BIOST 546 Homework 4

Ivy Zhang

2/28/2022

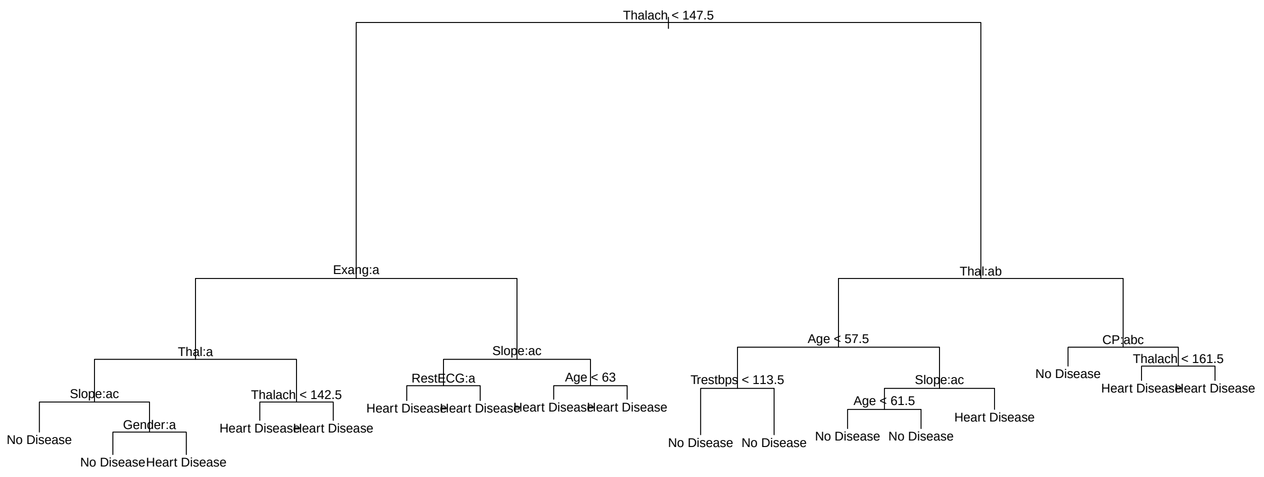
Problem 1

(a)

Disease	Number of Observation
No Disease	171
Heart Disease	200

In this dataset, we have a total sample size of 371 and 12 predictors. The number of observations in each class is shown in the previous table. The process of dividing data in to train set and test set will be shown in the code appendix.

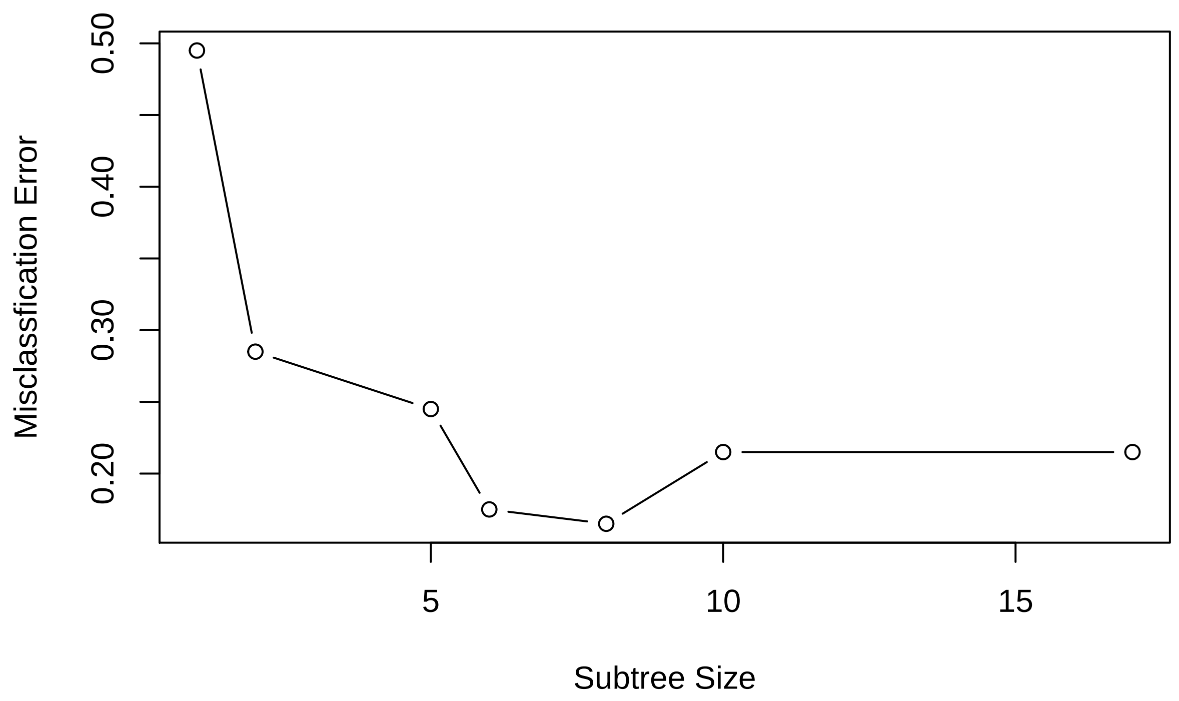
(b)

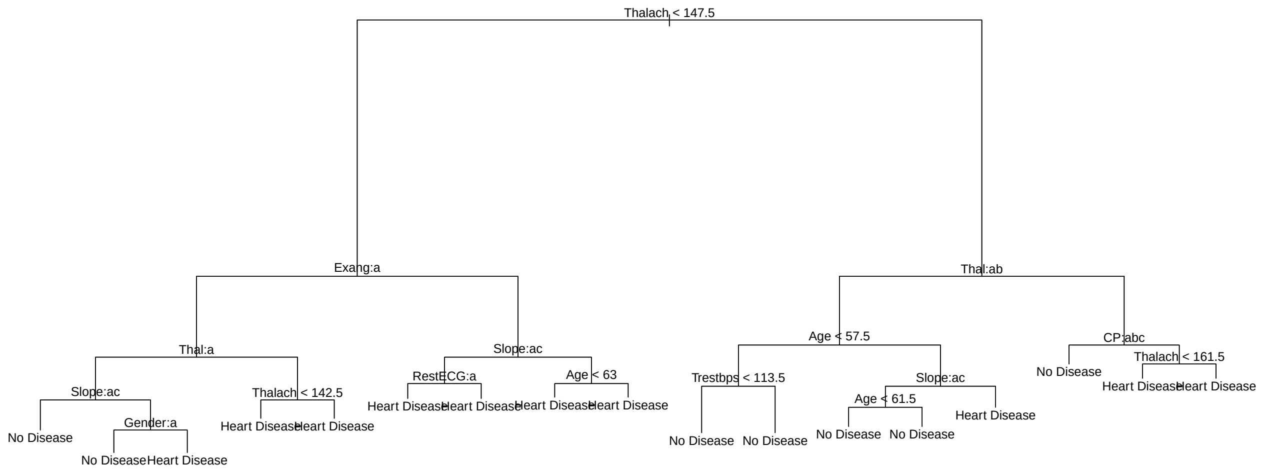


(c)

Based on the calculation, the training set misclassification error is 0.12 and the test set misclassification error is 0.228. We can see the test misclassification error is much higher than the trainning misclassification error. We may need to put overfitting into our concern. Both the misclassfication error are relatively high.

(d)





Based on the cross validation, the best subtree size will be 8.

(e)

Based on the calculation, the training set misclassification error is 0.155 and the test set misclassification error is 0.263. We can see training misclassification error increase and test misclassification error also increase if we compare the overgrown tree performance and prune tree performance. The increasing of the training misclassification error is because we prune the overgrown tree's branches, that lead to reduce the accuracy in predicting training set. The test set accuracy is also increase may due to the same reason. It may also due to the randomness. The prune tree may not do better in accuracy but can reducing the variance as an trade-off.

(f)

Based on the calculation, the training set misclassification error is 0 and the test set misclassification error is 0.228. We can see the bag model is giving a training miscalssification error of 0 and smaller testing mirror compared to the prune tree model. We may consider about the overfitting problem in doing bag model. Usually, bagged trees reduced the variance of the individual trees by using bootstrapping to average, and it also reducing the testing set misclassification error.

(g)

Based on the calculation, the training set misclassification error is 0 and the test set misclassification error is 0.211. We can see the random forest model with m = 4 (1/3 of the total number of predictors) is also giving a training set misclassification error of 0 but a relatively lower testing misclassification error compared to the bagged model. This method can further reduce the variance by "decorrelate" the trees.

(h)

The source of the randomness is from the bootstrapping process. Both the bagged tree and the random forest are using bootstrapping to split the training dataset and building model using each different training datasets and gave us different estimation, leading to the randomness. In addition, random forest also need to be randomly choosing predictors of the models, therefore lead further randomness to the fitting the models.

(i)

Based on the calculation, the training set misclassification error is 0 and the test set misclassification error is 0.193. The training set misclassification error is still 0 and the test set misclassification error seems to be the lowest among all the previous models.

Problem 2

(a)

Process will be shown in the appendix code.

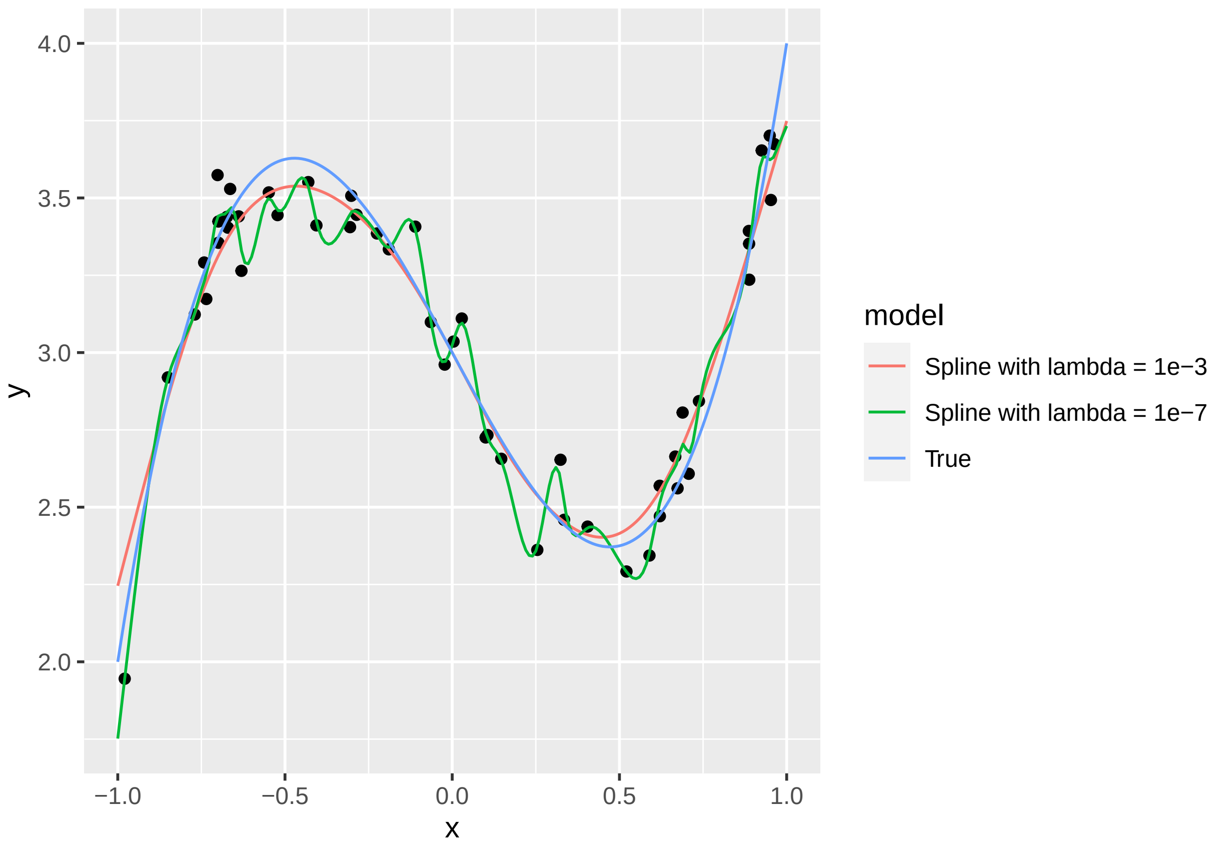
(b)

Process will be shown in the appendix code.

(c)

Process will be shown in the appendix code

(d)

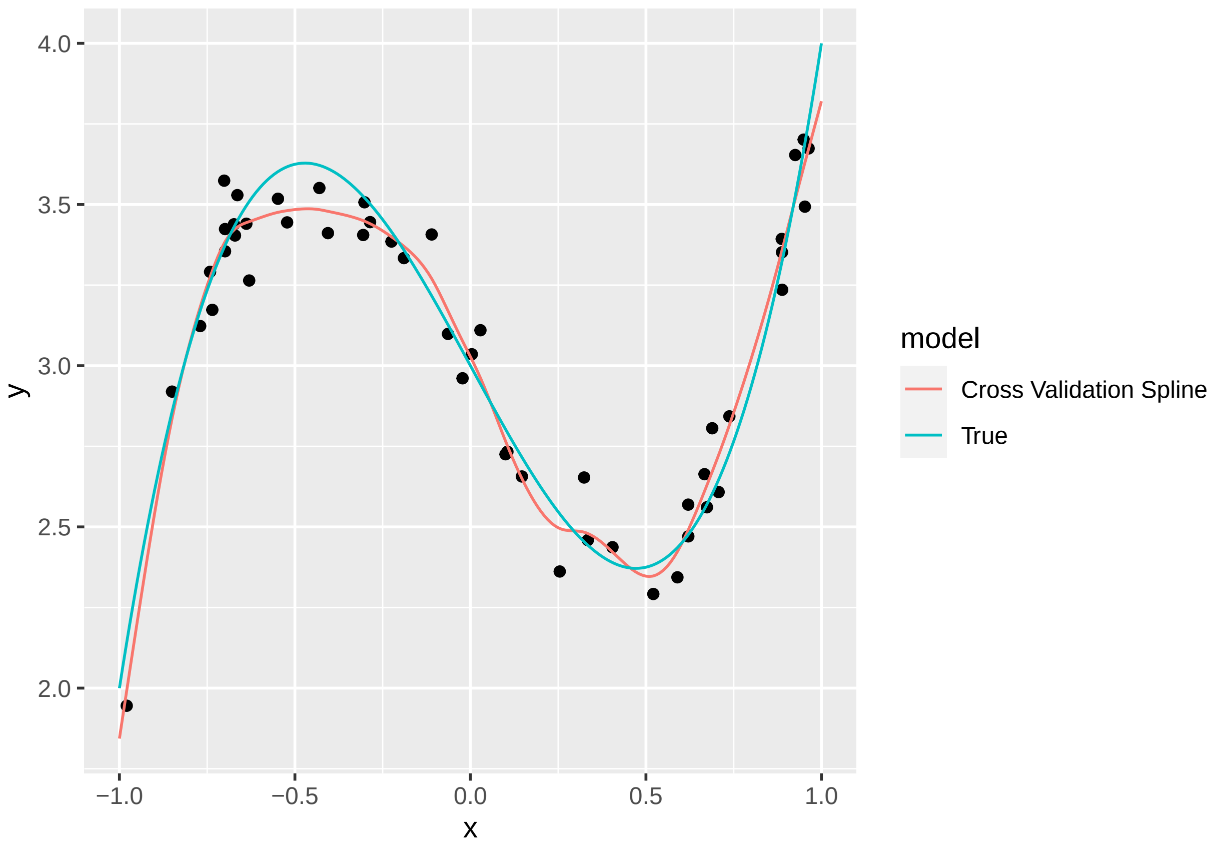


As we can see when the lambda become smaller(when lambda = 1e-7), the model become very flexible, it fitted with the point too much instead of showing the general trend of how y varies with x, and lead to the overfitting problem. When lambda equals to 1e-3, the predicted model is capturing the general shape and trend of the true function is. Therefore, we can see lambda value is directly associated with the flexiblitity of the model.

(e)

Process will be shown in the appendix code. The best lambda is 2.8789099×10^{-5}.

(f)



(g)

It seemed that the variance of the prediction of X = 0 using the smoothing spline models with lambda equals to 1e-3 in B = 1000 datasets bootstrapped is 0.00104 and will be 0.00432 using the lambda value equals to 1e-7. The variance of model using smaller lambda value is having larger variance in prediction value. It may due to the smaller lambda model have higher flexibility and may suffer with the overfitting problem.

```r
knitr::opts_chunk$set(echo = FALSE, warning = F, message = F)
#1a
library(tree)
library(dplyr)
library(knitr)
load("/Users/ivyyuezhang/Desktop/R hw/heart.RData")
n = nrow(full)
n_p = ncol(full) - 1
n_m = full%>%group_by(Disease)%>%summarize(`Number of Observation` = n())
kable(n_m)
set.seed(2)
train_ids = sample(nrow(full), 200)
test_ids = seq(nrow(full))[-train_ids]
full$Disease = as.factor(full$Disease)
data_all = full
data_train = full[train_ids,]
data_test = full[test_ids,]
#1b
tree.med<-tree(Disease~.,data_train)
plot(tree.med)
text(tree.med)
train_pre = predict(tree.med, newdata = data_train, type = "class")
test_pre = predict(tree.med, newdata = data_test, type = "class")
train_mis = mean(train_pre != data_train$Disease)
test_mis = mean(test_pre != data_test$Disease)
#1d
set.seed(2)
cv.med=cv.tree(tree.med,FUN=prune.misclass)
cv.med$mis= cv.med$dev/200
plot(cv.med$size,cv.med$mis,type="b", xlab = "Subtree Size", ylab = "Misclassfication Error")
best_size = cv.med$size[which.min(cv.med$dev)]
prune.med<-prune.tree(tree.med,best=best_size)
plot(tree.med) #Will draw prune trees by hand but plotting the overgrown tree
text(tree.med)
#1e
prune_train_pre = predict(prune.med, newdata = data_train, type = "class")
prune_test_pre = predict(prune.med, newdata = data_test, type = "class")
prune_train_mis = mean(prune_train_pre != data_train$Disease)
prune_test_mis = mean(prune_test_pre != data_test$Disease)
#1f
library(randomForest)
set.seed(2)
bag.med<-randomForest(Disease~.,data=data_train, mtry = 12, importance=TRUE)
bag_train_pre = predict(bag.med, newdata = data_train, type = "class")
bag_test_pre = predict(bag.med, newdata = data_test, type = "class")
bag_train_mis = mean(bag_train_pre != data_train$Disease)
bag_test_mis = mean(bag_test_pre != data_test$Disease)
#1g
set.seed(2)
rf.med<-randomForest(Disease~.,data=data_train,importance=TRUE, mtry = 4)
rf_train_pre = predict(rf.med, newdata = data_train, type = "class")
rf_test_pre = predict(rf.med, newdata = data_test, type = "class")
rf_train_mis = mean(rf_train_pre != data_train$Disease)
```

```r
rf_test_mis = mean(rf_test_pre != data_test$Disease)
#1i
library(gbm)
data_train$Disease = ifelse(data_train$Disease == "No Disease", 0, 1 )
data_test$Disease = ifelse(data_test$Disease == "No Disease", 0, 1 )
set.seed(2)
boost.med<-gbm(Disease~.,data=data_train,distribution="bernoulli",
               n.trees=500,interaction.depth=2,shrinkage = 0.1)
boost_train_pre = predict(boost.med, newdata = data_train, type = "response")
boost_test_pre = predict(boost.med, newdata = data_test, type = "response")
boost_train_label = ifelse(boost_train_pre<=0.5, 0, 1)
boost_test_label = ifelse(boost_test_pre <= 0.5, 0, 1)
boost_train_mis = mean(boost_train_label != data_train$Disease)
boost_test_mis = mean(boost_test_label != data_test$Disease)
#2a
set.seed(2)
X = runif(50, min = -1, max= 1)
epislon = rnorm(50, mean = 0, sd = 0.1)
#2b
Y = 3 - 2 * X + 3 * X^3 + epislon
#2c
library(splines)
lambda_3_model = smooth.spline(X, Y, lambda = 1e-3)
lambda_7_model = smooth.spline(X, Y, lambda = 1e-7)
#2d
library(ggplot2)
new_X  = seq(-1, 1, by = 0.01)
graph_matrix = data.frame(x = new_X, y = 3-2*new_X+3*new_X^3, model = "True")
graph_matrix = rbind(graph_matrix, data.frame(x = new_X, y = predict(lambda_3_model, new_X)$y,
                          model = "Spline with lambda = 1e-3"))
graph_matrix = rbind(graph_matrix, data.frame(x = new_X, y = predict(lambda_7_model, new_X)$y,
                          model = "Spline with lambda = 1e-7"))
graph = data.frame(x = X, y = Y)
ggplot(aes(x = x, y = y),data = graph)+
  geom_point()+
  geom_line(aes(x = x, y = y, col = model), data = graph_matrix)

#2e
cv.model = smooth.spline(X, Y,cv=TRUE)
#2f
graph_matrix = data.frame(x = new_X, y = 3-2*new_X+3*new_X^3, model = "True")
graph_matrix = rbind(graph_matrix, data.frame(x = new_X, y = predict(cv.model, new_X)$y,
                          model = "Cross Validation Spline"))
graph = data.frame(x = X, y = Y)
ggplot(aes(x = x, y = y),data = graph)+
  geom_point()+
  geom_line(aes(x = x, y = y, col = model), data = graph_matrix)

#2g
library(boot)
fit.fn = function (data,index) {
  data_boot = data[index,]
  lambda_3_model_b =with(data_boot, smooth.spline(x, y, lambda = 1e-3))
  lambda_7_model_b = with(data_boot, smooth.spline(x, y, lambda = 1e-7))
  return(c(predict(lambda_3_model_b,0)$y, predict(lambda_7_model_b,0)$y))
}
boot.out = boot(graph, fit.fn, R=1000)$t
var_3 = var(boot.out[,1])
var_7 = var(boot.out[,2])
```