TelecomChurn.R

#summarize the target variable  
library(caret)

## Loading required package: lattice

## Loading required package: ggplot2

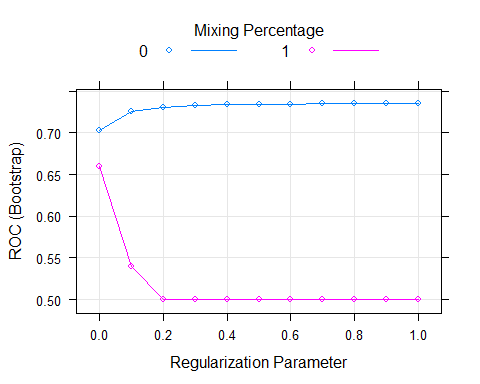
library(C50)  
data(churn)  
table(churnTrain$churn) / nrow(churnTrain)

##   
## yes no   
## 0.1449145 0.8550855

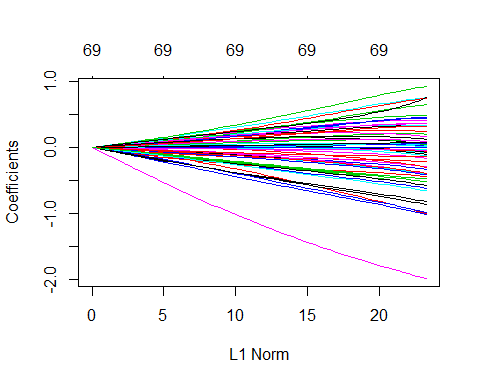
churn\_x <- churnTrain$churn  
churn\_y <- churnTest$churn  
  
########################################################  
#Make custom train/test indices  
########################################################  
  
#STEP1 first order of business is to create a reusable trainControl object#  
#you can use to reliably compare them.  
  
# Create custom indices: myFolds  
myFolds <- createFolds(churn\_y, k = 5)  
  
# Create reusable trainControl object: myControl  
myControl <- trainControl(  
 summaryFunction = twoClassSummary,  
 classProbs = TRUE, # IMPORTANT!  
 verboseIter = TRUE,  
 savePredictions = TRUE,  
 index = myFolds  
)  
########################################################  
#Fit the baseline model  
########################################################  
#glmnet, which penalizes linear and logistic regression models   
#on the size and number of coefficients to help prevent overfitting.  
# Fit glmnet model: model\_glmnet  
model\_glmnet <- train(  
 churn ~., data = churnTrain,  
 metric = "ROC",  
 method = "glmnet",  
 tuneGrid = expand.grid(  
 alpha = 0:1,  
 lambda = 0:10/10  
 ),  
 trControl = myControl  
)

## + Fold1: alpha=0, lambda=1   
## - Fold1: alpha=0, lambda=1   
## + Fold1: alpha=1, lambda=1   
## - Fold1: alpha=1, lambda=1   
## + Fold2: alpha=0, lambda=1   
## - Fold2: alpha=0, lambda=1   
## + Fold2: alpha=1, lambda=1   
## - Fold2: alpha=1, lambda=1   
## + Fold3: alpha=0, lambda=1   
## - Fold3: alpha=0, lambda=1   
## + Fold3: alpha=1, lambda=1   
## - Fold3: alpha=1, lambda=1   
## + Fold4: alpha=0, lambda=1   
## - Fold4: alpha=0, lambda=1   
## + Fold4: alpha=1, lambda=1   
## - Fold4: alpha=1, lambda=1   
## + Fold5: alpha=0, lambda=1   
## - Fold5: alpha=0, lambda=1   
## + Fold5: alpha=1, lambda=1   
## - Fold5: alpha=1, lambda=1   
## Aggregating results  
## Selecting tuning parameters  
## Fitting alpha = 0, lambda = 1 on full training set

#plot the results  
plot(model\_glmnet)



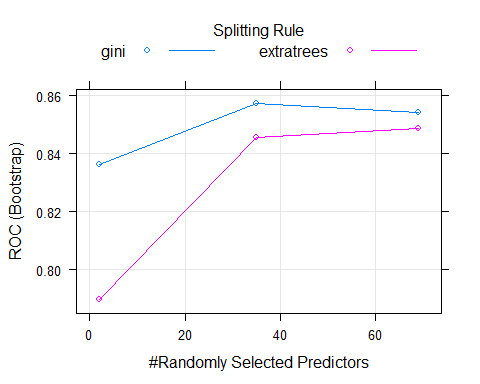
#plot the coefficients  
plot(model\_glmnet$finalModel)



########################################################  
#Random forest with custom trainControl  
########################################################  
#What RF's drawback ?  
#You no longer have model coefficients to help interpret the model.  
#RFcombines an ensemble of non-linear decision trees into a highly flexible  
library(ranger)  
#ranger package, which is a re-implementation of randomForest   
#that produces almost the exact same results, but is faster  
  
set.seed(42)  
churnTrain$churn <- factor(churnTrain$churn, levels = c("no", "yes"))  
model\_rf <- train(  
 churn ~., churnTrain,  
 metric = "ROC",  
 method = "ranger",  
 trControl = myControl  
)

## + Fold1: mtry= 2, min.node.size=1, splitrule=gini   
## - Fold1: mtry= 2, min.node.size=1, splitrule=gini   
## + Fold1: mtry=35, min.node.size=1, splitrule=gini   
## - Fold1: mtry=35, min.node.size=1, splitrule=gini   
## + Fold1: mtry=69, min.node.size=1, splitrule=gini   
## - Fold1: mtry=69, min.node.size=1, splitrule=gini   
## + Fold1: mtry= 2, min.node.size=1, splitrule=extratrees   
## - Fold1: mtry= 2, min.node.size=1, splitrule=extratrees   
## + Fold1: mtry=35, min.node.size=1, splitrule=extratrees   
## - Fold1: mtry=35, min.node.size=1, splitrule=extratrees   
## + Fold1: mtry=69, min.node.size=1, splitrule=extratrees   
## - Fold1: mtry=69, min.node.size=1, splitrule=extratrees   
## + Fold2: mtry= 2, min.node.size=1, splitrule=gini   
## - Fold2: mtry= 2, min.node.size=1, splitrule=gini   
## + Fold2: mtry=35, min.node.size=1, splitrule=gini   
## - Fold2: mtry=35, min.node.size=1, splitrule=gini   
## + Fold2: mtry=69, min.node.size=1, splitrule=gini   
## - Fold2: mtry=69, min.node.size=1, splitrule=gini   
## + Fold2: mtry= 2, min.node.size=1, splitrule=extratrees   
## - Fold2: mtry= 2, min.node.size=1, splitrule=extratrees   
## + Fold2: mtry=35, min.node.size=1, splitrule=extratrees   
## - Fold2: mtry=35, min.node.size=1, splitrule=extratrees   
## + Fold2: mtry=69, min.node.size=1, splitrule=extratrees   
## - Fold2: mtry=69, min.node.size=1, splitrule=extratrees   
## + Fold3: mtry= 2, min.node.size=1, splitrule=gini   
## - Fold3: mtry= 2, min.node.size=1, splitrule=gini   
## + Fold3: mtry=35, min.node.size=1, splitrule=gini   
## - Fold3: mtry=35, min.node.size=1, splitrule=gini   
## + Fold3: mtry=69, min.node.size=1, splitrule=gini   
## - Fold3: mtry=69, min.node.size=1, splitrule=gini   
## + Fold3: mtry= 2, min.node.size=1, splitrule=extratrees   
## - Fold3: mtry= 2, min.node.size=1, splitrule=extratrees   
## + Fold3: mtry=35, min.node.size=1, splitrule=extratrees   
## - Fold3: mtry=35, min.node.size=1, splitrule=extratrees   
## + Fold3: mtry=69, min.node.size=1, splitrule=extratrees   
## - Fold3: mtry=69, min.node.size=1, splitrule=extratrees   
## + Fold4: mtry= 2, min.node.size=1, splitrule=gini   
## - Fold4: mtry= 2, min.node.size=1, splitrule=gini   
## + Fold4: mtry=35, min.node.size=1, splitrule=gini   
## - Fold4: mtry=35, min.node.size=1, splitrule=gini   
## + Fold4: mtry=69, min.node.size=1, splitrule=gini   
## - Fold4: mtry=69, min.node.size=1, splitrule=gini   
## + Fold4: mtry= 2, min.node.size=1, splitrule=extratrees   
## - Fold4: mtry= 2, min.node.size=1, splitrule=extratrees   
## + Fold4: mtry=35, min.node.size=1, splitrule=extratrees   
## - Fold4: mtry=35, min.node.size=1, splitrule=extratrees   
## + Fold4: mtry=69, min.node.size=1, splitrule=extratrees   
## - Fold4: mtry=69, min.node.size=1, splitrule=extratrees   
## + Fold5: mtry= 2, min.node.size=1, splitrule=gini   
## - Fold5: mtry= 2, min.node.size=1, splitrule=gini   
## + Fold5: mtry=35, min.node.size=1, splitrule=gini   
## - Fold5: mtry=35, min.node.size=1, splitrule=gini   
## + Fold5: mtry=69, min.node.size=1, splitrule=gini   
## - Fold5: mtry=69, min.node.size=1, splitrule=gini   
## + Fold5: mtry= 2, min.node.size=1, splitrule=extratrees   
## - Fold5: mtry= 2, min.node.size=1, splitrule=extratrees   
## + Fold5: mtry=35, min.node.size=1, splitrule=extratrees   
## - Fold5: mtry=35, min.node.size=1, splitrule=extratrees   
## + Fold5: mtry=69, min.node.size=1, splitrule=extratrees   
## - Fold5: mtry=69, min.node.size=1, splitrule=extratrees   
## Aggregating results  
## Selecting tuning parameters  
## Fitting mtry = 35, splitrule = gini, min.node.size = 1 on full training set

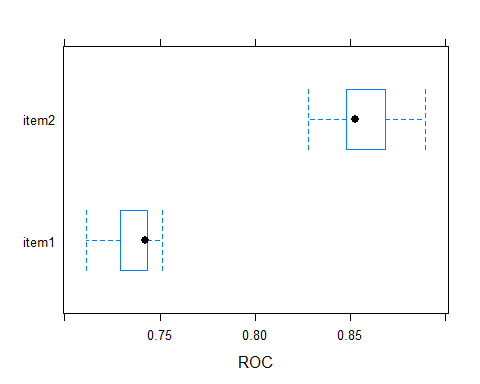
#plot RF  
plot(model\_rf)



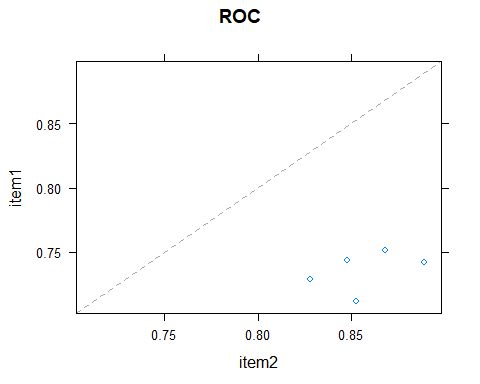
#What's the primary reason that train/test indices need to match when comparing two models?  
#Because otherwise you wouldn't be doing a fair comparison of your models and your results could be due to chance.  
  
########################################################  
#Create a resamples object  
########################################################  
#Now that you have fit two models to the churn dataset,  
#it's time to compare their out-of-sample predictions  
#and choose which one is the best model for your dataset.  
  
# Create model\_list  
model\_list <- list(item1 = model\_glmnet, item2 = model\_rf)  
# Pass model\_list to resamples(): resamples  
resamples <- resamples(model\_list)  
# Summarize the results  
summary(resamples)

##   
## Call:  
## summary.resamples(object = resamples)  
##   
## Models: item1, item2   
## Number of resamples: 5   
##   
## ROC   
## Min. 1st Qu. Median Mean 3rd Qu. Max. NA's  
## item1 0.7115070 0.7291777 0.7425147 0.7356026 0.7435125 0.7513009 0  
## item2 0.8281609 0.8480603 0.8530089 0.8574096 0.8684091 0.8894090 0  
##   
## Sens   
## Min. 1st Qu. Median Mean 3rd Qu. Max. NA's  
## item1 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0  
## item2 0.9620204 0.9761905 0.9855807 0.9805317 0.9874608 0.9914063 0  
##   
## Spec   
## Min. 1st Qu. Median Mean 3rd Qu. Max. NA's  
## item1 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 0  
## item2 0.2146119 0.2574032 0.3504464 0.3661792 0.4853933 0.5230415 0

########################################################  
#Create a box-and-whisker plot  
########################################################  
#box-and-whisker plot, which allows you to compare the distribution of predictive accuracy   
#(in this case AUC) for the two models.  
  
#you want the model with the higher median AUC, as well as a smaller range between min and max AUC.  
# Create bwplot  
bwplot(resamples, metric = "ROC")



#RF has an higher median AUC and a smaller ranger between min & max AUC  
########################################################  
#Create a scatter-plot  
########################################################  
#This plot shows you how similar the two models' performances are on different folds.  
#useful for identifying if one model is consistently better than the other across all folds  
#or if there are situations when the inferior model produces better predictions on a particular subset of the data  
# Create xyplot  
xyplot(resamples, metric = "ROC")



########################################################  
#Ensembling models  
########################################################  
library(caretEnsemble)

##   
## Attaching package: 'caretEnsemble'

## The following object is masked from 'package:ggplot2':  
##   
## autoplot

#caretEnsemble provides the caretList() function   
#for creating multiple caret models at once on the same dataset,  
#create a CaretList  
model\_list <- caretList(  
 churn ~. ,   
 data = churnTrain,  
 methodList = c("glm","rf"),  
 trControl = myControl)

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

## + Fold1: parameter=none   
## - Fold1: parameter=none   
## + Fold2: parameter=none

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## - Fold2: parameter=none   
## + Fold3: parameter=none

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## - Fold3: parameter=none   
## + Fold4: parameter=none

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## - Fold4: parameter=none   
## + Fold5: parameter=none   
## - Fold5: parameter=none   
## Aggregating results  
## Fitting final model on full training set

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

## + Fold1: mtry= 2   
## - Fold1: mtry= 2   
## + Fold1: mtry=35   
## - Fold1: mtry=35   
## + Fold1: mtry=69   
## - Fold1: mtry=69   
## + Fold2: mtry= 2   
## - Fold2: mtry= 2   
## + Fold2: mtry=35   
## - Fold2: mtry=35   
## + Fold2: mtry=69   
## - Fold2: mtry=69   
## + Fold3: mtry= 2   
## - Fold3: mtry= 2   
## + Fold3: mtry=35   
## - Fold3: mtry=35   
## + Fold3: mtry=69   
## - Fold3: mtry=69   
## + Fold4: mtry= 2   
## - Fold4: mtry= 2   
## + Fold4: mtry=35   
## - Fold4: mtry=35   
## + Fold4: mtry=69   
## - Fold4: mtry=69   
## + Fold5: mtry= 2   
## - Fold5: mtry= 2   
## + Fold5: mtry=35   
## - Fold5: mtry=35   
## + Fold5: mtry=69   
## - Fold5: mtry=69   
## Aggregating results  
## Selecting tuning parameters  
## Fitting mtry = 35 on full training set

# Create ensemble model: stack  
stack <- caretStack(model\_list, method = "glm")  
  
# Look at summary  
summary(stack)

##   
## Call:  
## NULL  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.5321 -0.3757 -0.2988 -0.2776 2.5650   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -3.25157 0.04501 -72.238 < 2e-16 \*\*\*  
## glm 0.30973 0.09248 3.349 0.000811 \*\*\*  
## rf 7.20106 0.14391 50.038 < 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: 12519.9 on 14997 degrees of freedom  
## Residual deviance: 8449.9 on 14995 degrees of freedom  
## AIC: 8455.9  
##   
## Number of Fisher Scoring iterations: 5