Random Forest

Carrie Cheng

2023-04-30

```
library(dplyr)
dat <- read.csv('brfss_final.csv')
outcome <- data.frame(dat$X,dat$MICHD,dat$CVDINFR4,dat$CVDCRHD4)
outcome %>% group_by(dat.MICHD) %>% summarise(count=n())
```

```
## # A tibble: 2 x 2
##   dat.MICHD count
##       <int> <int>
## 1         1 14580
## 2         2 14580
```

```
outcome %>% group_by(dat.CVDINFR4) %>% summarise(count=n())
```

```
## # A tibble: 4 x 2
##   dat.CVDINFR4 count
##          <int> <int>
## 1            1  9188
## 2            2 19802
## 3            7   160
## 4            9    10
```

```
outcome %>% group_by(dat.CVDCRHD4) %>% summarise(count=n())
```

```
## # A tibble: 4 x 2
##   dat.CVDCRHD4 count
##          <int> <int>
## 1            1  9729
## 2            2 18874
## 3            7   550
## 4            9     7
```

```
## remove the ones that responded don't know & not sure in CVDINFR4 & CVDCRHD4
dat <- dat[-which(dat$CVDINFR4 == 7 | dat$CVDINFR4 == 9),]
dat <- dat[-which(dat$CVDCRHD4 == 7 | dat$CVDCRHD4 == 9),]
# remove columns that has only 1 value for all rows
dat <- dat[ , -which(names(dat) %in% c("MEDSHEPB","TOLDCFS", "HAVECFS", "WORKCFS"))]
```

Drop columns with more than 5% data missing, impute the rest using KNN

```r
# convert outcome variables
dat$MICHD <- factor(2-dat$MICHD)
dat$CVDINFR4 <- factor(2-dat$CVDINFR4)
dat$CVDCRHD4 <- factor(2-dat$CVDCRHD4)
# i believe X is the index column, not needed
# remove weights
dat <- dat[, !colnames(dat) %in% c('X', 'LLCPWT2', 'LLCPWT', 'CLLCPWT','STRWT','WT2RAKE')]
dat <- dat[, !colnames(dat) %in% c('QSTVER', 'STSTR','RAWRAKE')] # remove based on knowledge
threshold <- .05
ncol(dat) # 190
```

```
## [1] 187
```

```r
dat <- dat[, colMeans(is.na(dat)) <= threshold]
ncol(dat) # 52 columns left
```

```
## [1] 49
```

```r
columns_to_impute <- colnames(dat)[colSums(is.na(dat)) > 0]
columns_to_impute
```

```
##  [1] "CPDEMO1B" "VETERAN3" "EMPLOY1"  "INCOME3"  "DEAF"     "BLIND"
##  [7] "DECIDE"   "DIFFWALK" "DIFFDRES" "DIFFALON" "USENOW3"  "METSTAT"
## [13] "URBSTAT"  "MSCODE"   "DRDXAR3"
```

```r
str(dat[,columns_to_impute])
```

```
## 'data.frame':    28433 obs. of  15 variables:
##  $ CPDEMO1B: int  1 1 8 1 1 8 8 1 1 2 ...
##  $ VETERAN3: int  2 2 2 2 1 2 1 2 2 2 ...
##  $ EMPLOY1 : int  8 7 2 7 7 7 7 8 7 7 ...
##  $ INCOME3 : int  77 3 99 77 7 99 5 77 5 10 ...
##  $ DEAF    : int  2 2 2 2 2 2 1 2 2 2 ...
##  $ BLIND   : int  1 2 2 2 2 2 2 2 2 2 ...
##  $ DECIDE  : int  1 2 1 2 1 2 2 2 2 2 ...
##  $ DIFFWALK: int  1 2 2 2 2 1 1 1 2 2 ...
##  $ DIFFDRES: int  2 2 2 2 2 1 2 2 2 2 ...
##  $ DIFFALON: int  1 2 2 2 2 1 1 2 2 2 ...
##  $ USENOW3 : int  3 3 3 3 3 3 3 3 3 3 ...
##  $ METSTAT : int  1 1 1 1 1 2 1 2 1 1 ...
##  $ URBSTAT : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ MSCODE  : int  2 1 3 1 3 2 2 5 2 3 ...
##  $ DRDXAR3 : int  1 2 1 1 2 1 1 2 1 1 ...
```

```r
complete_columns <- colnames(dat)[colSums(is.na(dat)) == 0 &
                                  !colnames(dat) %in% c('MICHD', 'CVDINFR4','CVDCRHD4')]
# miss_names <- paste0("miss_", columns_to_impute)
# dat[, miss_names] <- NA
# for (i in 1:nrow(dat)){
#   for (j in 1:length(miss_names)) {
```

```
#     dat[i, miss_names[j]] <- as.numeric(any(is.na(dat[i, columns_to_impute[j]])))
#   }
# }
for (c in columns_to_impute) {
    col <- dat[[c]]
    scaled <- scale(dat[, complete_columns])
    knn <- knn(
        train = scaled[!is.na(col), complete_columns],
        test  = scaled[is.na(col), complete_columns],
        cl    = dat[!is.na(col), c]
        )

    dat[is.na(col), c] = knn
}
colSums(is.na(dat))
```

```
##  GENHLTH PHYSHLTH MENTHLTH PRIMINSR PERSDOC3 MEDCOST1 CHECKUP1 CVDINFR4
##        0        0        0        0        0        0        0        0
## CVDCRHD4 CVDSTRK3 CHCSCNCR CHCOCNCR CHCCOPD3 ADDEPEV3 CHCKDNY2 DIABETE4
##        0        0        0        0        0        0        0        0
##  MARITAL RENTHOM1 NUMHHOL3 CPDEMO1B VETERAN3  EMPLOY1  INCOME3     DEAF
##        0        0        0        0        0        0        0        0
##    BLIND   DECIDE DIFFWALK DIFFDRES DIFFALON  USENOW3  QSTLANG  METSTAT
##        0        0        0        0        0        0        0        0
##  URBSTAT   MSCODE  DUALUSE  TOTINDA  RFHYPE6  CHOLCH3    MICHD  ASTHMS1
##        0        0        0        0        0        0        0        0
##  DRDXAR3     RACE      SEX    AGE80  CHLDCNT   EDUCAG  SMOKER3  CURECI1
##        0        0        0        0        0        0        0        0
## DROCDY3_
##        0
```

```
library(caret)
```

```
## Loading required package: ggplot2
```

```
## Loading required package: lattice
```

```
library(randomForest)
```

```
## randomForest 4.7-1.1
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
library(ggplot2)
library(ROCR)
set.seed(263)
train_index <- createDataPartition(dat$MICHD, p = 0.8, list = FALSE)
train <- dat[train_index, ]
test <- dat[-train_index, ]
# train$weights <- ifelse(as.numeric(train$MICHD) == 1,
#                         1/mean(as.numeric(train$MICHD) == 1),
#                         1/(1-mean(as.numeric(train$MICHD) == 1)))
# train$weights <- as.numeric(train$weights)
# test$weights <- ifelse(as.numeric(test$MICHD) == 1,
#                         1/mean(as.numeric(test$MICHD) == 1),
#                         1/(1-mean(as.numeric(test$MICHD) == 1)))
# test$weights <- as.numeric(test$weights)
# index_weight <- which(names(train) == "weights")
summary(train)
```

```
##     GENHLTH         PHYSHLTH        MENTHLTH        PRIMINSR
##  Min.   :1.000   Min.   : 1.00   Min.   : 1.00   Min.   : 1.000
##  1st Qu.:2.000   1st Qu.:20.00   1st Qu.:30.00   1st Qu.: 3.000
##  Median :3.000   Median :88.00   Median :88.00   Median : 3.000
##  Mean   :2.941   Mean   :59.02   Mean   :65.33   Mean   : 7.463
##  3rd Qu.:4.000   3rd Qu.:88.00   3rd Qu.:88.00   3rd Qu.: 3.000
##  Max.   :9.000   Max.   :99.00   Max.   :99.00   Max.   :99.000
##     PERSDOC3        MEDCOST1        CHECKUP1     CVDINFR4  CVDCRHD4
##  Min.   :1.000   Min.   :1.000   Min.   :1.000   0:15844   0:15091
##  1st Qu.:1.000   1st Qu.:2.000   1st Qu.:1.000   1: 6903   1: 7656
##  Median :1.000   Median :2.000   Median :1.000
##  Mean   :1.498   Mean   :1.972   Mean   :1.246
##  3rd Qu.:2.000   3rd Qu.:2.000   3rd Qu.:1.000
##  Max.   :9.000   Max.   :9.000   Max.   :9.000
##     CVDSTRK3        CHCSCNCR        CHCOCNCR        CHCCOPD3
##  Min.   :1.000   Min.   :1.000   Min.   :1.000   Min.   :1.000
##  1st Qu.:2.000   1st Qu.:2.000   1st Qu.:2.000   1st Qu.:2.000
##  Median :2.000   Median :2.000   Median :2.000   Median :2.000
##  Mean   :1.921   Mean   :1.835   Mean   :1.837   Mean   :1.867
##  3rd Qu.:2.000   3rd Qu.:2.000   3rd Qu.:2.000   3rd Qu.:2.000
##  Max.   :9.000   Max.   :9.000   Max.   :9.000   Max.   :9.000
##     ADDEPEV3        CHCKDNY2        DIABETE4        MARITAL
##  Min.   :1.000   Min.   :1.000   Min.   :1.000   Min.   :1.000
##  1st Qu.:2.000   1st Qu.:2.000   1st Qu.:1.000   1st Qu.:1.000
##  Median :2.000   Median :2.000   Median :3.000   Median :2.000
##  Mean   :1.854   Mean   :1.938   Mean   :2.532   Mean   :2.173
##  3rd Qu.:2.000   3rd Qu.:2.000   3rd Qu.:3.000   3rd Qu.:3.000
##  Max.   :9.000   Max.   :9.000   Max.   :9.000   Max.   :9.000
##     RENTHOM1        NUMHHOL3        CPDEMO1B        VETERAN3
##  Min.   :1.000   Min.   :1.000   Min.   :1.000   Min.   :1.000
##  1st Qu.:1.000   1st Qu.:1.000   1st Qu.:1.000   1st Qu.:2.000
##  Median :1.000   Median :2.000   Median :1.000   Median :2.000
##  Mean   :1.248   Mean   :1.789   Mean   :2.345   Mean   :1.832
```

```
##  3rd Qu.:1.000   3rd Qu.:2.000   3rd Qu.:2.000   3rd Qu.:2.000
##  Max.   :9.000   Max.   :9.000   Max.   :9.000   Max.   :9.000
##     EMPLOY1         INCOME3           DEAF            BLIND
##  Min.   :1.000   Min.   : 1.00   Min.   :1.000   Min.   :1.000
##  1st Qu.:5.000   1st Qu.: 5.00   1st Qu.:2.000   1st Qu.:2.000
##  Median :7.000   Median : 7.00   Median :2.000   Median :2.000
##  Mean   :5.701   Mean   :25.92   Mean   :1.878   Mean   :1.967
##  3rd Qu.:7.000   3rd Qu.:11.00   3rd Qu.:2.000   3rd Qu.:2.000
##  Max.   :9.000   Max.   :99.00   Max.   :9.000   Max.   :9.000
##      DECIDE         DIFFWALK        DIFFDRES        DIFFALON
##  Min.   :1.000   Min.   :1.000   Min.   :1.000   Min.   :1.000
##  1st Qu.:2.000   1st Qu.:1.000   1st Qu.:2.000   1st Qu.:2.000
##  Median :2.000   Median :2.000   Median :2.000   Median :2.000
##  Mean   :1.957   Mean   :1.758   Mean   :1.971   Mean   :1.934
##  3rd Qu.:2.000   3rd Qu.:2.000   3rd Qu.:2.000   3rd Qu.:2.000
##  Max.   :9.000   Max.   :9.000   Max.   :9.000   Max.   :9.000
##     USENOW3         QSTLANG         METSTAT         URBSTAT
##  Min.   :1.000   Min.   :1.000   Min.   :1.000   Min.   :1.000
##  1st Qu.:3.000   1st Qu.:1.000   1st Qu.:1.000   1st Qu.:1.000
##  Median :3.000   Median :1.000   Median :1.000   Median :1.000
##  Mean   :3.027   Mean   :1.007   Mean   :1.381   Mean   :1.199
##  3rd Qu.:3.000   3rd Qu.:1.000   3rd Qu.:2.000   3rd Qu.:1.000
##  Max.   :9.000   Max.   :2.000   Max.   :2.000   Max.   :2.000
##      MSCODE         DUALUSE         TOTINDA         RFHYPE6
##  Min.   :1.000   Min.   :1.000   Min.   :1.000   Min.   :1.000
##  1st Qu.:1.000   1st Qu.:1.000   1st Qu.:1.000   1st Qu.:1.000
##  Median :3.000   Median :1.000   Median :1.000   Median :2.000
##  Mean   :3.031   Mean   :2.326   Mean   :1.362   Mean   :1.665
##  3rd Qu.:5.000   3rd Qu.:1.000   3rd Qu.:2.000   3rd Qu.:2.000
##  Max.   :5.000   Max.   :9.000   Max.   :9.000   Max.   :9.000
##     CHOLCH3      MICHD         ASTHMS1         DRDXAR3           RACE
##  Min.   :1.000   0:11664   Min.   :1.00   Min.   :1.000   Min.   :1.000
##  1st Qu.:1.000   1:11083   1st Qu.:3.00   1st Qu.:1.000   1st Qu.:1.000
##  Median :1.000             Median :3.00   Median :1.000   Median :1.000
##  Mean   :1.429             Mean   :2.81   Mean   :1.476   Mean   :1.632
##  3rd Qu.:1.000             3rd Qu.:3.00   3rd Qu.:2.000   3rd Qu.:1.000
##  Max.   :9.000             Max.   :9.00   Max.   :2.000   Max.   :9.000
##       SEX             AGE80           CHLDCNT          EDUCAG        SMOKER3
##  Min.   :1.000   Min.   :18.00   Min.   :1.000   Min.   :1.00   Min.   :1.00
##  1st Qu.:1.000   1st Qu.:65.00   1st Qu.:1.000   1st Qu.:2.00   1st Qu.:3.00
##  Median :2.000   Median :72.00   Median :1.000   Median :3.00   Median :4.00
##  Mean   :1.571   Mean   :69.94   Mean   :1.272   Mean   :2.99   Mean   :3.63
##  3rd Qu.:2.000   3rd Qu.:80.00   3rd Qu.:1.000   3rd Qu.:4.00   3rd Qu.:4.00
##  Max.   :2.000   Max.   :80.00   Max.   :9.000   Max.   :9.00   Max.   :9.00
##     CURECI1         DROCDY3_
##  Min.   :1.000   Min.   :  0.00
##  1st Qu.:1.000   1st Qu.:  0.00
##  Median :1.000   Median :  0.00
##  Mean   :1.399   Mean   : 69.66
##  3rd Qu.:1.000   3rd Qu.: 17.00
##  Max.   :9.000   Max.   :900.00
```

Apply PCA

```
### get factor loadings for the train set
train <- apply(train, 2, as.numeric)
train <- as.data.frame(train)
col_index <- which(names(train) %in% c("MICHD", "CVDINFR4", "CVDCRHD4", "weights"))
pc <- prcomp(train[, -col_index], scale. = T)
lambda <- pc$sdev^2
M <- min(which(cumsum(lambda)/sum(lambda) > 0.85))
print(M)
```

```
## [1] 32
```

```
loadings <- pc$x[, 1:M]

train_pca <- data.frame(loadings, MICHD = train$MICHD)
```

```
### get factor loadings for the test set
pc_test <- prcomp(test[, -col_index], scale. = T)
loadings_test <- pc_test$x[, 1:M]

test_pca <- data.frame(loadings_test, MICHD = test$MICHD)
```

Parameter Tuning

Let's tune number of trees ntrees and number of features selected to place split mtry. In the following, let's use 10-fold cross-validation.

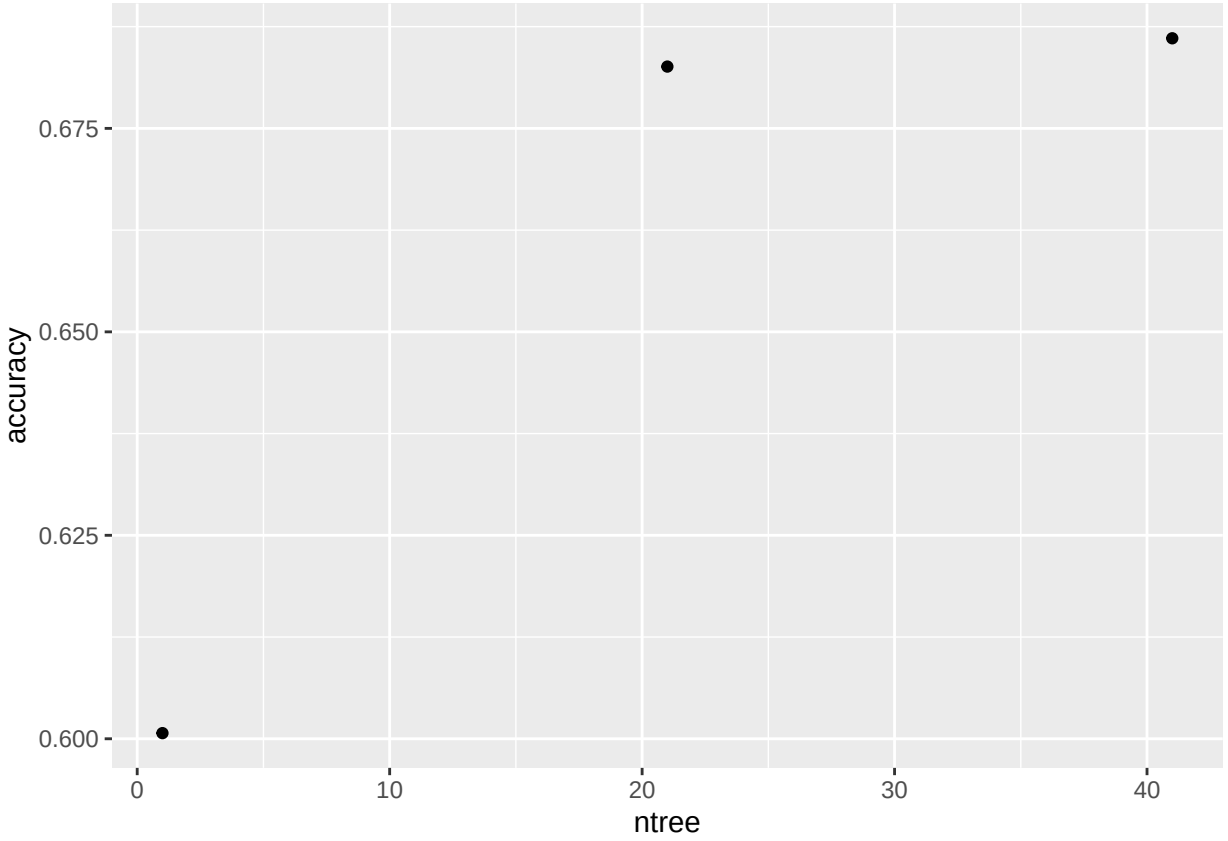
```
## get index of the other two outcomes

index_michd <- which(names(train) == "MICHD")
index_infr <- which(names(train) == "CVDINFR4")
index_crhd <- which(names(train) == "CVDCRHD4")
```

Tune number of trees

Let's set mtry = 10.

```
ntree <- seq(1, 51, by = 20)
accuracy <- sapply(ntree, function(n){
  train(as.factor(MICHD) ~ ., method = "rf",
        data = train_pca,
        tuneGrid = data.frame(mtry = 10),
        ntree = n, trControl = trainControl(method = "cv", number = 10))$results$Accuracy
})

qplot(ntree, accuracy)
```

```
## Warning: `qplot()` was deprecated in ggplot2 3.4.0.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

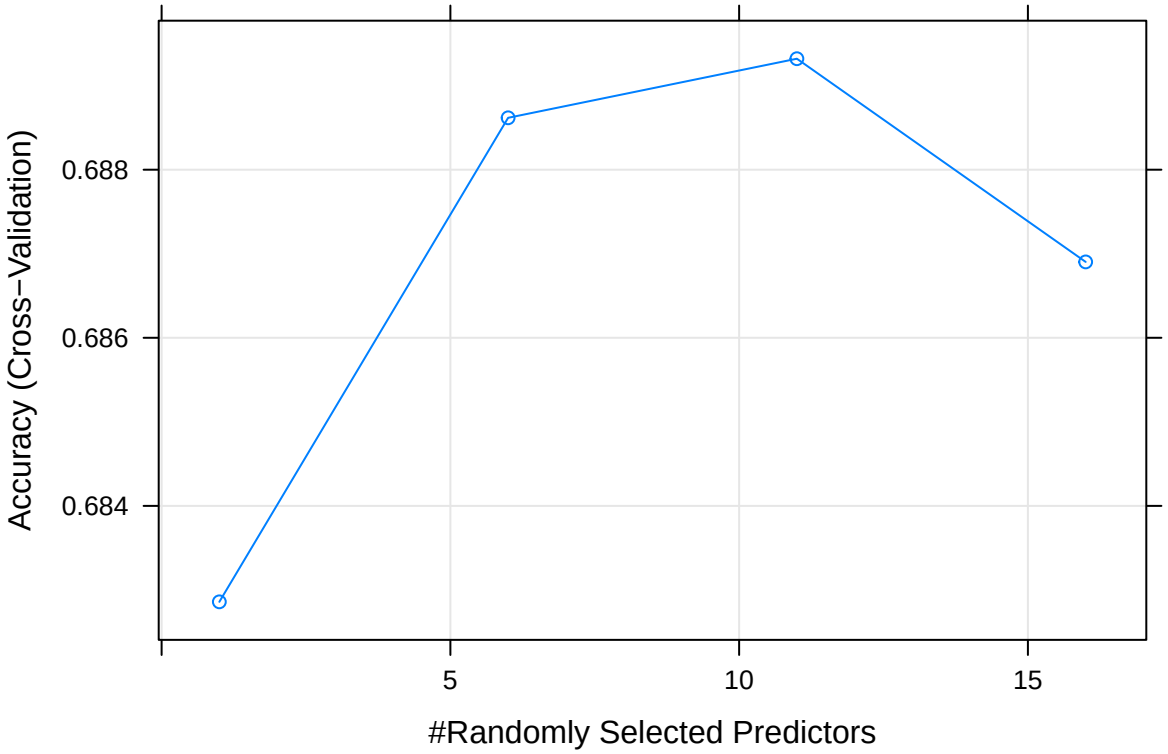



```
best_ntree <- ntree[which(accuracy == max(accuracy))]
best_ntree <- min(best_ntree)
print(paste("The best ntree is", best_ntree))
```

```
## [1] "The best ntree is 41"
```

Tune mtry

```
train_rf <- train(as.factor(MICHD) ~ ., method = "rf",
                  data = train_pca,
                  tuneGrid = data.frame(mtry = seq(1, 20, by = 5)),
                  ntree = best_ntree,
                  nodesize = 10, trControl = trainControl(method = "cv", number = 10))

plot(train_rf)
```




```
best_mtry <- train_rf$bestTune

result_cv <- train_rf$results

print(paste("The best mtry is ", best_mtry))
```

```
## [1] "The best mtry is  11"
```

Use the best model to train random forest

The below is the confusion matrix on the test set.

```
rf_best <- randomForest(as.factor(MICHD) ~.,
                        data = train_pca,
                        mtry = best_mtry[[1]], ntree = best_ntree, nodesize = 10)

pred_test <- predict(rf_best, test_pca)
cm_test <- confusionMatrix(as.factor(pred_test), as.factor(test_pca$MICHD))

cm_test
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1930  896
##          1  986 1874
```

```
##
##                Accuracy : 0.669
##                  95% CI : (0.6566, 0.6812)
##     No Information Rate : 0.5128
##     P-Value [Acc > NIR] : < 2e-16
##
##                   Kappa : 0.3381
##
##  Mcnemar's Test P-Value : 0.04021
##
##             Sensitivity : 0.6619
##             Specificity : 0.6765
##          Pos Pred Value : 0.6829
##          Neg Pred Value : 0.6552
##              Prevalence : 0.5128
##          Detection Rate : 0.3394
##    Detection Prevalence : 0.4970
##       Balanced Accuracy : 0.6692
##
##        'Positive' Class : 0
##
```

```
metric_test <- c(cm_test$overall[["Accuracy"]],
                 cm_test$byClass[c("Sensitivity","Specificity")])

cat(paste("The overall accuracy using the best tuned random forest model is",
      metric_test[1], "\n",
      "Sensitivity is", metric_test[2], "\n",
      "Specificity is", metric_test[3]))
```

```
## The overall accuracy using the best tuned random forest model is 0.669011607456912
##  Sensitivity is 0.661865569272977
##  Specificity is 0.676534296028881
```

ROC curve

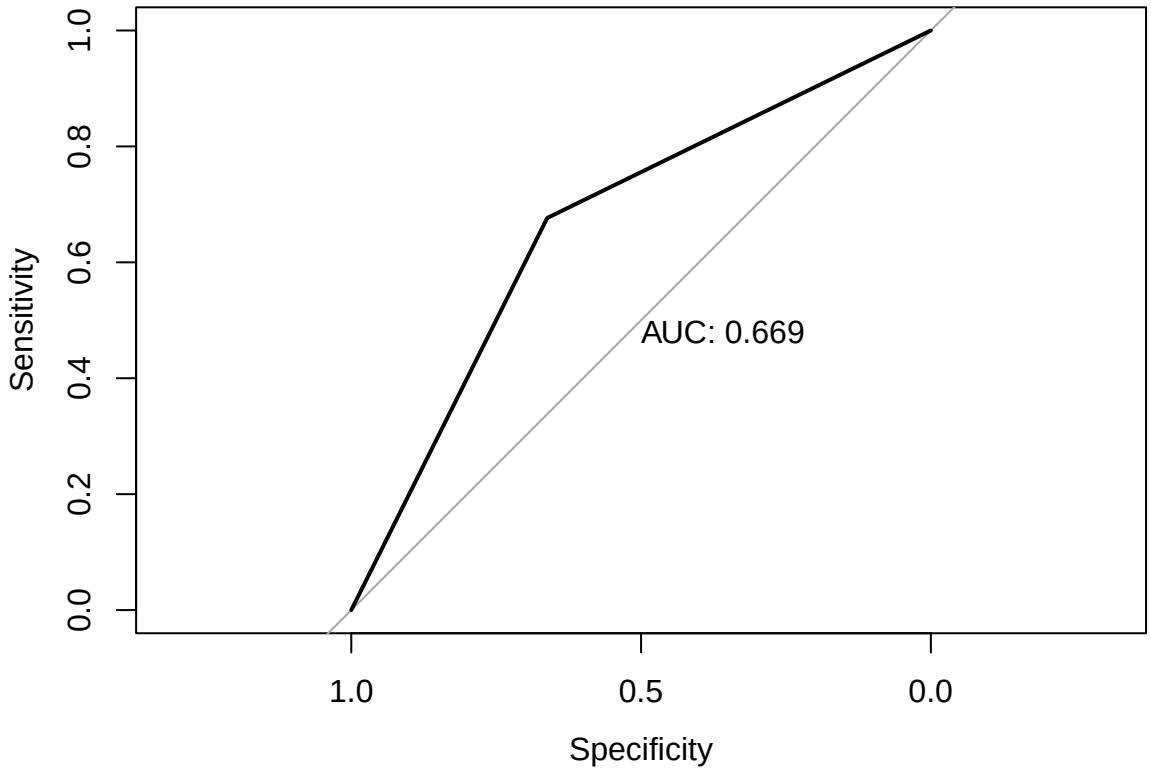
```
library(pROC)
```

```
## Type 'citation("pROC")' for a citation.

##
## Attaching package: 'pROC'

## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```
roc_rf <- roc(as.numeric(test$MICHD) ~ as.numeric(pred_test),
              plot = TRUE, print.auc = TRUE)
```

```
## Setting levels: control = 1, case = 2
```

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




```
print(paste("AUC is", as.numeric(roc_rf$auc)))
```

```
## [1] "AUC is 0.669199932650929"
```

Importance Features

```
importance(rf_best)
```

```
##      MeanDecreaseGini
## PC1         1054.0741
## PC2          783.0769
## PC3          343.7225
## PC4          536.6740
## PC5          427.6975
## PC6          247.1356
## PC7          272.3550
## PC8          540.5986
## PC9          214.9752
## PC10         216.0115
## PC11         230.7977
## PC12         204.5897
```

```
## PC13         200.3967
## PC14         222.8896
## PC15         218.0884
## PC16         260.5821
## PC17         207.3853
## PC18         205.2519
## PC19         188.5568
## PC20         213.1544
## PC21         208.1936
## PC22         204.3037
## PC23         195.2351
## PC24         218.7295
## PC25         248.1405
## PC26         218.0383
## PC27         208.4982
## PC28         214.7951
## PC29         218.4504
## PC30         208.5454
## PC31         204.7015
## PC32         227.5594
```

```
varImpPlot(rf_best)
```