## Random Forests

libraries

options(tidyverse.quiet=FALSE)  
library(tidyverse)

## Warning: package 'tidyverse' was built under R version 3.5.2

## -- Attaching packages ----------------------------------------------------- tidyverse 1.2.1 --

## v ggplot2 3.1.0 v purrr 0.3.0  
## v tibble 2.0.1 v dplyr 0.7.8  
## v tidyr 0.8.2 v stringr 1.3.1  
## v readr 1.3.1 v forcats 0.3.0

## Warning: package 'tibble' was built under R version 3.5.2

## Warning: package 'readr' was built under R version 3.5.2

## Warning: package 'purrr' was built under R version 3.5.2

## Warning: package 'dplyr' was built under R version 3.5.2

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

library(caret)

## Warning: package 'caret' was built under R version 3.5.2

## Loading required package: lattice

##   
## Attaching package: 'caret'

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

library(ranger)

## Warning: package 'ranger' was built under R version 3.5.2

read-in dataset

blood = read\_csv("Blood.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()  
## )

str(blood)

## Classes 'spec\_tbl\_df', 'tbl\_df', 'tbl' and 'data.frame': 748 obs. of 5 variables:  
## $ Mnths\_Since\_Last : num 2 0 1 2 1 4 2 1 2 5 ...  
## $ TotalDonations : num 50 13 16 20 24 4 7 12 9 46 ...  
## $ Total\_Donated : num 12500 3250 4000 5000 6000 1000 1750 3000 2250 11500 ...  
## $ Mnths\_Since\_First: num 98 28 35 45 77 4 14 35 22 98 ...  
## $ DonatedMarch : num 1 1 1 1 0 0 1 0 1 1 ...  
## - attr(\*, "spec")=  
## .. cols(  
## .. Mnths\_Since\_Last = col\_double(),  
## .. TotalDonations = col\_double(),  
## .. Total\_Donated = col\_double(),  
## .. Mnths\_Since\_First = col\_double(),  
## .. DonatedMarch = col\_double()  
## .. )

convert DonatedMarch variable to a factor and recode

blood = blood %>% mutate(DonatedMarch = as\_factor(as.character(DonatedMarch))) %>%  
mutate(DonatedMarch = fct\_recode(DonatedMarch, "Yes" = "1", "No" = "0"))  
str(blood)

## Classes 'tbl\_df', 'tbl' and 'data.frame': 748 obs. of 5 variables:  
## $ Mnths\_Since\_Last : num 2 0 1 2 1 4 2 1 2 5 ...  
## $ TotalDonations : num 50 13 16 20 24 4 7 12 9 46 ...  
## $ Total\_Donated : num 12500 3250 4000 5000 6000 1000 1750 3000 2250 11500 ...  
## $ Mnths\_Since\_First: num 98 28 35 45 77 4 14 35 22 98 ...  
## $ DonatedMarch : Factor w/ 2 levels "Yes","No": 1 1 1 1 2 2 1 2 1 1 ...

Task 1: Split the dataset into training(70%) and testing(30%) sets using seed 1234

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

Task 2: Create a random forest model on the training set

fit\_control = trainControl(method = "cv",   
 number = 10) #set up 10 fold cross-validation  
set.seed(1234)   
rf\_fit = train(DonatedMarch ~.,   
 data = trainBlood,   
 method = "ranger",   
 importance = "permutation",   
 num.trees = 100,  
 trControl = fit\_control)

Task 3: Use varImp to determine most/least important variables in the model

varImp(rf\_fit)

## ranger variable importance  
##   
## Overall  
## Mnths\_Since\_Last 100.0  
## Mnths\_Since\_First 56.4  
## Total\_Donated 29.6  
## TotalDonations 0.0

rf\_fit

## Random Forest   
##   
## 524 samples  
## 4 predictor  
## 2 classes: 'Yes', 'No'   
##   
## No pre-processing  
## Resampling: Cross-Validated (10 fold)   
## Summary of sample sizes: 471, 472, 471, 472, 472, 471, ...   
## Resampling results across tuning parameters:  
##   
## mtry splitrule Accuracy Kappa   
## 2 gini 0.7707184 0.2877282  
## 2 extratrees 0.7765602 0.2836371  
## 3 gini 0.7534107 0.2317846  
## 3 extratrees 0.7688679 0.2770361  
## 4 gini 0.7515965 0.2395862  
## 4 extratrees 0.7651306 0.2681866  
##   
## 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 model is Mnths\_Since\_Last and the least important variable is TotalDonations

Task 4: Use the model to develop predictions on the training set.

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

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

Task 5: Use the model to create a confusion matrix

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

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction Yes No  
## Yes 82 8  
## No 43 391  
##   
## Accuracy : 0.9027   
## 95% CI : (0.874, 0.9267)  
## No Information Rate : 0.7615   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.7036   
## Mcnemar's Test P-Value : 1.927e-06   
##   
## Sensitivity : 0.6560   
## Specificity : 0.9799   
## Pos Pred Value : 0.9111   
## Neg Pred Value : 0.9009   
## Prevalence : 0.2385   
## Detection Rate : 0.1565   
## Detection Prevalence : 0.1718   
## Balanced Accuracy : 0.8180   
##   
## 'Positive' Class : Yes   
##

The accuracy for the training set is: 0.9027 The Sensitivity is: 0.6560 The Specificity is: 0.9799

Task 6: How does the accuracy of the confusion matrix compare to a naive model? The naive accuracy (0.7615) is not as good as the accuracy of the confusion matrix (0.9027)

Task 7:

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

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

confusion matrix for testing set

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

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction Yes No  
## Yes 15 15  
## No 38 156  
##   
## Accuracy : 0.7634   
## 95% CI : (0.7022, 0.8175)  
## No Information Rate : 0.7634   
## P-Value [Acc > NIR] : 0.536778   
##   
## Kappa : 0.2297   
## Mcnemar's Test P-Value : 0.002512   
##   
## Sensitivity : 0.28302   
## Specificity : 0.91228   
## Pos Pred Value : 0.50000   
## Neg Pred Value : 0.80412   
## Prevalence : 0.23661   
## Detection Rate : 0.06696   
## Detection Prevalence : 0.13393   
## Balanced Accuracy : 0.59765   
##   
## 'Positive' Class : Yes   
##

The model doesn’t perform as well on the testing set as it does on the training set; however, it is not entirely innacurrate. The niave model has the same level of accuracy as our model of 0.7634. Accuracy: 0.7634 Sensitivity: 0.28302 Specificity: 0.91228