td_boosting

Tvisha R. Devavarapu

2023-05-06

```
library(tidyverse)
library(dplyr)
library(ggplot2)

library(caret)
library(rpart)
library(rpart.plot)
library(party)
library(partykit)
library(pROC)

library(randomForest)
library(ranger)
library(gbm)
library(pdp)

set.seed(17)
```

1. Load Data

```
load("final_data.RData")
```

2. Train/test split

```
final_data <- final_data %>%
  mutate(binary_recovery_time = as.factor(binary_recovery_time))

# Here, we set the positive class to be low.
# 0 = low = positive class
# 1 = high = negative class
levels(final_data$binary_recovery_time) = c("low", "high")

set.seed(2)
training_rows <- createDataPartition(final_data$binary_recovery_time,
                                     p = 0.8,
                                     list = F)
```

```
train_x <- model.matrix(binary_recovery_time~., final_data %>% select(-id, -recovery_time))[training_rou
train_y <- final_data$binary_recovery_time[training_rows]
test_x <- model.matrix(binary_recovery_time~., final_data %>% select(-id, -recovery_time))[-training_rou
test_y <- final_data$binary_recovery_time[-training_rows]
```

```
training_set <- final_data[training_rows,]
```

Classification: Random Forest with boosting

```
ctrl.c <- trainControl(method = "cv", number = 10)

set.seed(2)
gbm_grid <- expand.grid(
  # upper bound for number of trees
  n.trees = c(50, 100, 200, 500, 1000, 2000),
  # similar to number of splits in the tree
  # number of layers in the tree
  interaction.depth = 1:5,
  shrinkage = c(0.005,0.01,0.015),
  # min obs allowed in a node
  n.minobsinnode = c(1,5))

library(doParallel)
```

```
## Loading required package: foreach
```

```
##
## Attaching package: 'foreach'
```

```
## The following objects are masked from 'package:purrr':
##
##     accumulate, when
```

```
## Loading required package: iterators
```

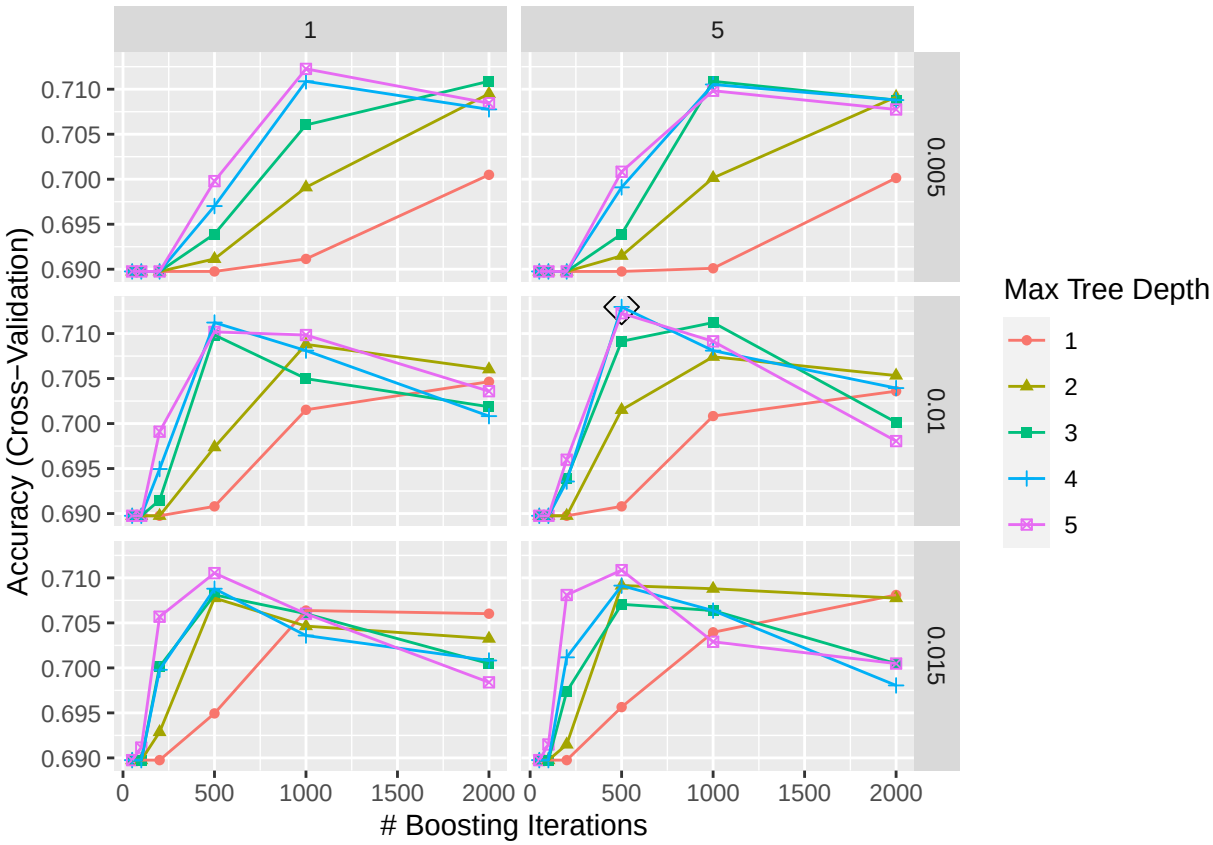
```
## Loading required package: parallel
```

```
Mycluster = makeCluster(detectCores() - 2)
registerDoParallel(Mycluster)

boost_fit <- train(train_x,
                   train_y,
                   method = "gbm",
                   tuneGrid = gbm_grid,
                   trControl = ctrl.c,
                   distribution = "adaboost",
                   #metric = "ROC",
                   verbose = FALSE)

stopCluster(Mycluster)
registerDoSEQ()
```

```
ggplot(boost_fit, highlight = TRUE)
```

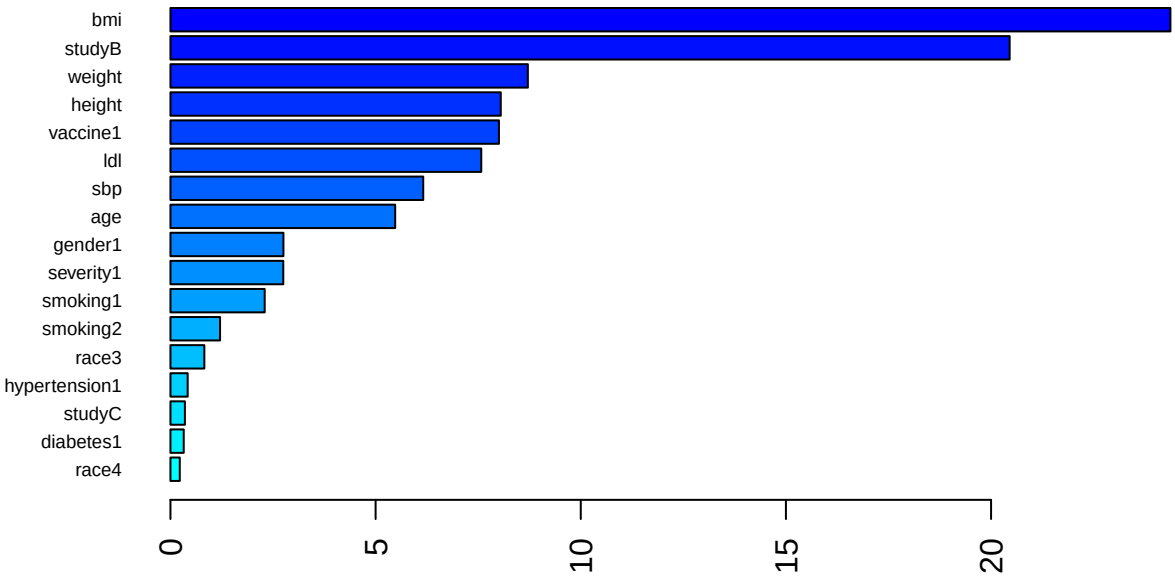



```
boost_fit$bestTune
```

```
##     n.trees interaction.depth shrinkage n.minobsinnode
## 106     500                 4      0.01              5
```

variable importance

```
summary(boost_fit$finalModel,las = 2, cBars = 17, cex.names = 0.6)
```




```
##                          var     rel.inf
## bmi                      bmi 24.37052777
## studyB                studyB 20.45243388
## weight                weight  8.71029731
## height                height  8.04982577
## vaccine1            vaccine1  8.00572107
## ldl                      ldl  7.57484863
## sbp                      sbp  6.16092664
## age                      age  5.47703636
## gender1              gender1  2.75357894
## severity1          severity1  2.74995110
## smoking1            smoking1  2.29700139
## smoking2            smoking2  1.20859525
## race3                  race3  0.82622068
## hypertension1 hypertension1  0.41987580
## studyC                studyC  0.35222659
## diabetes1          diabetes1  0.32341450
## race4                  race4  0.22773196
## race2                  race2  0.03978636
```

```
trboost_prediction <- predict(boost_fit$finalModel, newdata = train_x, type = "response")
```

```
## Using 500 trees...
```

```
# As predictions are made for the positive class, we set a threshold 0.5 and assign anything above to b
trboost_prediction[trboost_prediction > 0.50] = "low"
trboost_prediction[trboost_prediction < 0.50] = "high"

boost_tr_mse <- mean(trboost_prediction != train_y)
boost_tr_mse
```

```
## [1] 0.2690443
```

```
boost_prediction <- predict(boost_fit$finalModel, newdata = test_x, type = "response")
```

```
## Using 500 trees...
```

```
# As predictions are made for the positive class, we set a threshold 0.5 and assign anything above to b
boost_prediction[boost_prediction > 0.50] = "low"
boost_prediction[boost_prediction < 0.50] = "high"

boost_ts_mse <- mean(boost_prediction != test_y)
boost_ts_mse
```

```
## [1] 0.2884882
```