0.1377246
```

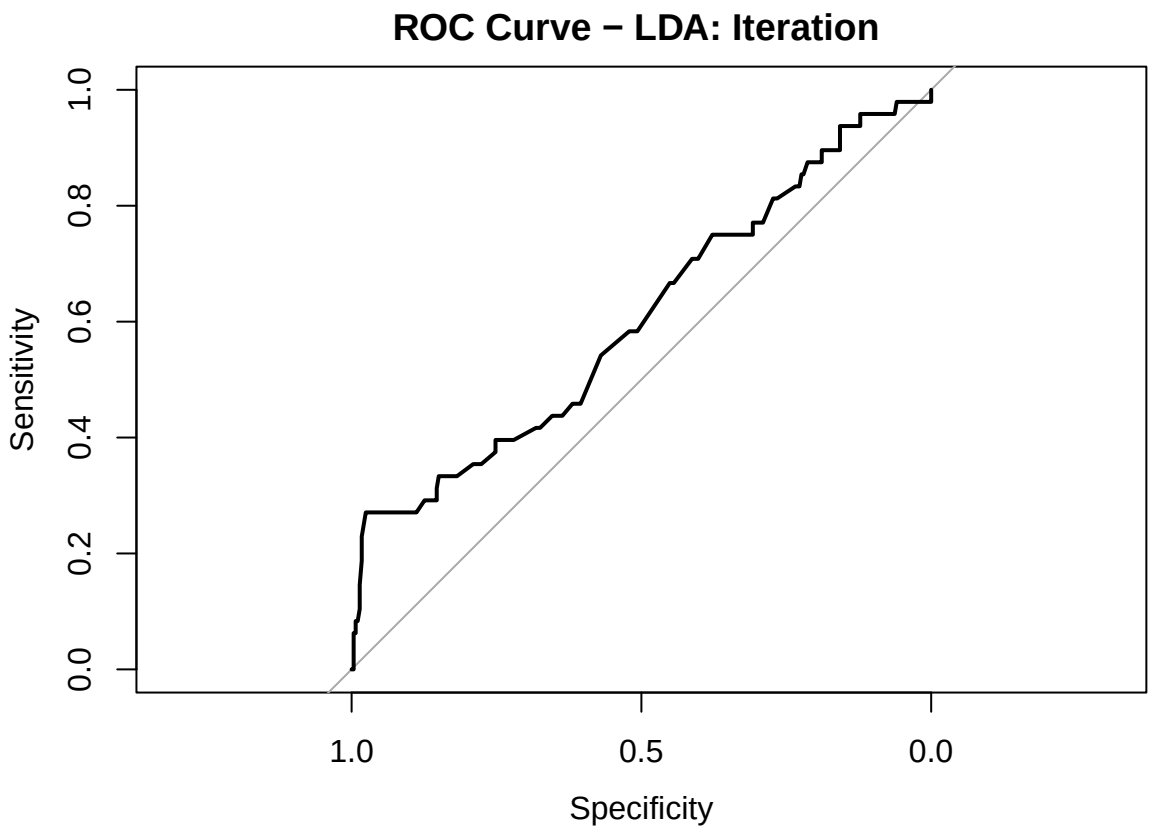
```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```




```
## Area under the curve: 0.6037
```

```
##          observed
## predicted   0   1
##         0 286  43
##         1   1   3
## [1] 0.1321321
```

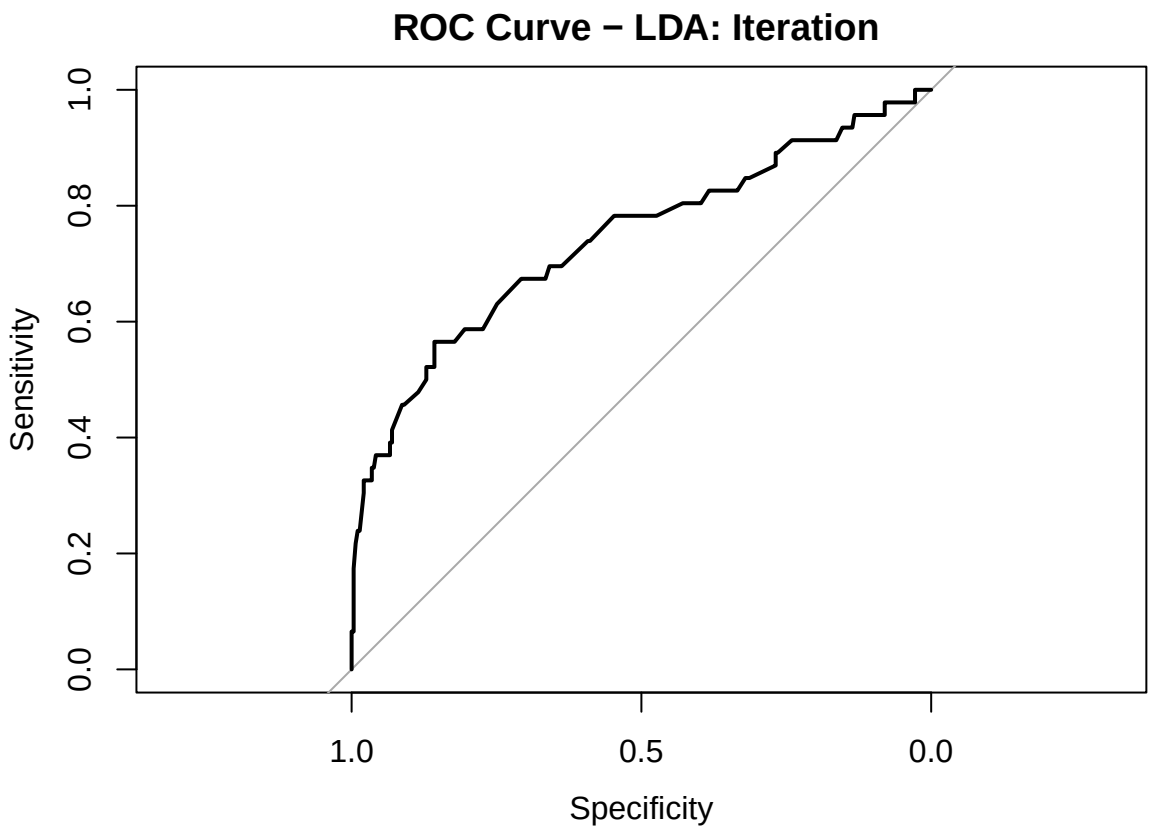
```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```




```
## Area under the curve: 0.741
```

```
##           observed
## predicted   0   1
##         0 291  42
##         1   0   0
## [1] 0.1261261
```

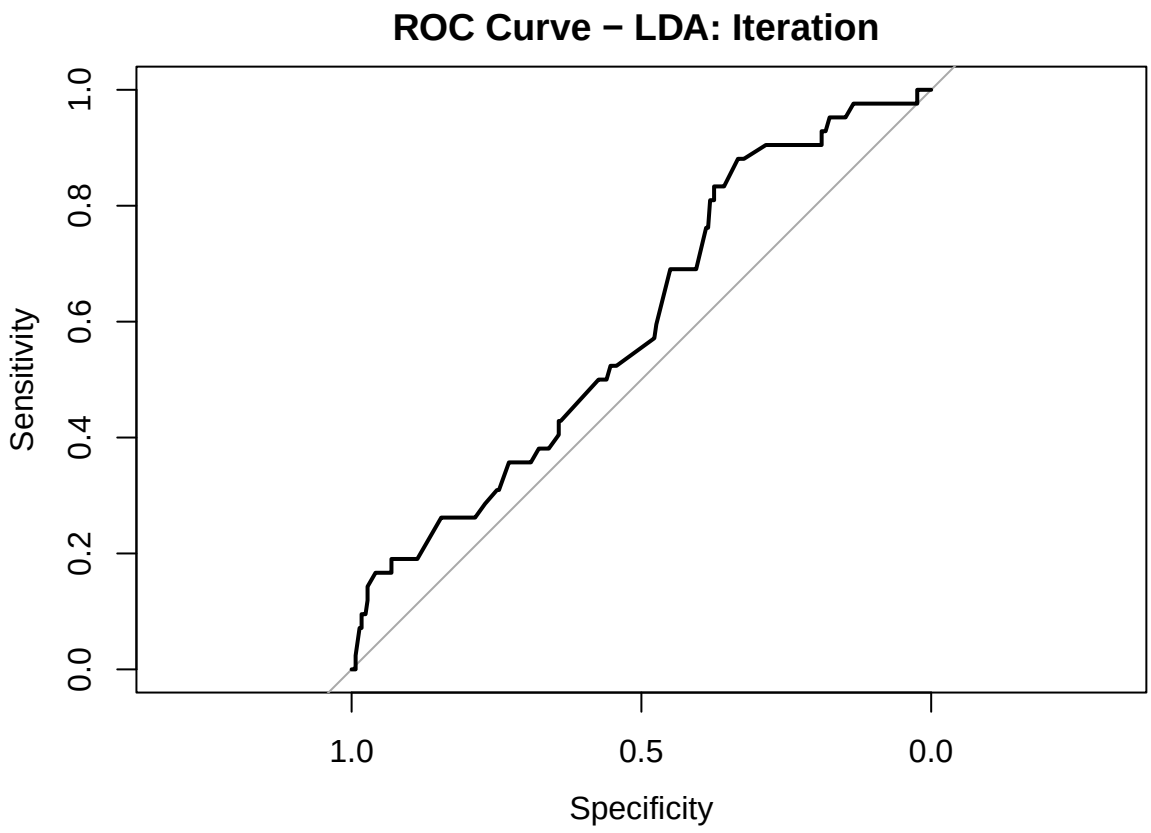
```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```




```
## Area under the curve: 0.5945
```

```
##          observed
## predicted   0   1
##         0 275  52
##         1   2   4
## [1] 0.1621622
```

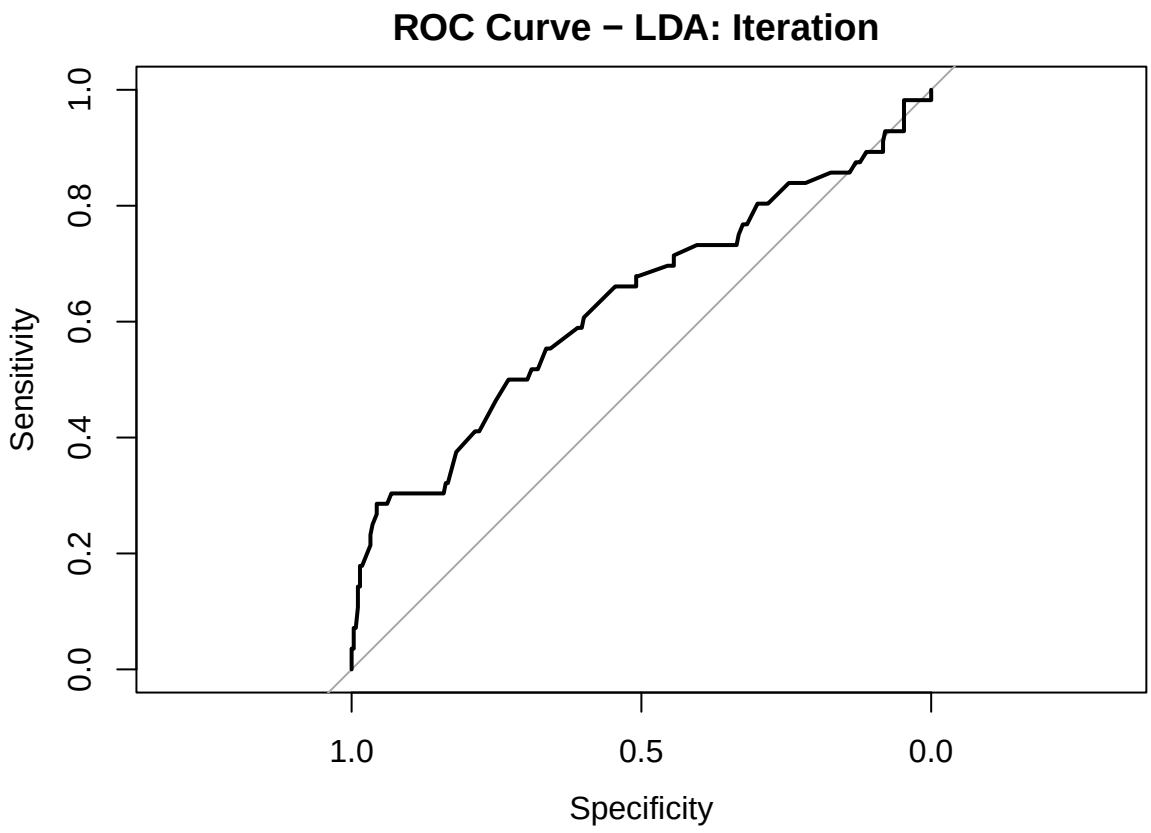
```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```




```
## Area under the curve: 0.6321
```

```
##          observed
## predicted   0   1
##         0 294  37
##         1   2   0
## [1] 0.1171171
```

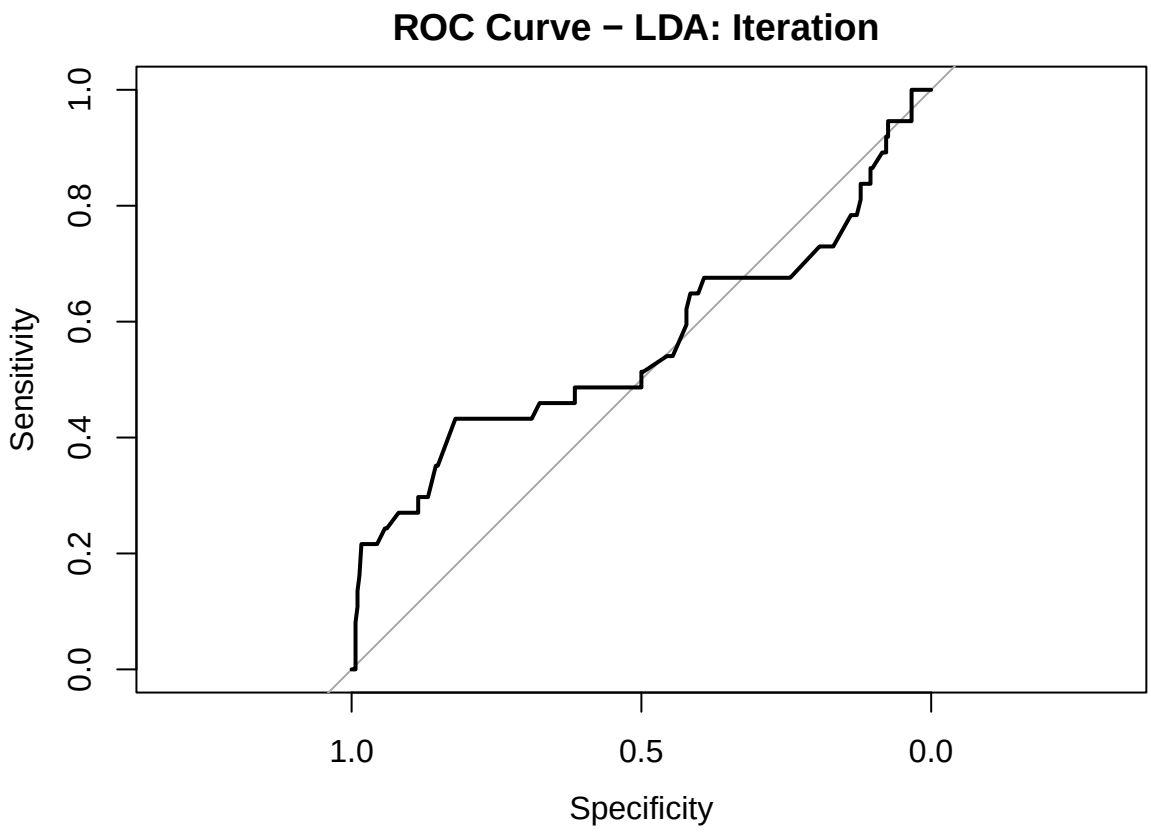
```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```



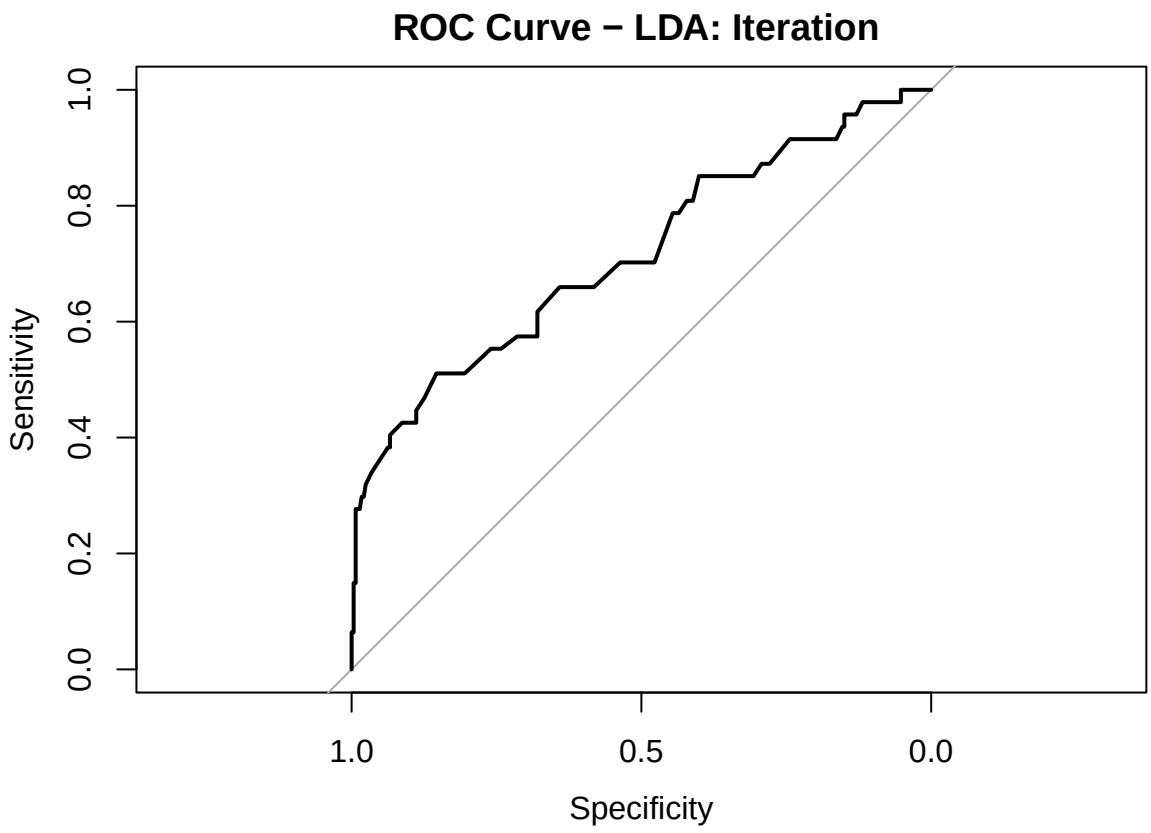

```
## Area under the curve: 0.5595
```

```
##          observed
## predicted   0   1
##         0 287  44
##         1   0   3
## [1] 0.1317365
```

```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```




```
## Area under the curve: 0.7153
```




```
totalMisclassificationRate = c(totalMisclassificationRate, mean(runningTotal))
print("Average Misclassification Rate")
```

```
## [1] "Average Misclassification Rate"
```

```
totalMisclassificationRate
```

```
## [1] 0.141918
```

```
print("Average AUC")
```

```
## [1] "Average AUC"
```

```
averageAUC = c(averageAUC, mean(auc))
print(averageAUC)
```

```
## [1] 0.6374061
```

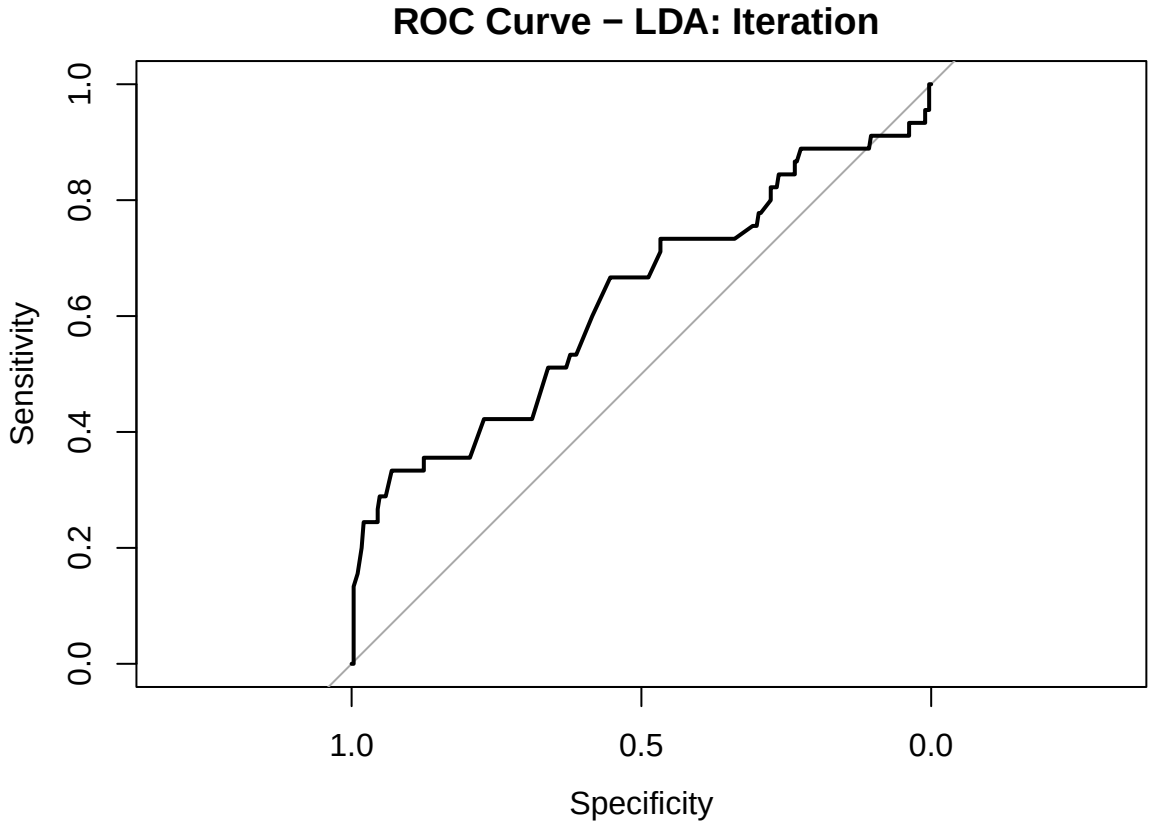
QDA

```
#Randomly shuffle the data
Customer_telecom <- Customer_telecom[sample(nrow(Customer_telecom)), ]

#Create 10 equally size folds
folds <- cut(seq(1, nrow(Customer_telecom)), breaks = 10, labels = FALSE)

totalMisclassificationRate <- c()
runningTotal <- c()

averageAUC = c()
auc = c()

for (i in 1:10) {
  testIndexes <- which(folds == i, arr.ind = TRUE)
  test_data <- Customer_telecom[testIndexes,]
  train_data <- Customer_telecom[-testIndexes,]
  m_qda <-
    qda(churn1 ~ number.vmail.messages +
          total.intl.calls +
          customer.service.calls,
        data = train_data)
  m_pred <- predict(m_qda, test_data)
  conf <-
    table(list(predicted = m_pred$class, observed = test_data$churn1))
  confusionMatrix(conf)
  print(conf)
  totalPreds = sum(conf)
  incorrectPreds <- conf[1, 2] + conf[2, 1]
  misclassificationRate <- incorrectPreds / totalPreds
  print(misclassificationRate)
  runningTotal <- c(runningTotal, misclassificationRate)

  roc_score = roc(response = test_data$churn1,
                  predictor = m_pred$posterior[, "1"])
  auc <- c(auc, roc_score$auc)
  print(roc_score$auc)
  plot(roc_score, main = "ROC Curve - LDA: Iteration ")
}
```

```
##          observed
## predicted   0   1
##         0 283  34
##         1   6  11
## [1] 0.1197605

## Setting levels: control = 0, case = 1

## Setting direction: controls < cases

## Area under the curve: 0.6275
```

```
##          observed
## predicted   0   1
##         0 266  52
##         1   6   9
## [1] 0.1741742
```

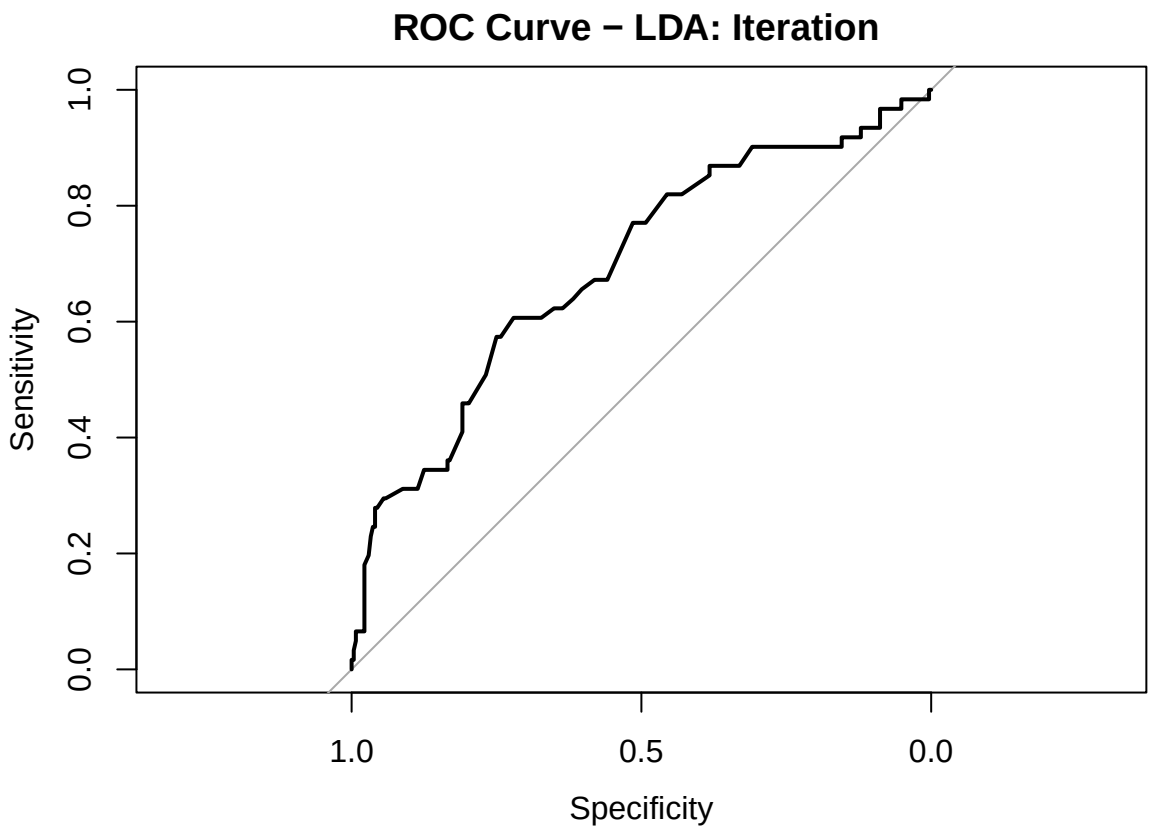
```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```




```
## Area under the curve: 0.6936
```

```
##          observed
## predicted   0   1
##         0 281  41
##         1   4   7
## [1] 0.1351351
```

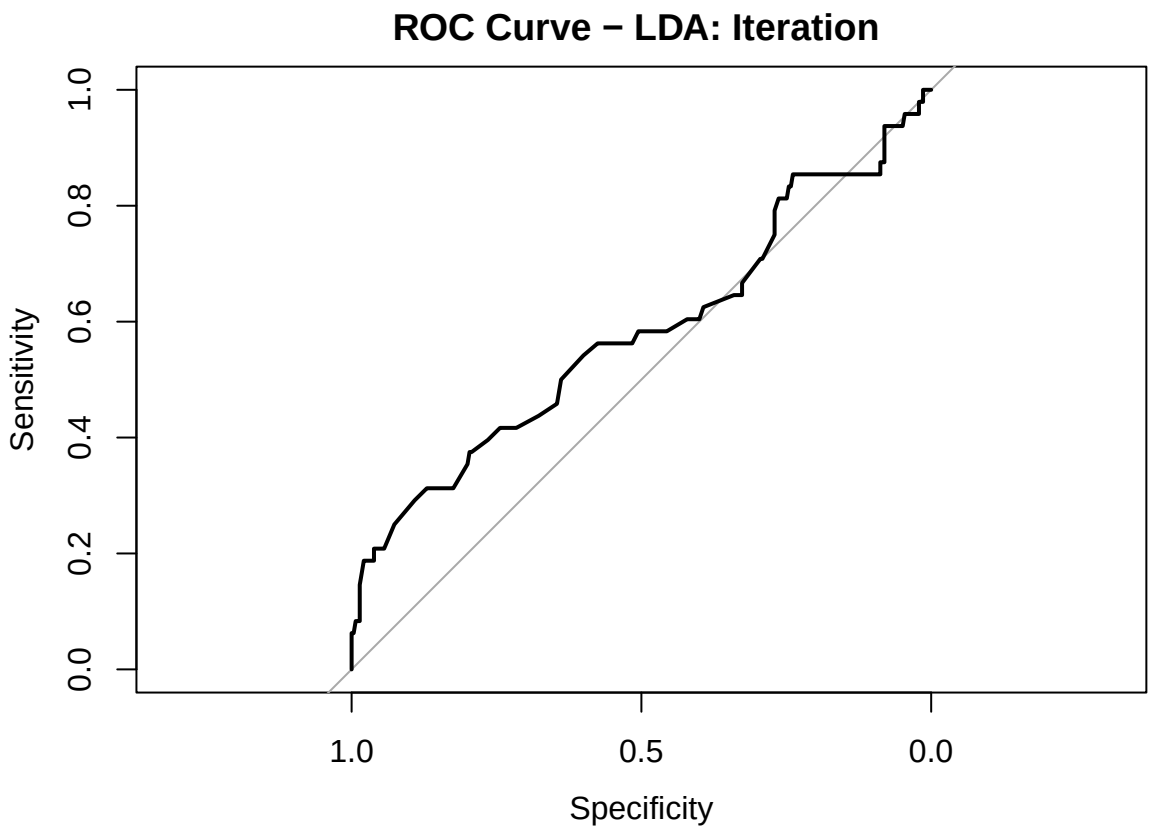
```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```




```
## Area under the curve: 0.5792
```

```
##          observed
## predicted   0   1
##         0 267  46
##         1   6  14
## [1] 0.1561562
```

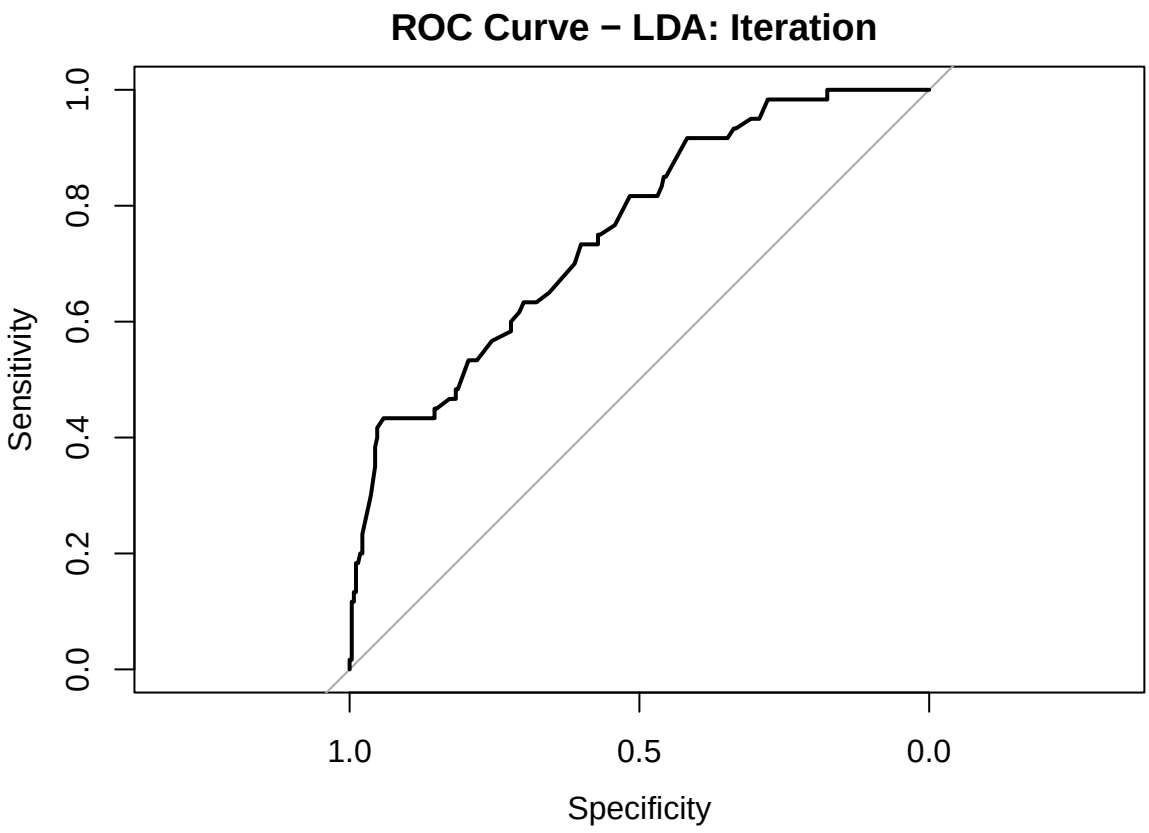
```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```




```
## Area under the curve: 0.7552
```

```
##          observed
## predicted   0   1
##         0 282  40
##         1   8   4
## [1] 0.1437126
```

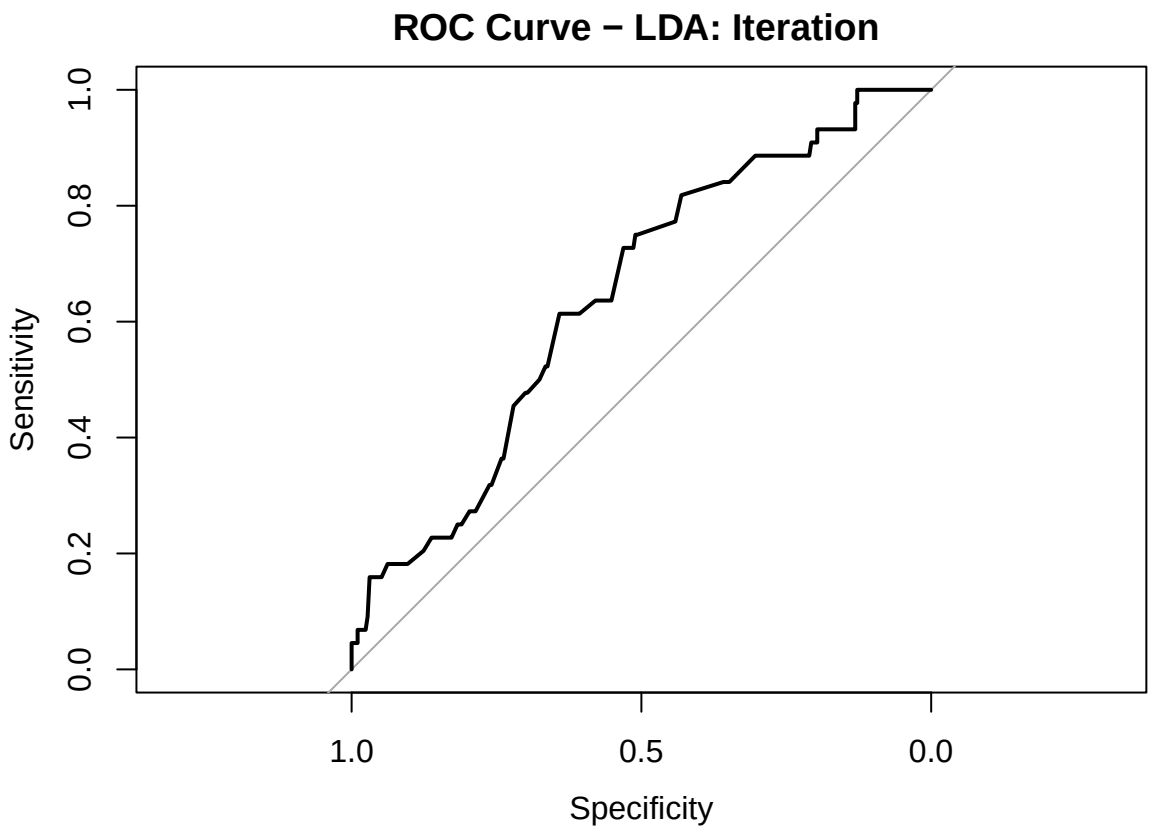
```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```




```
## Area under the curve: 0.6432
```

```
##          observed
## predicted   0   1
##         0 285  34
##         1   5   9
## [1] 0.1171171
```

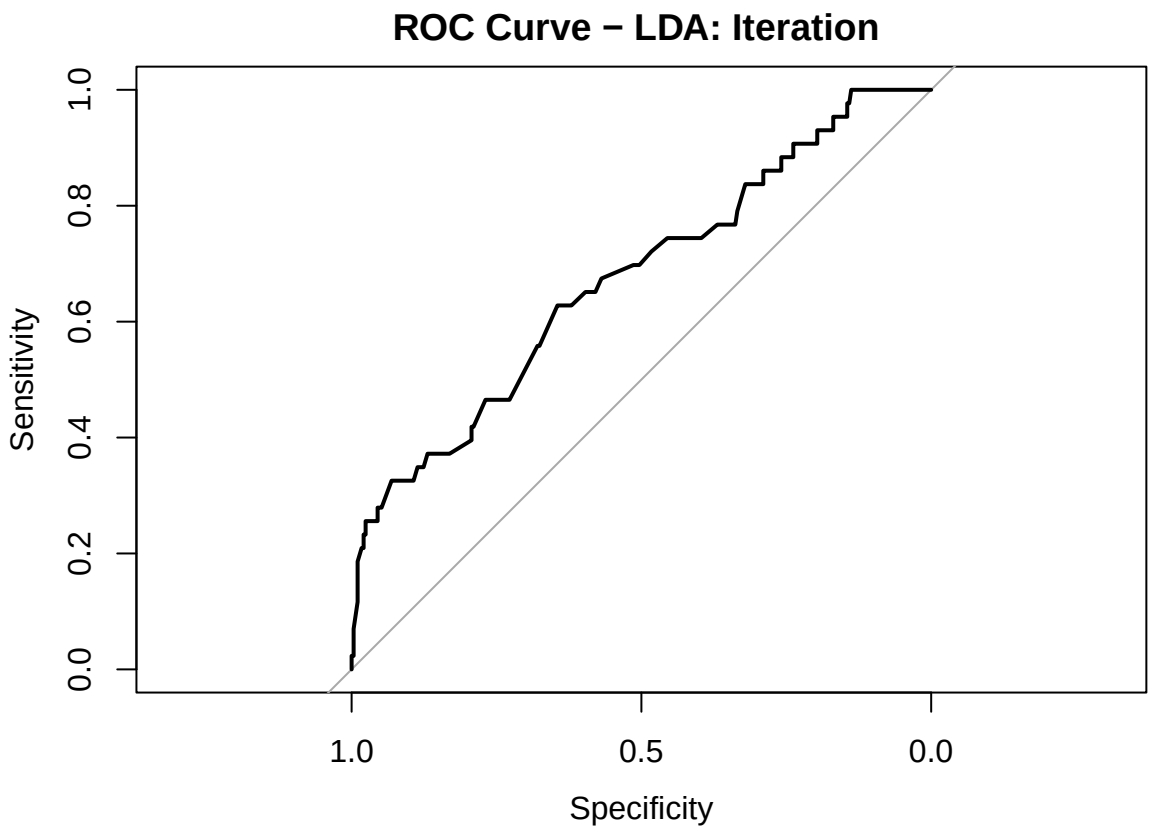
```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```




```
## Area under the curve: 0.6731
```

```
##          observed
## predicted   0   1
##         0 274  41
##         1  11   7
## [1] 0.1561562
```

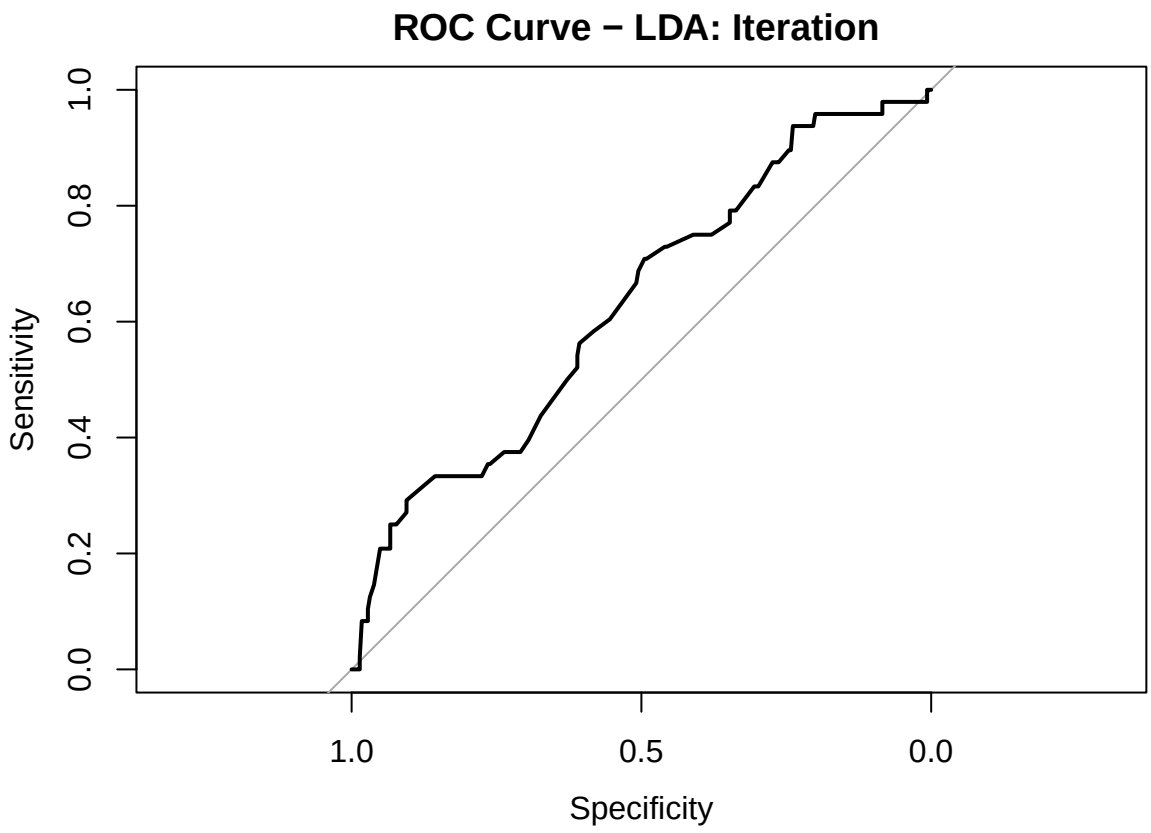
```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```




```
## Area under the curve: 0.6285
```

```
##          observed
## predicted   0   1
##         0 277  38
##         1   6  12
## [1] 0.1321321
```

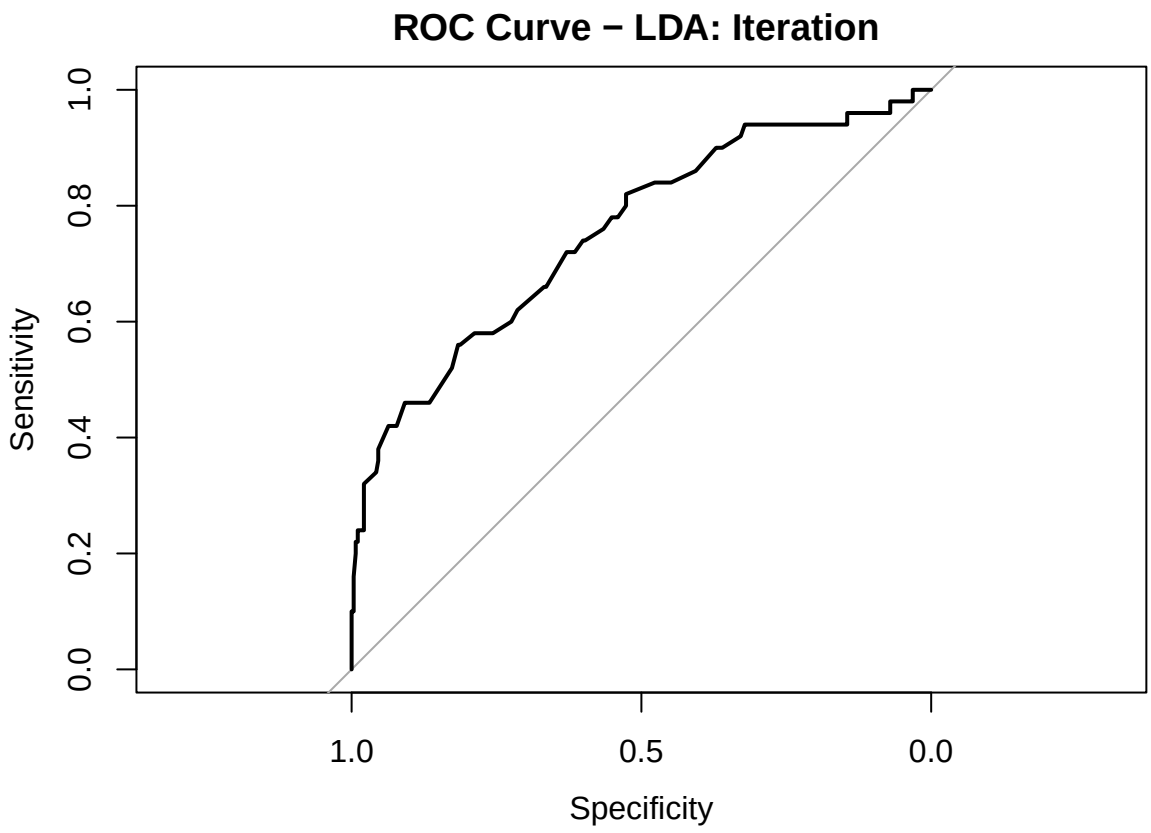
```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```




```
## Area under the curve: 0.7534
```

```
##          observed
## predicted   0   1
##         0 290  30
##         1   7   6
## [1] 0.1111111
```

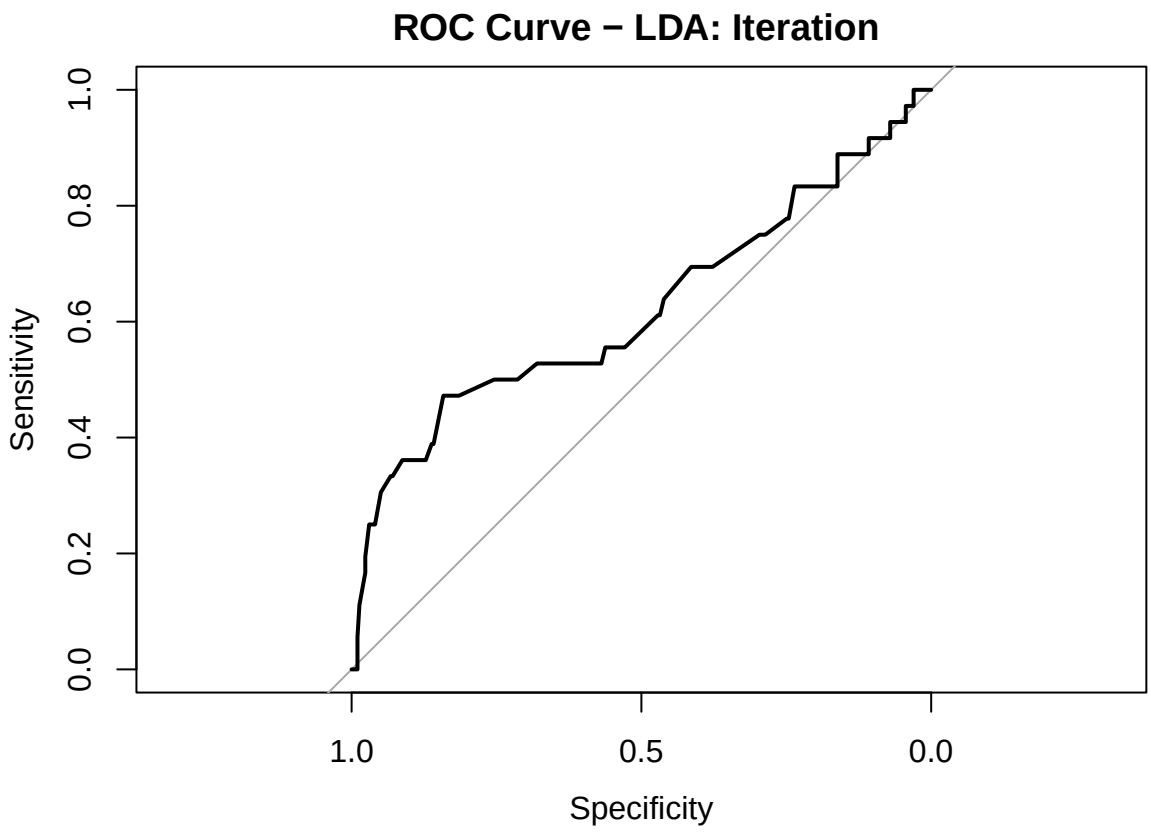
```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```



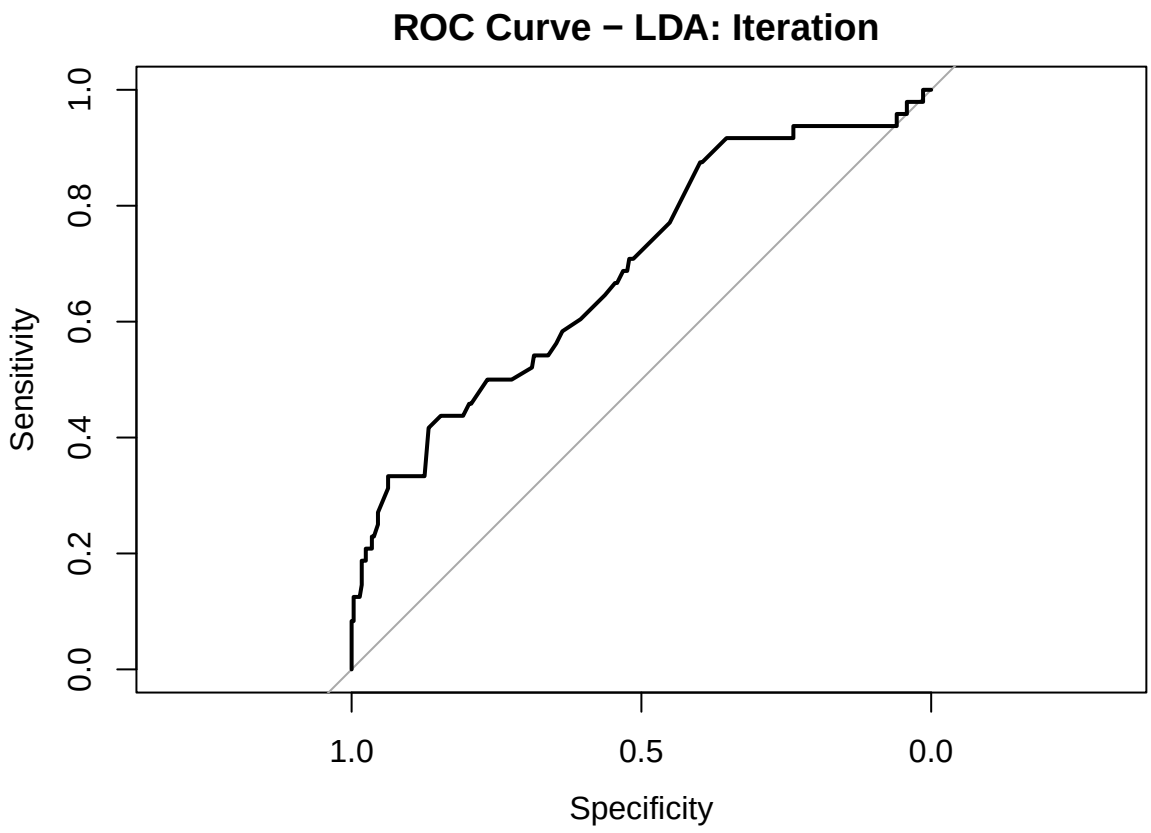

```
## Area under the curve: 0.6227
```

```
##          observed
## predicted   0   1
##         0 281  41
##         1   5   7
## [1] 0.1377246
```

```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```




```
## Area under the curve: 0.6894
```




```
totalMisclassificationRate = c(totalMisclassificationRate, mean(runningTotal))
print("Average Misclassification Rate")
```

```
## [1] "Average Misclassification Rate"
```

```
totalMisclassificationRate
```

```
## [1] 0.138318
```

```
print("Average AUC")
```

```
## [1] "Average AUC"
```

```
averageAUC = c(averageAUC, mean(auc))
print(averageAUC)
```

```
## [1] 0.6665668
```

Naive Bayes

```
#Randomly shuffle the data
Customer_telecom <-
  Customer_telecom[sample(nrow(Customer_telecom)),]

#Create 10 equally size folds
folds <-
  cut(seq(1, nrow(Customer_telecom)), breaks = 10, labels = FALSE)

totalMisclassificationRate <- c()
runningTotal <- c()

averageAUC = c()
auc = c()

for (i in 1:10) {
  testIndexes <- which(folds == i, arr.ind = TRUE)
  test_data <- Customer_telecom[testIndexes, ]
  train_data <- Customer_telecom[-testIndexes, ]
  m_naive <-
    naiveBayes(churn1 ~ number.vmail.messages + total.intl.calls + customer.service.calls,
               data = train_data)
  m_pred <- predict(m_naive, test_data)

  conf <-
    table(list(predicted = m_pred, observed = test_data$churn1))
  confusionMatrix(conf)
  print(conf)
  totalPreds = sum(conf)
  incorrectPreds <- conf[1, 2] + conf[2, 1]
  misclassificationRate <- incorrectPreds / totalPreds
  print(misclassificationRate)
  runningTotal <- c(runningTotal, misclassificationRate)

  #roc_score = roc(response = test_data$churn1, predictor = m_pred["1"])
  #auc <- c(auc,roc_score$auc)
  #print(roc_score$auc)
  #plot(roc_score, main="ROC Curve - LDA: Iteration ")
}
```

```
##          observed
## predicted   0   1
##         0 278  44
##         1   6   6
## [1] 0.1497006
##          observed
## predicted   0   1
##         0 276  40
##         1   7  10
## [1] 0.1411411
##          observed
## predicted   0   1
```

```
##         0 279  39
##         1   5  10
## [1] 0.1321321
##          observed
## predicted   0   1
##         0 283  38
##         1   4   8
## [1] 0.1261261
##          observed
## predicted   0   1
##         0 279  41
##         1   5   9
## [1] 0.1377246
##          observed
## predicted   0   1
##         0 274  42
##         1   6  11
## [1] 0.1441441
##          observed
## predicted   0   1
##         0 281  40
##         1   4   8
## [1] 0.1321321
##          observed
## predicted   0   1
##         0 286  33
##         1   7   7
## [1] 0.1201201
##          observed
## predicted   0   1
##         0 274  47
##         1   5   7
## [1] 0.1561562
##          observed
## predicted   0   1
##         0 281  34
##         1  10   9
## [1] 0.1317365
```

```
totalMisclassificationRate = c(totalMisclassificationRate, mean(runningTotal))
print("Average Misclassification Rate")
```

```
## [1] "Average Misclassification Rate"
```

```
totalMisclassificationRate
```

```
## [1] 0.1371114
```

```
#print("Average AUC")
#averageAUC=c(averageAUC, mean(auc))
#print(averageAUC)
```

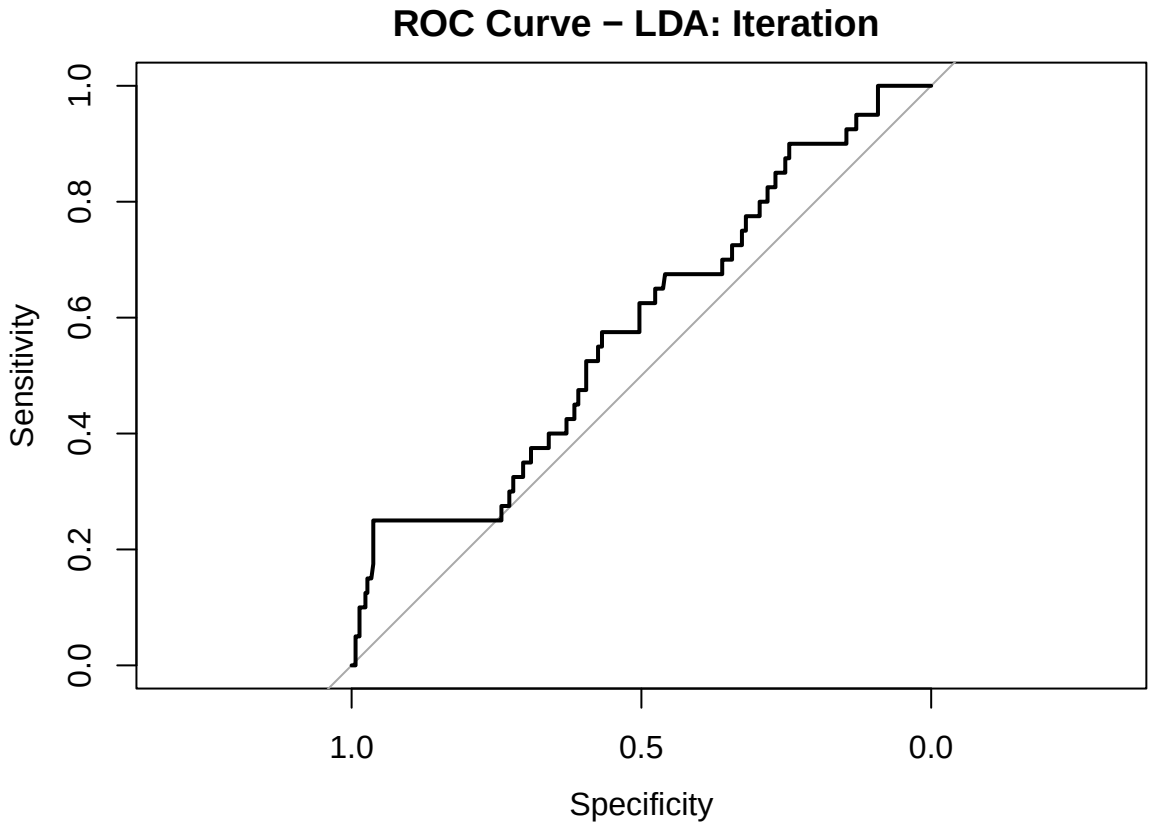
##Experiments

LDA with next most significant predictor

```
#Randomly shuffle the data
Customer_telecom <-
  Customer_telecom[sample(nrow(Customer_telecom)),]

#Create 10 equally size folds
folds <-
  cut(seq(1, nrow(Customer_telecom)), breaks = 10, labels = FALSE)

totalMisclassificationRate = c()
runningTotal = c()

averageAUC = c()
auc = c()

#K-fold Cross Validation
for (i in 1:10) {
  testIndexes <- which(folds == i, arr.ind = TRUE)
  test_data <- Customer_telecom[testIndexes,]
  train_data <- Customer_telecom[-testIndexes,]
  m_lda2 <-
    lda(
      churn1 ~ number.vmail.messages +
        total.intl.calls +
        customer.service.calls +
        total.day.calls,
      data = train_data
    )
  m_pred <- predict(m_lda2, test_data)
  conf <-
    table(list(predicted = m_pred$class, observed = test_data$churn1))
  confusionMatrix(conf)
  print(conf)
  totalPreds = sum(conf)
  incorrectPreds <- conf[1, 2] + conf[2, 1]
  misclassificationRate <- incorrectPreds / totalPreds
  print(misclassificationRate)
  runningTotal <- c(runningTotal, misclassificationRate)

  roc_score = roc(response = test_data$churn1,
                  predictor = m_pred$posterior[, "1"])
  auc <- c(auc, roc_score$auc)
  print(roc_score$auc)
  plot(roc_score, main = "ROC Curve - LDA: Iteration ")
}
```

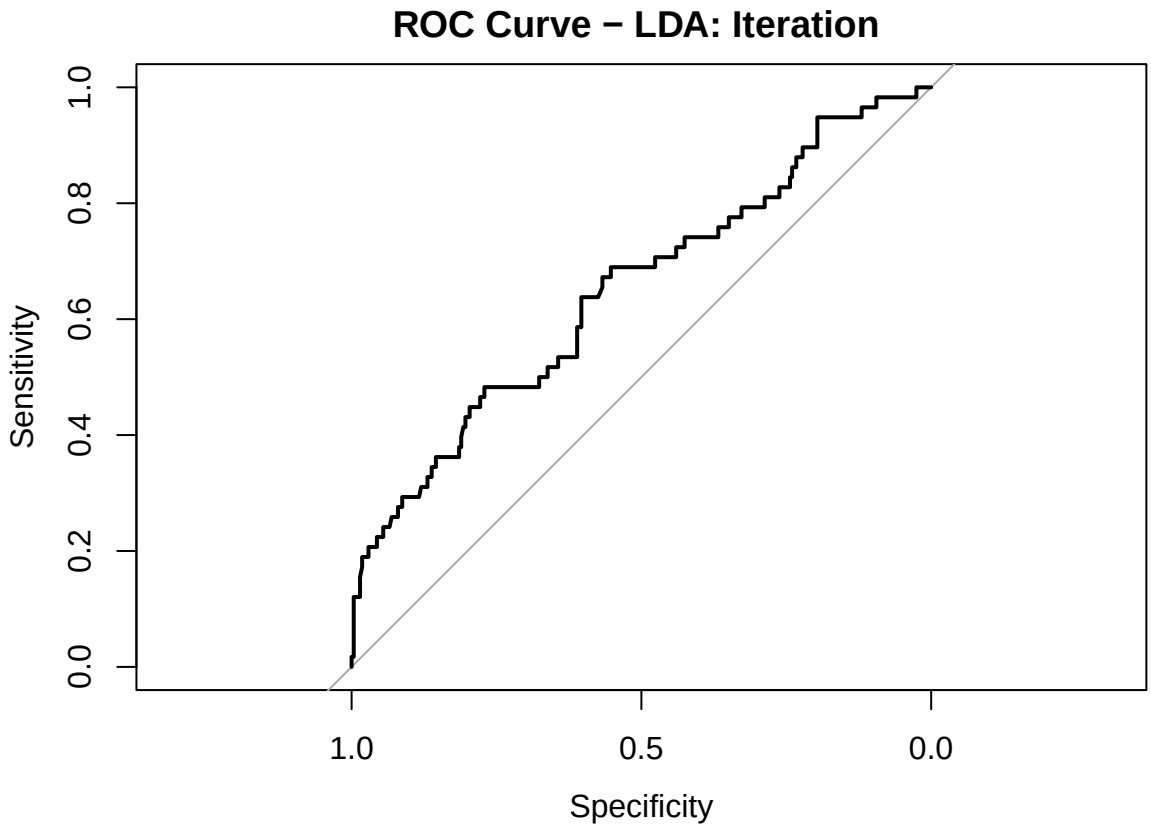
```
##           observed
## predicted   0   1
##         0 291  38
```

```
##         1   3   2
## [1] 0.1227545

## Setting levels: control = 0, case = 1

## Setting direction: controls < cases

## Area under the curve: 0.5822

##          observed
## predicted   0   1
##         0 275  57
##         1   0   1
## [1] 0.1711712

## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```

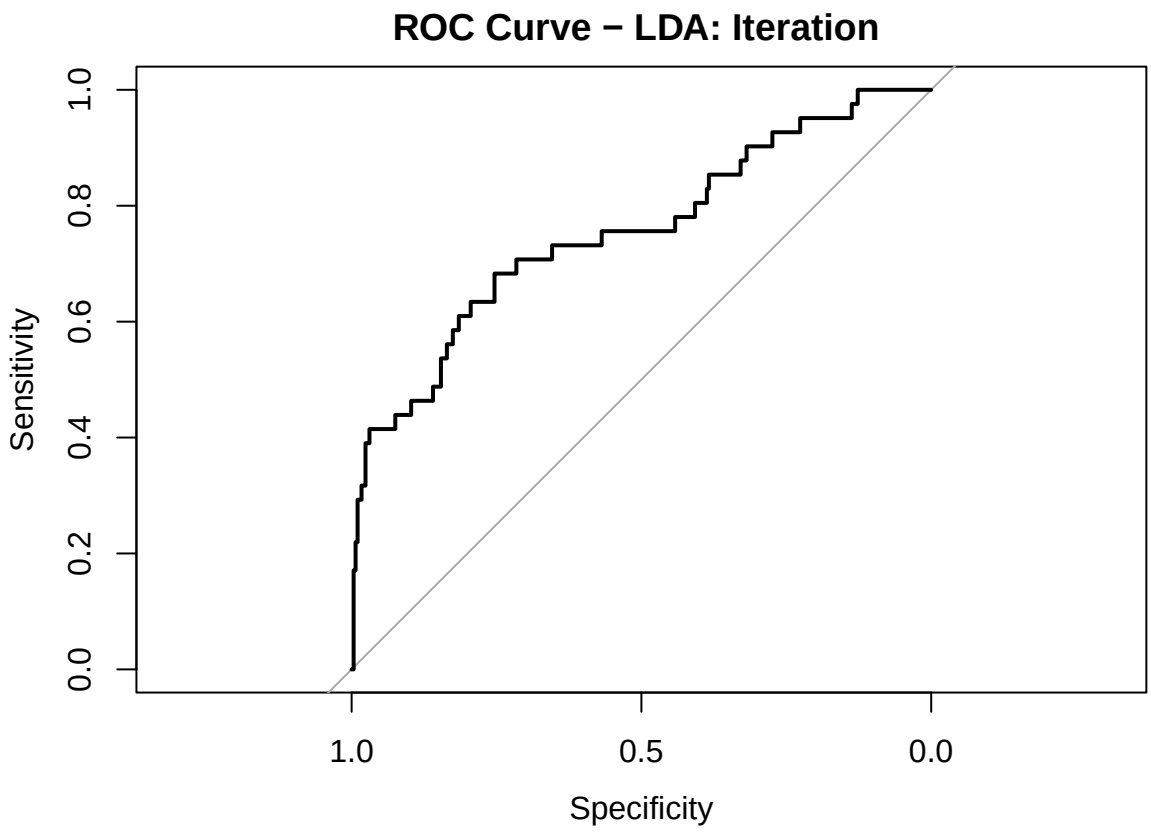



```
## Area under the curve: 0.6487

##          observed
## predicted   0   1
##         0 291  41
##         1   1   0
## [1] 0.1261261
```

```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```



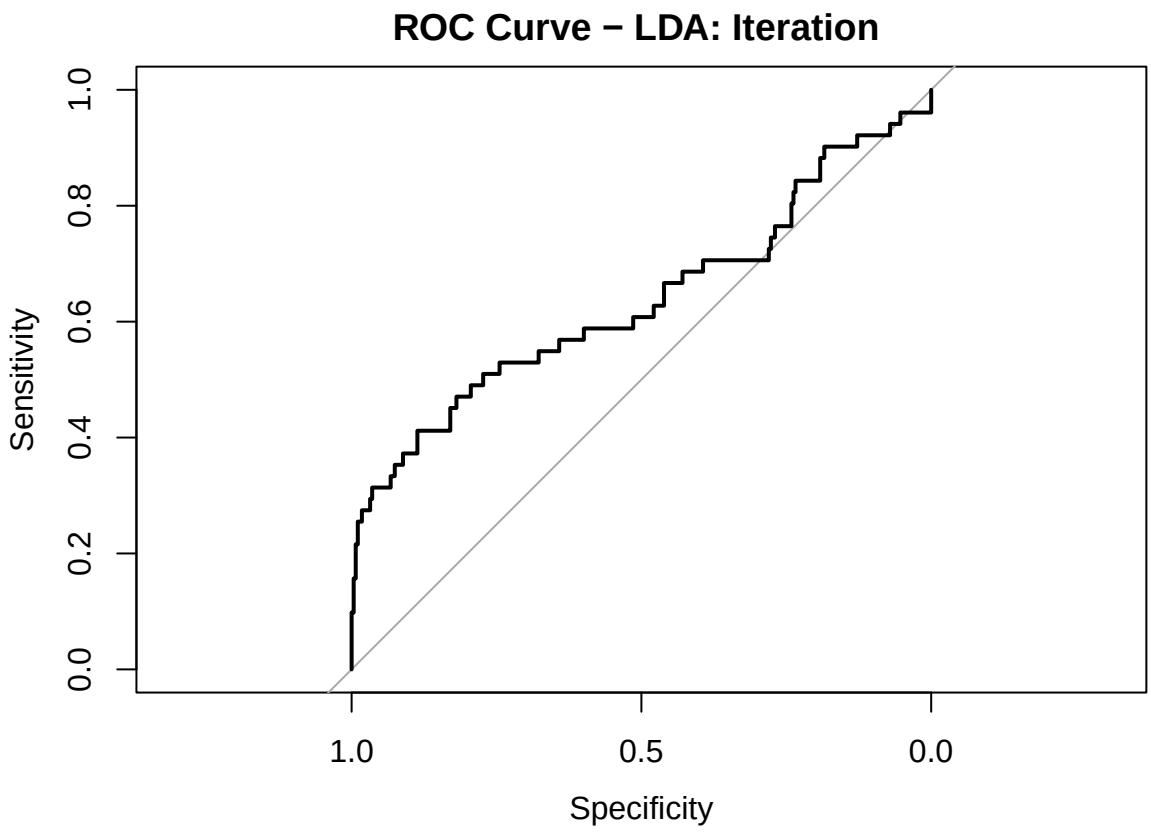

```
## Area under the curve: 0.7544

##           observed
## predicted   0   1
##         0 282  46
##         1   0   5
## [1] 0.1381381

## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```




```
## Area under the curve: 0.6363
```

```
##          observed
## predicted   0   1
##         0 286  45
##         1   0   3
## [1] 0.1347305
```

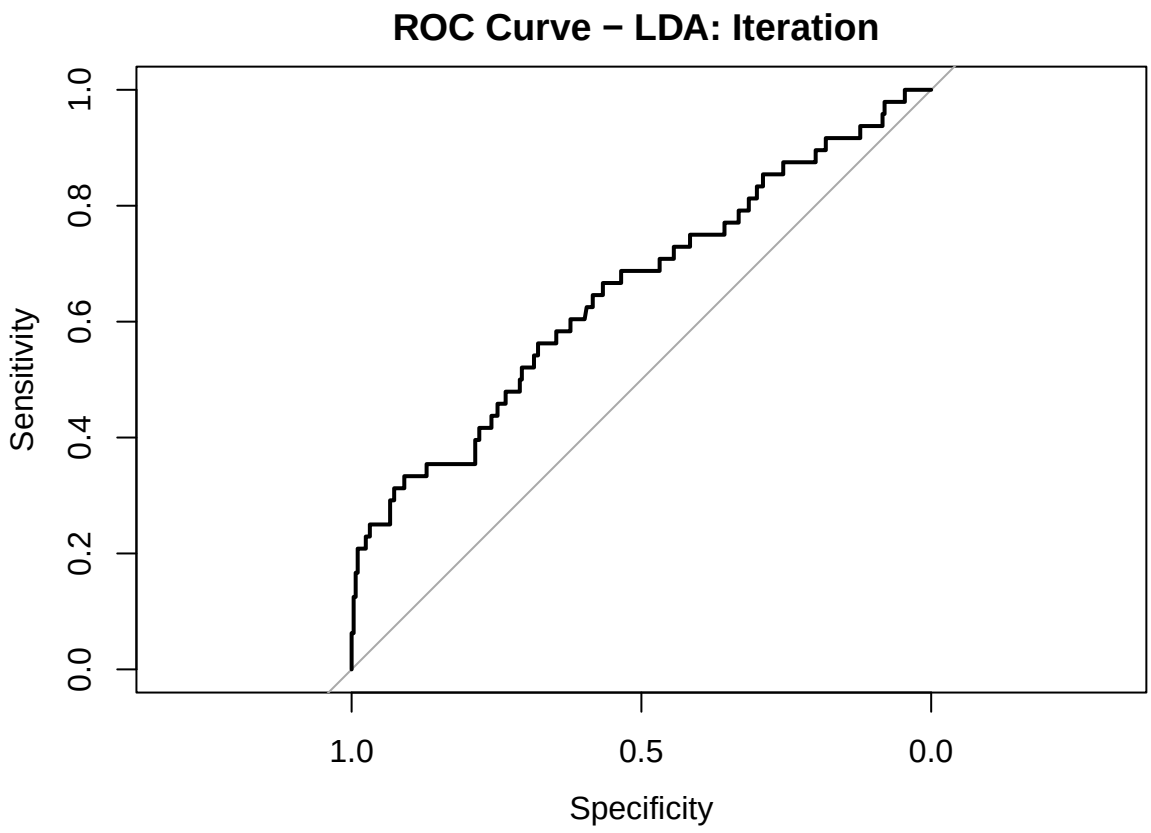
```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```




```
## Area under the curve: 0.6518
```

```
##          observed
## predicted   0   1
##         0 278  51
##         1   3   1
## [1] 0.1621622
```

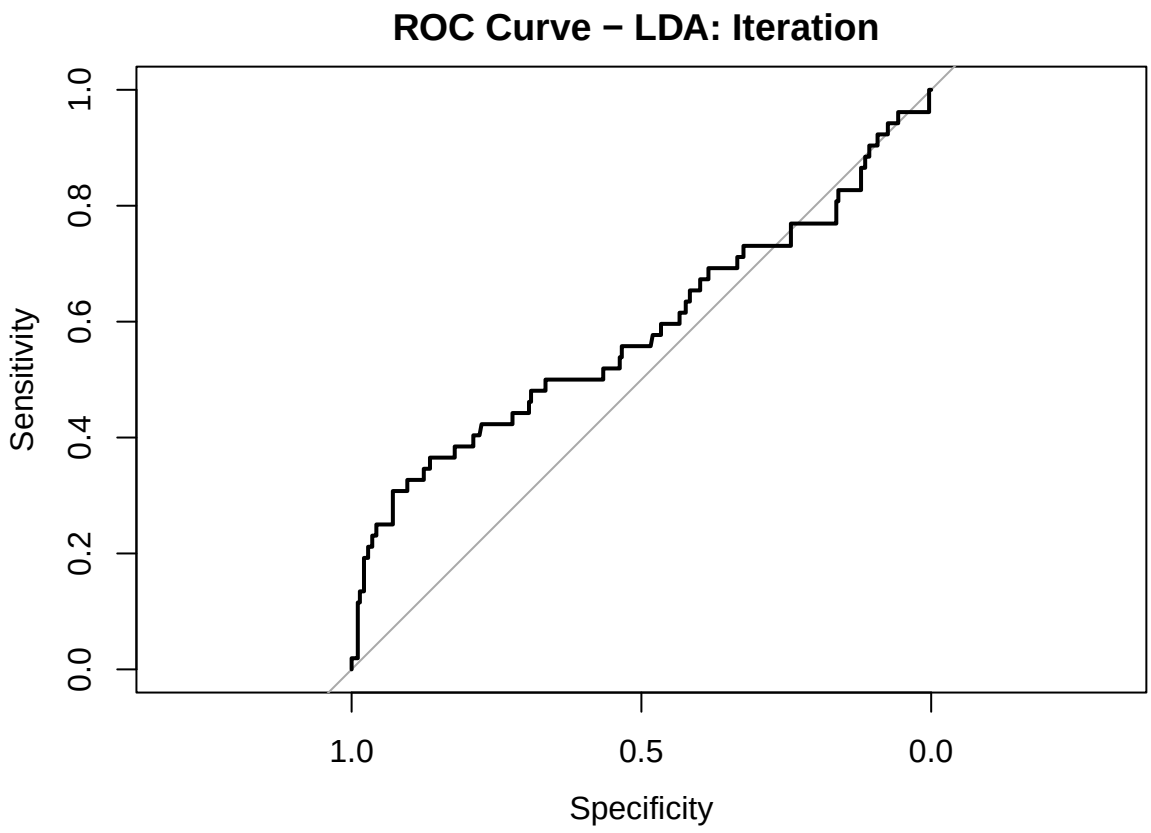
```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```




```
## Area under the curve: 0.5831
```

```
##          observed
## predicted   0   1
##         0 277  53
##         1   1   2
## [1] 0.1621622
```

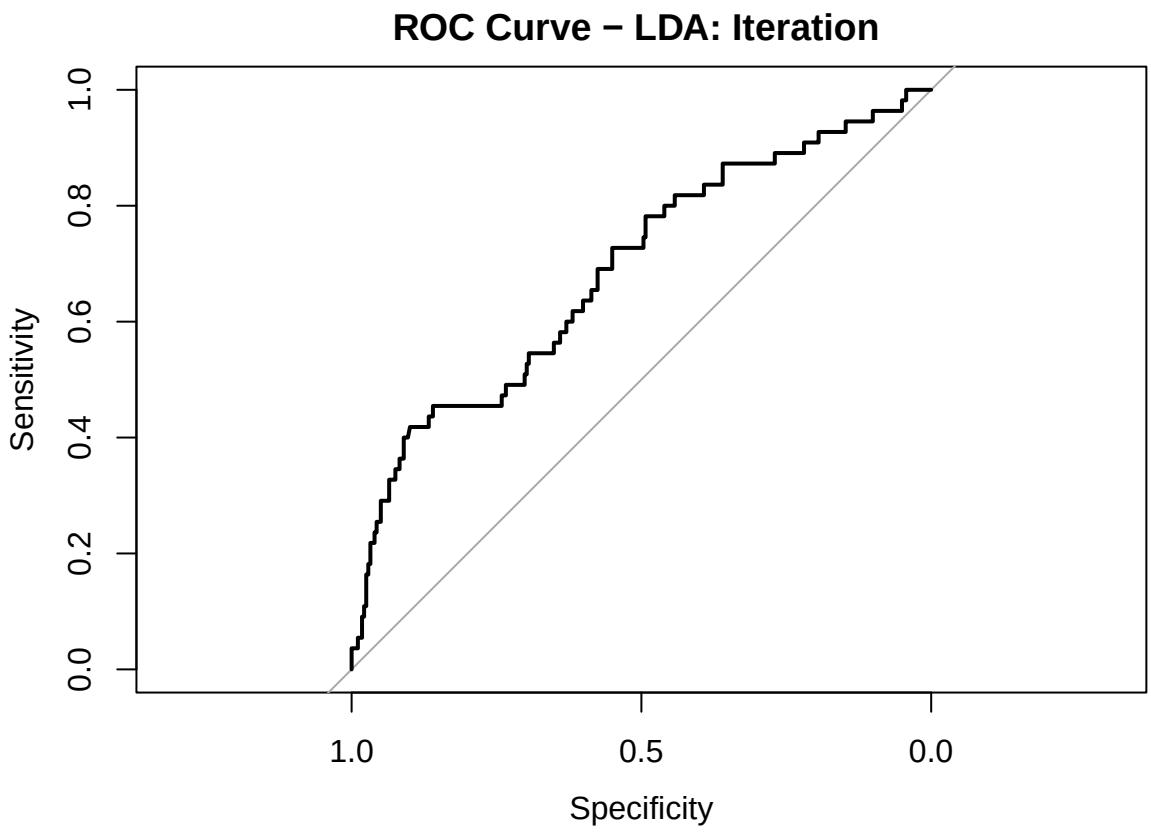
```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```




```
## Area under the curve: 0.6874
```

```
##          observed
## predicted   0   1
##         0 283  48
##         1   0   2
## [1] 0.1441441
```

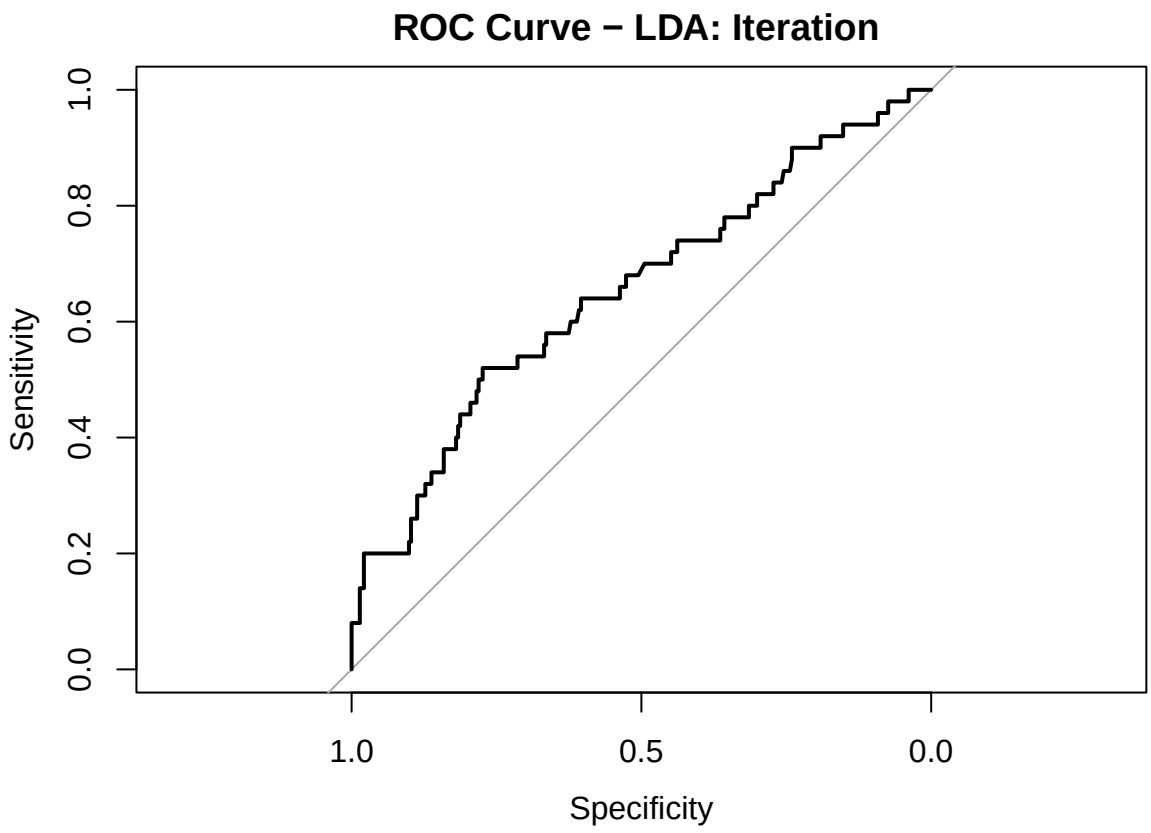
```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```




```
## Area under the curve: 0.6518
```

```
##          observed
## predicted   0   1
##         0 289  40
##         1   2   2
## [1] 0.1261261
```

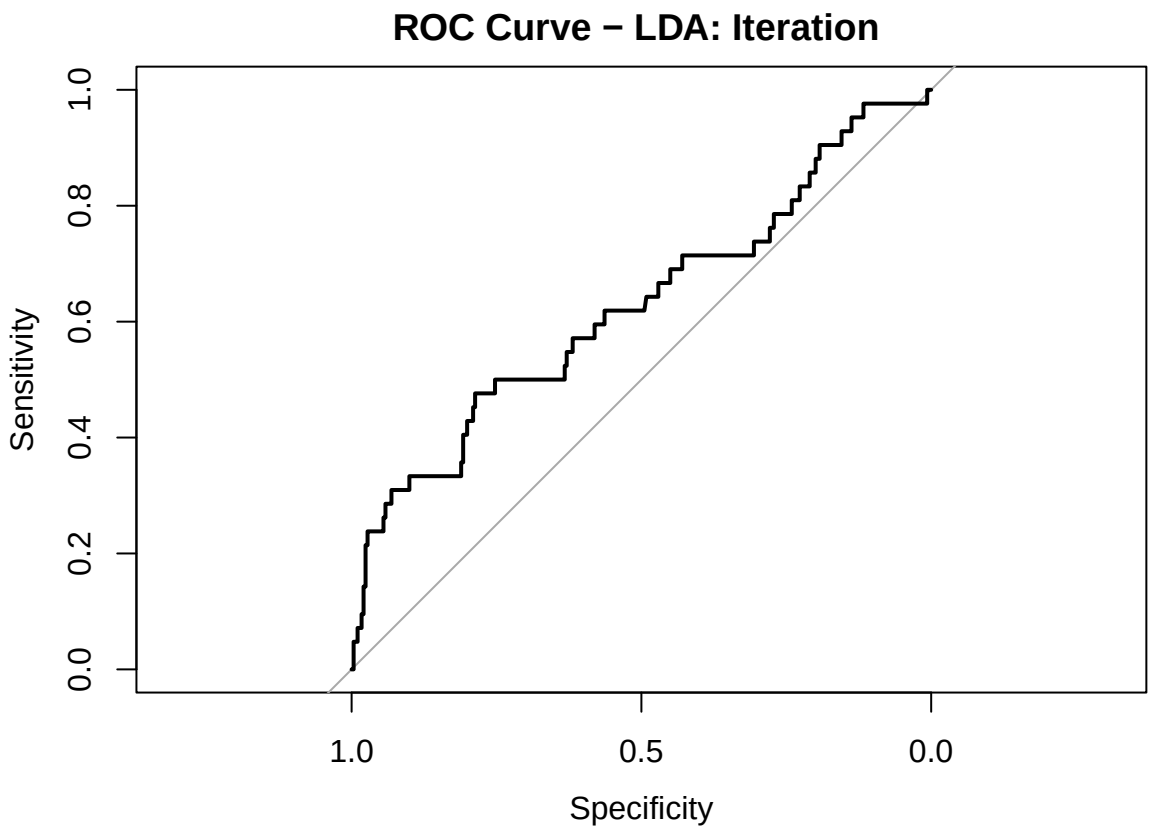
```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```



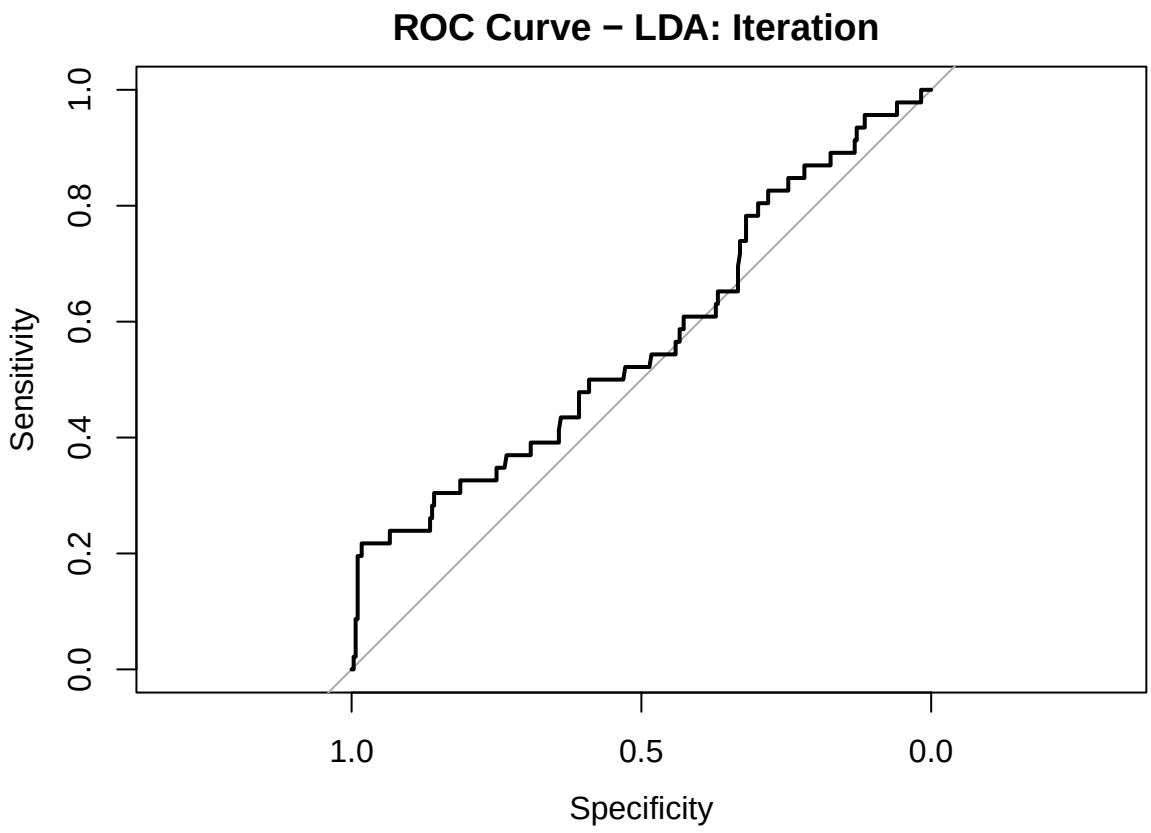

```
## Area under the curve: 0.6264
```

```
##          observed
## predicted   0   1
##         0 287  45
##         1   1   1
## [1] 0.1377246
```

```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```




```
## Area under the curve: 0.5694
```




```
totalMisclassificationRate = c(totalMisclassificationRate, mean(runningTotal))
print("Average Misclassification Rate")
```

```
## [1] "Average Misclassification Rate"
```

```
totalMisclassificationRate
```

```
## [1] 0.142524
```

```
print("Average AUC")
```

```
## [1] "Average AUC"
```

```
averageAUC = c(averageAUC, mean(auc))
print(averageAUC)
```

```
## [1] 0.6391538
```

By adding one more predictor total.day.calls (the potential predictor with the fourth smallest p-value from LRM), the average misclassification rate tend to increase slightly, which may be caused by overfitting - so we stop by including only the original 3 predictors in our model.

LDA All potentially relevant predictors

```
#Randomly shuffle the data
Customer_telecom <-
  Customer_telecom[sample(nrow(Customer_telecom)),]

#Create 10 equally size folds
folds <-
  cut(seq(1, nrow(Customer_telecom)), breaks = 10, labels = FALSE)

totalMisclassificationRate = c()
runningTotal = c()

averageAUC = c()
auc = c()

#K-fold Cross Validation
for (i in 1:10) {
  testIndexes <- which(folds == i, arr.ind = TRUE)
  test_data <- Customer_telecom[testIndexes, ]
  train_data <- Customer_telecom[-testIndexes, ]
  m_lda2 <-
    lda(
      churn1 ~ state +
        account.length +
        international.plan +
        voice.mail.plan +
        number.vmail.messages +
        total.day.minutes +
        total.eve.minutes +
        total.night.minutes +
        total.intl.minutes +
        customer.service.calls,
      data = train_data
    )
  m_pred <- predict(m_lda2, test_data)
  conf <-
    table(list(predicted = m_pred$class, observed = test_data$churn1))
  confusionMatrix(conf)
  print(conf)
  totalPreds = sum(conf)
  incorrectPreds <- conf[1, 2] + conf[2, 1]
  misclassificationRate <- incorrectPreds / totalPreds
  print(misclassificationRate)
  runningTotal <- c(runningTotal, misclassificationRate)

  roc_score = roc(response = test_data$churn1,
                  predictor = m_pred$posterior[, "1"])
  auc <- c(auc, roc_score$auc)
  print(roc_score$auc)
  plot(roc_score, main = "ROC Curve - LDA: Iteration ")
}
```

```
##          observed
## predicted   0   1
##         0 263  43
##         1  16  12
## [1] 0.1766467
```

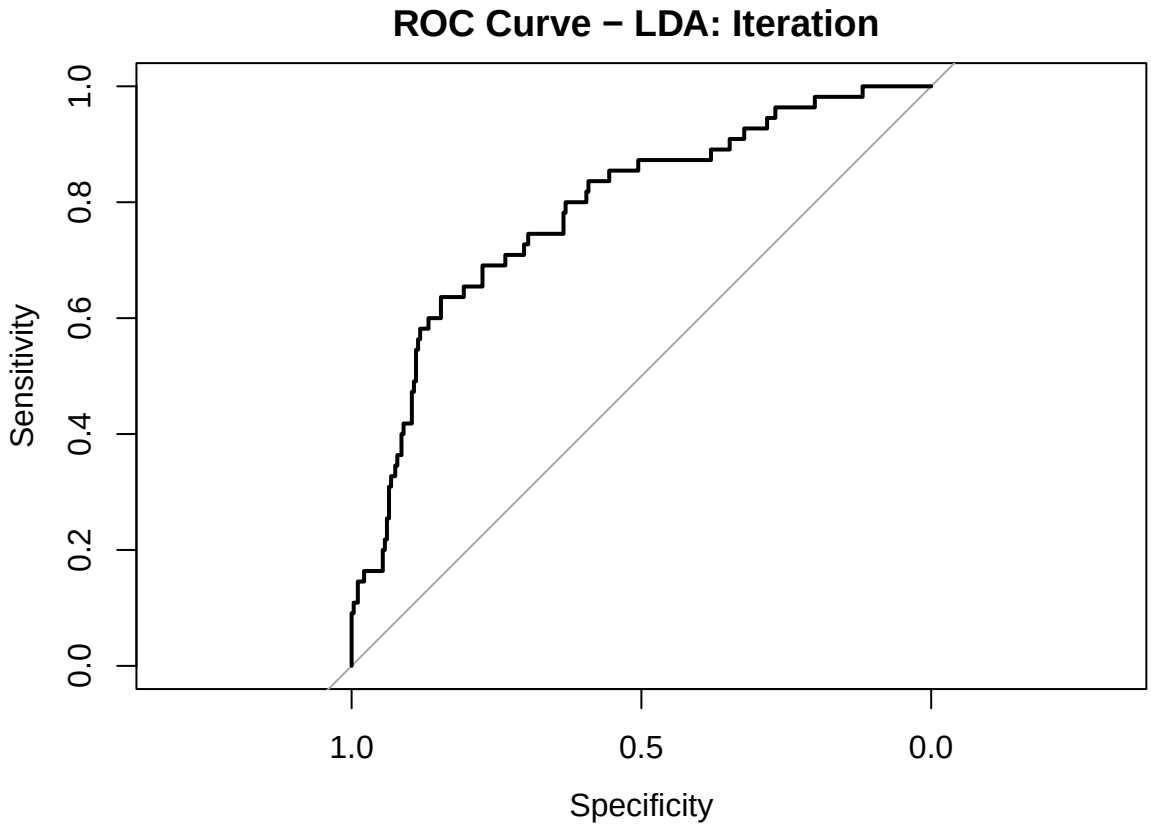
```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

```
## Area under the curve: 0.784
```

```
##          observed
## predicted   0   1
##         0 261  40
##         1  18  14
## [1] 0.1741742
```

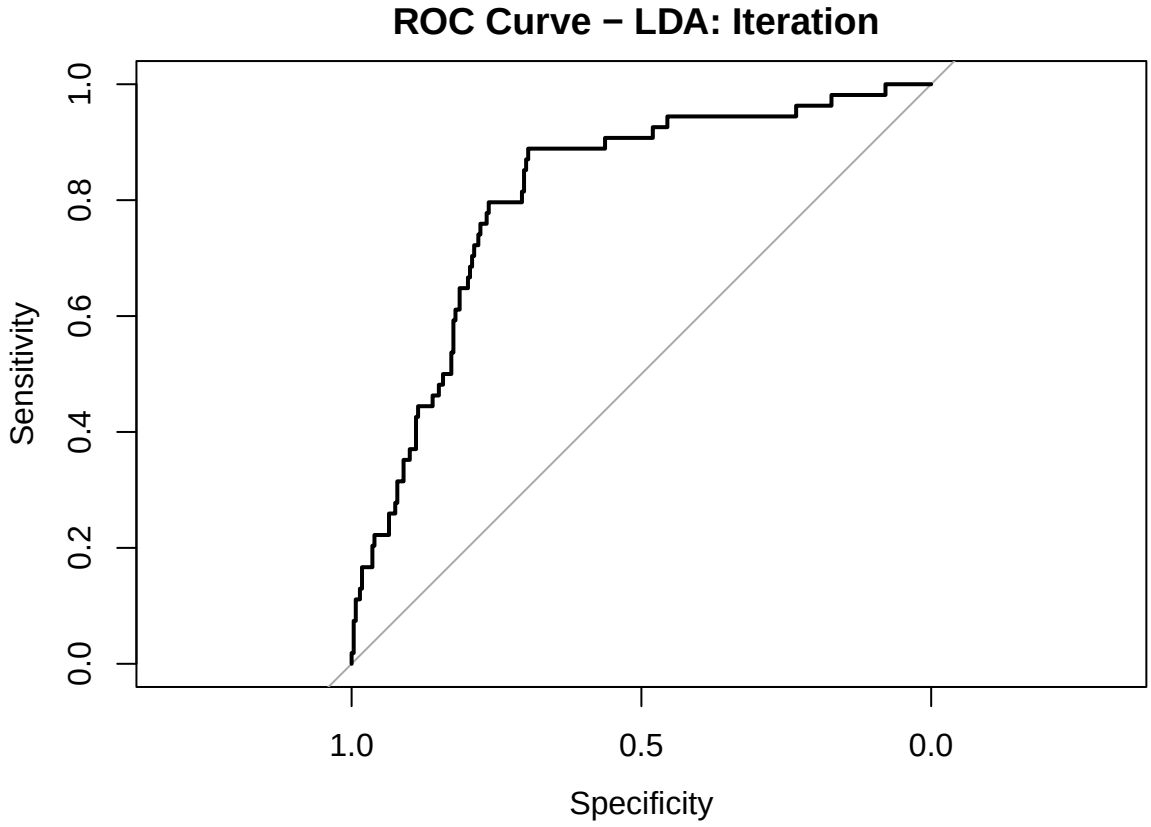
```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```




```
## Area under the curve: 0.8075
```

```
##          observed
## predicted   0   1
##         0 279  25
##         1  12  17
## [1] 0.1111111
```

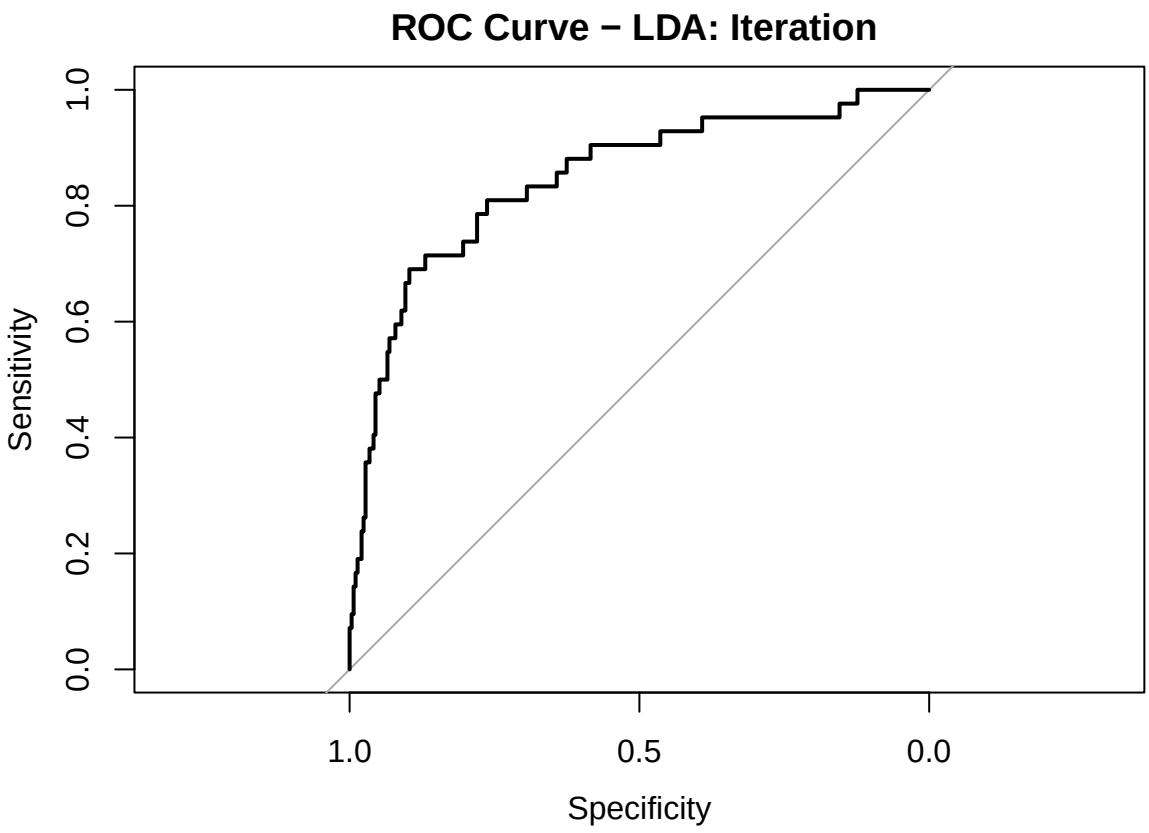
```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```




```
## Area under the curve: 0.8461
```

```
##          observed
## predicted   0   1
##         0 270  36
##         1  19   8
## [1] 0.1651652
```

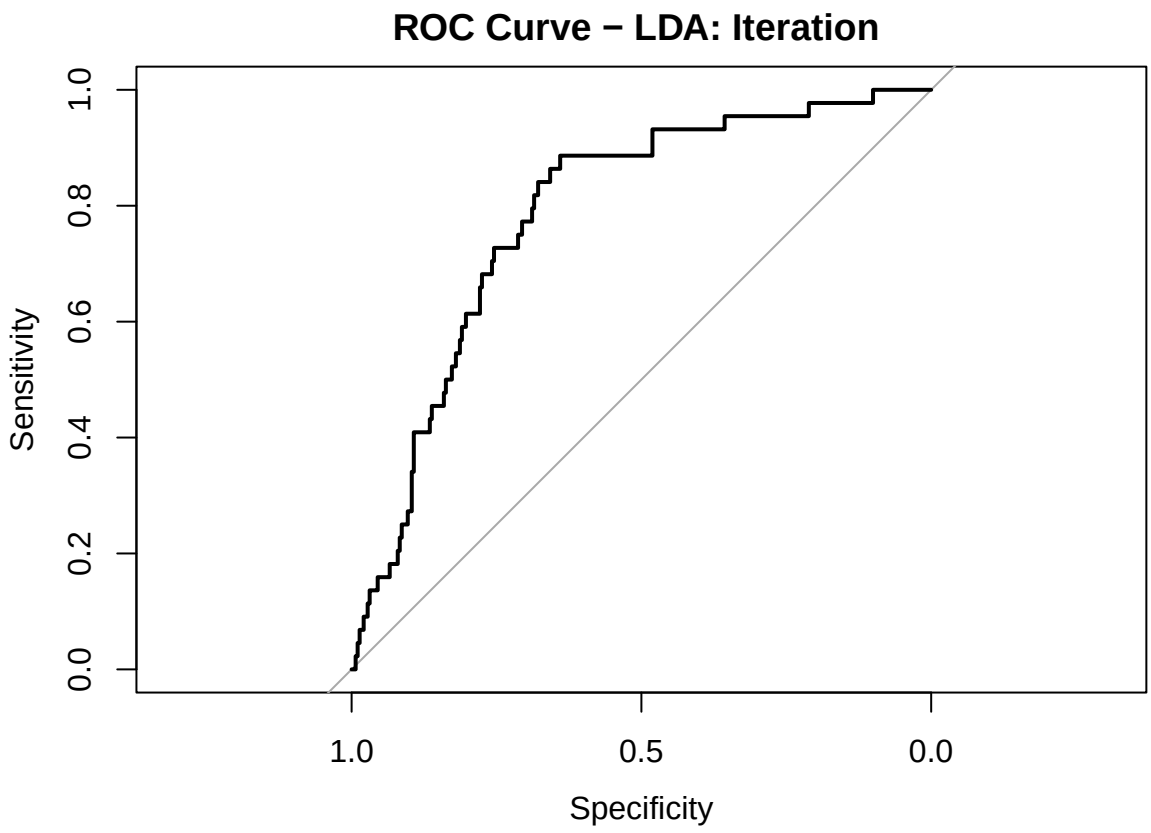
```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```




```
## Area under the curve: 0.7845
```

```
##          observed
## predicted   0   1
##         0 264  36
##         1  20  14
## [1] 0.1676647
```

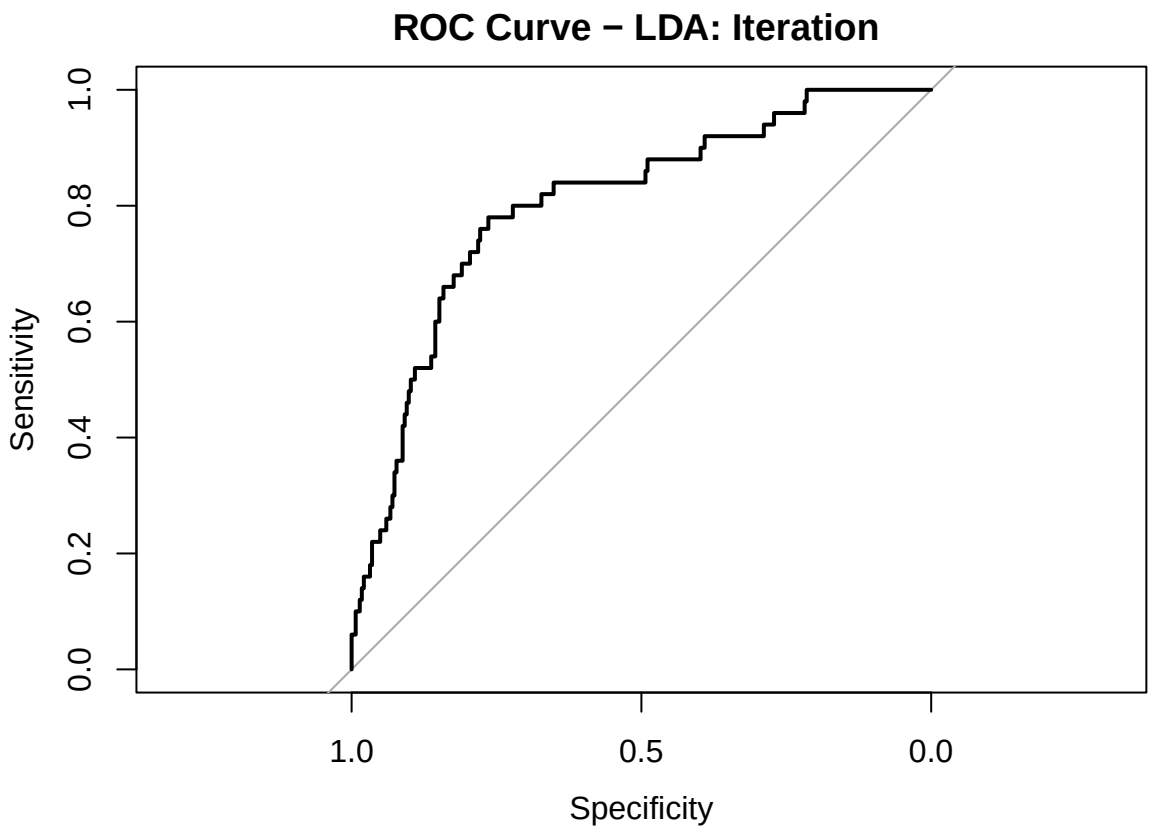
```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```




```
## Area under the curve: 0.8026
```

```
##          observed
## predicted   0   1
##         0 272  34
##         1   9  18
## [1] 0.1291291
```

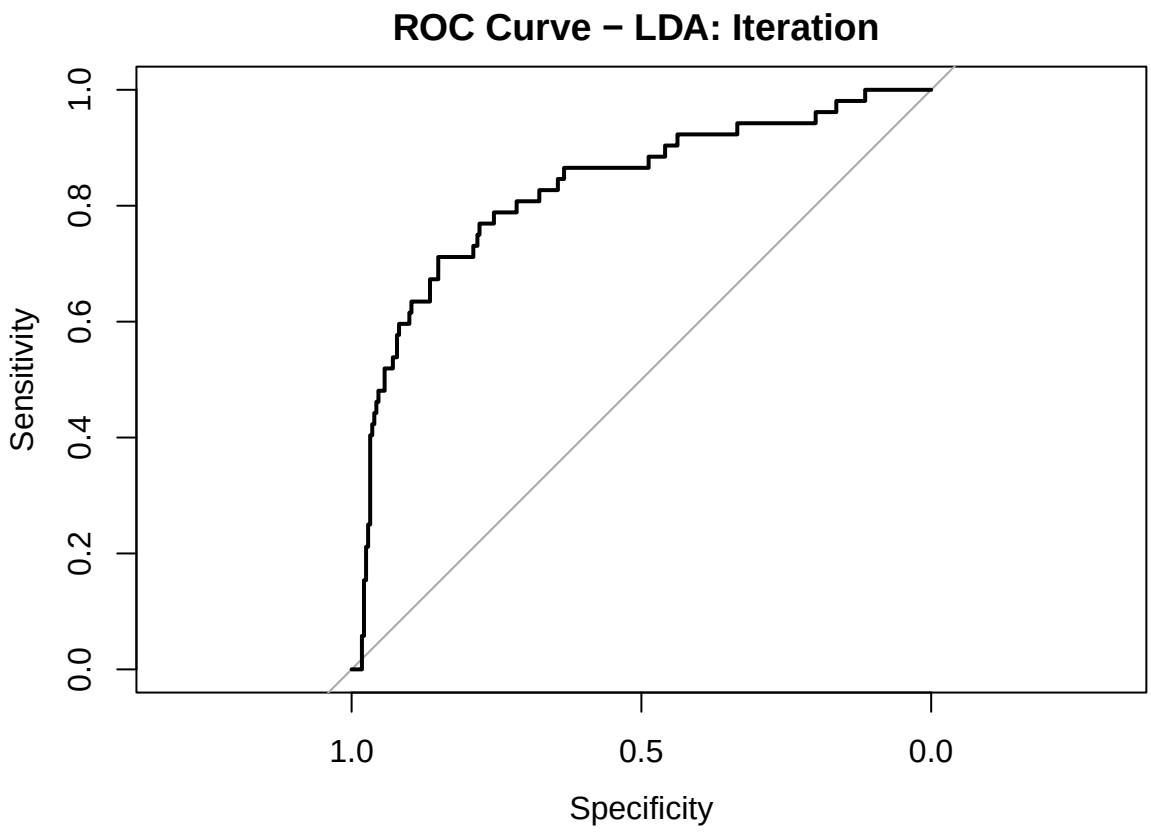
```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```




```
## Area under the curve: 0.8282
```

```
##          observed
## predicted   0   1
##         0 279  29
##         1  11  14
## [1] 0.1201201
```

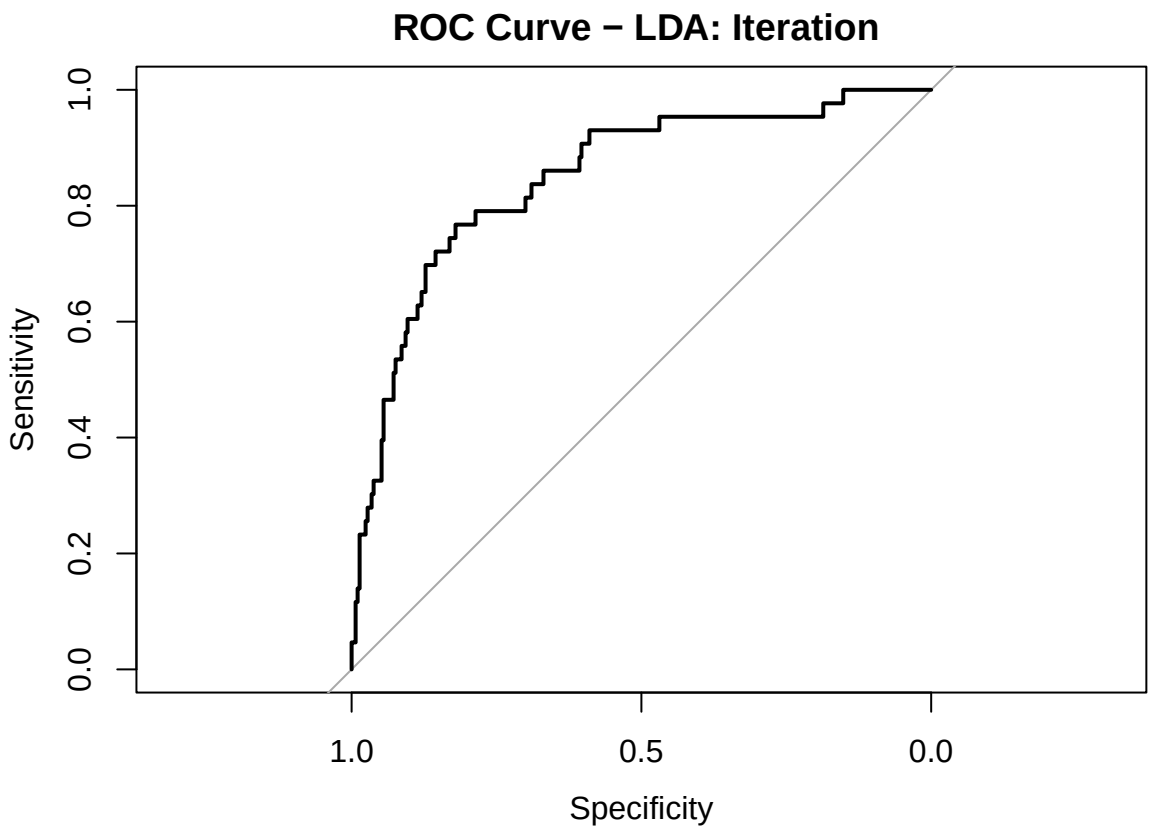
```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```




```
## Area under the curve: 0.8475
```

```
##          observed
## predicted   0   1
##         0 271  39
##         1  12  11
## [1] 0.1531532
```

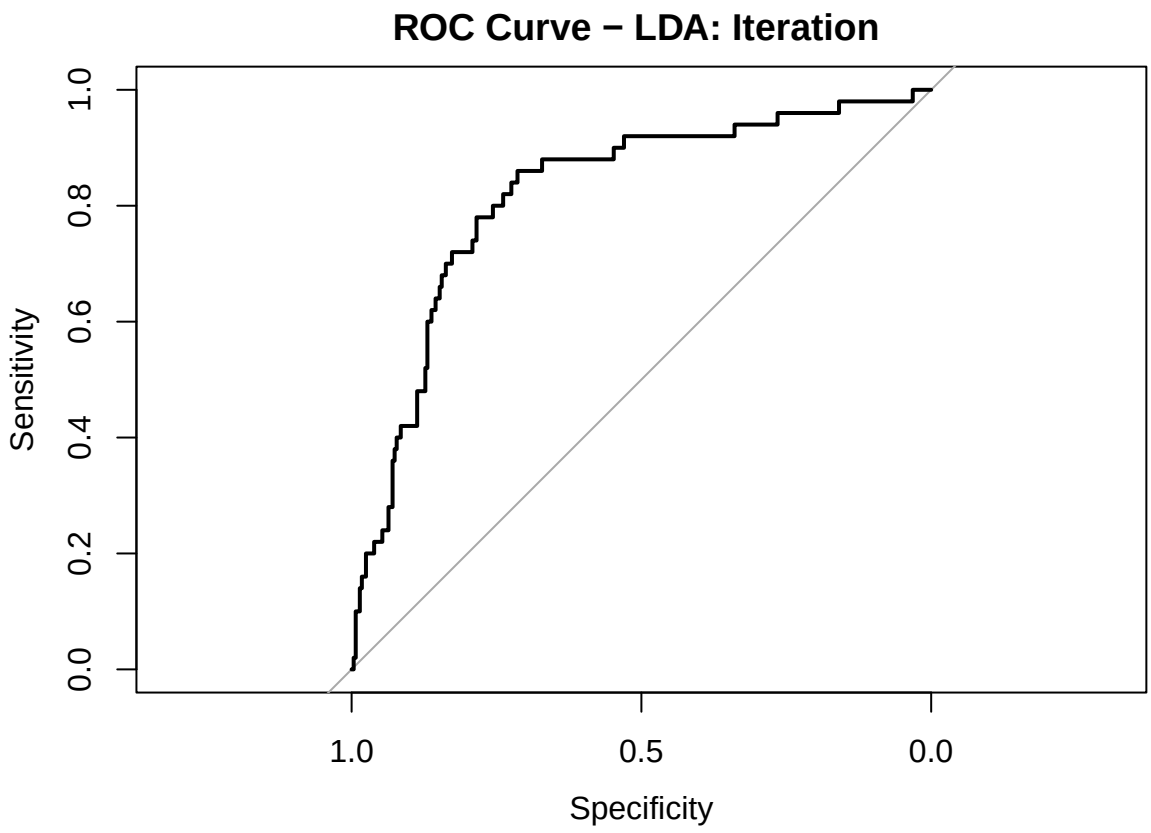
```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```




```
## Area under the curve: 0.8186
```

```
##          observed
## predicted   0   1
##         0 272  41
##         1  10  10
## [1] 0.1531532
```

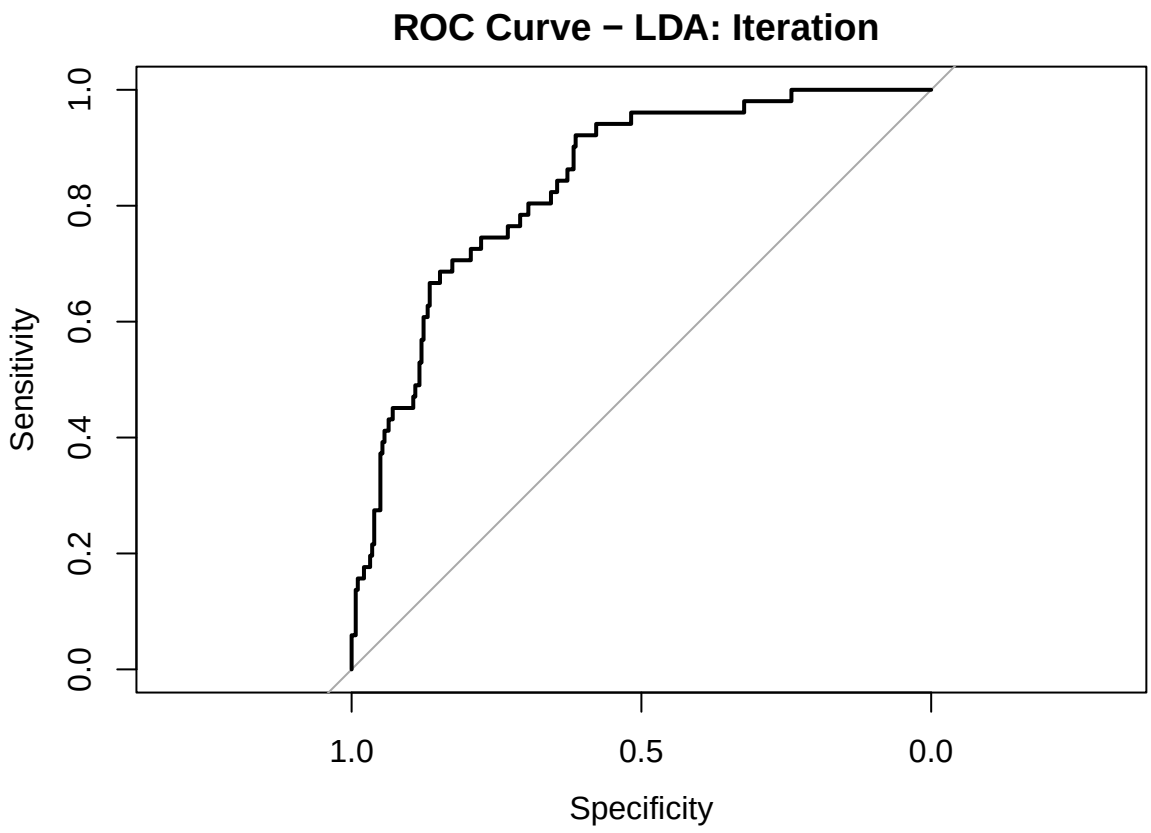
```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```



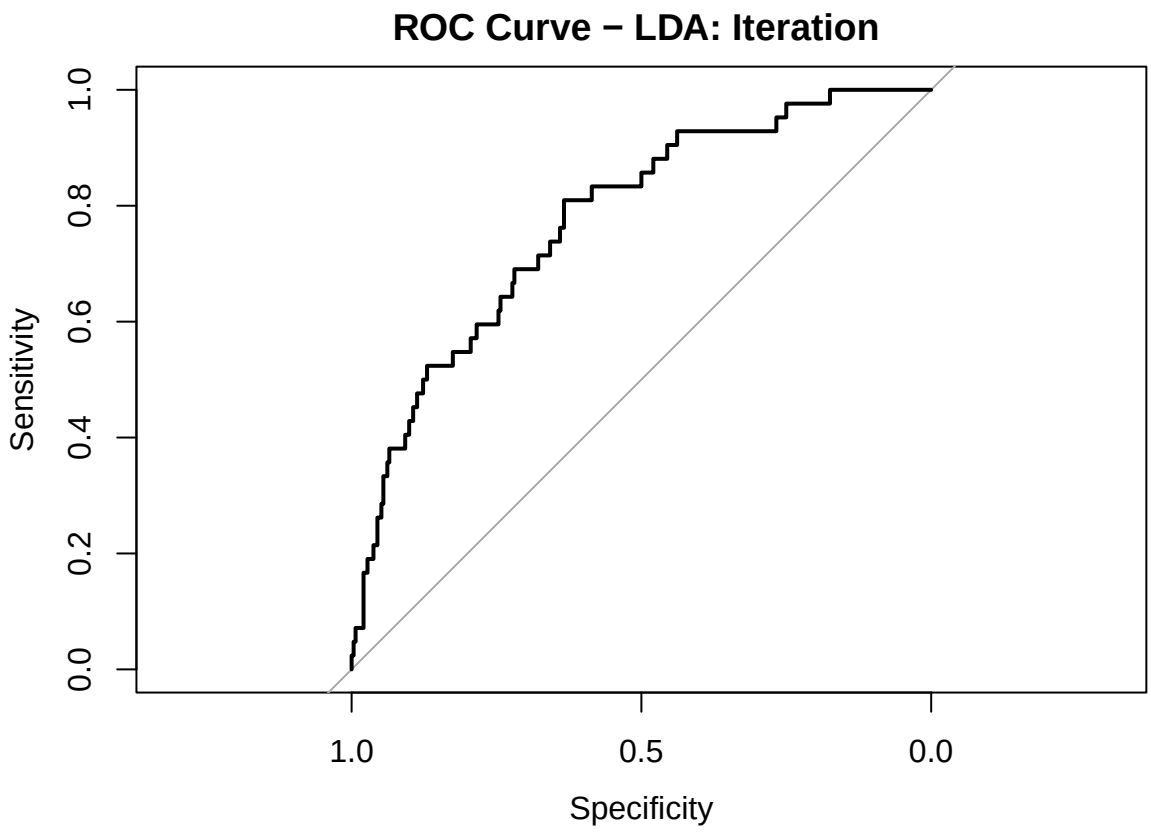

```
## Area under the curve: 0.838
```

```
##          observed
## predicted   0   1
##         0 277  30
##         1  15  12
## [1] 0.1347305
```

```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```




```
## Area under the curve: 0.7745
```




```
totalMisclassificationRate = c(totalMisclassificationRate, mean(runningTotal))
print("Average Misclassification Rate")
```

```
## [1] "Average Misclassification Rate"
```

```
totalMisclassificationRate
```

```
## [1] 0.1485048
```

```
print("Average AUC")
```

```
## [1] "Average AUC"
```

```
averageAUC = c(averageAUC, mean(auc))
print(averageAUC)
```

```
## [1] 0.8131459
```

Using all predictors except those that would not be meaningful (such as phone number), we get an average misclassification rate that is higher than our selected predictors. This is likely an indicator that the model is capturing unimportant observations.

LDA All call metrics

```
#Randomly shuffle the data
Customer_telecom <- Customer_telecom[sample(nrow(Customer_telecom)), ]

#Create 10 equally size folds
folds <- cut(seq(1, nrow(Customer_telecom)), breaks = 10, labels = FALSE)

totalMisclassificationRate = c()
runningTotal = c()

averageAUC = c()
auc = c()

#K-fold Cross Validation
for (i in 1:10) {
  testIndexes <- which(folds == i, arr.ind = TRUE)
  test_data <- Customer_telecom[testIndexes,]
  train_data <- Customer_telecom[-testIndexes,]
  m_lda2 <-
    lda(
      churn1 ~ number.vmail.messages +
        total.intl.calls +
        customer.service.calls +
        total.day.calls +
        total.night.calls,
      data = train_data
    )
  m_pred <- predict(m_lda2, test_data)
  conf <-
    table(list(predicted = m_pred$class, observed = test_data$churn1))
  confusionMatrix(conf)
  print(conf)
  totalPreds = sum(conf)
  incorrectPreds <- conf[1, 2] + conf[2, 1]
  misclassificationRate <- incorrectPreds / totalPreds
  print(misclassificationRate)
  runningTotal <- c(runningTotal, misclassificationRate)

  roc_score = roc(response = test_data$churn1,
                  predictor = m_pred$posterior[, "1"])
  auc <- c(auc, roc_score$auc)
  print(roc_score$auc)
  plot(roc_score, main = "ROC Curve - LDA: Iteration ")
}
```

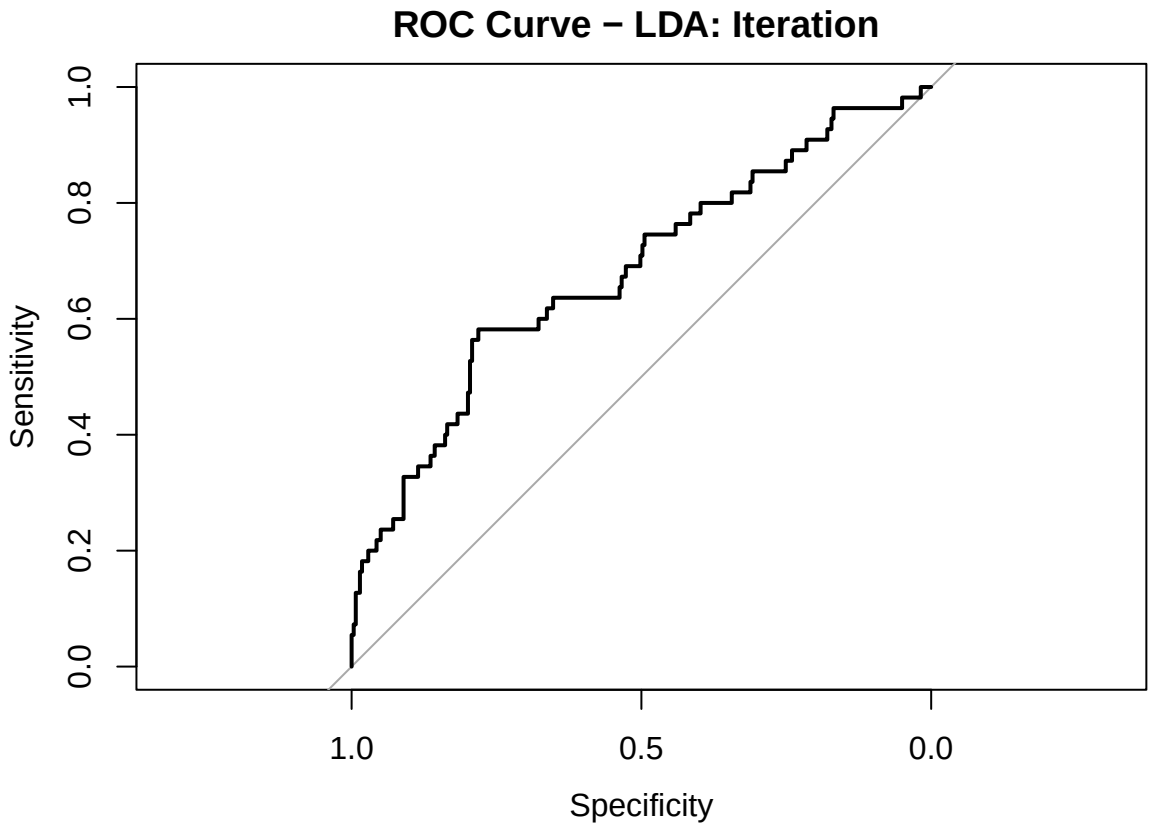
```
##          observed
## predicted   0   1
##         0 279  53
##         1   0   2
## [1] 0.1586826

## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

```
## Area under the curve: 0.6804
```

```
##          observed
## predicted   0   1
##         0 286  43
##         1   1   3
## [1] 0.1321321
```

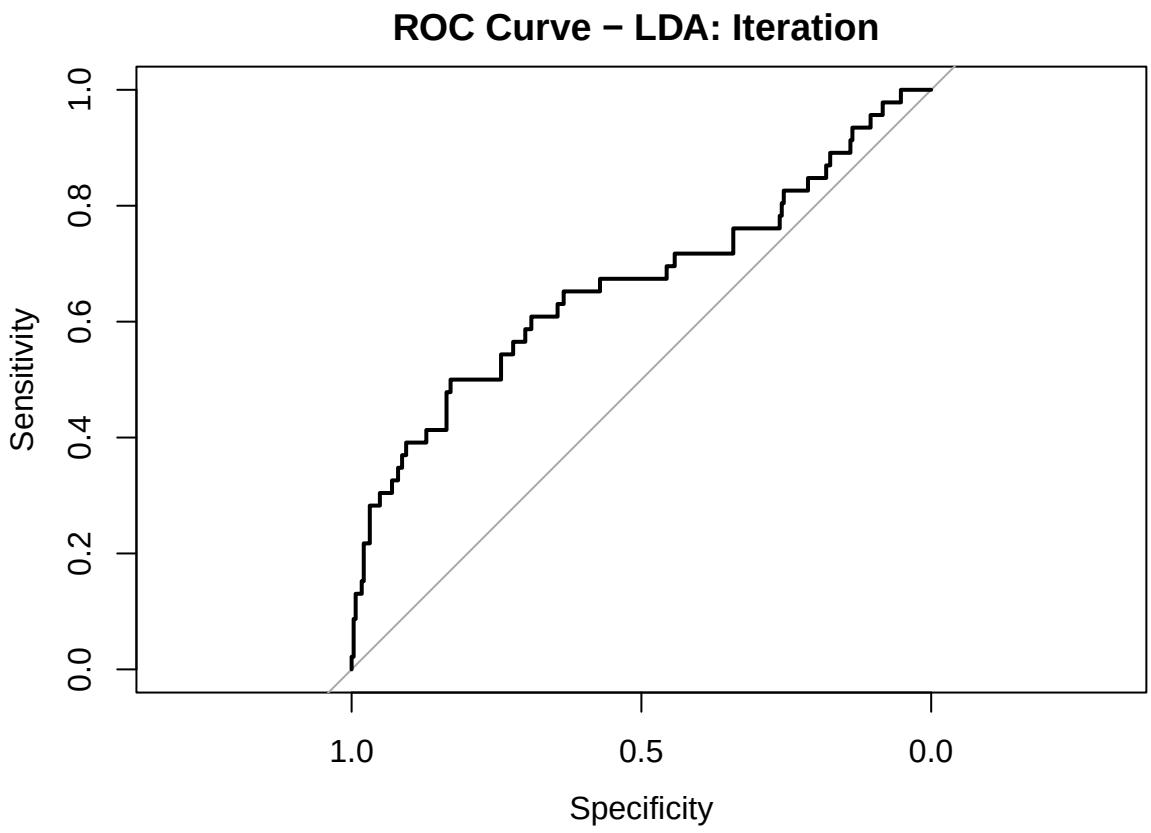
```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```




```
## Area under the curve: 0.6634
```

```
##          observed
## predicted   0   1
##         0 286  47
##         1   0   0
## [1] 0.1411411
```

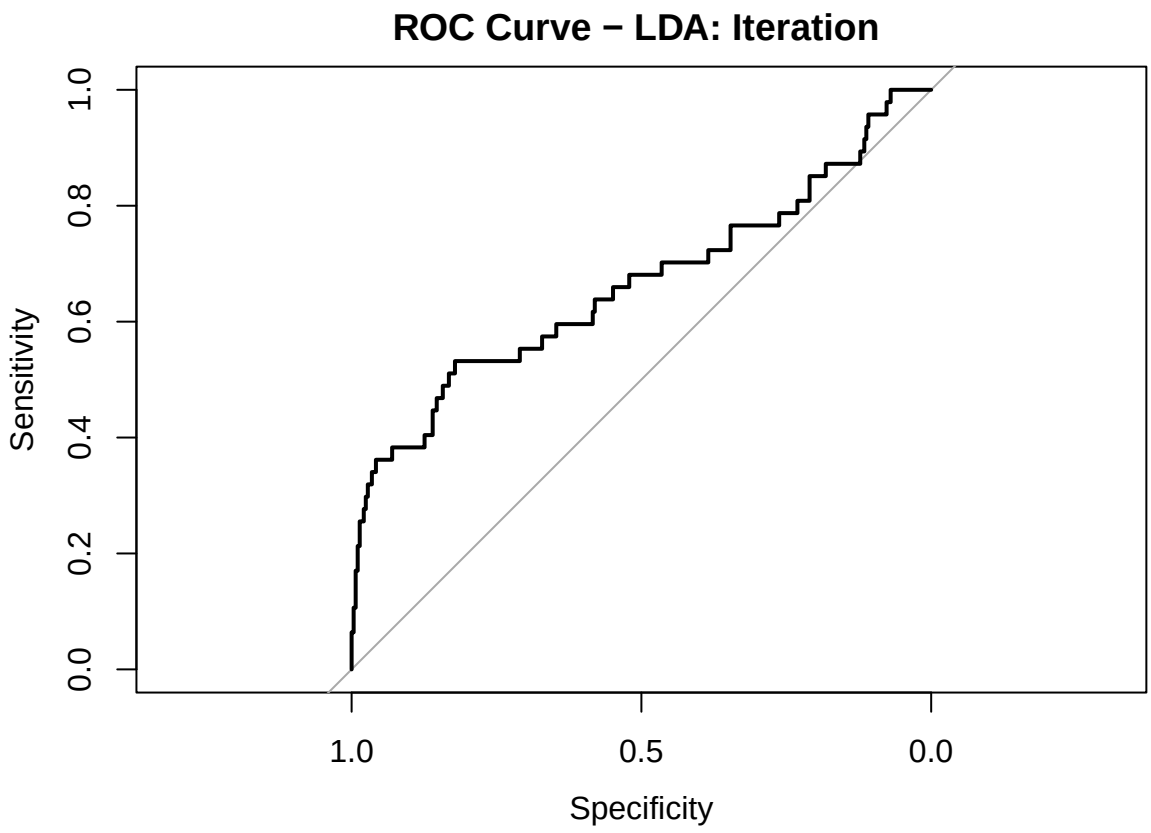
```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```




```
## Area under the curve: 0.6628
```

```
##          observed
## predicted   0   1
##         0 290  38
##         1   5   0
## [1] 0.1291291
```

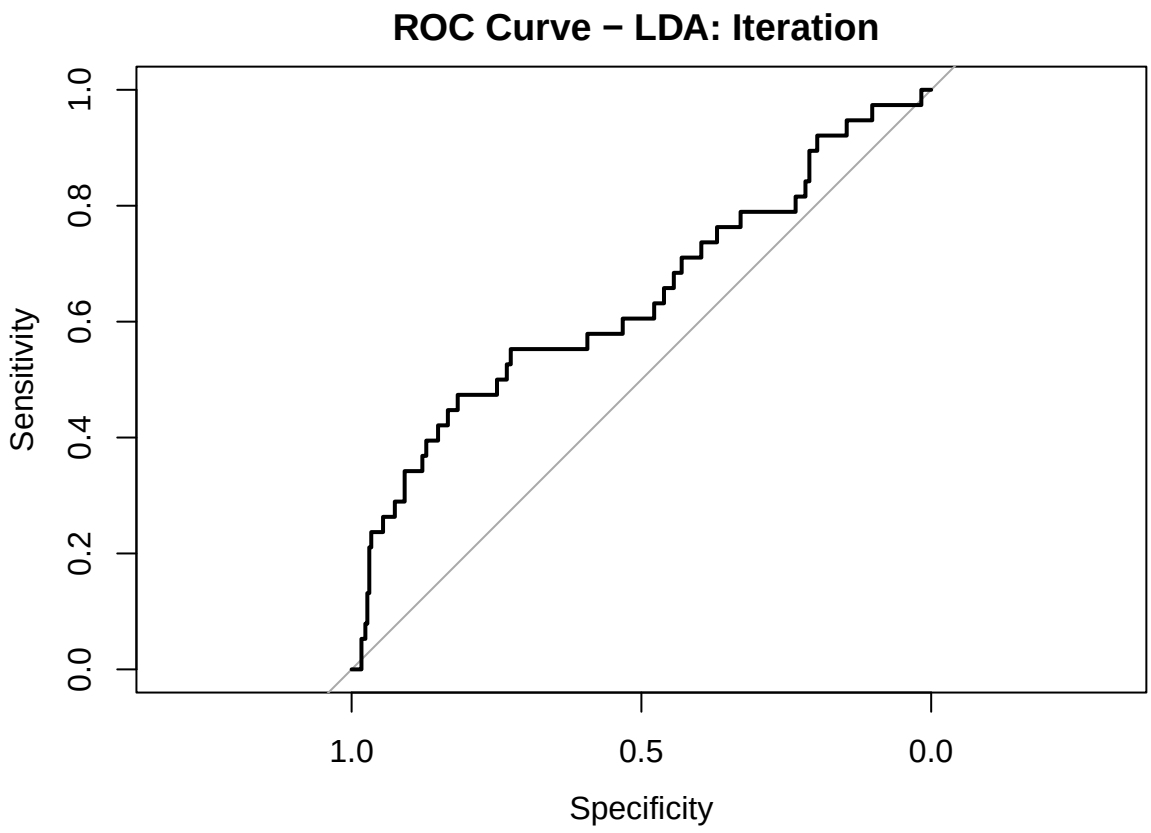
```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```




```
## Area under the curve: 0.6388
```

```
##          observed
## predicted   0   1
##         0 288  45
##         1   0   1
## [1] 0.1347305
```

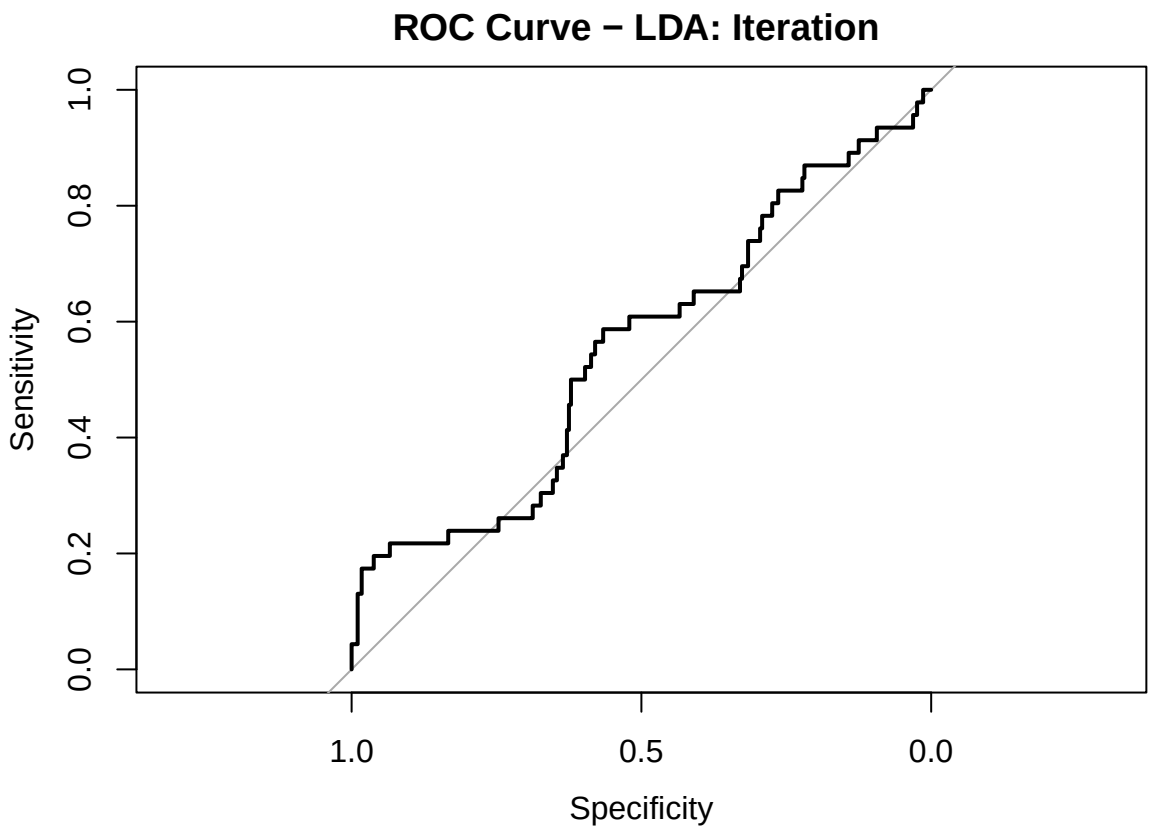
```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```




```
## Area under the curve: 0.5527
```

```
##          observed
## predicted   0   1
##         0 283  45
##         1   0   5
## [1] 0.1351351
```

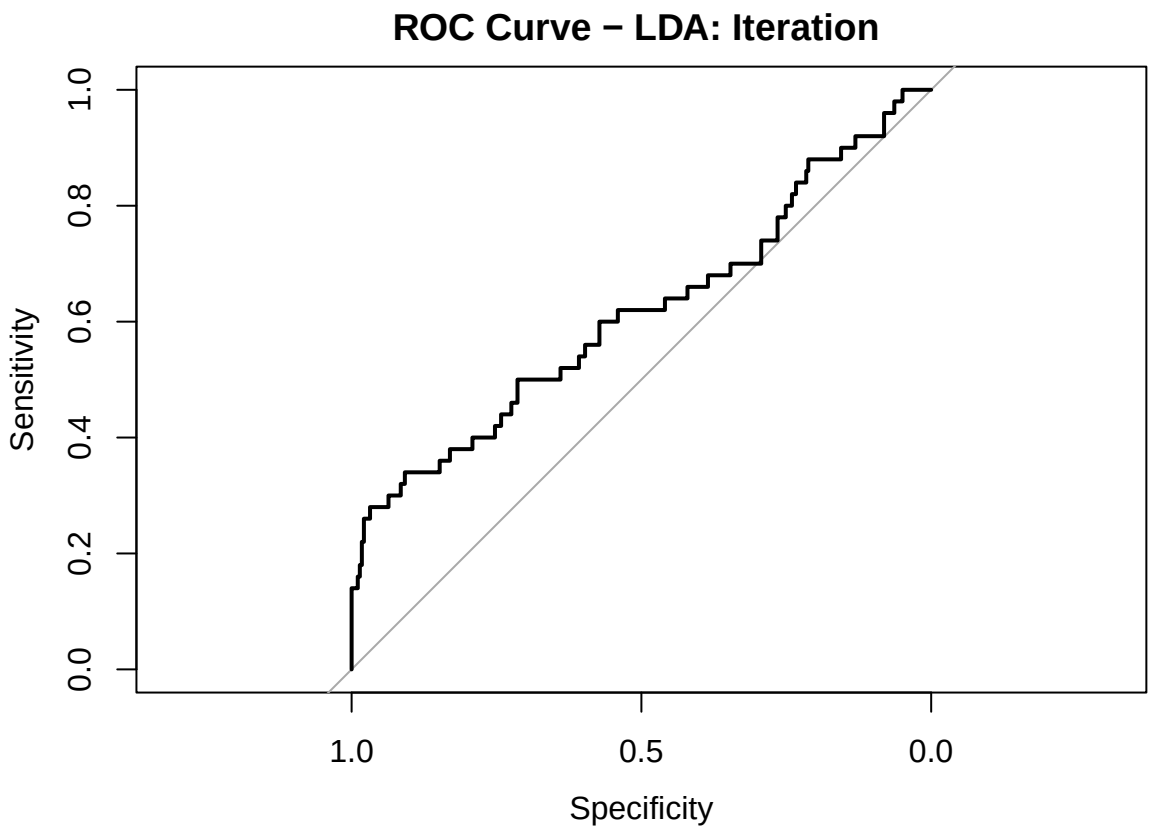
```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```




```
## Area under the curve: 0.6143
```

```
##          observed
## predicted   0   1
##         0 279  51
##         1   1   2
## [1] 0.1561562
```

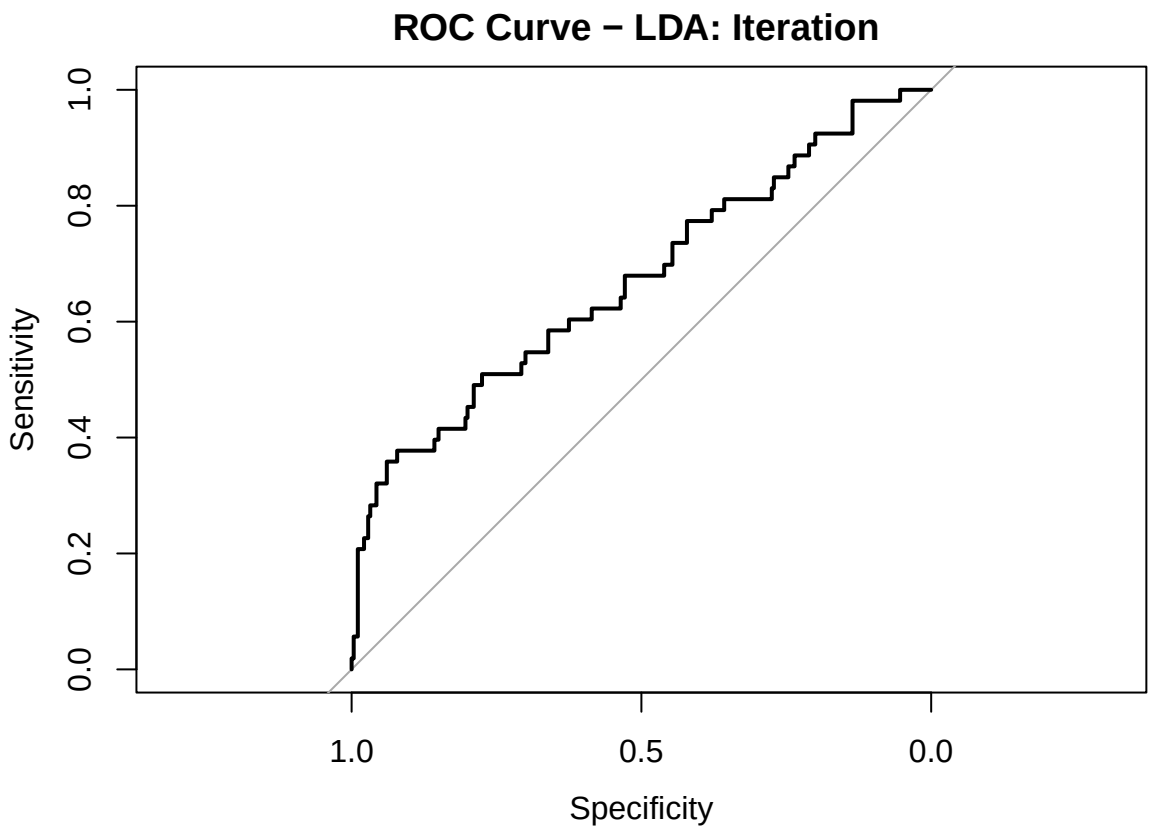
```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```




```
## Area under the curve: 0.6706
```

```
##          observed
## predicted   0   1
##         0 287  40
##         1   3   3
## [1] 0.1291291
```

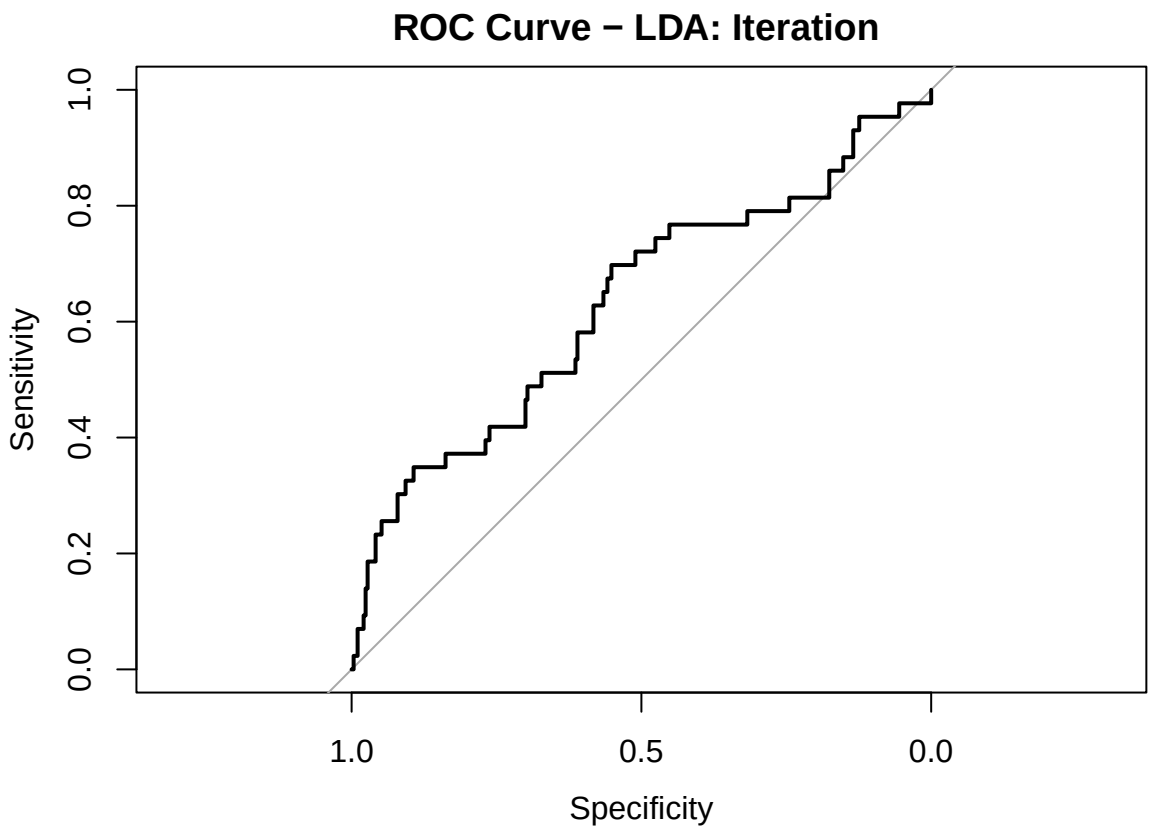
```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```




```
## Area under the curve: 0.6308
```

```
##          observed
## predicted   0   1
##         0 279  49
##         1   1   4
## [1] 0.1501502
```

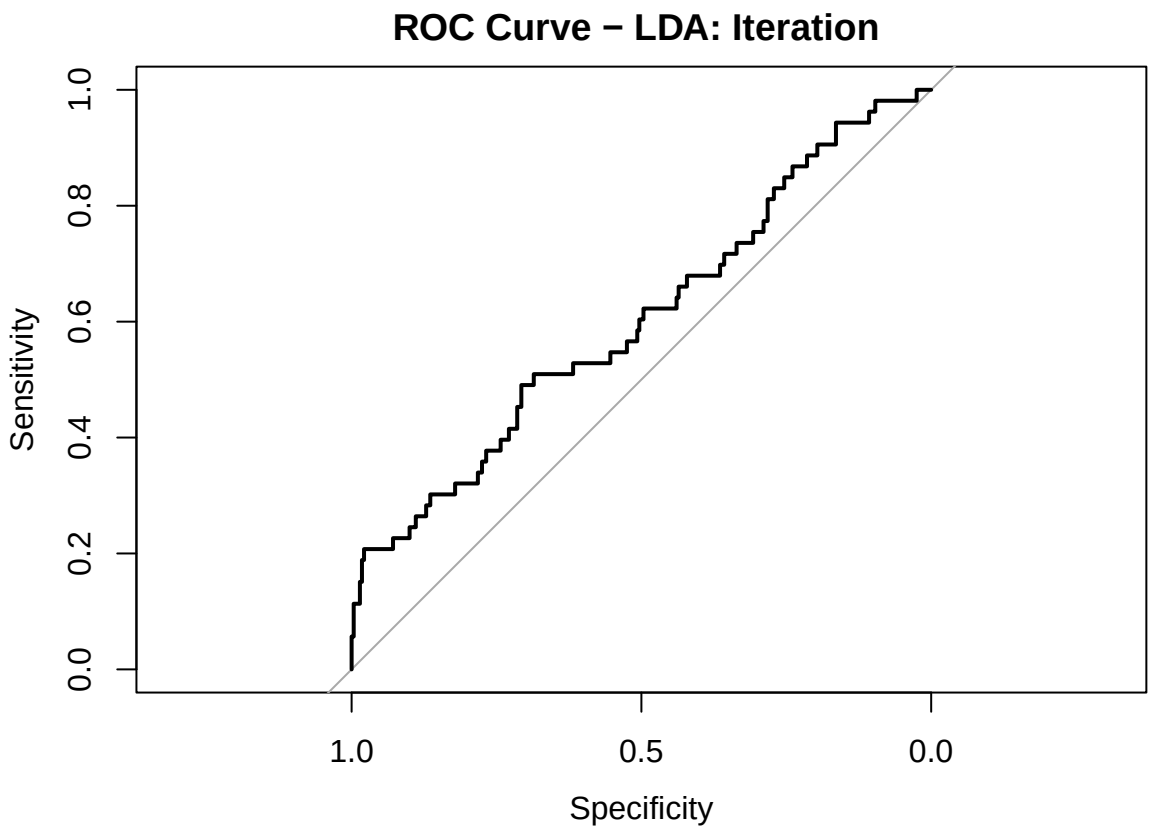
```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```



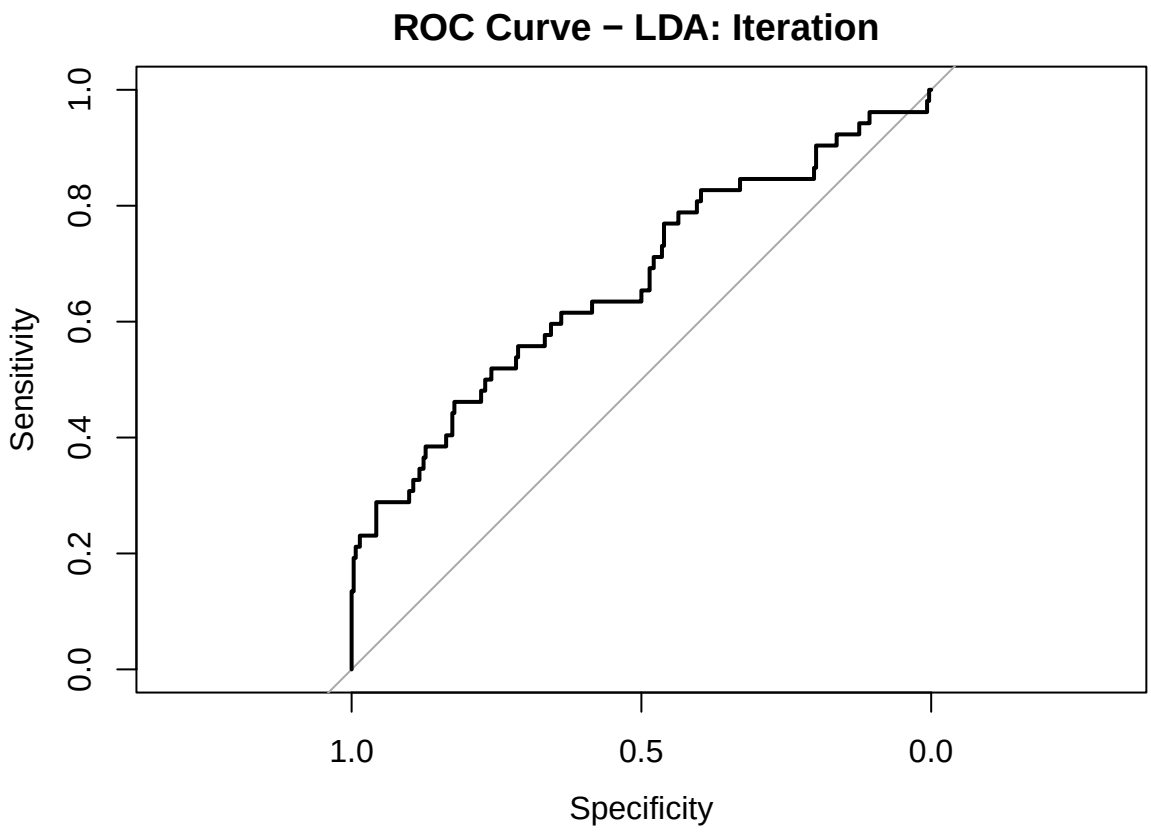

```
## Area under the curve: 0.6029
```

```
##          observed
## predicted   0   1
##         0 282  51
##         1   0   1
## [1] 0.1526946
```

```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```




```
## Area under the curve: 0.6686
```




```
totalMisclassificationRate = c(totalMisclassificationRate, mean(runningTotal))
print("Average Misclassification Rate")
```

```
## [1] "Average Misclassification Rate"
```

```
totalMisclassificationRate
```

```
## [1] 0.1419081
```

```
print("Average AUC")
```

```
## [1] "Average AUC"
```

```
averageAUC = c(averageAUC, mean(auc))
print(averageAUC)
```

```
## [1] 0.6385162
```

Using the top predictors associated with call metrics, we get a very similar rate as with the top three predictors. This likely means that the significance of the additional predictors is already captured in the top three predictors.