5054 Assignment 2

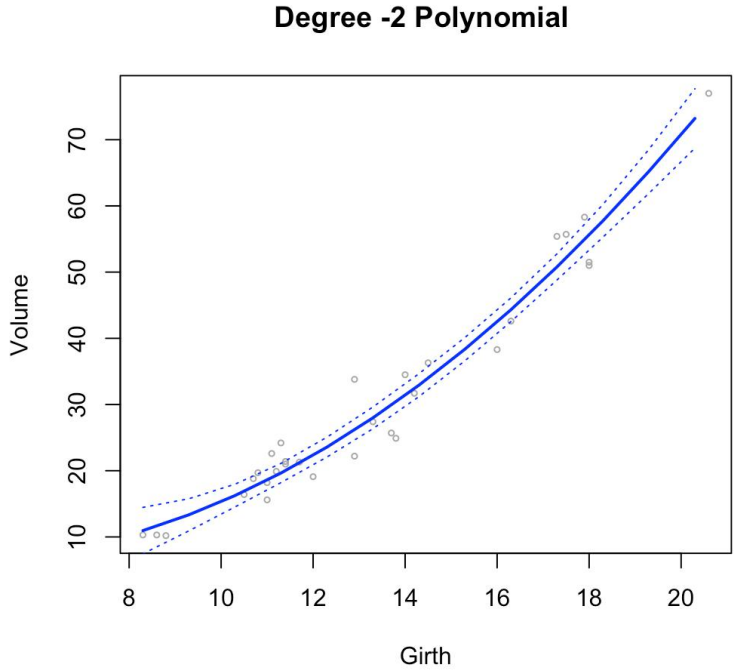
ZHANG Juntao - 20908272

November 17, 2022

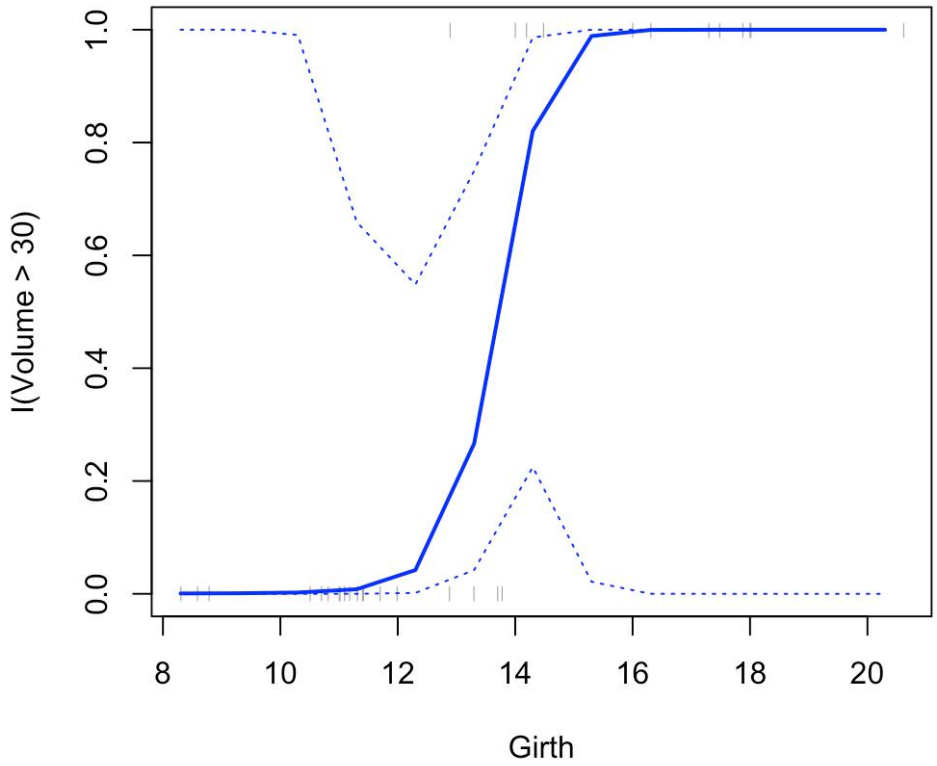
Problem 1: Investigation of the Diameter, Height and Volume for Black Cherry Trees

1. Fit four polynomial models (degree=1,2,3,4) of Girth and Volume, and choose the best one model with the largest Adjust R-square value.
Firstly, we fit these four models, get their Adjust R-square values and compare:
Adjust R-square value of polynomial model with degree=1: 0.9331
Adjust R-square value of polynomial model with degree=2: 0.9588
Adjust R-square value of polynomial model with degree=3: 0.9586
Adjust R-square value of polynomial model with degree=4: 0.9577

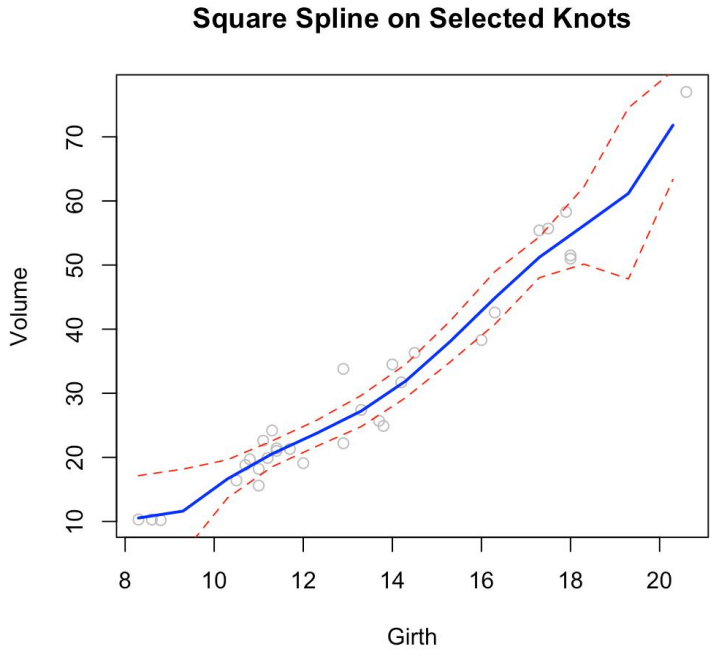
From the above results, we can get that polynomial model with degree=2 has the best performance, i.e. has the largest Adjust R-square value. So we choose this model to predict, plot the polynomial function of thismodel and also the confifidence bands with ±2 standard error. As the follow figure shows:



2. Use a polynomial logistic regression model with degree=2 to predict whether the Volume is larger or not than 30 using the variable Girth. Plot the function P(Volume > 30) with respect to Girth and the confidence bands with ±2 standard error.
Get the result as following figure shows:



3. Fit a regression spline with degree=2 to predict the Volume using the variable Girth at knots 10, 14, 18. Plot the function and also the confidence bands with ±2 standard error.
Get the following reuslut:

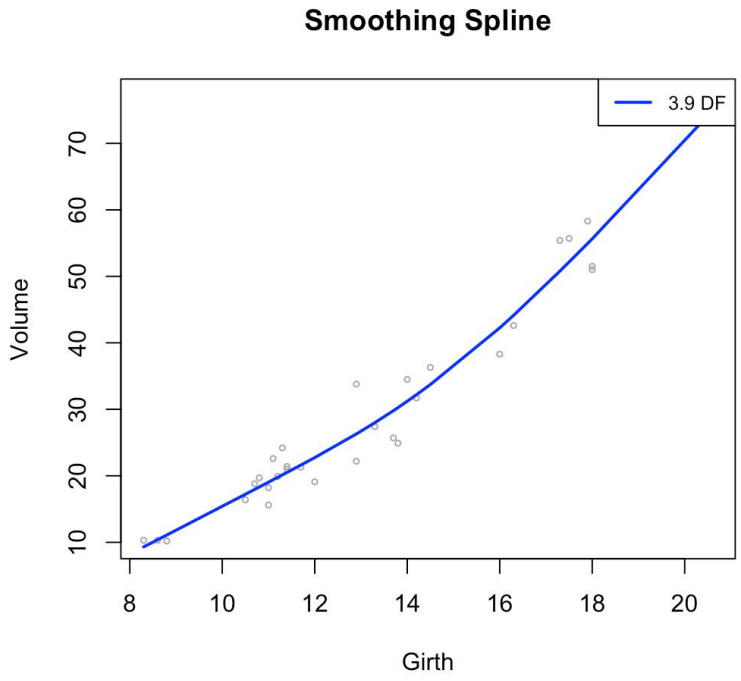


4. Fit a smoothing spline to predict the Volume using Girth, choosing the smoothing level by Cross-Validation. And get the result:

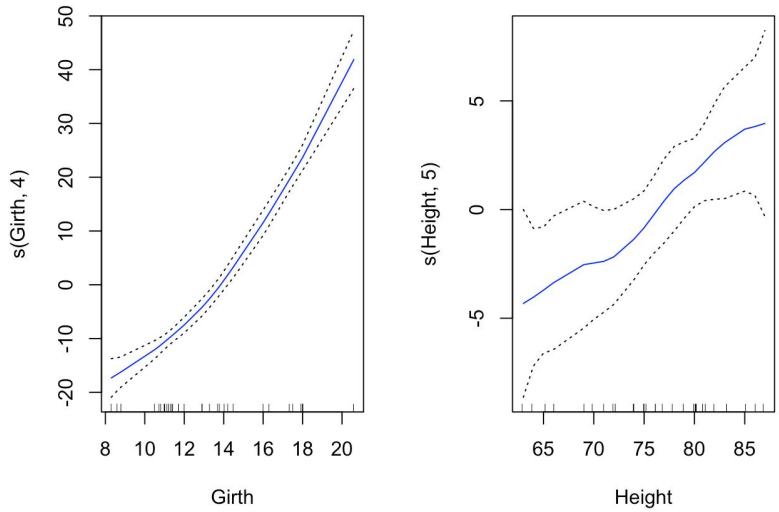
```
> fit2=smooth.spline(Girth,Volume,cv=TRUE)  # select the smoothness level by cross- validation;
Warning message:
In smooth.spline(Girth, Volume, cv = TRUE) :
  用有重复的'x'来做交叉验证好象不太对
> fit2$df
[1] 3.87138
> lines(fit2,col="blue",lwd=2)
> legend("topright",legend=c("3.9 DF"),col=c("blue"),lty=1,lwd=2,cex=.8)
```

So we choose 3.87138 as the smoothing level from Cross-Validation, i.e. the degree of freedom we used is 3.87138

Then, plot the function and get the following figure:

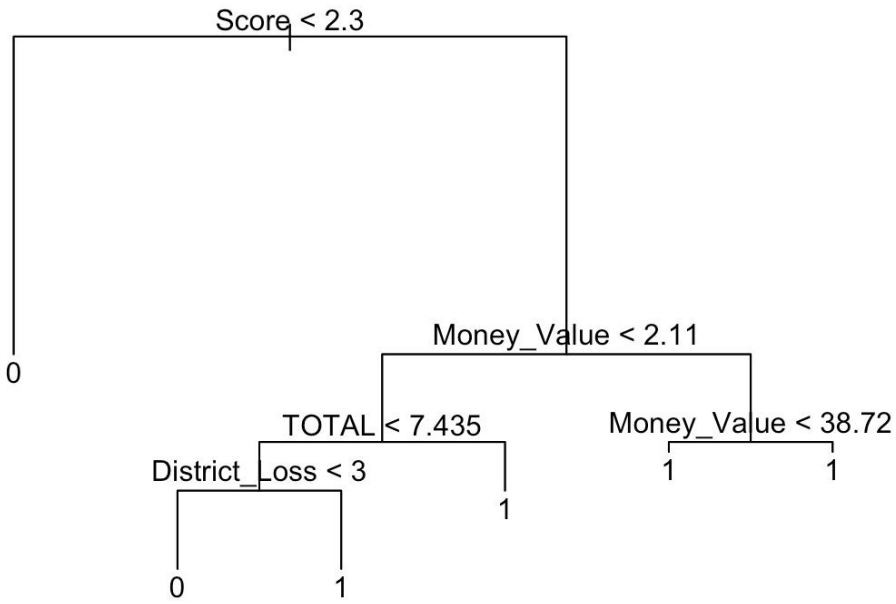


5. Use Girth and Height to predict the Volume by a GAM, where the individual function on Girth is a smoothing spline with df=4 and the function on Height is a smoothing spline with df=5. Plot the functions and the confifidence bands, and get the results as following figure shows:



Problem 2: Audit Risk

1. Use the train dataset to fit a classifification tree, and plot this tree, as follow figure shows:



Then report the training error, the training error is 0.06944 as following shows.

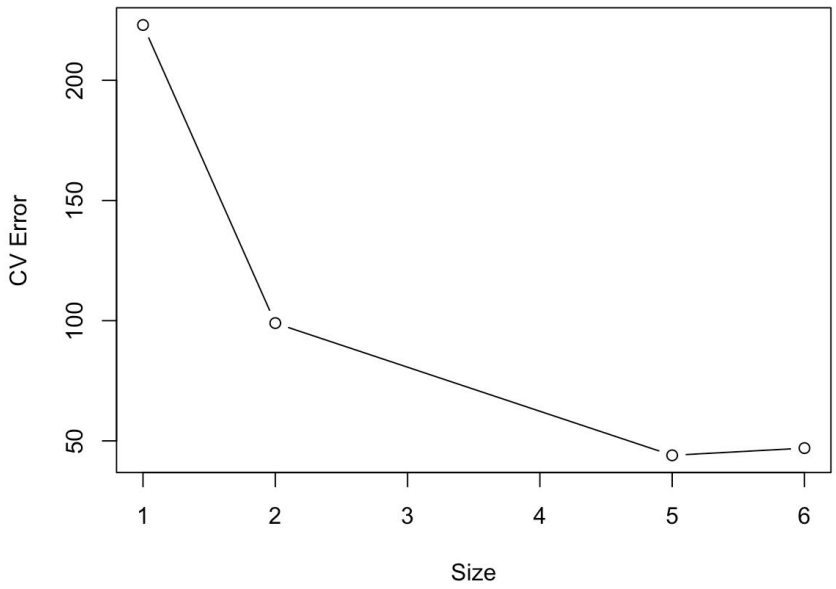
```
> summary(tree.audit_train)

Classification tree:
tree(formula = Risk ~ ., data = audit_train)
Variables actually used in tree construction:
[1] "Score"         "Money_Value"   "TOTAL"         "District_Loss"
Number of terminal nodes:  6
Residual mean deviance:  0.486 = 277 / 570
Misclassification error rate: 0.06944 = 40 / 576
```

After that, test the performance on the test dataset, the confusion matrix is as follow figure shows.

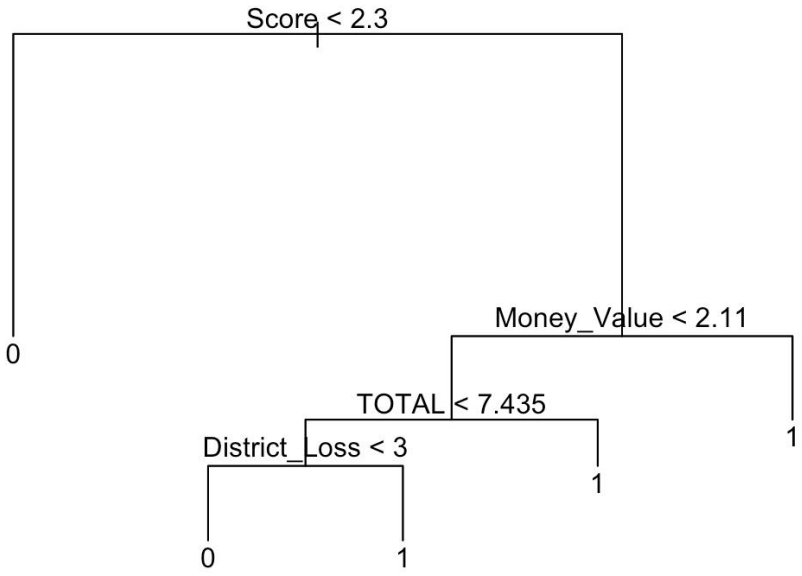
```
> table(p1, audit_test$Risk)

p1    0   1
  0 105   5
  1   6  83
```

From the above figure, we can get the testing error is: (5+6)/199=0.05528

2. Use CV to prune the tree in Step 1 on the train dataset. Plot the train error versus the tree size:



As we can see, the CV error is the smallest when tree size is 5. So prune the tree to obtain the five-node tree, and plot the pruned tree which has the best train error, as follow figure shows:



Use this pruned tree to predict in test dataset, and get the result as follow figure shows. We can get that the test error is: (5+6)/199 = 0.05528, which is same with the original tree.

tree.pred	0	1
0	105	5
1	6	83

3. Use random forest on the train dataset to build a classifier to predict the risk where setting m=13 and ntree=25. Report the training error:

```
> rf.audit_train=randomForest(Risk~.,data=audit_train,mtry=13,ntree=25,na.action=na.omit)
> rf.audit_train

Call:
 randomForest(formula = Risk ~ ., data = audit_train, mtry = 13,      ntree = 25, na.action = na.omit)
               Type of random forest: classification
                     Number of trees: 25
No. of variables tried at each split: 13

        OOB estimate of  error rate: 9.03%
Confusion matrix:
    0   1 class.error
0 330  23  0.06515581
1  29 194  0.13004484
```

As we can see, the total training error is: (23+29)/576 = 0.09028

4. Repeat Step 3 with five different choices m = 8, 12, 14, 16, 18 and choose the one with smallest mis-classification error on the train dataset.
Setting random seed to guarantee the running result is same every time.

(1) when m=8, the total training error is: (21+26)/576 = 0.08159722

```
> set.seed(7)
> rf.audittrain1=randomForest(Risk~.,data=audit_train, mtry=8,ntree=25,na.action=na.omit)
> rf.audittrain1

Call:
 randomForest(formula = Risk ~ ., data = audit_train, mtry = 8,      ntree = 25, na.action = na.omit)
               Type of random forest: classification
                     Number of trees: 25
No. of variables tried at each split: 8

        OOB estimate of  error rate: 8.16%
Confusion matrix:
    0   1 class.error
0 332  21  0.05949008
1  26 197  0.11659193
```

(2) when m=12, the total training error is: (26+29)/576 = 0.09548611

```
> set.seed(7)
> rf.audittrain2=randomForest(Risk~.,data=audit_train, mtry=12,ntree=25,na.action=na.omit)
> rf.audittrain2

Call:
 randomForest(formula = Risk ~ ., data = audit_train, mtry = 12,      ntree = 25, na.action = na.omit)
               Type of random forest: classification
                     Number of trees: 25
No. of variables tried at each split: 12

        OOB estimate of  error rate: 9.55%
Confusion matrix:
    0   1 class.error
0 327  26  0.07365439
1  29 194  0.13004484
```

(3) when m=14, the total training error is: (27+29)/576 = 0.09722222

```
> set.seed(7)
> rf.audittrain3=randomForest(Risk~.,data=audit_train, mtry=14,ntree=25,na.action=na.omit)
> rf.audittrain3

Call:
 randomForest(formula = Risk ~ ., data = audit_train, mtry = 14,      ntree = 25, na.action = na.omit)
               Type of random forest: classification
                     Number of trees: 25
No. of variables tried at each split: 14

        OOB estimate of  error rate: 9.72%
Confusion matrix:
    0   1 class.error
0 326  27  0.07648725
1  29 194  0.13004484
```

(4) when m=16, the total training error is: (25+25)/576 = 0.08680556

```
> set.seed(7)
> rf.audittrain4=randomForest(Risk~.,data=audit_train, mtry=16,ntree=25,na.action=na.omit)
> rf.audittrain4

Call:
 randomForest(formula = Risk ~ ., data = audit_train, mtry = 16,      ntree = 25, na.action = na.omit)
               Type of random forest: classification
                     Number of trees: 25
No. of variables tried at each split: 16

        OOB estimate of  error rate: 8.68%
Confusion matrix:
    0   1 class.error
0 328  25  0.07082153
1  25 198  0.11210762
```

(5) when m=18, the total training error is: (33+29)/576 = 0.1076389

```
> set.seed(7)
> rf.audittrain5=randomForest(Risk~.,data=audit_train, mtry=18,ntree=25,na.action=na.omit)
> rf.audittrain5

Call:
 randomForest(formula = Risk ~ ., data = audit_train, mtry = 18,      ntree = 25, na.action = na.omit)
               Type of random forest: classification
                     Number of trees: 25
No. of variables tried at each split: 18

        OOB estimate of  error rate: 10.76%
Confusion matrix:
    0   1 class.error
0 324  29  0.08215297
1  33 190  0.14798206
```

From the above results, we can see that when m=8, the model has the smallest mis-classification error on the train dataset, which is 0.08159722

Then test this model's (m=8, n=25) performance on the test dataset.

```
> yhat_test1.rf = predict(rf.audittrain1,newdata=audit_test,type="class")
> table(yhat_test1.rf,audit_test$Risk)

yhat_test1.rf   0   1
            0 108   6
            1   3  82
```

As we can see, the testing error is: (3+6)/199 = 0.04522613

5. Compare the above methods:

Decesion tree and pruned decesion tree have same testing error in test dataset, and random forest with m=8, n=25 has smaller testing error in test dataset. This shows that random forest models has more flexibility than decesion tree, actually random forest is a voting system consist of a sequence of decesion trees. Through selecting parameters of random forest resonablly, we can get better prediction and less overfitting effect than decesion tree.

Also, the running result of random forest can be different each time, which caused by random sampling. So I set random seed in my code to guarantee the running result is same each time.