# Module 5

## BAN 502

### Stephen Kiser

library(tidyverse)

## -- Attaching packages ------------------------------- tidyverse 1.3.0 --

## v ggplot2 3.3.0 v purrr 0.3.3  
## v tibble 3.0.1 v dplyr 0.8.5  
## v tidyr 1.0.2 v stringr 1.4.0  
## v readr 1.3.1 v forcats 0.5.0

## -- Conflicts ---------------------------------- tidyverse\_conflicts() --  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

library(caret)

## Loading required package: lattice

##   
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':  
##   
## lift

library(nnet)  
library(rpart)  
library(ranger)  
library(caretEnsemble)

##   
## Attaching package: 'caretEnsemble'

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

library(xgboost)

##   
## Attaching package: 'xgboost'

## The following object is masked from 'package:dplyr':  
##   
## slice

fin <- read\_csv("2018Fin.csv")

## Warning: Missing column names filled in: 'X1' [1]

## Parsed with column specification:  
## cols(  
## .default = col\_double(),  
## X1 = col\_character(),  
## Sector = col\_character()  
## )

## See spec(...) for full column specifications.

#str(fin)  
#summary(fin)

fin <- fin %>% select(c(Class, `Revenue Growth`, `EPS Diluted`, `EBITDA Margin`, priceBookValueRatio, debtEquityRatio, debtRatio, `PE ratio`, Sector, `Revenue Growth`, returnOnAssets, returnOnEquity, returnOnCapitalEmployed, quickRatio))  
  
  
fin <- fin %>% mutate(Class = as.factor(Class)) %>% mutate(Class = fct\_recode(Class,   
 "No" = "0",  
 "Yes" = "1"  
 ))  
fin <- fin %>% mutate(Sector = as.factor(Sector))  
  
fin <- fin %>% drop\_na()  
  
str(fin)

## tibble [2,655 x 13] (S3: tbl\_df/tbl/data.frame)  
## $ Class : Factor w/ 2 levels "No","Yes": 2 2 2 2 2 2 2 2 2 2 ...  
## $ Revenue Growth : num [1:2655] 0.112 0.032 0.129 0.495 0.159 ...  
## $ EPS Diluted : num [1:2655] 2.53 0.66 4.48 11.51 11.91 ...  
## $ EBITDA Margin : num [1:2655] 0.31 0.453 0.456 0.638 0.316 0.423 0.359 0.079 0.319 0.086 ...  
## $ priceBookValueRatio : num [1:2655] 2.16 1.03 2.86 1.89 10.18 ...  
## $ debtEquityRatio : num [1:2655] 1.56 1.108 0.353 0.144 1.068 ...  
## $ debtRatio : num [1:2655] 0.444 0.473 0.206 0.107 0.313 ...  
## $ PE ratio : num [1:2655] 13.3 23.3 10.3 4.3 18.9 ...  
## $ Sector : Factor w/ 11 levels "Basic Materials",..: 3 5 10 10 10 10 3 10 3 3 ...  
## $ returnOnAssets : num [1:2655] 0.3033 0.0446 0.3444 0.4163 1.0497 ...  
## $ returnOnEquity : num [1:2655] 0.1638 0.0478 0.2824 0.4377 0.5556 ...  
## $ returnOnCapitalEmployed: num [1:2655] 0.0531 0.0339 0.1444 0.3052 0.217 ...  
## $ quickRatio : num [1:2655] 0.54 0.632 1.105 2.134 0.995 ...

summary(fin)

## Class Revenue Growth EPS Diluted EBITDA Margin   
## No : 942 Min. : -1.000 Min. :-359825.0 Min. :-8809.838   
## Yes:1713 1st Qu.: 0.006 1st Qu.: -0.6 1st Qu.: -0.014   
## Median : 0.083 Median : 0.6 Median : 0.100   
## Mean : 5.313 Mean : -153.4 Mean : -12.820   
## 3rd Qu.: 0.204 3rd Qu.: 2.4 3rd Qu.: 0.205   
## Max. :12739.000 Max. : 83.3 Max. : 1060.404   
##   
## priceBookValueRatio debtEquityRatio debtRatio PE ratio   
## Min. : 0.00 Min. :-251.0270 Min. : 0.00000 Min. : 0.00   
## 1st Qu.: 1.08 1st Qu.: 0.0018 1st Qu.: 0.04685 1st Qu.: 0.00   
## Median : 2.06 Median : 0.4204 Median : 0.23510 Median : 10.62   
## Mean : 40.46 Mean : 0.5979 Mean : 0.27977 Mean : 24.45   
## 3rd Qu.: 4.00 3rd Qu.: 1.0385 3rd Qu.: 0.39575 3rd Qu.: 22.17   
## Max. :79083.10 Max. : 637.2299 Max. :24.35520 Max. :3842.00   
##   
## Sector returnOnAssets returnOnEquity   
## Healthcare :535 Min. :-193.67920 Min. :-34772.46   
## Technology :529 1st Qu.: -0.15660 1st Qu.: -0.15   
## Industrials :434 Median : 0.05850 Median : 0.07   
## Consumer Cyclical:416 Mean : -0.04213 Mean : -13.42   
## Energy :199 3rd Qu.: 0.19590 3rd Qu.: 0.18   
## Basic Materials :180 Max. : 143.27930 Max. : 44.12   
## (Other) :362   
## returnOnCapitalEmployed quickRatio   
## Min. :-108.2549 Min. : 0.00099   
## 1st Qu.: -0.0178 1st Qu.: 0.77087   
## Median : 0.0402 Median : 1.25872   
## Mean : -0.4895 Mean : 2.49180   
## 3rd Qu.: 0.0937 3rd Qu.: 2.45151   
## Max. : 10.9396 Max. :68.22496   
##

fin = fin %>% filter(`Revenue Growth` <= 1)   
fin = fin %>% filter(`EPS Diluted` >= -10, `EPS Diluted` <= 10)   
fin = fin %>% filter(`EBITDA Margin` >= -5, `EBITDA Margin` <= 5)  
fin = fin %>% filter(priceBookValueRatio >= 0, priceBookValueRatio <= 5)   
fin = fin %>% filter(debtEquityRatio >= -1, debtEquityRatio <= 2)  
fin = fin %>% filter(debtRatio <= 1)   
fin = fin %>% filter(`PE ratio` <= 100)   
fin = fin %>% filter(returnOnAssets >= -5, returnOnAssets <= 5)  
fin = fin %>% filter(returnOnEquity >= -5, returnOnEquity <= 5)  
fin = fin %>% filter(returnOnCapitalEmployed >= -2, returnOnCapitalEmployed <= 2)   
fin = fin %>% filter(quickRatio <= 20)

### Task 1

set.seed(12345)  
train.rows <- createDataPartition(y=fin$Class, p=0.7, list = FALSE)  
  
train <- dplyr::slice(fin, train.rows)  
test <- dplyr::slice(fin, -train.rows)

### Task 2

fitControl <- trainControl(method = "cv", number = 10)  
  
  
  
nnetGrid = expand.grid(size = 1:12,  
 decay = c(0.5, 0.1, 1e-2, 1e-3, 1e-4, 1e-5, 1e-6, 1e-7))  
  
set.seed(1234)  
nnetBasic <- train(x=as.data.frame(fin[-1]), y = fin$Class,  
 method = "nnet",  
 trControl = fitControl,  
 tuneGrid = nnetGrid,  
 trace = FALSE  
 )

nnetBasic

## Neural Network   
##   
## 1486 samples  
## 12 predictor  
## 2 classes: 'No', 'Yes'   
##   
## No pre-processing  
## Resampling: Cross-Validated (10 fold)   
## Summary of sample sizes: 1337, 1337, 1337, 1337, 1337, 1338, ...   
## Resampling results across tuning parameters:  
##   
## size decay Accuracy Kappa   
## 1 1e-07 0.6574596 0.20256344  
## 1 1e-06 0.6756575 0.22772721  
## 1 1e-05 0.6642073 0.19742368  
## 1 1e-04 0.6614683 0.23258163  
## 1 1e-03 0.6749184 0.23341365  
## 1 1e-02 0.6735806 0.21928334  
## 1 1e-01 0.6681888 0.21111098  
## 1 5e-01 0.6735806 0.21259938  
## 2 1e-07 0.6641711 0.14235668  
## 2 1e-06 0.6507845 0.09951177  
## 2 1e-05 0.6574687 0.16293390  
## 2 1e-04 0.6769454 0.18005129  
## 2 1e-03 0.6648649 0.18512555  
## 2 1e-02 0.6803963 0.21208227  
## 2 1e-01 0.6830038 0.24028916  
## 2 5e-01 0.6695629 0.20071722  
## 3 1e-07 0.6473699 0.12946398  
## 3 1e-06 0.6581172 0.17659492  
## 3 1e-05 0.6434020 0.14533158  
## 3 1e-04 0.6527889 0.15836216  
## 3 1e-03 0.6682070 0.19885474  
## 3 1e-02 0.6703202 0.21014127  
## 3 1e-01 0.6689371 0.20346504  
## 3 5e-01 0.6608244 0.17483720  
## 4 1e-07 0.6427081 0.14134086  
## 4 1e-06 0.6466624 0.16381183  
## 4 1e-05 0.6440051 0.15690219  
## 4 1e-04 0.6527934 0.17559597  
## 4 1e-03 0.6426628 0.14030208  
## 4 1e-02 0.6689098 0.19303477  
## 4 1e-01 0.6810493 0.24324211  
## 4 5e-01 0.6776438 0.21972864  
## 5 1e-07 0.6500726 0.14595851  
## 5 1e-06 0.6433385 0.16068858  
## 5 1e-05 0.6461001 0.15122630  
## 5 1e-04 0.6581580 0.19593328  
## 5 1e-03 0.6446717 0.14627487  
## 5 1e-02 0.6500635 0.18301192  
## 5 1e-01 0.6642119 0.19719958  
## 5 5e-01 0.6695719 0.20246857  
## 6 1e-07 0.6649011 0.18492900  
## 6 1e-06 0.6655496 0.19748663  
## 6 1e-05 0.6588019 0.19038325  
## 6 1e-04 0.6615364 0.20138621  
## 6 1e-03 0.6588246 0.18466992  
## 6 1e-02 0.6494105 0.17125199  
## 6 1e-01 0.6594957 0.19462561  
## 6 5e-01 0.6688962 0.20490013  
## 7 1e-07 0.6527118 0.18090039  
## 7 1e-06 0.6399646 0.16468924  
## 7 1e-05 0.6452975 0.19130749  
## 7 1e-04 0.6433022 0.15013543  
## 7 1e-03 0.6561536 0.18542670  
## 7 1e-02 0.6588427 0.19371613  
## 7 1e-01 0.6453519 0.16649087  
## 7 5e-01 0.6749773 0.21137120  
## 8 1e-07 0.6494241 0.16644299  
## 8 1e-06 0.6588337 0.17971558  
## 8 1e-05 0.6540722 0.19571175  
## 8 1e-04 0.6393434 0.15229513  
## 8 1e-03 0.6695402 0.21321830  
## 8 1e-02 0.6641574 0.21252044  
## 8 1e-01 0.6682614 0.22182084  
## 8 5e-01 0.6749410 0.21307293  
## 9 1e-07 0.6494286 0.19020922  
## 9 1e-06 0.6527662 0.18965604  
## 9 1e-05 0.6494649 0.17721026  
## 9 1e-04 0.6379966 0.16312102  
## 9 1e-03 0.6574914 0.20928361  
## 9 1e-02 0.6486668 0.17892745  
## 9 1e-01 0.6722293 0.22984042  
## 9 5e-01 0.6689280 0.20153808  
## 10 1e-07 0.6561174 0.19586911  
## 10 1e-06 0.6426673 0.17964841  
## 10 1e-05 0.6379739 0.16317732  
## 10 1e-04 0.6393343 0.16744075  
## 10 1e-03 0.6460321 0.17925945  
## 10 1e-02 0.6446808 0.17354505  
## 10 1e-01 0.6601941 0.20235333  
## 10 5e-01 0.6735988 0.21601590  
## 11 1e-07 0.6420234 0.16097972  
## 11 1e-06 0.6621984 0.21969060  
## 11 1e-05 0.6581535 0.19253464  
## 11 1e-04 0.6426673 0.17668466  
## 11 1e-03 0.6460366 0.18701495  
## 11 1e-02 0.6540541 0.19279713  
## 11 1e-01 0.6588246 0.20580962  
## 11 5e-01 0.6763423 0.21826675  
## 12 1e-07 0.6318747 0.14632495  
## 12 1e-06 0.6567930 0.20163239  
## 12 1e-05 0.6702521 0.24404420  
## 12 1e-04 0.6460457 0.18944229  
## 12 1e-03 0.6386133 0.17345047  
## 12 1e-02 0.6581671 0.20386991  
## 12 1e-01 0.6521631 0.18897569  
## 12 5e-01 0.6594368 0.18376113  
##   
## Accuracy was used to select the optimal model using the largest value.  
## The final values used for the model were size = 2 and decay = 0.1.

### Task 3

predNet <- predict(nnetBasic, train)  
  
confusionMatrix(predNet, train$Class, positive = "Yes")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction No Yes  
## No 128 93  
## Yes 233 587  
##   
## Accuracy : 0.6868   
## 95% CI : (0.6577, 0.7149)  
## No Information Rate : 0.6532   
## P-Value [Acc > NIR] : 0.01187   
##   
## Kappa : 0.2396   
##   
## Mcnemar's Test P-Value : 1.377e-14   
##   
## Sensitivity : 0.8632   
## Specificity : 0.3546   
## Pos Pred Value : 0.7159   
## Neg Pred Value : 0.5792   
## Prevalence : 0.6532   
## Detection Rate : 0.5639   
## Detection Prevalence : 0.7877   
## Balanced Accuracy : 0.6089   
##   
## 'Positive' Class : Yes   
##

The model’s accuracy is 68.68% which is not bad, but it has room for improvement. The reason for this accuracy is because of all the data we removed. The specificity is 0.3546 which is means there are a lot of false positive results. The sensitivity is 0.8632 which is a better number in our prediction than 0.3546.

### Task 4

predNet.test <- predict(nnetBasic, test)  
  
confusionMatrix(predNet.test, test$Class, positive = "Yes")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction No Yes  
## No 62 28  
## Yes 92 263  
##   
## Accuracy : 0.7303   
## 95% CI : (0.6865, 0.771)  
## No Information Rate : 0.6539   
## P-Value [Acc > NIR] : 0.0003372   
##   
## Kappa : 0.3396   
##   
## Mcnemar's Test P-Value : 8.867e-09   
##   
## Sensitivity : 0.9038   
## Specificity : 0.4026   
## Pos Pred Value : 0.7408   
## Neg Pred Value : 0.6889   
## Prevalence : 0.6539   
## Detection Rate : 0.5910   
## Detection Prevalence : 0.7978   
## Balanced Accuracy : 0.6532   
##   
## 'Positive' Class : Yes   
##

The testing set has a better accuracy than our training set. This is because there is less data which means outliers can have a bigger impact to our numbers. Sensitivity is 0.9038 and specificity is 0.4026. Even though specificity increased it means our model is will have more errors for negative results.

### Task 5

control <- trainControl(method = "cv", number = 5,   
 savePredictions = "final",  
 classProbs = TRUE,   
 summaryFunction = twoClassSummary,   
 index = createResample(train$Class))  
  
set.seed(111)  
model\_list <- caretList(x=as.data.frame(train[,-1]), y = train$Class,  
 metric = "ROC",  
 trControl = control,  
 methodList = c("glm", "rpart"),  
 tuneList=list(   
 ranger = caretModelSpec(method="ranger", max.depth = 5, tuneGrid = expand.grid(mtry = 1:12,   
 splitrule = c("gini","extratrees","hellinger"),   
 min.node.size=1:5)),   
 nn = caretModelSpec(method="nnet",   
 tuneGrid =   
 expand.grid(size = 1:23,   
 decay = c(0.5, 0.1, 1e-2, 1e-3, 1e-4, 1e-5, 1e-6, 1e-7)),trace=FALSE)))

### Task 6

modelCor(resamples(model\_list))

## ranger nn glm rpart  
## ranger 1.0000000 0.7614202 0.7168607 0.2230454  
## nn 0.7614202 1.0000000 0.9423411 -0.1122145  
## glm 0.7168607 0.9423411 1.0000000 -0.1492275  
## rpart 0.2230454 -0.1122145 -0.1492275 1.0000000

The correlation in our models in the ensemble are best glm and nn with 0.9423411. There is no good correlation with rpart in this ensemble.

### Task 7

ensemble <- caretEnsemble(  
 model\_list,  
 metric = "ROC",  
 trControl = control  
)  
  
  
pred\_ensemble = predict(ensemble, train, type = "raw")  
confusionMatrix(pred\_ensemble, train$Class)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction No Yes  
## No 121 68  
## Yes 240 612  
##   
## Accuracy : 0.7041   
## 95% CI : (0.6754, 0.7317)  
## No Information Rate : 0.6532   
## P-Value [Acc > NIR] : 0.0002714   
##   
## Kappa : 0.2648   
##   
## Mcnemar's Test P-Value : < 2.2e-16   
##   
## Sensitivity : 0.3352   
## Specificity : 0.9000   
## Pos Pred Value : 0.6402   
## Neg Pred Value : 0.7183   
## Prevalence : 0.3468   
## Detection Rate : 0.1162   
## Detection Prevalence : 0.1816   
## Balanced Accuracy : 0.6176   
##   
## 'Positive' Class : No   
##

pred\_ensemble\_test = predict(ensemble, test, type = "raw")  
confusionMatrix(pred\_ensemble\_test, test$Class)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction No Yes  
## No 53 30  
## Yes 101 261  
##   
## Accuracy : 0.7056   
## 95% CI : (0.6609, 0.7476)  
## No Information Rate : 0.6539   
## P-Value [Acc > NIR] : 0.01175   
##   
## Kappa : 0.2704   
##   
## Mcnemar's Test P-Value : 9.6e-10   
##   
## Sensitivity : 0.3442   
## Specificity : 0.8969   
## Pos Pred Value : 0.6386   
## Neg Pred Value : 0.7210   
## Prevalence : 0.3461   
## Detection Rate : 0.1191   
## Detection Prevalence : 0.1865   
## Balanced Accuracy : 0.6205   
##   
## 'Positive' Class : No   
##

The model has a 70.41% accuracy on the training set and a 70.56% on the testing set. The sensitivity of the training set is 0.3352 and in the testing set it is 0.3442. The specificity of the training set is 0.900 and in the testing set it is 0.8969. We got an increase in the accuracy and specificity, but a decrease in sensitivity for both sets. This means that both sets will have more errors in the true positives.

### Task 8

control2 <- trainControl(method = "cv", number = 10,   
 savePredictions = "final",  
 classProbs = TRUE,   
 summaryFunction = twoClassSummary,   
 index = createResample(train$Class))  
  
stack = caretStack(  
 model\_list,  
 method = "glm",  
 metric = "ROC",  
 trControl = control2  
 )  
  
print(stack)

## A glm ensemble of 4 base models: ranger, nn, glm, rpart  
##   
## Ensemble results:  
## Generalized Linear Model   
##   
## 3858 samples  
## 4 predictor  
## 2 classes: 'No', 'Yes'   
##   
## No pre-processing  
## Resampling: Cross-Validated (10 fold)   
## Summary of sample sizes: 1041, 1041, 1041, 1041, 1041, 1041, ...   
## Resampling results:  
##   
## ROC Sens Spec   
## 0.6809224 0.1699754 0.9179756

summary(stack)

##   
## Call:  
## NULL  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.0238 -1.1529 0.6208 0.8983 1.7675   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 2.88766 0.15734 18.353 < 2e-16 \*\*\*  
## ranger -4.02570 0.71132 -5.660 1.52e-08 \*\*\*  
## nn -3.05087 0.80343 -3.797 0.000146 \*\*\*  
## glm 0.90710 0.66221 1.370 0.170745   
## rpart -0.05245 0.19366 -0.271 0.786507   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 4964.9 on 3857 degrees of freedom  
## Residual deviance: 4538.2 on 3853 degrees of freedom  
## AIC: 4548.2  
##   
## Number of Fisher Scoring iterations: 4

### Task 9

train\_dummy = dummyVars(" ~ .", data = train)  
  
train\_xgb = data.frame(predict(train\_dummy, newdata = train))  
  
  
test\_dummy = dummyVars(" ~ .", data = test)  
test\_xgb = data.frame(predict(test\_dummy, newdata = test))  
  
  
train\_xgb <- train\_xgb %>% dplyr::select(-Class.No)  
  
test\_xgb <- test\_xgb %>% dplyr::select(-Class.No)  
  
str(train\_xgb)

## 'data.frame': 1041 obs. of 23 variables:  
## $ Class.Yes : num 1 1 1 1 1 1 1 0 1 1 ...  
## $ X.Revenue.Growth. : num 0.1115 0.1289 0.4309 0.3735 0.0636 ...  
## $ X.EPS.Diluted. : num 2.53 4.48 -0.97 7.57 2.85 0.85 3.67 1.87 1.56 3.23 ...  
## $ X.EBITDA.Margin. : num 0.31 0.456 -0.981 0.531 0.355 0.438 0.248 0.34 0.323 0.312 ...  
## $ priceBookValueRatio : num 2.16 2.86 3.27 4.48 1.13 ...  
## $ debtEquityRatio : num 1.56 0.353 0 0 0.959 ...  
## $ debtRatio : num 0.444 0.206 0 0 0.332 ...  
## $ X.PE.ratio. : num 13.3 10.3 0 17.1 10 ...  
## $ Sector.Basic.Materials : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ Sector.Communication.Services: num 0 0 0 0 1 0 0 0 0 0 ...  
## $ Sector.Consumer.Cyclical : num 1 0 0 0 0 0 0 0 0 0 ...  
## $ Sector.Consumer.Defensive : num 0 0 0 0 0 0 1 0 0 0 ...  
## $ Sector.Energy : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ Sector.Financial.Services : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ Sector.Healthcare : num 0 0 0 0 0 0 0 1 0 0 ...  
## $ Sector.Industrials : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ Sector.Real.Estate : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ Sector.Technology : num 0 1 1 1 0 1 0 0 1 1 ...  
## $ Sector.Utilities : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ returnOnAssets : num 0.303 0.344 -0.787 0.325 0.143 ...  
## $ returnOnEquity : num 0.164 0.282 -0.543 0.263 0.105 ...  
## $ returnOnCapitalEmployed : num 0.0531 0.1444 -0.694 0.3165 0.0352 ...  
## $ quickRatio : num 0.54 1.105 5.583 6.94 0.492 ...

str(test\_xgb)

## 'data.frame': 445 obs. of 23 variables:  
## $ Class.Yes : num 1 1 0 1 1 1 1 1 1 1 ...  
## $ X.Revenue.Growth. : num 0.032 0.188 0.381 -0.152 0.078 ...  
## $ X.EPS.Diluted. : num 0.66 4.88 1.29 4.17 1.53 0.8 5.28 5.32 8.36 4.44 ...  
## $ X.EBITDA.Margin. : num 0.453 0.172 0.622 0.465 0.205 0.047 0.075 0.429 0.3 0.287 ...  
## $ priceBookValueRatio : num 1.027 1.505 0.979 3.761 1.382 ...  
## $ debtEquityRatio : num 1.108 0.197 0.453 1.278 0.444 ...  
## $ debtRatio : num 0.473 0.109 0.258 0.429 0.228 ...  
## $ X.PE.ratio. : num 23.3 14 11 14.9 23.4 ...  
## $ Sector.Basic.Materials : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ Sector.Communication.Services: num 0 0 0 0 0 0 0 0 0 0 ...  
## $ Sector.Consumer.Cyclical : num 0 0 0 0 0 0 0 0 1 1 ...  
## $ Sector.Consumer.Defensive : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ Sector.Energy : num 1 1 1 0 1 0 1 1 0 0 ...  
## $ Sector.Financial.Services : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ Sector.Healthcare : num 0 0 0 1 0 0 0 0 0 0 ...  
## $ Sector.Industrials : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ Sector.Real.Estate : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ Sector.Technology : num 0 0 0 0 0 1 0 0 0 0 ...  
## $ Sector.Utilities : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ returnOnAssets : num 0.0446 0.0876 0.0613 0.1866 0.1535 ...  
## $ returnOnEquity : num 0.0478 0.1087 0.0904 0.2551 0.0591 ...  
## $ returnOnCapitalEmployed : num 0.0339 0.0669 0.0578 0.2003 0.0321 ...  
## $ quickRatio : num 0.632 0.486 1.387 3.151 0.79 ...

set.seed(999)  
  
ctrl <- trainControl(method = "cv", number = 5)  
  
tgrid <- expand.grid(  
 nrounds = 100,  
 max\_depth = c(1,2,3,4),  
 eta = c(0.01, 0.1, 0.2, 0.3),  
 gamma = 0,  
 colsample\_bytree = c(0.6, 0.8, 1),  
 min\_child\_weight = 1,  
 subsample = c(0.8, 1)  
)  
  
fitxgb <- train(as.factor(Class.Yes)~.,  
 data= train\_xgb,  
 method = "xgbTree",  
 tuneGrid = tgrid,  
 trControl = ctrl)

predxgbtrain <- predict(fitxgb, train\_xgb)  
confusionMatrix(as.factor(train\_xgb$Class.Yes),predxgbtrain, positive = "1")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 114 247  
## 1 63 617  
##   
## Accuracy : 0.7022   
## 95% CI : (0.6734, 0.7299)  
## No Information Rate : 0.83   
## P-Value [Acc > NIR] : 1   
##   
## Kappa : 0.2534   
##   
## Mcnemar's Test P-Value : <2e-16   
##   
## Sensitivity : 0.7141   
## Specificity : 0.6441   
## Pos Pred Value : 0.9074   
## Neg Pred Value : 0.3158   
## Prevalence : 0.8300   
## Detection Rate : 0.5927   
## Detection Prevalence : 0.6532   
## Balanced Accuracy : 0.6791   
##   
## 'Positive' Class : 1   
##

predxgbtest = predict(fitxgb, test\_xgb)  
confusionMatrix(as.factor(test\_xgb$Class.Yes), predxgbtest, positive = "1")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 50 104  
## 1 31 260  
##   
## Accuracy : 0.6966   
## 95% CI : (0.6516, 0.739)  
## No Information Rate : 0.818   
## P-Value [Acc > NIR] : 1   
##   
## Kappa : 0.2455   
##   
## Mcnemar's Test P-Value : 5.763e-10   
##   
## Sensitivity : 0.7143   
## Specificity : 0.6173   
## Pos Pred Value : 0.8935   
## Neg Pred Value : 0.3247   
## Prevalence : 0.8180   
## Detection Rate : 0.5843   
## Detection Prevalence : 0.6539   
## Balanced Accuracy : 0.6658   
##   
## 'Positive' Class : 1   
##

In the training set our accuracy is now 70.22% and in the testing set it is 69.66%. Now our sensitivity for the trianing set is 0.7141, and 0.7143 in the testing set. The specificity is 0.6441 in the training set and 0.6173 in the testing set. This model has decreased the specifity from the prevous model, but has increased our sensitivity for both. Making the models have a better all around predictions instead of what previous models had.