CS 598: Homework 4

Fall 2019, Apoorva (apoorva6)

Contents

Question 1 Tuning Random Forests in Virtual Twins

Personalized medicine draws a lot of attention in medical research. The goal of personalized medicine is to make a tailored decision for each patient, such that his/her clinical outcome can be optimized. Let's consider data modified from the SIDES method. In this dataset, 470 patients and 13 variables are observed. You can download the data from our website. The variables are listed below.

- `Health`: health outcome (larger the better)
- `THERAPY`: 1 for active treatment, 0 for the control treatment
- `TIMFIRST`: Time from first sepsis-organ fail to start drug
- `AGE`: Patient age in years
- `BLLPLAT`: Baseline local platelets
- `blSOFA`: Sum of baseline sofa score (cardiovascular, hematology, hepatorenal, and respiration scores)
- `BLLCREAT`: Base creatinine
- `ORGANNUM`: Number of baseline organ failures
- `PRAPACHE`: Pre-infusion apache-ii score
- `BLGCS`: Base GLASGOW coma scale score
- `BLIL6`: Baseline serum IL-6 concentration
- `BLADL`: Baseline activity of daily living score
- `BLLBILI`: Baseline local bilirubin
- `BEST`: The true best treatment suggested by Doctors. *Response variable*

For each patient, sepsis was observed during their hospital stay. Hence, they need to choose one of the two treatments (indicated by variable `THERAPY`) to prevent further adverse events. After the treatment, their health outcome (`health`) were measured, with a larger value being the better outcome. However, since treatments were assigned randomly, we are not able to suggest better treatment for a new patient. A strategy called Virtual Twins was proposed by Foster et al. (2011) to tackle this problem. We consider a simpler version of the method. We fit two random forests to model the outcome `health`: one model uses all patients who received treatment 1, and another model for all patients who received treatment 0. Denote these two models as $\widehat{f}_1(x)$ and $\widehat{f}_0(x)$, respectively. When a new patient arrives, we use both models to predict the outcomes and see which model gives a better health status. We will suggest the treatment label associated with the model that gives a larger prediction value. In other words, for a new x^*, we compare $\widehat{f}_1(x^*)$ and $\widehat{f}_0(x^*)$ and suggest the better lable. The goal for this question is to select tuning parameters for random forest such that it will suggest the best treatment for a patient. Perform the following:

- Randomly split the data into 75% for training and 25% for testing.
- For the training data, fit the virtual twins model and then use the testing data to suggest the best treatment.
  - You should not use the variable `BEST` when fitting the models
  - Pick three different `mtry` values and three different `nodesize`, leave all other tuning parameters as default
  - After predicting the best treatment in the testing data, compare it to the truth `BEST`

```
library(randomForest)
# Read csv
set.seed(1)
sepsis <- read.csv("Sepsis.csv")

#Diving the data based on test and control groups
treatment_1 <- subset(sepsis, THERAPY ==1)
treatment_0 <- subset(sepsis, THERAPY ==0)

#Sampling to split into test and train
sample_1 <- sample(1:nrow(treatment_1),0.75 * nrow(treatment_1))
sample_0 <- sample(1:nrow(treatment_0),0.75 * nrow(treatment_0))
train_1 <- treatment_1[sample_1,-3]
train_0 <- treatment_0[sample_0,-3]
test <- sepsis[-c(train_0$X,train_1$X),-c(1,3)]


error_table <- c()
for(mtry in c(2,5,7))
{
  test.err <- c()
  for (node in c(3,4,5))
  {
    model_1 <- randomForest(Health ~ .- BEST , data = train_1[,-1], mtry=mtry, nodesize=node)
    pred_1 <- predict(model_1,test[,-1])

    model_0 <- randomForest(Health ~ .- BEST , data = train_0[,-1], mtry=mtry, nodesize=node)
    pred_0 <- predict(model_0,test[,-1])
    test_best_pred <- ifelse(pred_0 >pred_1,0,1)

    test.err <- c(test.err,  mean((test$BEST-test_best_pred)^2))
  }
  error_table <- rbind(error_table,test.err)
}

colnames(error_table) <- c("Node=3","Node=4","Node=5")
rownames(error_table) <- c("mtry=2","mtry=5","mtry=7")
accuracy <- (1-error_table)*100
accuracy
```

```
##          Node=3   Node=4   Node=5
## mtry=2 83.19328 85.71429 82.35294
## mtry=5 83.19328 82.35294 83.19328
## mtry=7 83.19328 82.35294 83.19328
```

- Repeat this entire process 100 times and average the prediction errors
- Summarize your results, including the model performance and the effect of tuning parameters. Intuitively demonstrate them.

```
error_avg <- matrix(0, nrow=3, ncol=3)
for (i in 1:100)
    {
  set.seed(i)
    #Sampling to split into test and train
    sample_1 <- sample(1:nrow(treatment_1),0.75 * nrow(treatment_1))
    sample_0 <- sample(1:nrow(treatment_0),0.75 * nrow(treatment_0))
    train_1 <- treatment_1[sample_1,-3]
    train_0 <- treatment_0[sample_0,-3]
    test <- sepsis[-c(train_0$X,train_1$X),-c(1,3)]

      error_table <- c()
      for(mtry in c(2,5,7))
      {
         test.err <- c()
         for (node in c(3,4,5))
         {

         model_1 <- randomForest(Health ~ .- BEST , data = train_1[,-1], mtry=mtry, nodesize=node)
         pred_1 <- predict(model_1,test[,-1])

         model_0 <- randomForest(Health ~ .- BEST , data = train_0[,-1], mtry=mtry, nodesize=node)
         pred_0 <- predict(model_0,test[,-1])
         test_best_pred <- ifelse(pred_0 >pred_1,0,1)

         test.err <- c(test.err,  mean((test$BEST-test_best_pred)^2))
        }
      error_table <- rbind(error_table,test.err)
      }
      error_avg <- (error_avg + error_table)/2

}
colnames(error_avg) <- c("Node=3","Node=4","Node=5")
rownames(error_avg) <- c("mtry=2","mtry=5","mtry=7")
accuracy <- (1-error_avg)*100
accuracy
```

```
##           Node=3   Node=4   Node=5
## mtry=2 73.83028 74.40730 75.89141
## mtry=5 76.07915 77.61505 76.71730
## mtry=7 76.12935 77.21351 78.81301
```

From the above set of values of tuning parameters, it appears that mtry =7 and node=5 produces the highest accuracy. We will use these set of tuning pararmeters to build our final twin model below.

Question 2 Second Step in Virtual Twins

The second step in a virtual twins model is to use a single tree model (CART) to describe the choice of the best treatment. Perform the following: * Based on your optimal tuning parameter, fit the Virtual Twins model described in Question 1. Again, you should not use the `BEST` variable. * For each subject, obtain the predicted best treatment of the training data itself * Treating the label of best treatment as the outcome, and fit a single tree model to predict it. Be careful which variables should be removed from this model fitting. * Consider tuning the tree model using the cost-complexity tuning.

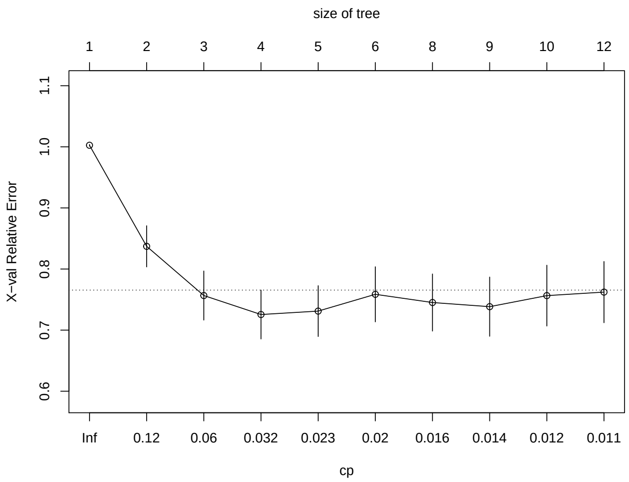
```
# Vitual Twins model on entire data
model1 <- randomForest(Health ~ .  , data = treatment_1[,-c(1,3,15)] , mtry=7, nodesize=5)
pred_sepsis_1 <- predict(model1,sepsis[,-c(1,3,15)])

model0 <- randomForest(Health ~ . , data = treatment_0[,-c(1,3,15)] , mtry=7, nodesize=5)
pred_sepsis_0 <- predict(model0,sepsis[,-c(1,3,15)])
```

```
best_pred_treatment <- ifelse(pred_sepsis_0 >pred_sepsis_1,0,1)

sepsis_1 <- cbind(sepsis,best_pred_treatment)

#Building single tree using our prediction from Twin model as response
library(rpart)
library(rpart.plot)
tree_model_CART <- rpart(best_pred_treatment ~ . , data = sepsis_1[,-c(1,2,3,15)])
plotcp(tree_model_CART)
```



```
#View cp values at every node split
printcp(tree_model_CART)
```

```
##
## Regression tree:
## rpart(formula = best_pred_treatment ~ ., data = sepsis_1[, -c(1,
##     2, 3, 15)])
##
## Variables actually used in tree construction:
## [1] AGE      BLADL    BLIL6    BLLBILI  BLLCREAT PRAPACHE TIMFIRST
##
## Root node error: 117.49/470 = 0.24998
##
## n= 470
##
##           CP nsplit rel error  xerror      xstd
## 1  0.168298      0   1.00000 1.00255 0.0013241
## 2  0.083801      1   0.83170 0.83715 0.0336995
## 3  0.042574      2   0.74790 0.75661 0.0401631
## 4  0.023459      3   0.70533 0.72553 0.0400058
## 5  0.022927      4   0.68187 0.73113 0.0415846
## 6  0.017899      5   0.65894 0.75861 0.0450771
## 7  0.014963      7   0.62314 0.74518 0.0467234
## 8  0.012294      8   0.60818 0.73839 0.0484581
## 9  0.012123      9   0.59589 0.75646 0.0497639
## 10 0.010000     11   0.57164 0.76223 0.0500988
```

```
#Pruning the tree by using the min value of cp error
tree_model_CART <- prune(tree_model_CART,cp= 0.03)
```

```r
p1 <- predict(tree_model_CART,sepsis_1[,-c(1,2,3,15)])

#Checking accuracy of the single CART tree
acc <- 1 - (mean((sepsis$BEST-p1)^2))
acc
```

```
## [1] 0.8689631
```

The cp value for pruning is chosen to be between the range of cp that has the lowest xerror. Since there are are more than 10 variables considered to build the tree. The accuracy of the model on training data after applying the twin models and single CART tree is 0.8689631