Capstone Project - Humana Competition case

## R Markdownsqr

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document.

## 1. Introduction

Humana is a leading health care company that offers a wide range of insurance products and health and wellness services. Social Determinants of Health (SDoH) are a key component of Humana’s integrated value-based health ecosystem. 60% of what creates health has to do with the interplay between our socio-economic and community environments and lifestyle behaviors. Humana is seeking that “broader view“ of its members to better understand the whole person and to assist them in new ways towards achieving their best health.  
In the absence of regular, universal screening for SDoH, Humana needs to utilize robust data and advanced data science to understand which of our members are struggling with SDoH. This analysis will focus only on Transportation Challenges which is one of the major factors of SDoH. The Expectations: 1. Predictive model -Since screening all Medicare members is challenging, having an effective predictive model to accurately identify members most likely struggling with Transportation Challenges is valuable. Data is provided and can be supplemented with publicly available data. 2. Proposed solutions–It is likely that members struggling with Transportation Challenges are not homogeneous and hence there are perhaps different solutions for different segments of members.

## 2. Load the Dataset of Humana

rm(list = ls())  
HumanaComp<-read.csv("2020\_Competition\_Training.csv")  
#Loading the test data   
HumanaTest<-read.csv("2020\_Competition\_Holdout.csv")

## 3. Data Exploration and Visualization

##Understanding the Data Structure

#glimpse(HumanaComp)

* There are total 69572 observations and are of type number, integer and character.
* There are total 826 variables. The 2nd variable - “transportation\_issues” indicates the customers have (1) transportation issue or not(0).

### Missing Values:

# Show the % of missing variables in columns  
MissingColumns <- colMeans(is.na(HumanaComp))  
min(MissingColumns)\*100

## [1] 0

max(MissingColumns)\*100

## [1] 99.6651

# Show the list of columns with missing variables more than 40%  
colnames(HumanaComp)[colMeans(is.na(HumanaComp)) > 0.40]

## [1] "hedis\_ami" "hedis\_cmc\_ldc\_c\_control"  
## [3] "hedis\_cmc\_ldc\_c\_screen"

The amount of data missing from columns is from 0% to 99.67%.

# Show the % of missing variables in rows  
MissingRows <- rowMeans(is.na(HumanaComp))  
min(MissingRows)\*100

## [1] 0

max(MissingRows)\*100

## [1] 15.3753

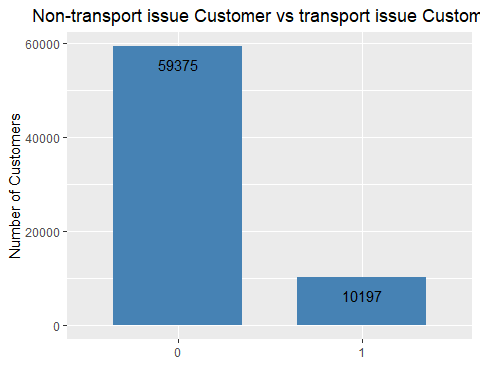
# Show the list of rows with missing variables more than 40%  
rownames(HumanaComp)[rowMeans(is.na(HumanaComp)) > 0.40]

## character(0)

the amount of data missing from rows range is 0% to 15.38%

### Visualize Members with vs without transportation issue

# Create a bar plot of number of customers that has transpiration issue and those do not   
  
ggplot(HumanaComp, aes(x=factor(transportation\_issues))) +  
 geom\_bar(stat="count", width=0.7, fill="steelblue") +  
 labs(title="Non-transport issue Customer vs transport issue Customers") +  
 labs(x="", y="Number of Customers") +  
 theme(plot.title = element\_text(hjust = 0.5)) +  
 geom\_text(stat='count', aes(label=..count..), vjust=2)

 Customers without transportation issue: 59375 Customers with transportation issue: 10197

##4. Data Pre-processing:

#Eliminate the duplicate columns  
Train\_Data1 <- HumanaComp[!duplicated(as.list(HumanaComp))] #733 variables  
  
# Eliminate near zero variance variables  
nzv\_cols\_r <- nearZeroVar(Train\_Data1)  
Train\_Data2 <- Train\_Data1[,-nzv\_cols\_r] #404 variables  
  
#Lets check the reduced dataframe   
dim(Train\_Data2)

## [1] 69572 404

#Dropping of the columns with more than 40% missing values  
library(VIM)

## Loading required package: colorspace

##   
## Attaching package: 'colorspace'

## The following object is masked from 'package:pROC':  
##   
## coords

## Loading required package: grid

## VIM is ready to use.

## Suggestions and bug-reports can be submitted at: https://github.com/statistikat/VIM/issues

##   
## Attaching package: 'VIM'

## The following object is masked from 'package:datasets':  
##   
## sleep

Train\_Data3<-Train\_Data2[,colMeans(is.na(Train\_Data2))<0.4] #401 variables  
  
#Lets check the reduced dataframe   
dim(Train\_Data3)

## [1] 69572 401

#Impute NA values  
  
#Character columns with NULL values  
Train\_char<- Train\_Data3[sapply(Train\_Data3,is.character)]  
#Imputing the missing values using KNN  
Train\_char <-kNN(Train\_char, k=4)  
  
anyNA(Train\_char)

## [1] FALSE

Train\_char<-Train\_char[-c(1,6,7,8,9,15)]  
  
library(dplyr)  
#all other columns other than character ones  
Train\_num<-Train\_Data3[!sapply(Train\_Data3,is.character)]  
  
# Impute the NA values using medianImpute  
impute\_model\_r <- preProcess(Train\_num, method = "medianImpute")  
Train\_num<- predict(impute\_model\_r, Train\_num)  
  
anyNA(Train\_num)

## [1] FALSE

# Eliminate the highly correlated variables  
corr <- cor(Train\_num)  
hc <- findCorrelation(corr, cutoff=0.50)   
hc <- sort(hc)  
Train\_num2 = Train\_num[,-c(hc)] #214 variables  
  
Train\_data4<-cbind(Train\_char,Train\_num2)   
dim(Train\_data4)

## [1] 69572 238

#Final reduced variables - 238   
# we will proceed with the 214 number variables and not with the 24 character variables

This process model reduced from 826 variable to 238 (24 - character & 214 Number)

Feature Engineering

### Lasso Regression:

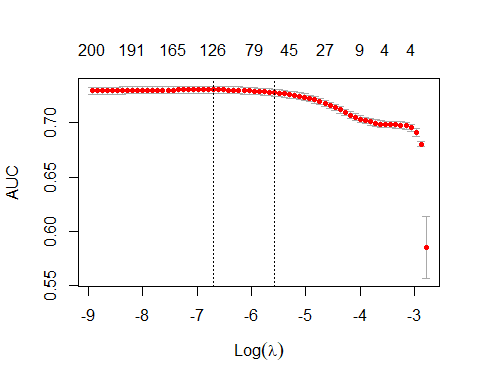
#Scaling the data  
#install.packages("normalr")  
library(normalr)

## Warning: package 'normalr' was built under R version 4.0.5

a<-subset(Train\_num2, select = -c(transportation\_issues))  
a<-normalise(a)  
  
X<- as.matrix(a)  
Y <- as.vector(as.factor(Train\_num2$transportation\_issues))  
LassoModel <- cv.glmnet(X, Y, alpha = 1, family = "binomial", nfolds = 10, type.measure = "auc")  
  
#Summary of Lasso Model  
summary(LassoModel)

## Length Class Mode   
## lambda 67 -none- numeric   
## cvm 67 -none- numeric   
## cvsd 67 -none- numeric   
## cvup 67 -none- numeric   
## cvlo 67 -none- numeric   
## nzero 67 -none- numeric   
## call 7 -none- call   
## name 1 -none- character  
## glmnet.fit 13 lognet list   
## lambda.min 1 -none- numeric   
## lambda.1se 1 -none- numeric   
## index 2 -none- numeric

#Plot AUC  
plot(LassoModel)



#Coefficient at the minimum lambda value  
LassoModel\_coefs <- coef(LassoModel, s = "lambda.min")  
  
#Turn the coefficient values into a data frame  
LassoModel\_coefs <- data.frame(name = LassoModel\_coefs@Dimnames[[1]][LassoModel\_coefs@i + 1], coefficient = LassoModel\_coefs@x)  
  
# Get the absolute value of all the coefficients  
LassoModel\_coefs$coefficient <- abs(LassoModel\_coefs$coefficient)  
  
# Orders the data frame by decreasing value of coefficients in the data frame  
LassoModel\_coefs <- LassoModel\_coefs[order(LassoModel\_coefs$coefficient, decreasing = TRUE), ]  
#LassoModel\_coefs  
  
# Remove the intercept from the data frame  
LassoModel\_coefs<-filter(LassoModel\_coefs, name!="(Intercept)")  
  
# Select the top 20 variables  
LassoModel\_coefs\_top\_20 <- LassoModel\_coefs[1:20, ]  
LassoModel\_coefs\_top\_20

## name coefficient  
## 1 credit\_num\_nonagn1stmorg\_collectio 92142475723  
## 2 credit\_num\_heloc\_severederog 52648765130  
## 3 credit\_num\_agency1stmorg\_collectio 34356118141  
## 4 credit\_num\_mtg\_90to119dpd 24505864109  
## 5 credit\_num\_nonagn1stmorg\_bankruptc 23523598208  
## 6 med\_er\_visit\_ct\_pmpm 1231562300  
## 7 rx\_gpi2\_17\_pmpm\_ct 1108501561  
## 8 betos\_o1a\_ind 789593263  
## 9 cms\_dual\_eligible\_ind 520731734  
## 10 rx\_mail\_ind 419231744  
## 11 submcc\_ben\_othr\_pmpm\_ct 349702393  
## 12 cms\_disabled\_ind 323407718  
## 13 credit\_hh\_mtg\_severederog 296531126  
## 14 cons\_retail\_buyer 290454551  
## 15 cmsd2\_skn\_radiation\_ind 283413445  
## 16 betos\_y2\_ind 275652414  
## 17 submcc\_sor\_ear\_pmpm\_ct 271820850  
## 18 bh\_cdto\_ind 236353206  
## 19 betos\_m5b\_ind 236250244  
## 20 betos\_t1a\_ind 235038458

# Select the remaining variables for PCA  
LassoModel\_coefs\_PCA\_variables <- LassoModel\_coefs[-c(1:20), ]  
  
# Turn names into a vector  
LassoModel\_coefs\_top\_20 <- as.vector(LassoModel\_coefs\_top\_20$name)  
LassoModel\_coefs\_PCA\_variables <- as.vector(LassoModel\_coefs\_PCA\_variables$name)  
  
# Add "transportation\_issue" to vector  
LassoModel\_coefs\_top\_20 <- c(LassoModel\_coefs\_top\_20,"transportation\_issues")  
LassoModel\_coefs\_top\_20

## [1] "credit\_num\_nonagn1stmorg\_collectio" "credit\_num\_heloc\_severederog"   
## [3] "credit\_num\_agency1stmorg\_collectio" "credit\_num\_mtg\_90to119dpd"   
## [5] "credit\_num\_nonagn1stmorg\_bankruptc" "med\_er\_visit\_ct\_pmpm"   
## [7] "rx\_gpi2\_17\_pmpm\_ct" "betos\_o1a\_ind"   
## [9] "cms\_dual\_eligible\_ind" "rx\_mail\_ind"   
## [11] "submcc\_ben\_othr\_pmpm\_ct" "cms\_disabled\_ind"   
## [13] "credit\_hh\_mtg\_severederog" "cons\_retail\_buyer"   
## [15] "cmsd2\_skn\_radiation\_ind" "betos\_y2\_ind"   
## [17] "submcc\_sor\_ear\_pmpm\_ct" "bh\_cdto\_ind"   
## [19] "betos\_m5b\_ind" "betos\_t1a\_ind"   
## [21] "transportation\_issues"

Lasso returned a total of around 143 variables as important to the default target variable.

### Principle Component Analysis:

set.seed(123)  
# Select other than the top 20 important variables from the lasso variable selection process  
#Append them with the character columns  
#PCAModel <- cbind(Train\_num2%>%dplyr::select(LassoModel\_coefs\_PCA\_variables), Train\_char)  
PCAModel<-Train\_num2%>%dplyr::select(LassoModel\_coefs\_PCA\_variables)

## Note: Using an external vector in selections is ambiguous.  
## i Use `all\_of(LassoModel\_coefs\_PCA\_variables)` instead of `LassoModel\_coefs\_PCA\_variables` to silence this message.  
## i See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.  
## This message is displayed once per session.

# Create a pre-processing model that eliminates near zero variance variables, highly correlated variables, and then does the imputation of missing values with the median and PCA.  
preProcessModel\_PCA <- preProcess(PCAModel, method = c("YeoJohnson", "center", "scale", "pca"), thresh = 0.75)  
PCAModel <- predict(preProcessModel\_PCA, PCAModel)  
preProcessModel\_PCA

## Created from 69572 samples and 106 variables  
##   
## Pre-processing:  
## - centered (106)  
## - ignored (0)  
## - principal component signal extraction (106)  
## - scaled (106)  
## - Yeo-Johnson transformation (20)  
##   
## Lambda estimates for Yeo-Johnson transformation:  
## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## -2.41331 -1.00192 0.28688 -0.02517 0.45248 2.85211   
##   
## PCA needed 63 components to capture 75 percent of the variance

69 Components needed to capture 75 percent of the variance leftover in the variables.

Splitting Training and Test Data:

# Merge the Lasso and PCA dataframe  
Train\_Data5 <- cbind.data.frame(Train\_num2%>%dplyr::select(LassoModel\_coefs\_top\_20), PCAModel)

## Note: Using an external vector in selections is ambiguous.  
## i Use `all\_of(LassoModel\_coefs\_top\_20)` instead of `LassoModel\_coefs\_top\_20` to silence this message.  
## i See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.  
## This message is displayed once per session.

# Create training and test set  
set.seed(123)  
index <- createDataPartition(Train\_Data5$transportation\_issues, p = 0.70, list = FALSE)  
train <- Train\_Data5[index, ]  
test <- Train\_Data5[-index, ]  
  
# Factorize Default variable.  
train$transportation\_issues <- as.factor(train$transportation\_issues)  
test$transportation\_issues <- as.factor(test$transportation\_issues)  
  
#glimpse(train)

##5. Modeling

### Random Forest Model:

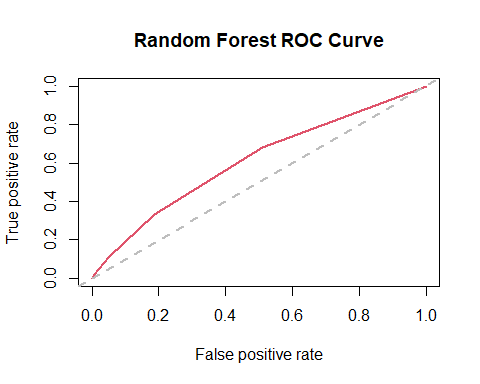
set.seed(123)  
RandomForestModel <- randomForest(transportation\_issues ~ ., data = train, ntree = 5, mtry = 5)  
print(RandomForestModel)

##   
## Call:  
## randomForest(formula = transportation\_issues ~ ., data = train, ntree = 5, mtry = 5)   
## Type of random forest: classification  
## Number of trees: 5  
## No. of variables tried at each split: 5  
##   
## OOB estimate of error rate: 21.76%  
## Confusion matrix:  
## 0 1 class.error  
## 0 33113 4289 0.1146730  
## 1 5249 1180 0.8164567

#Confusion Matrix  
RF <- data.frame(actual = test$transportation\_issue,predict(RandomForestModel, newdata = test, type = "prob"))  
RF$predict <- ifelse(RF$X0 > 0.50, 0, 1)  
CM <- confusionMatrix(as.factor(RF$predict), as.factor(RF$actual))  
CM

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 16975 2702  
## 1 870 324  
##   
## Accuracy : 0.8289   
## 95% CI : (0.8237, 0.8339)  
## No Information Rate : 0.855   
## P-Value [Acc > NIR] : 1   
##   
## Kappa : 0.0779   
##   
## Mcnemar's Test P-Value : <2e-16   
##   
## Sensitivity : 0.9512   
## Specificity : 0.1071   
## Pos Pred Value : 0.8627   
## Neg Pred Value : 0.2714   
## Prevalence : 0.8550   
## Detection Rate : 0.8133   
## Detection Prevalence : 0.9428   
## Balanced Accuracy : 0.5292   
##   
## 'Positive' Class : 0   
##

#Plot AUC  
set.seed(123)  
Predict <- prediction(RF$X1,test$transportation\_issue)  
auc <- performance(Predict, "auc")  
Predict1 <- performance(Predict, "tpr", "fpr")  
plot(Predict1, main = "Random Forest ROC Curve", col = 2, lwd = 2)  
abline(a=0, b=1, lwd=2, lty=2, col="gray")

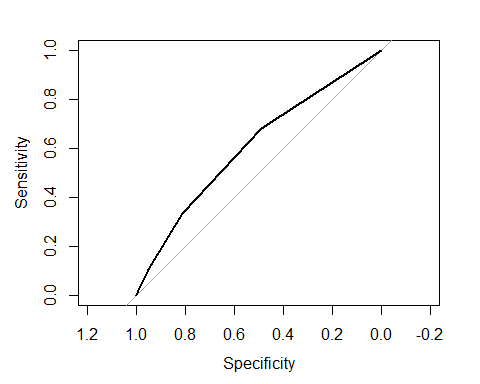


rf.roc <- roc(test$transportation\_issue,RF$X1)

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

## Setting direction: controls < cases

plot(rf.roc)



auc(rf.roc)

## Area under the curve: 0.609

Here we again split the cleaned data into train and test.

Split Data by 70 - 30,Train-Test ratio.

set.seed(123)  
split=sample.split(Train\_Data5$transportation\_issues,SplitRatio = 0.7)  
Train = subset(Train\_Data5,split==TRUE)  
Test = subset(Train\_Data5,split==FALSE)

### Elastic Net Method

x<- as.matrix(subset(Train, select = -c(transportation\_issues)))  
y<-as.vector(Train$transportation\_issues)  
ElasticNet <- cv.glmnet(x, y, type.measure = "auc", family = "binomial", alpha = 0.8,)  
summary(ElasticNet)

## Length Class Mode   
## lambda 61 -none- numeric   
## cvm 61 -none- numeric   
## cvsd 61 -none- numeric   
## cvup 61 -none- numeric   
## cvlo 61 -none- numeric   
## nzero 61 -none- numeric   
## call 6 -none- call   
## name 1 -none- character  
## glmnet.fit 13 lognet list   
## lambda.min 1 -none- numeric   
## lambda.1se 1 -none- numeric   
## index 2 -none- numeric

max(ElasticNet$cvm)

## [1] 0.7359303

predictions <- predict(ElasticNet, newx = as.matrix(Test[,-21]), s = "lambda.min", type = "response") %>% as.vector()  
pred\_class <- predict(ElasticNet, newx = as.matrix(Test[,-21]), s = "lambda.min", type = "class") %>%   
 as.vector() %>% factor(levels = c("0", "1"))  
  
Roc\_Result <- roc(Test$transportation\_issues, predictions)

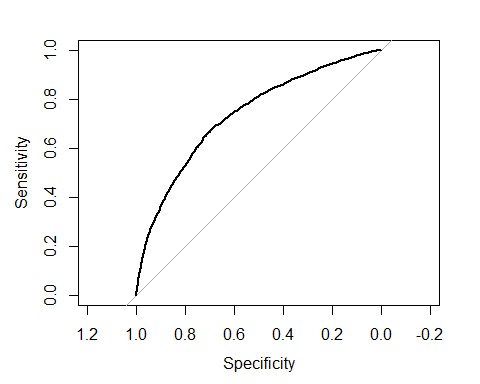
## Setting levels: control = 0, case = 1

## Setting direction: controls < cases

Roc\_Result

##   
## Call:  
## roc.default(response = Test$transportation\_issues, predictor = predictions)  
##   
## Data: predictions in 17813 controls (Test$transportation\_issues 0) < 3059 cases (Test$transportation\_issues 1).  
## Area under the curve: 0.7361

plot(Roc\_Result)



auc(Roc\_Result)

## Area under the curve: 0.7361

confusionMatrix(pred\_class, as.factor(Test$transportation\_issues))

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 17654 2852  
## 1 159 207  
##   
## Accuracy : 0.8557   
## 95% CI : (0.8509, 0.8605)  
## No Information Rate : 0.8534   
## P-Value [Acc > NIR] : 0.1764   
##   
## Kappa : 0.0924   
##   
## Mcnemar's Test P-Value : <2e-16   
##   
## Sensitivity : 0.99107   
## Specificity : 0.06767   
## Pos Pred Value : 0.86092   
## Neg Pred Value : 0.56557   
## Prevalence : 0.85344   
## Detection Rate : 0.84582   
## Detection Prevalence : 0.98246   
## Balanced Accuracy : 0.52937   
##   
## 'Positive' Class : 0   
##

### Logistic Regression

Train\_Data\_00 <- Train\_Data5[Train\_Data5$transportation\_issues == 0, ]  
Train\_Data\_11 <- Train\_Data5[Train\_Data5$transportation\_issues == 1, ]  
index\_1 <- sample(length(Train\_Data\_11$transportation\_issues == 1))  
index\_0 <- sample(length(Train\_Data\_00$transportation\_issues == 0))  
Train\_Data\_00 <- Train\_Data\_00[index\_0, ]  
Train\_Data\_11 <- Train\_Data\_11[index\_1, ]  
Data\_Full <- as.data.frame(rbind(Train\_Data\_00, Train\_Data\_11))

split <- createDataPartition(Data\_Full$transportation\_issues,p=0.70,list = FALSE)  
TrainM2 <- Data\_Full[split,]  
TestM2 <- Data\_Full[-split,]  
TrainM2$default <- TrainM2$default  
TestM2$default <- TestM2$default

CV\_MC <- trainControl(method = "cv",number = 5, summaryFunction = twoClassSummary,classProbs = TRUE,verboseIter = TRUE)  
CV\_M2 <- train(make.names(transportation\_issues)~.,data=TrainM2,method="glmnet",trControl=CV\_MC)

## Warning in train.default(x, y, weights = w, ...): The metric "Accuracy" was not  
## in the result set. ROC will be used instead.

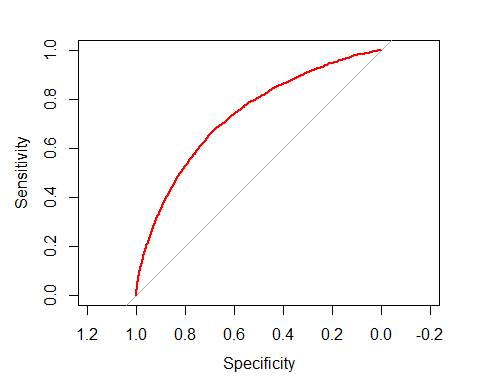
## + Fold1: alpha=0.10, lambda=0.01354   
## - Fold1: alpha=0.10, lambda=0.01354   
## + Fold1: alpha=0.55, lambda=0.01354   
## - Fold1: alpha=0.55, lambda=0.01354   
## + Fold1: alpha=1.00, lambda=0.01354   
## - Fold1: alpha=1.00, lambda=0.01354   
## + Fold2: alpha=0.10, lambda=0.01354   
## - Fold2: alpha=0.10, lambda=0.01354   
## + Fold2: alpha=0.55, lambda=0.01354   
## - Fold2: alpha=0.55, lambda=0.01354   
## + Fold2: alpha=1.00, lambda=0.01354   
## - Fold2: alpha=1.00, lambda=0.01354   
## + Fold3: alpha=0.10, lambda=0.01354   
## - Fold3: alpha=0.10, lambda=0.01354   
## + Fold3: alpha=0.55, lambda=0.01354   
## - Fold3: alpha=0.55, lambda=0.01354   
## + Fold3: alpha=1.00, lambda=0.01354   
## - Fold3: alpha=1.00, lambda=0.01354   
## + Fold4: alpha=0.10, lambda=0.01354   
## - Fold4: alpha=0.10, lambda=0.01354   
## + Fold4: alpha=0.55, lambda=0.01354   
## - Fold4: alpha=0.55, lambda=0.01354   
## + Fold4: alpha=1.00, lambda=0.01354   
## - Fold4: alpha=1.00, lambda=0.01354   
## + Fold5: alpha=0.10, lambda=0.01354   
## - Fold5: alpha=0.10, lambda=0.01354   
## + Fold5: alpha=0.55, lambda=0.01354   
## - Fold5: alpha=0.55, lambda=0.01354   
## + Fold5: alpha=1.00, lambda=0.01354   
## - Fold5: alpha=1.00, lambda=0.01354   
## Aggregating results  
## Selecting tuning parameters  
## Fitting alpha = 0.1, lambda = 0.0135 on full training set

Pred\_M2 <- predict(CV\_M2,TestM2[,-21])  
  
Pred\_M2ROC <- predict(CV\_M2,TestM2[,-21], type = "prob")  
Roc\_Result\_M2 <- roc(TestM2$transportation\_issues, Pred\_M2ROC[,2])

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

## Setting direction: controls < cases

plot(Roc\_Result\_M2,col = "red", lwd = 2)



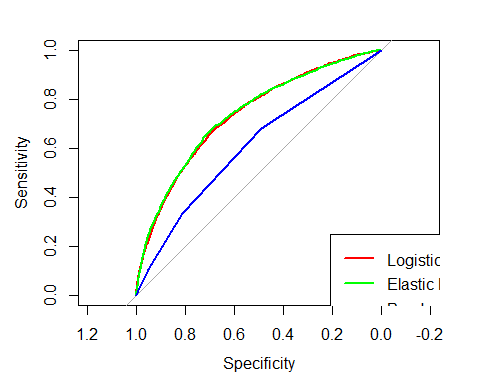
auc(Roc\_Result\_M2)

## Area under the curve: 0.7332

levels(TestM2$transportation\_issues) <- c("X0","X1")  
#confusionMatrix(Pred\_M2, TestM2$transportation\_issues)

### Comparing ROC of all the Models

plot(Roc\_Result\_M2,col="red")  
plot(Roc\_Result,col="green",add=TRUE)  
plot(rf.roc,col="blue",add=TRUE)  
legend(0.21,0.25, c('Logistic Regression','Elastic Net','Random Forest'),lty=c(1,1),  
lwd=c(2,2),col=c('red','green','blue'))



As we can see the logistic model gives the best results, lets apply that to full dataset.

CV\_M3 <- train(make.names(transportation\_issues)~.,data=Train\_num2,method="glmnet",trControl=CV\_MC)

## Warning in train.default(x, y, weights = w, ...): The metric "Accuracy" was not  
## in the result set. ROC will be used instead.

## + Fold1: alpha=0.10, lambda=0.01288   
## - Fold1: alpha=0.10, lambda=0.01288   
## + Fold1: alpha=0.55, lambda=0.01288   
## - Fold1: alpha=0.55, lambda=0.01288   
## + Fold1: alpha=1.00, lambda=0.01288   
## - Fold1: alpha=1.00, lambda=0.01288   
## + Fold2: alpha=0.10, lambda=0.01288   
## - Fold2: alpha=0.10, lambda=0.01288   
## + Fold2: alpha=0.55, lambda=0.01288   
## - Fold2: alpha=0.55, lambda=0.01288   
## + Fold2: alpha=1.00, lambda=0.01288   
## - Fold2: alpha=1.00, lambda=0.01288   
## + Fold3: alpha=0.10, lambda=0.01288   
## - Fold3: alpha=0.10, lambda=0.01288   
## + Fold3: alpha=0.55, lambda=0.01288   
## - Fold3: alpha=0.55, lambda=0.01288   
## + Fold3: alpha=1.00, lambda=0.01288   
## - Fold3: alpha=1.00, lambda=0.01288   
## + Fold4: alpha=0.10, lambda=0.01288   
## - Fold4: alpha=0.10, lambda=0.01288   
## + Fold4: alpha=0.55, lambda=0.01288   
## - Fold4: alpha=0.55, lambda=0.01288   
## + Fold4: alpha=1.00, lambda=0.01288   
## - Fold4: alpha=1.00, lambda=0.01288   
## + Fold5: alpha=0.10, lambda=0.01288   
## - Fold5: alpha=0.10, lambda=0.01288   
## + Fold5: alpha=0.55, lambda=0.01288   
## - Fold5: alpha=0.55, lambda=0.01288   
## + Fold5: alpha=1.00, lambda=0.01288   
## - Fold5: alpha=1.00, lambda=0.01288   
## Aggregating results  
## Selecting tuning parameters  
## Fitting alpha = 0.1, lambda = 0.0129 on full training set

summary(CV\_M3)

## Length Class Mode   
## a0 67 -none- numeric   
## beta 14271 dgCMatrix S4   
## df 67 -none- numeric   
## dim 2 -none- numeric   
## lambda 67 -none- numeric   
## dev.ratio 67 -none- numeric   
## nulldev 1 -none- numeric   
## npasses 1 -none- numeric   
## jerr 1 -none- numeric   
## offset 1 -none- logical   
## classnames 2 -none- character  
## call 5 -none- call   
## nobs 1 -none- numeric   
## lambdaOpt 1 -none- numeric   
## xNames 213 -none- character  
## problemType 1 -none- character  
## tuneValue 2 data.frame list   
## obsLevels 2 -none- character  
## param 0 -none- list

Lets apply logistic regression on test dataset

Test Data and Imputation of the NA values

#HumanaTest is the test data  
  
#all other columns other than character ones  
HumanaTest2<-HumanaTest[!sapply(HumanaTest,is.character)]  
  
# Impute the NA values using medianImpute  
impute\_model\_r <- preProcess(HumanaTest2, method = "medianImpute")  
HumanaTest2<- predict(impute\_model\_r, HumanaTest2)  
  
anyNA(HumanaTest2)

## [1] FALSE

Predicting Test data on full train data set

Pred\_M4 <- predict(CV\_M3,HumanaTest2)  
summary(Pred\_M4)

## X0 X1   
## 17305 376

There are 376 members have transporation issues