# Module 4 - Random Forests

## Cooper, Sarah

library(tidyverse)

library(caret)

library(ranger)

Blood\_1\_ <- read\_csv("C:/Users/Sarah/Downloads/Blood (1).csv")

## Parsed with column specification:  
## cols(  
## Mnths\_Since\_Last = col\_double(),  
## TotalDonations = col\_double(),  
## Total\_Donated = col\_double(),  
## Mnths\_Since\_First = col\_double(),  
## DonatedMarch = col\_double()  
## )

Blood <- Blood\_1\_  
Blood = Blood %>% mutate(DonatedMarch = as\_factor(as.character(DonatedMarch))) %>%  
mutate(DonatedMarch = fct\_recode(DonatedMarch,  
"No" = "0",  
"Yes" = "1"))

# Task 1

set.seed(1234)  
train.rows = createDataPartition(y = Blood$DonatedMarch, p=0.7, list = FALSE)  
train = Blood[train.rows,]  
test = Blood[-train.rows,]

# Task 2

fit\_control = trainControl(method = "cv", number = 10)  
  
set.seed(123)  
rf\_fit = train(x=Blood[,-5], y=Blood$DonatedMarch, method = "ranger", importance = "permutation", trControl = fit\_control, num.trees = 100)

# Task 3

varImp(rf\_fit)

## ranger variable importance  
##   
## Overall  
## Total\_Donated 100.00  
## Mnths\_Since\_First 20.33  
## TotalDonations 17.08  
## Mnths\_Since\_Last 0.00

rf\_fit

## Random Forest   
##   
## 748 samples  
## 4 predictor  
## 2 classes: 'Yes', 'No'   
##   
## No pre-processing  
## Resampling: Cross-Validated (10 fold)   
## Summary of sample sizes: 674, 674, 673, 673, 673, 673, ...   
## Resampling results across tuning parameters:  
##   
## mtry splitrule Accuracy Kappa   
## 2 gini 0.7620901 0.2512608  
## 2 extratrees 0.7794414 0.2659173  
## 3 gini 0.7406847 0.2094663  
## 3 extratrees 0.7567387 0.2353788  
## 4 gini 0.7460901 0.2246573  
## 4 extratrees 0.7474234 0.2139096  
##   
## Tuning parameter 'min.node.size' was held constant at a value of 1  
## Accuracy was used to select the optimal model using the largest value.  
## The final values used for the model were mtry = 2, splitrule = extratrees  
## and min.node.size = 1.

*The most important variable in the dataset is Total\_Donated and the least important is Mnths\_Since\_Last.*

# Task 4

predRF = predict(rf\_fit, train)  
head(predRF)

## [1] Yes Yes Yes No No Yes  
## Levels: Yes No

# Task 5

confusionMatrix(predRF, train$DonatedMarch, positive = "Yes")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction Yes No  
## Yes 77 7  
## No 48 392  
##   
## Accuracy : 0.895   
## 95% CI : (0.8656, 0.9199)  
## No Information Rate : 0.7615   
## P-Value [Acc > NIR] : 4.371e-15   
##   
## Kappa : 0.6744   
##   
## Mcnemar's Test P-Value : 6.906e-08   
##   
## Sensitivity : 0.6160   
## Specificity : 0.9825   
## Pos Pred Value : 0.9167   
## Neg Pred Value : 0.8909   
## Prevalence : 0.2385   
## Detection Rate : 0.1469   
## Detection Prevalence : 0.1603   
## Balanced Accuracy : 0.7992   
##   
## 'Positive' Class : Yes   
##

*Accuracy = 90%, Sensitivity = 62%, and Specificity = 98%*

# Task 6

*The actual accuracy of the model at 90% is far better than the naive model at 76%.*

# Task 7

predRF\_test = predict(rf\_fit, newdata = test)  
confusionMatrix(predRF\_test, test$DonatedMarch, positive = "Yes")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction Yes No  
## Yes 26 3  
## No 27 168  
##   
## Accuracy : 0.8661   
## 95% CI : (0.8144, 0.9078)  
## No Information Rate : 0.7634   
## P-Value [Acc > NIR] : 9.092e-05   
##   
## Kappa : 0.5606   
##   
## Mcnemar's Test P-Value : 2.679e-05   
##   
## Sensitivity : 0.4906   
## Specificity : 0.9825   
## Pos Pred Value : 0.8966   
## Neg Pred Value : 0.8615   
## Prevalence : 0.2366   
## Detection Rate : 0.1161   
## Detection Prevalence : 0.1295   
## Balanced Accuracy : 0.7365   
##   
## 'Positive' Class : Yes   
##

*When using the model on the testing set the results are similar to that of the training set. The only sharp contrast is the Sensitivity rating of 49% on the test model versus 62% on the train model.*

# Task 8

*I believe these datasets would be acceptable for real-world use. I would be cautious if implementing any firm decisions based off these datasets alone but that’s merely due to their accuracy rates not crossing into the 90%+ range. With high confidence you could predit the DonatedMarch variable using these models.*