# lab7_ch11.R

sathvik

2021-03-06

```
# Chapter 11
# 171EC146: Sathvik S Prabhu

library(caret)
```

```
## Loading required package: lattice
```

```
## Loading required package: ggplot2
```

```
library(C50)
library(irr)
```

```
## Loading required package: lpSolve
```

```
# Creating a simple tuned model
credit <- read.csv("/home/sathvik/EC8/ML/Lab/Lab5/credit.csv")
set.seed(300)
credit$default<-as.factor(credit$default)
m <- train(default ~ ., data = credit, method = "C5.0")
m
```

```
## C5.0
##
## 1000 samples
##   20 predictor
##    2 classes: '1', '2'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 1000, 1000, 1000, 1000, 1000, 1000, ...
## Resampling results across tuning parameters:
##
##   model  winnow  trials  Accuracy   Kappa
##   rules  FALSE    1      0.7027014  0.2920803
##   rules  FALSE   10      0.7243269  0.3453188
##   rules  FALSE   20      0.7294474  0.3499065
##   rules   TRUE    1      0.6923914  0.2744217
##   rules   TRUE   10      0.7263089  0.3420603
##   rules   TRUE   20      0.7360215  0.3558315
##   tree   FALSE    1      0.6951701  0.2677459
##   tree   FALSE   10      0.7372294  0.3271180
##   tree   FALSE   20      0.7395288  0.3333668
##   tree    TRUE    1      0.6946925  0.2680391
##   tree    TRUE   10      0.7364909  0.3303587
##   tree    TRUE   20      0.7420638  0.3427882
```

```
##
## Accuracy was used to select the optimal model using the largest value.
## The final values used for the model were trials = 20, model = tree and winnow
##  = TRUE.
```

```
p <- predict(m, credit)
table(p, credit$default)
```

```
##
## p     1   2
##   1 676  79
##   2  24 221
```

```
head(predict(m, credit))
```

```
## [1] 1 2 1 1 2 1
## Levels: 1 2
```

```
head(predict(m, credit, type = "prob"))
```

```
##           1          2
## 1 0.8720819 0.12791809
## 2 0.3284062 0.67159380
## 3 1.0000000 0.00000000
## 4 0.7563177 0.24368229
## 5 0.4531722 0.54682783
## 6 0.9085110 0.09148904
```

```
# Customizing the tuning process
# . The oneSE function
# chooses the simplest candidate within one standard error of the best performance,
# and tolerance uses the simplest candidate within a user-specified percentage.
ctrl <- trainControl(method = "cv", number = 10,
                     selectionFunction = "oneSE")
grid <- expand.grid(.model = "tree",
                    .trials = c(1, 5, 10, 15, 20, 25, 30, 35),
                    .winnow = "FALSE")
grid
```

```
##   .model .trials .winnow
## 1   tree       1   FALSE
## 2   tree       5   FALSE
## 3   tree      10   FALSE
## 4   tree      15   FALSE
## 5   tree      20   FALSE
## 6   tree      25   FALSE
## 7   tree      30   FALSE
## 8   tree      35   FALSE
```

```
set.seed(300)
m <- train(default ~ ., data = credit, method = "C5.0",metric = "Kappa",
             trControl = ctrl,tuneGrid = grid)
```

```
## Warning in Ops.factor(x$winnow): '!' not meaningful for factors
```

```
m
```

```
## C5.0
##
```

```
## 1000 samples
##   20 predictor
##    2 classes: '1', '2'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 900, 900, 900, 900, 900, 900, ...
## Resampling results across tuning parameters:
##
##   trials  Accuracy  Kappa
##    1      0.708     0.2824182
##    5      0.740     0.3604766
##   10      0.751     0.3604338
##   15      0.752     0.3716050
##   20      0.752     0.3655537
##   25      0.755     0.3775273
##   30      0.756     0.3761998
##   35      0.758     0.3818372
##
## Tuning parameter 'model' was held constant at a value of tree
## Tuning
##  parameter 'winnow' was held constant at a value of FALSE
## Kappa was used to select the optimal model using  the one SE rule.
## The final values used for the model were trials = 5, model = tree and winnow
##  = FALSE.
```

```
# Bagging (Bootstrap aggregating)
library(ipred)
set.seed(300)
mybag <- bagging(default ~ ., data = credit, nbagg = 25) # 25 decision trees
credit_pred <- predict(mybag, credit)
table(credit_pred, credit$default)
```

```
##
## credit_pred   1   2
##           1 699   3
##           2   1 297
```

```
library(caret)
set.seed(300)
ctrl <- trainControl(method = "cv", number = 10)
train(default ~ ., data = credit, method = "treebag",
        trControl = ctrl)
```

```
## Bagged CART
##
## 1000 samples
##   20 predictor
##    2 classes: '1', '2'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 900, 900, 900, 900, 900, 900, ...
## Resampling results:
##
```

```
##   Accuracy  Kappa
##   0.743     0.3543981
```

```
str(svmBag)
```

```
## List of 3
##  $ fit      :function (x, y, ...)
##  $ pred     :function (object, x)
##  $ aggregate:function (x, type = "class")
```

```
svmBag$fit
```

```
## function (x, y, ...)
## {
##     loadNamespace("kernlab")
##     out <- kernlab::ksvm(as.matrix(x), y, prob.model = is.factor(y),
##         ...)
##     out
## }
## <bytecode: 0x55e2b9a8db20>
## <environment: namespace:caret>
```

```
library(kernlab)
```

```
##
## Attaching package: 'kernlab'
```

```
## The following object is masked from 'package:ggplot2':
##
##     alpha
```

```
credit$default<-as.factor(credit$default)
bagctrl <- bagControl(fit = svmBag$fit,
                      predict = svmBag$pred,
                      aggregate = svmBag$aggregate)
set.seed(300)
# svmbag <- train(default ~ ., data = credit, "bag",
#                   trControl = ctrl, bagControl = bagctrl)
# svmbag

# Boosting

# Random Forests
# bagging with random featureselection

# Training
library(randomForest)
```

```
## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```r
set.seed(300)
rf <- randomForest(default ~ ., data = credit)
rf
```

```
##
## Call:
##  randomForest(formula = default ~ ., data = credit)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 4
##
##         OOB estimate of  error rate: 23.2%
## Confusion matrix:
##     1   2 class.error
## 1 648  52  0.07428571
## 2 180 120  0.60000000
```

```r
# Evaluation
library(caret)
ctrl <- trainControl(method = "repeatedcv",
                     number = 10, repeats = 10)
# mtry defines how many features are randomly selected at each split.
grid_rf <- expand.grid(.mtry = c(2, 4, 8, 16))
set.seed(300)
m_rf <- train(default ~ ., data = credit, method = "rf",metric = "Kappa",
              trControl = ctrl, tuneGrid = grid_rf)

grid_c50 <- expand.grid(.model = "tree",
                        .trials = c(10, 20, 30, 40),
                        .winnow = "FALSE")
set.seed(300)
m_c50 <- train(default ~ ., data = credit, method = "C5.0",
               metric = "Kappa", trControl = ctrl,
               tuneGrid = grid_c50)
```

```
## Warning in Ops.factor(x$winnow): '!' not meaningful for factors
```

```r
# Comparing RF and C50
m_rf
```

```
## Random Forest
##
## 1000 samples
##   20 predictor
##    2 classes: '1', '2'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 10 times)
## Summary of sample sizes: 900, 900, 900, 900, 900, 900, ...
## Resampling results across tuning parameters:
##
##   mtry  Accuracy  Kappa
##    2    0.7202    0.09787729
##    4    0.7486    0.27551507
##    8    0.7550    0.32980623
```

```
##   16    0.7601    0.36418906
##
## Kappa was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 16.
```

```
m_c50
```

```
## C5.0
##
## 1000 samples
##   20 predictor
##    2 classes: '1', '2'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 10 times)
## Summary of sample sizes: 900, 900, 900, 900, 900, 900, ...
## Resampling results across tuning parameters:
##
##   trials  Accuracy  Kappa
##   10      0.7381    0.3314105
##   20      0.7463    0.3530474
##   30      0.7491    0.3602113
##   40      0.7553    0.3753119
##
## Tuning parameter 'model' was held constant at a value of tree
## Tuning
##  parameter 'winnow' was held constant at a value of FALSE
## Kappa was used to select the optimal model using the largest value.
## The final values used for the model were trials = 40, model = tree and winnow
##  = FALSE.
```