Assignment 2 - Logistic Regression

Eoin Flynn

5 March 2018

Bond University
Data Science

Contents

Functions

This section will hold all of the functions that will be used throughout this markdown.

```r
# Create a decision tree
createDecisionTreeModel <- function(formula, dataset, maxdepth) {
    suppressMessages(library(party))
    decisionTreeModel <- ctree(formula, data = dataset, controls = ctree_control(maxdepth = maxdepth))

    return(decisionTreeModel)
}

# Change rows to factors
setRowAsFactor <- function(dataset, columns) {
    for (column in columns) {
        dataset[, column] <- as.factor(dataset[, column])
    }
    return(dataset)
}

# Create a logistic regression model
createLogisticRegressionModel <- function(formula, family = binomial,
    dataset) {
    logisticRegressionModel = glm(formula, family = family, data = dataset)

    return(logisticRegressionModel)
}

# Create a prediction dataframe
createPrediction_df <- function(model, dataset, predictionType = "response",
    oneClass, zeroClass) {
    # Run the prediction
    prediction <- predict(model, dataset, type = predictionType)
    # Convert to a dataframe
    prediction_df <- data.frame(prediction)
    # Rename the column to reference easier
    colnames(prediction_df) <- "probabilities"
    # Add a row for the classification
    prediction_df$classification <- rep(zeroClass, nrow(prediction_df))
    # Convert all probabilites above 0.5 to be the affirmative
    # class
    prediction_df$classification[prediction_df$probabilities >
        0.5] <- oneClass
    prediction_df$classification <- as.factor(prediction_df$classification)


    return(prediction_df)
}

# Get model performance for plotting ROC curve
getModelPerformance <- function(model, dataset, outcomeColumn,
    type = "response", xAxis = "tpr", yAxis = "fpr") {
    suppressMessages(library(ROCR))
    # Create a predict variable
    predict <- predict(model, dataset, type = type)
    # Create a predicition variable
    predicition <- prediction(predict, outcomeColumn)
    # Create the performance variable
    performance <- performance(predicition, xAxis, yAxis)

    return(performance)
}

# Plot ROC curves
plotROCCurves <- function(model1, model2, main, model1Colour = "#009900",
    model2Colour = "#FF8000", model1Name, model2Name, legendLocation = "bottomright") {
    plot(model1, main = main, col = model1Colour, print.auc = TRUE)
    plot(model2, add = T, col = model2Colour)
    legend(legendLocation, legend = paste(rep(c(model1Name, model2Name))),
        col = c(model1Colour, model2Colour), cex = 0.8, fill = c(model1Colour,
            model2Colour))
}

# Calculate AUC. Returns as a decimal
getAUC <- function(outcomeColumn, dataset, model, oneClass, zeroClass) {
    suppressMessages(library(ModelMetrics))
    prediction_df <- createPrediction_df(model, dataset, oneClass = oneClass,
        zeroClass = zeroClass)
    auc <- auc(outcomeColumn, prediction_df$classification)

    return(auc)
}

# Create a confusion matrix
createConfusionMatrix <- function(model, dataset, outcomeColumn,
    oneClass, zeroClass) {
    suppressMessages(library(caret))
    prediction_df <- createPrediction_df(model, dataset, oneClass = oneClass,
        zeroClass = zeroClass)
    userConfusionMatrix <- table(outcomeColumn, prediction_df$classification)

    return(userConfusionMatrix)
}
```

Data

In this section we will load in our data and do some basic data exploration.

```
suppressMessages(library(RMySQL))

USER <- "root"
PASSWORD <- "A13337995"
HOST <- "localhost"
DBNAME <- "world"

statement <- "Select * from world.customerChurn"
```

```
db <- dbConnect(MySQL(), user = USER, password = PASSWORD, host = HOST,
    dbname = DBNAME, port = 3306)
customerDataset <- dbGetQuery(db, statement = statement)
dbDisconnect(db)
```

```
## [1] TRUE
```

```
# Loops through and changes all relevant rows to factors and
# returns the dataset post modification


customerDataset <- setRowAsFactor(customerDataset, c("gender",
    "SeniorCitizen", "Partner", "Dependents", "PhoneService",
    "MultipleLines", "InternetService", "OnlineSecurity", "OnlineBackup",
    "DeviceProtection", "TechSupport", "StreamingTV", "StreamingMovies",
    "Contract", "PaperlessBilling", "PaymentMethod", "Churn"))

# Drop the columns that will not be needed
customerDataset = customerDataset[, -which(names(customerDataset) %in%
    c("customerID"))]
```

Split Data

We will now split our data into test and training sets. The purpose of this is to create a sample of data that the model has never seen before in order to gauge its accuracy. The training set will consist of 80% of the data while the remaining 20% will constitute the test set.

```
suppressMessages(library(caTools))

# Set the seed to reproducability
set.seed(12216)

# Create our two datasets
sample <- sample.split(customerDataset, SplitRatio = 0.8)
train_df <- subset(customerDataset, sample == TRUE)
test_df <- subset(customerDataset, sample == FALSE)

# We can now see that the data is split approximately 80:20
print(sprintf("The full dataset has %s observations", NROW(customerDataset)))
```

```
## [1] "The full dataset has 7032 observations"
```

```
print(sprintf("The training dataset has %s observations", NROW(train_df)))
```

```
## [1] "The training dataset has 5628 observations"
```

```
print(sprintf("The testing dataset has %s observations", NROW(test_df)))
```

```
## [1] "The testing dataset has 1404 observations"
```

```
# Check to see how many customers churned in each dataset
table(train_df$Churn)
```

```
##
##   No  Yes
## 4120 1508
```

```r
table(test_df$Churn)
```

```
##
##   No  Yes
## 1043  361
```

```r
# We can see that each dataset holds approximately the same
# proportion of customers who churned
print(sprintf("%.2f%% of the training set churned", ((NROW(subset(train_df,
    Churn == "Yes")))/NROW(train_df) * 100)))
```

```
## [1] "26.79% of the training set churned"
```

```r
print(sprintf("%.2f%% of the testing set churned", ((NROW(subset(test_df,
    Churn == "Yes")))/NROW(test_df) * 100)))
```

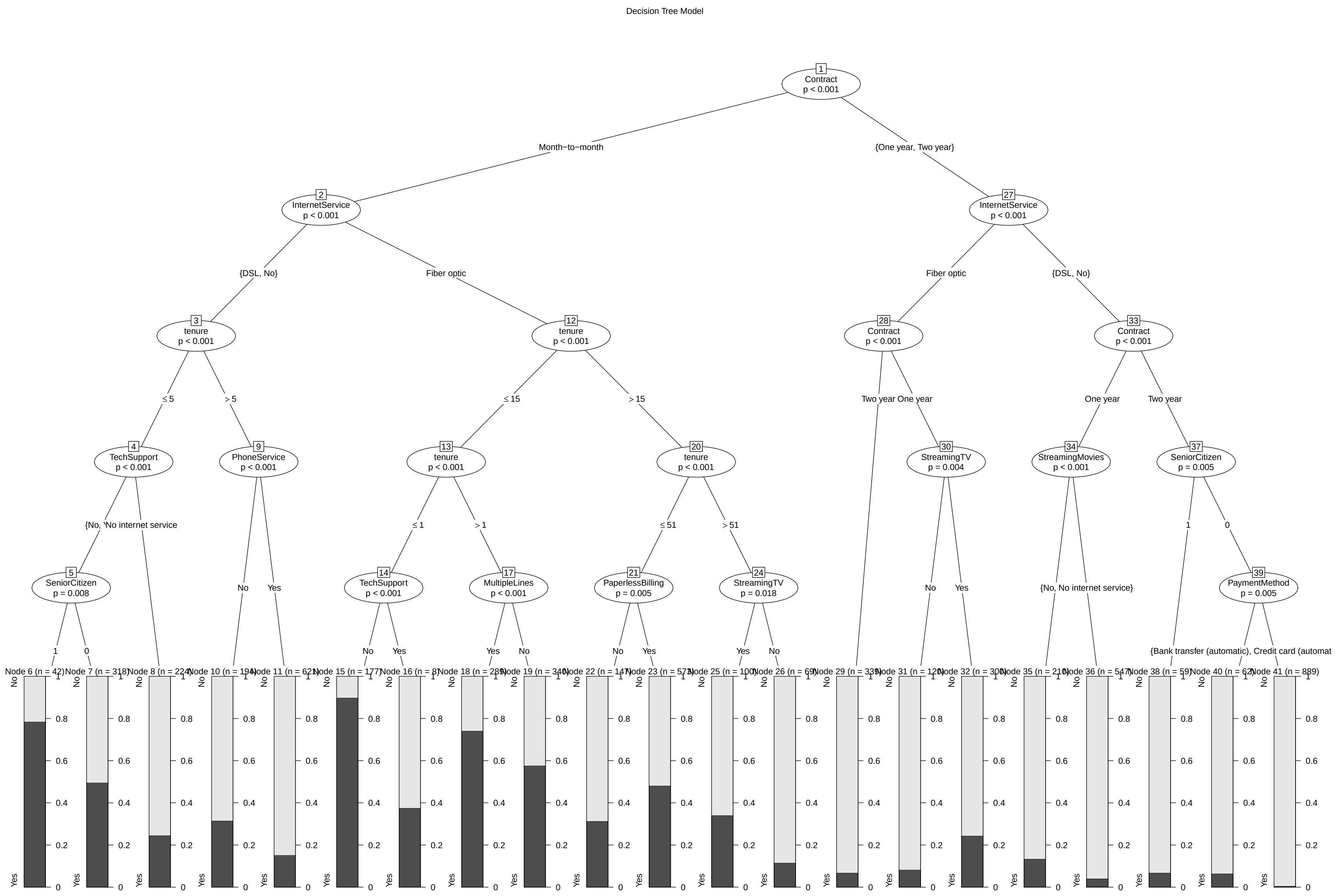
```
## [1] "25.71% of the testing set churned"
```

Decision Tree

We want to first make a decision tree to determine which variables are best able to predict whether a customer will churn. We already know from our previous report that the optimal model is however this time the tree will only be run on the training dataset.

```r
# Create and plot the decision tree
decisionTreeModel = createDecisionTreeModel(formula = Churn ~
    ., dataset = train_df, maxdepth = 5)
```

```r
plot(decisionTreeModel, main = "Decision Tree Model", type = "extended",
    newpage = TRUE)
```


Decision Tree Model
1 Contract p < 0.001
Month−to−month
{One year, Two year}
2 InternetService p < 0.001
{DSL, No}
Fiber optic
27 InternetService p < 0.001
Fiber optic
{DSL, No}
3 tenure p < 0.001
≤ 5
> 5
12 tenure p < 0.001
≤ 15
> 15
28 Contract p < 0.001
Two year
One year
33 Contract p < 0.001
One year
Two year
4 TechSupport p < 0.001
{No, '
No internet service
9 PhoneService p < 0.001
No
Yes
13 tenure p < 0.001
≤ 1
> 1
20 tenure p < 0.001
≤ 51
> 51
30 StreamingTV p = 0.004
No
Yes
34 StreamingMovies p < 0.001
{No, No internet service}
37 SeniorCitizen p = 0.005
1
0
5 SeniorCitizen p = 0.008
1
0
14 TechSupport p < 0.001
No
Yes
17 MultipleLines p < 0.001
Yes
No
21 PaperlessBilling p = 0.005
No
Yes
24 StreamingTV p = 0.018
Yes
No
39 PaymentMethod p = 0.005
{Bank transfer (automatic), Credit card (automat
Node 6 (n = 42)
Node 7 (n = 318)
Node 8 (n = 224)
Node 10 (n = 194)
Node 11 (n = 621)
Node 15 (n = 177)
Node 16 (n = 8)
Node 18 (n = 289)
Node 19 (n = 340)
Node 22 (n = 147)
Node 23 (n = 573)
Node 25 (n = 100)
Node 26 (n = 69)
Node 29 (n = 339)
Node 31 (n = 120)
Node 32 (n = 300)
Node 35 (n = 210)
Node 36 (n = 547)
Node 38 (n = 59)
Node 40 (n = 62)
Node 41 (n = 889)
No
Yes
0
0.2
0.4
0.6
0.8
1

∞

From looking at the decision tree it is clear that the top three variables are Contract, InternetService, and Tenure. Below those three levels, other variables such as StreamingTV, and TechSupport become relevant. To develop the best performing model with this dataset we will create a regression using only those three top level variables, and then a second using all variables. The results from each model will then be compared before the best model is presented to management.

Logistic Regression

We will now create multiple logistic regressions using GLM package. We will compare and then optimize before providing a final model to management for business use.

All Variables Logistic Regression

In this section we will create a logistic regression using all variables in the training dataset and look at statistical significance of each. Later in the report we will assess the accuracy of this model against others.

```
# We will start by first making a regression using all
# variables
allVariablesLogisticRegressionModel <- createLogisticRegressionModel(formula = Churn ~
    ., dataset = train_df)

# Print a summary of the regression
print("Model Summary")
```

```
## [1] "Model Summary"
```

```
summary(allVariablesLogisticRegressionModel)
```

```
##
## Call:
## glm(formula = formula, family = family, data = dataset)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.9271  -0.6763  -0.2846   0.7434   3.4193
##
## Coefficients: (7 not defined because of singularities)
##                                       Estimate Std. Error z value
## (Intercept)                          1.168e+00  9.050e-01   1.290
## genderMale                           5.431e-02  7.237e-02   0.750
## SeniorCitizen1                       2.279e-01  9.429e-02   2.417
## PartnerYes                           3.171e-02  8.752e-02   0.362
## DependentsYes                       -1.351e-01  1.002e-01  -1.348
## tenure                              -5.790e-02  6.865e-03  -8.434
## PhoneServiceYes                      1.985e-01  7.223e-01   0.275
## MultipleLinesNo phone service               NA         NA      NA
## MultipleLinesYes                     4.521e-01  1.974e-01   2.290
## InternetServiceFiber optic           1.765e+00  8.883e-01   1.987
## InternetServiceNo                   -1.900e+00  8.981e-01  -2.116
## OnlineSecurityNo internet service           NA         NA      NA
## OnlineSecurityYes                   -2.001e-01  1.994e-01  -1.003
## OnlineBackupNo internet service             NA         NA      NA
## OnlineBackupYes                      6.721e-02  1.948e-01   0.345
## DeviceProtectionNo internet service          NA         NA      NA
## DeviceProtectionYes                    1.476e-01  1.965e-01   0.751
## TechSupportNo internet service                NA         NA      NA
## TechSupportYes                        -1.518e-01  2.011e-01  -0.755
## StreamingTVNo internet service                NA         NA      NA
## StreamingTVYes                         6.178e-01  3.625e-01   1.704
## StreamingMoviesNo internet service            NA         NA      NA
## StreamingMoviesYes                     5.822e-01  3.650e-01   1.595
## ContractOne year                      -6.773e-01  1.197e-01  -5.657
## ContractTwo year                      -1.424e+00  2.001e-01  -7.114
## PaperlessBillingYes                    3.382e-01  8.284e-02   4.083
## PaymentMethodCredit card (automatic) -5.187e-02  1.273e-01  -0.407
## PaymentMethodElectronic check          3.014e-01  1.056e-01   2.855
## PaymentMethodMailed check             -4.768e-02  1.280e-01  -0.372
## MonthlyCharges                        -4.160e-02  3.531e-02  -1.178
## TotalCharges                           2.991e-04  7.846e-05   3.813
##                                       Pr(>|z|)
## (Intercept)                           0.197001
## genderMale                            0.452970
## SeniorCitizen1                        0.015645 *
## PartnerYes                            0.717142
## DependentsYes                         0.177760
## tenure                                 < 2e-16 ***
## PhoneServiceYes                       0.783462
## MultipleLinesNo phone service               NA
## MultipleLinesYes                      0.022016 *
## InternetServiceFiber optic            0.046872 *
## InternetServiceNo                     0.034383 *
## OnlineSecurityNo internet service           NA
## OnlineSecurityYes                     0.315685
## OnlineBackupNo internet service             NA
## OnlineBackupYes                       0.730109
## DeviceProtectionNo internet service         NA
## DeviceProtectionYes                   0.452539
## TechSupportNo internet service              NA
## TechSupportYes                        0.450396
## StreamingTVNo internet service              NA
## StreamingTVYes                        0.088343 .
## StreamingMoviesNo internet service          NA
## StreamingMoviesYes                    0.110691
## ContractOne year                      1.54e-08 ***
## ContractTwo year                      1.13e-12 ***
## PaperlessBillingYes                   4.45e-05 ***
## PaymentMethodCredit card (automatic) 0.683705
## PaymentMethodElectronic check         0.004305 **
## PaymentMethodMailed check             0.709541
## MonthlyCharges                        0.238771
## TotalCharges                          0.000138 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 6542.0  on 5627  degrees of freedom
## Residual deviance: 4674.1  on 5604  degrees of freedom
## AIC: 4722.1
##
## Number of Fisher Scoring iterations: 6
```

```
# Convert Betas into odds ratio
print("Odds Ratios")
```

```
## [1] "Odds Ratios"
```

```
exp(coef(allVariablesLogisticRegressionModel))
```

```
##                           (Intercept)                            genderMale
##                             3.2142616                             1.0558152
##                        SeniorCitizen1                            PartnerYes
##                             1.2559792                             1.0322134
##                         DependentsYes                                tenure
##                             0.8736378                             0.9437414
##                       PhoneServiceYes         MultipleLinesNo phone service
##                             1.2195730                                    NA
##                      MultipleLinesYes            InternetServiceFiber optic
##                             1.5716412                             5.8440322
##                     InternetServiceNo    OnlineSecurityNo internet service
##                             0.1495586                                    NA
##                     OnlineSecurityYes      OnlineBackupNo internet service
##                             0.8186677                                    NA
##                       OnlineBackupYes  DeviceProtectionNo internet service
##                             1.0695253                                    NA
##                   DeviceProtectionYes       TechSupportNo internet service
##                             1.1590444                                    NA
##                        TechSupportYes       StreamingTVNo internet service
##                             0.8591911                                    NA
##                        StreamingTVYes   StreamingMoviesNo internet service
##                             1.8547853                                    NA
##                    StreamingMoviesYes                      ContractOne year
##                             1.7900570                             0.5079813
##                      ContractTwo year                   PaperlessBillingYes
##                             0.2408675                             1.4024619
## PaymentMethodCredit card (automatic)        PaymentMethodElectronic check
##                             0.9494527                             1.3517921
##            PaymentMethodMailed check                        MonthlyCharges
##                             0.9534397                             0.9592580
##                          TotalCharges
##                             1.0002992
```

From the model's summary we can see that our top three variables identified earlier are all have a high statistical significance. TotalCharges is also statistically different from zero however since the decision tree deemed that it had no predictive information we will not be including it in our model building

Top Three Logistic Regression

We will now create our logistic regression using only the top three variables from our decision tree (Contract, InternetService, and Tenure)

```r
# We will start by first making a regression using all
# variables
topThreeLogisticRegressionModel <- createLogisticRegressionModel(formula = Churn ~
    Contract + InternetService + tenure, dataset = train_df)

# Print a summary of the regression
print("Model Summary")
```

```
## [1] "Model Summary"
```

```r
summary(topThreeLogisticRegressionModel)
```

```
##
## Call:
## glm(formula = formula, family = family, data = dataset)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.5640  -0.6746  -0.3077   0.8350   3.1500
##
## Coefficients:
##                              Estimate Std. Error z value Pr(>|z|)
## (Intercept)                 -0.257388   0.070388  -3.657 0.000255 ***
## ContractOne year            -0.870290   0.113829  -7.646 2.08e-14 ***
## ContractTwo year            -1.740473   0.191348  -9.096  < 2e-16 ***
## InternetServiceFiber optic  1.163022   0.080631  14.424  < 2e-16 ***
## InternetServiceNo           -1.121017   0.131645  -8.515  < 2e-16 ***
## tenure                      -0.031107   0.002167 -14.353  < 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: 6542.0  on 5627  degrees of freedom
## Residual deviance: 4852.3  on 5622  degrees of freedom
## AIC: 4864.3
##
## Number of Fisher Scoring iterations: 6
```

```r
# Convert Betas into odds ratio
print("Odds Ratios")
```

```
## [1] "Odds Ratios"
```

```r
exp(coef(topThreeLogisticRegressionModel))
```

```
##                (Intercept)           ContractOne year
##                  0.7730685                  0.4188300
##           ContractTwo year InternetServiceFiber optic
##                  0.1754374                  3.1995868
##          InternetServiceNo                     tenure
##                  0.3259482                  0.9693719
```

Model Comparison

Now that we have create all three models we will test their accuracy against the training dataset and also the test dataset. One of the easiest ways to see which model is more accurate is to use an ROC curve and measure the area under each curve, the larger the area under the curve the more accurate the model is. We will first test each model using the training dataset before testing each one individually with the test dataset to gauge how robust they are. Once we have established whether the models are robust we will create a pair of confusion matricies using the test dataset to see which model had the highest percentage of churns predicted correctly.

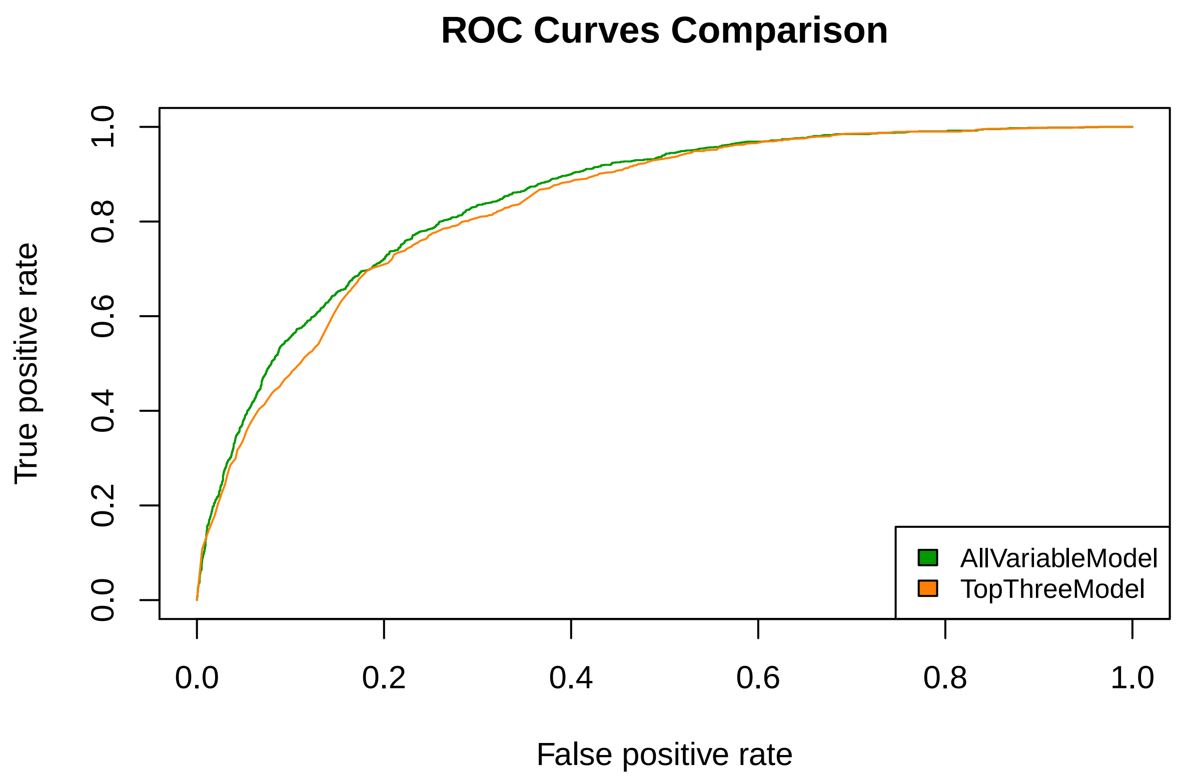
```r
suppressMessages(library(caret))
suppressMessages(library(ROCR))

# Get the performance of each model
allVariableModelPerformance <- getModelPerformance(allVariablesLogisticRegressionModel,
    dataset = train_df, outcomeColumn = train_df$Churn)
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type =
## ifelse(type == : prediction from a rank-deficient fit may be misleading
```

```r
topThreeModelPerformance <- getModelPerformance(topThreeLogisticRegressionModel,
    dataset = train_df, outcomeColumn = train_df$Churn)

# Plot the ROC curves
plotROCCurves(allVariableModelPerformance, topThreeModelPerformance,
    main = "ROC Curves Comparison", model1Name = "AllVariableModel",
    model2Name = "TopThreeModel")
```



```r
# Get the AUC
allVariableModelAUC <- getAUC(train_df$Churn, train_df, allVariablesLogisticRegressionModel,
    oneClass = "Yes", zeroClass = "No")
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type =
## ifelse(type == : prediction from a rank-deficient fit may be misleading
```

```r
topThreeModelAUC <- getAUC(train_df$Churn, train_df, topThreeLogisticRegressionModel,
    oneClass = "Yes", zeroClass = "No")
# Print the AUC
print(sprintf("The AUC for the allVariableModel is %.2f%%", allVariableModelAUC *
    100))
```

```
## [1] "The AUC for the allVariableModel is 73.03%"
```

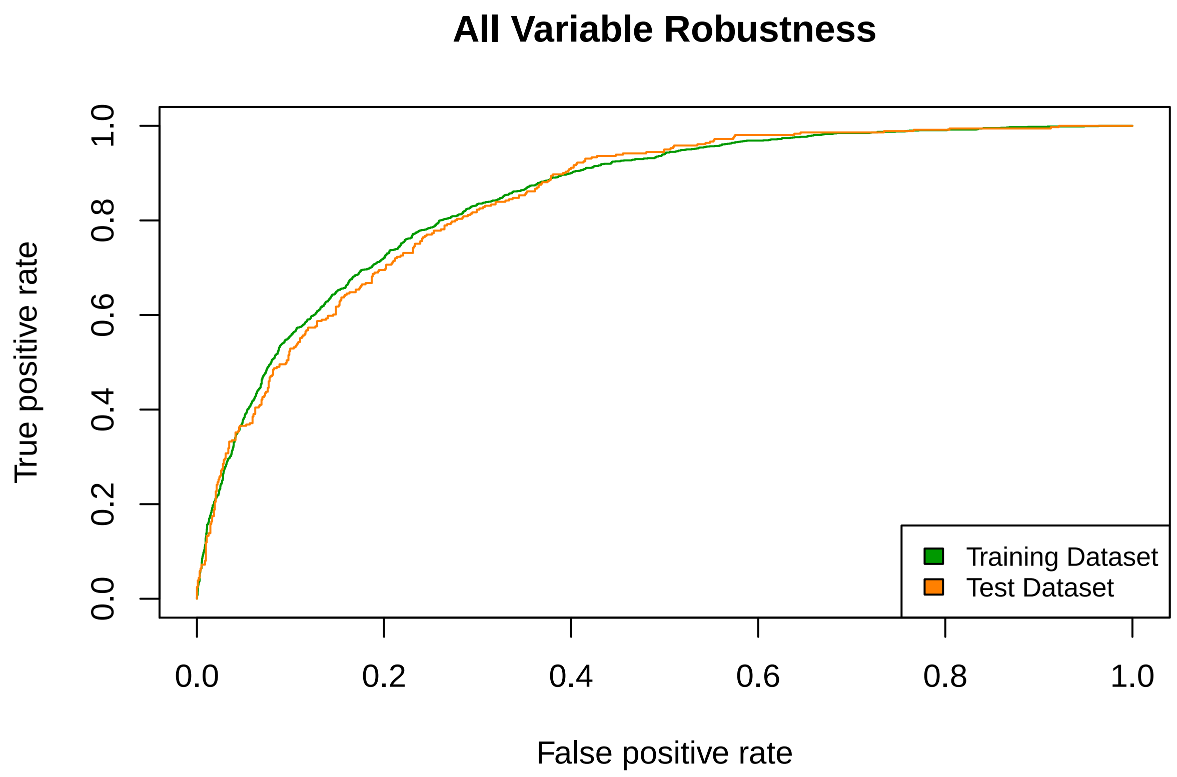
```r
print(sprintf("The AUC for the topThreeModel is %.2f%%", topThreeModelAUC *
    100))
```

```
## [1] "The AUC for the topThreeModel is 69.21%"
```

```r
# Check the allVariableModel for overfitting
allVariableModelPerformanceTest <- getModelPerformance(allVariablesLogisticRegressionModel,
    dataset = test_df, outcomeColumn = test_df$Churn)
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type =
## ifelse(type == : prediction from a rank-deficient fit may be misleading
```

```r
plotROCCurves(allVariableModelPerformance, allVariableModelPerformanceTest,
    main = "All Variable Robustness", model1Name = "Training Dataset",
    model2Name = "Test Dataset")
```



```
allVariableModelAUCTest <- getAUC(test_df$Churn, test_df, allVariablesLogisticRegressionModel,
    oneClass = "Yes", zeroClass = "No")
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type =
## ifelse(type == : prediction from a rank-deficient fit may be misleading
```

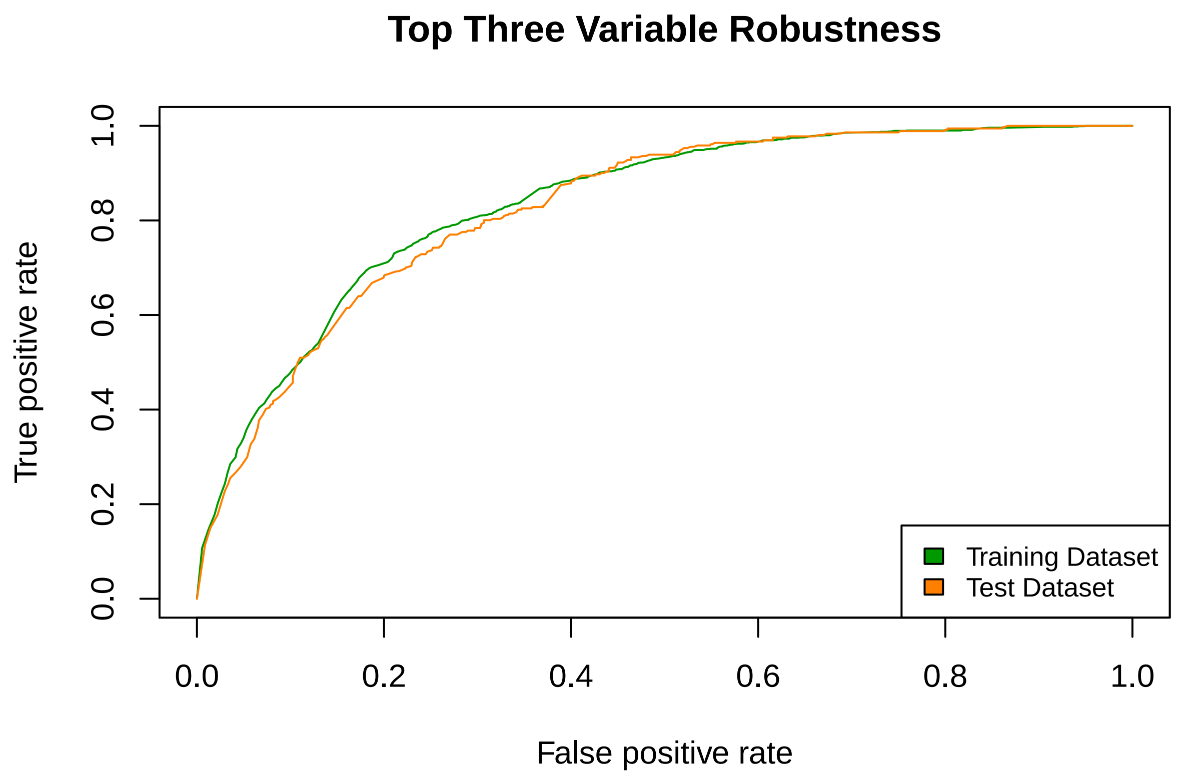
```
print(sprintf("The AUC for the training dataset using the allVariableModel is %.2f%%",
    allVariableModelAUC * 100))
```

```
## [1] "The AUC for the training dataset using the allVariableModel is 73.03%"
```

```
print(sprintf("The AUC for the test dataset using the allVariableModel is %.2f%%",
    allVariableModelAUCTest * 100))
```

```
## [1] "The AUC for the test dataset using the allVariableModel is 72.13%"
```

```
# Check the topThreeModel for overfitting
topThreeModelPerformanceTest <- getModelPerformance(topThreeLogisticRegressionModel,
    dataset = test_df, outcomeColumn = test_df$Churn)
plotROCCurves(topThreeModelPerformance, topThreeModelPerformanceTest,
    main = "Top Three Variable Robustness", model1Name = "Training Dataset",
    model2Name = "Test Dataset")
```



```r
topThreeModelAUCTest <- getAUC(test_df$Churn, test_df, topThreeLogisticRegressionModel,
    oneClass = "Yes", zeroClass = "No")
print(sprintf("The AUC for the training dataset using the topThreeModel is %.2f%%",
    topThreeModelAUC * 100))
```

```
## [1] "The AUC for the training dataset using the topThreeModel is 69.21%"
```

```r
print(sprintf("The AUC for the test dataset using the topThreeModel is %.2f%%",
    topThreeModelAUCTest * 100))
```

```
## [1] "The AUC for the test dataset using the topThreeModel is 69.97%"
```

```r
# Create confusion matricies to compare predicitons
print("All Variable Model Confusion Matrix")
```

```
## [1] "All Variable Model Confusion Matrix"
```

```r
createConfusionMatrix(allVariablesLogisticRegressionModel, test_df,
    test_df$Churn, oneClass = "Yes", zeroClass = "No")
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type =
## ifelse(type == : prediction from a rank-deficient fit may be misleading

##
## outcomeColumn  No Yes
##           No  924 119
##           Yes 160 201
```

```r
print("Top Three Variable Confusion Matrix")
```

```
## [1] "Top Three Variable Confusion Matrix"
```

```
print(createConfusionMatrix(topThreeLogisticRegressionModel,
    test_df, test_df$Churn, oneClass = "Yes", zeroClass = "No"))
```

```
##
## outcomeColumn  No Yes
##           No  928 115
##           Yes 177 184
```