Random Forests

# Random Forests

## BAN502

### Danielle Harris

library(tidyverse)

## ── Attaching packages ──────────────── tidyverse 1.2.1 ──

## ✔ ggplot2 3.1.0 ✔ purrr 0.2.5  
## ✔ tibble 1.4.2 ✔ dplyr 0.7.7  
## ✔ tidyr 0.8.2 ✔ stringr 1.3.1  
## ✔ readr 1.1.1 ✔ forcats 0.3.0

## ── Conflicts ─────────────────── tidyverse\_conflicts() ──  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ dplyr::lag() masks stats::lag()

library(caret)

## 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

Blood= read\_csv("Blood.csv")

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

Blood = Blood %>% mutate(DonatedMarch= as.factor(DonatedMarch)) %>%   
 mutate (DonatedMarch= fct\_recode(DonatedMarch, "no"="0", "yes"= "1"))

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

fit\_control = trainControl(method = "cv",   
 number = 10)   
  
set.seed(123)   
rf\_fit = train(DonatedMarch ~.,   
 data = Blood,   
 method = "ranger",   
 importance = "permutation",   
 num.trees = 100,  
 trControl = fit\_control)

varImp(rf\_fit)

## ranger variable importance  
##   
## Overall  
## TotalDonations 100.00  
## Total\_Donated 72.61  
## Mnths\_Since\_First 22.65  
## Mnths\_Since\_Last 0.00

rf\_fit

## Random Forest   
##   
## 748 samples  
## 4 predictor  
## 2 classes: 'no', 'yes'   
##   
## No pre-processing  
## Resampling: Cross-Validated (10 fold)   
## Summary of sample sizes: 673, 673, 673, 673, 673, 673, ...   
## Resampling results across tuning parameters:  
##   
## mtry splitrule Accuracy Kappa   
## 2 gini 0.7647207 0.2536439  
## 2 extratrees 0.7847387 0.2969495  
## 3 gini 0.7500360 0.2266758  
## 3 extratrees 0.7553514 0.2184775  
## 4 gini 0.7527387 0.2221681  
## 4 extratrees 0.7353333 0.1913524  
##   
## 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 variable “TotalDonations” is the most important variable, while the least important variable is “Mnths\_Since\_Last.”

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

## [1] yes yes no no yes yes  
## Levels: no yes

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

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction no yes  
## no 395 54  
## yes 4 71  
##   
## Accuracy : 0.8893   
## 95% CI : (0.8593, 0.9149)  
## No Information Rate : 0.7615   
## P-Value [Acc > NIR] : 7.715e-14   
##   
## Kappa : 0.6468   
## Mcnemar's Test P-Value : 1.243e-10   
##   
## Sensitivity : 0.5680   
## Specificity : 0.9900   
## Pos Pred Value : 0.9467   
## Neg Pred Value : 0.8797   
## Prevalence : 0.2385   
## Detection Rate : 0.1355   
## Detection Prevalence : 0.1431   
## Balanced Accuracy : 0.7790   
##   
## 'Positive' Class : yes   
##

## The accuracy of the model is 89%, the sensitivity is 58% and the specificity is 98%.

## The naive model displays 76% accuracy.

predRF\_test = predict(rf\_fit, newdata = test)

confusionMatrix(predRF\_test, test$DonatedMarch, positive = "yes")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction no yes  
## no 170 24  
## yes 1 29  
##   
## Accuracy : 0.8884   
## 95% CI : (0.8397, 0.9265)  
## No Information Rate : 0.7634   
## P-Value [Acc > NIR] : 1.568e-06   
##   
## Kappa : 0.6366   
## Mcnemar's Test P-Value : 1.083e-05   
##   
## Sensitivity : 0.5472   
## Specificity : 0.9942   
## Pos Pred Value : 0.9667   
## Neg Pred Value : 0.8763   
## Prevalence : 0.2366   
## Detection Rate : 0.1295   
## Detection Prevalence : 0.1339   
## Balanced Accuracy : 0.7707   
##   
## 'Positive' Class : yes   
##

# The model for test data has an accuracy of 88%, a sensitivity of 52% and a specificity of 99%.