# Ariel Romero

## Parameter Selection and Neural Networks

library(tidyverse)

## -- Attaching packages -------------

## v ggplot2 3.1.0 v purrr 0.3.2   
## v tibble 2.1.1 v dplyr 0.8.0.1  
## v tidyr 0.8.3 v stringr 1.4.0   
## v readr 1.3.1 v forcats 0.4.0

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

library(caret)

## Loading required package: lattice

##   
## Attaching package: 'caret'

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

library(nnet)  
  
parole <- read\_csv("parole.csv")

## Parsed with column specification:  
## cols(  
## male = col\_double(),  
## race = col\_double(),  
## age = col\_double(),  
## state = col\_double(),  
## time.served = col\_double(),  
## max.sentence = col\_double(),  
## multiple.offenses = col\_double(),  
## crime = col\_double(),  
## violator = col\_double()  
## )

parole = parole %>% mutate(male = as\_factor(as.character(male))) %>%  
mutate(male = fct\_recode(male,  
"female" = "0",  
"male" = "1")) %>% mutate(race = as\_factor(as.character(race))) %>%  
mutate(race = fct\_recode(race,  
"white" = "1",  
"other" = "2")) %>% mutate(state = as\_factor(as.character(state))) %>%  
mutate(state = fct\_recode(state,  
"Kentucky" = "2",  
"Louisiana" = "3",  
"Virginia" = "4",  
"other" = "1")) %>% mutate(crime = as\_factor(as.character(crime))) %>%  
mutate(crime = fct\_recode(crime,  
"larceny" = "2",  
"drug-related" = "3",  
"driving-related" = "4",  
"other" = "1")) %>% mutate(multiple.offenses = as\_factor(as.character(multiple.offenses))) %>%  
mutate(multiple.offenses = fct\_recode(multiple.offenses,  
"otherwise" = "0",  
"multiple offenses" = "1")) %>% mutate(violator = as\_factor(as.character(violator))) %>%  
mutate(violator = fct\_recode(violator,  
"non-violator" = "0",  
"violator" = "1"))  
  
set.seed(12345)  
train.rows = createDataPartition(y = parole$violator, p=0.7, list = FALSE)  
train = parole[train.rows,]   
test = parole[-train.rows,]  
  
fitControl = trainControl(method = "cv",   
 number = 10)  
  
nnetGrid <- expand.grid(size = 12, decay = 0.1)  
  
set.seed(1234)  
nnetBasic = train(violator ~ .,   
 parole,  
 method = "nnet",  
 tuneGrid = nnetGrid,  
 trControl = fitControl,  
 verbose = FALSE,  
 trace = FALSE)

nnetBasic

## Neural Network   
##   
## 675 samples  
## 8 predictor  
## 2 classes: 'non-violator', 'violator'   
##   
## No pre-processing  
## Resampling: Cross-Validated (10 fold)   
## Summary of sample sizes: 608, 607, 609, 607, 607, 607, ...   
## Resampling results:  
##   
## Accuracy Kappa   
## 0.8799266 0.3222643  
##   
## Tuning parameter 'size' was held constant at a value of 12  
##   
## Tuning parameter 'decay' was held constant at a value of 0.1

predTrain = predict(nnetBasic,train)  
  
confusionMatrix(predTrain,train$violator)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction non-violator violator  
## non-violator 409 24  
## violator 9 31  
##   
## Accuracy : 0.9302   
## 95% CI : (0.9034, 0.9515)  
## No Information Rate : 0.8837   
## P-Value [Acc > NIR] : 0.0005254   
##   
## Kappa : 0.6149   
##   
## Mcnemar's Test P-Value : 0.0148061   
##   
## Sensitivity : 0.9785   
## Specificity : 0.5636   
## Pos Pred Value : 0.9446   
## Neg Pred Value : 0.7750   
## Prevalence : 0.8837   
## Detection Rate : 0.8647   
## Detection Prevalence : 0.9154   
## Balanced Accuracy : 0.7711   
##   
## 'Positive' Class : non-violator   
##

fitControl2 = trainControl(method = "cv",   
 number = 10)  
  
nnetGrid2 = expand.grid(size = seq(from = 1, to = 12, by = 1),  
 decay = seq(from = 0.1, to = 0.5, by = 0.1))  
  
set.seed(1234)  
nnetBasic2 = train(violator ~ .,   
 parole,  
 method = "nnet",  
 tuneGrid = nnetGrid,  
 trControl = fitControl,  
 verbose = FALSE,  
 trace = FALSE)

nnetBasic2

## Neural Network   
##   
## 675 samples  
## 8 predictor  
## 2 classes: 'non-violator', 'violator'   
##   
## No pre-processing  
## Resampling: Cross-Validated (10 fold)   
## Summary of sample sizes: 608, 607, 609, 607, 607, 607, ...   
## Resampling results:  
##   
## Accuracy Kappa   
## 0.8799266 0.3222643  
##   
## Tuning parameter 'size' was held constant at a value of 12  
##   
## Tuning parameter 'decay' was held constant at a value of 0.1

predTrain2 = predict(nnetBasic2,train)  
  
confusionMatrix(predTrain2,train$violator)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction non-violator violator  
## non-violator 409 24  
## violator 9 31  
##   
## Accuracy : 0.9302   
## 95% CI : (0.9034, 0.9515)  
## No Information Rate : 0.8837   
## P-Value [Acc > NIR] : 0.0005254   
##   
## Kappa : 0.6149   
##   
## Mcnemar's Test P-Value : 0.0148061   
##   
## Sensitivity : 0.9785   
## Specificity : 0.5636   
## Pos Pred Value : 0.9446   
## Neg Pred Value : 0.7750   
## Prevalence : 0.8837   
## Detection Rate : 0.8647   
## Detection Prevalence : 0.9154   
## Balanced Accuracy : 0.7711   
##   
## 'Positive' Class : non-violator   
##

predTest = predict(nnetBasic,train)  
  
confusionMatrix(predTest, train$violator)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction non-violator violator  
## non-violator 409 24  
## violator 9 31  
##   
## Accuracy : 0.9302   
## 95% CI : (0.9034, 0.9515)  
## No Information Rate : 0.8837   
## P-Value [Acc > NIR] : 0.0005254   
##   
## Kappa : 0.6149   
##   
## Mcnemar's Test P-Value : 0.0148061   
##   
## Sensitivity : 0.9785   
## Specificity : 0.5636   
## Pos Pred Value : 0.9446   
## Neg Pred Value : 0.7750   
## Prevalence : 0.8837   
## Detection Rate : 0.8647   
## Detection Prevalence : 0.9154   
## Balanced Accuracy : 0.7711   
##   
## 'Positive' Class : non-violator   
##

predTest2 = predict(nnetBasic2, test)  
  
confusionMatrix(predTest2, test$violator)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction non-violator violator  
## non-violator 178 16  
## violator 1 7  
##   
## Accuracy : 0.9158   
## 95% CI : (0.8687, 0.9502)  
## No Information Rate : 0.8861   
## P-Value [Acc > NIR] : 0.108358   
##   
## Kappa : 0.4174   
##   
## Mcnemar's Test P-Value : 0.000685   
##   
## Sensitivity : 0.9944   
## Specificity : 0.3043   
## Pos Pred Value : 0.9175   
## Neg Pred Value : 0.8750   
## Prevalence : 0.8861   
## Detection Rate : 0.8812   
## Detection Prevalence : 0.9604   
## Balanced Accuracy : 0.6494   
##   
## 'Positive' Class : non-violator   
##

There does appear to be overfitting on both models created in Tasks 2 and 4. Both models give the same results for the testing and training datasets.