## Module 4 Assignment 2

### Hackett, Evan

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 0.2.5  
## v tibble 1.4.2 v dplyr 0.7.7  
## v tidyr 0.8.2 v stringr 1.3.1  
## v readr 1.1.1 v forcats 0.3.0

## -- 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

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()  
## )

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

splitting Blood dataset into training/testing

set.seed(1234)  
train.rows = createDataPartition(y = Blood$DonatedMarch, p=0.7, list = FALSE) #70% in training  
train = Blood[train.rows,]   
test = Blood[-train.rows,]

Random forest

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

Check out random forest details

varImp(rf\_fit)

## ranger variable importance  
##   
## Overall  
## Total\_Donated 100.000  
## TotalDonations 38.494  
## Mnths\_Since\_First 7.657  
## Mnths\_Since\_Last 0.000

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, 471, 472, 472, 471, 472, ...   
## Resampling results across tuning parameters:  
##   
## mtry splitrule Accuracy Kappa   
## 2 gini 0.7804790 0.3105144  
## 2 extratrees 0.7880987 0.3133046  
## 3 gini 0.7804790 0.3284588  
## 3 extratrees 0.7747097 0.2923162  
## 4 gini 0.7689768 0.2939497  
## 4 extratrees 0.7727504 0.2903873  
##   
## 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 is the Total\_Donated variable. The Least important variables it the Mnths\_Since\_Last variable.

Predictions

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

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

Confusion matrix

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

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction yes no  
## yes 81 5  
## no 44 394  
##   
## Accuracy : 0.9065   
## 95% CI : (0.8783, 0.93)  
## No Information Rate : 0.7615   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.7117   
## Mcnemar's Test P-Value : 5.681e-08   
##   
## Sensitivity : 0.6480   
## Specificity : 0.9875   
## Pos Pred Value : 0.9419   
## Neg Pred Value : 0.8995   
## Prevalence : 0.2385   
## Detection Rate : 0.1546   
## Detection Prevalence : 0.1641   
## Balanced Accuracy : 0.8177   
##   
## 'Positive' Class : yes   
##

The accuracy of this model is significantly higher than that of the naive model. The accuracy rate of this model is 90.65% which is greater than that of the naive model which has an accuracy rate of 76.15%, this difference is statistically different and we know this as the p value is less than .05.

Predictions on testing set

predRFTest = predict(rf\_fit, test)  
head(predRFTest)

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

Confusion matrix

confusionMatrix(predRFTest, test$DonatedMarch, positive = "yes")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction yes no  
## yes 15 12  
## no 38 159  
##   
## Accuracy : 0.7768   
## 95% CI : (0.7165, 0.8296)  
## No Information Rate : 0.7634   
## P-Value [Acc > NIR] : 0.351547   
##   
## Kappa : 0.2562   
## Mcnemar's Test P-Value : 0.000407   
##   
## Sensitivity : 0.28302   
## Specificity : 0.92982   
## Pos Pred Value : 0.55556   
## Neg Pred Value : 0.80711   
## Prevalence : 0.23661   
## Detection Rate : 0.06696   
## Detection Prevalence : 0.12054   
## Balanced Accuracy : 0.60642   
##   
## 'Positive' Class : yes   
##

The model does not perform as well on the test data set. The accuracy rate has decreaed to 77.68% which is just marginally better than the naive model of 76.34% and the variance is not statistically significant as the p value is greater than .05.