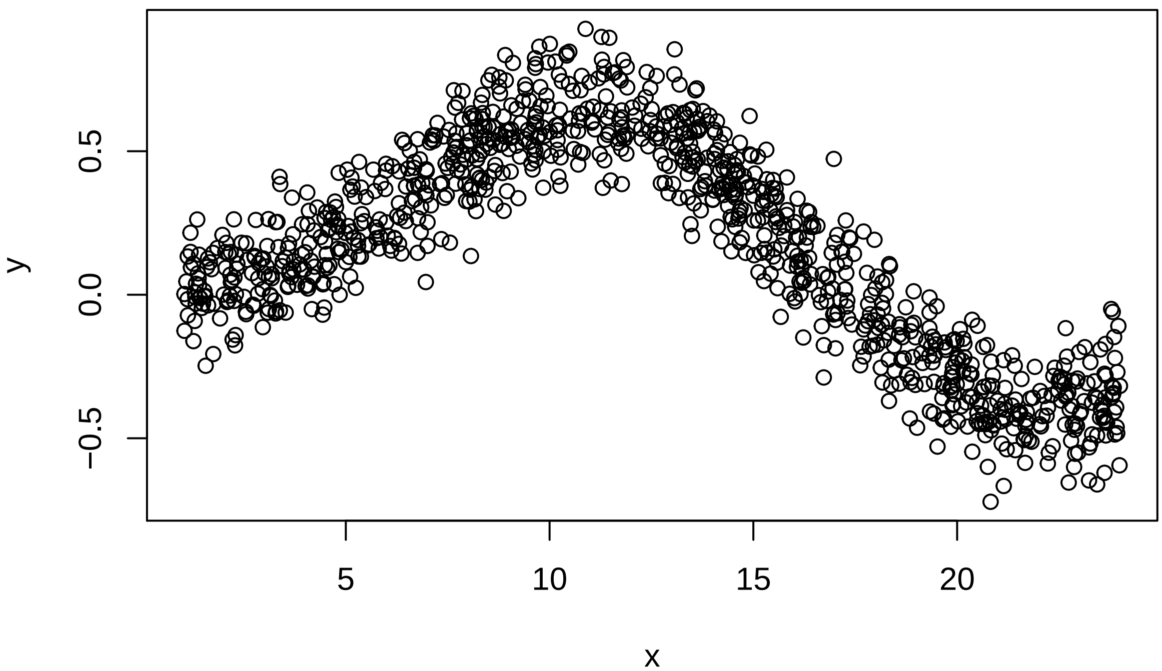
HW4: Chapter 8

Marshall Tuck

8/2/2020

1. Read in dataset hump1000.csv.

```
#Set seed
set.seed(42)

#Read in data, output plot
df3<- read.table("hump1000.csv", header=TRUE, sep=',')
plot(df3$x, df3$y, xlab="x", ylab="y")
```



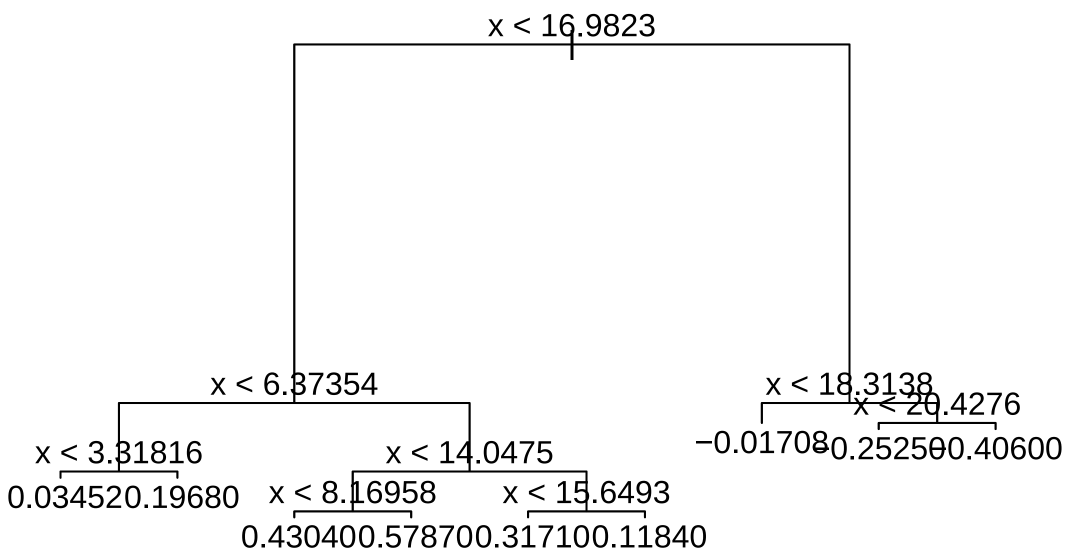

```
testindex<-sample(1:nrow(df3), 400)
df3_test<-df3[testindex,]
df3_train<-df3[-testindex,]
```

a. Fit a best tree model using cross validation. Plot the data and fitted model. What is estimated RMSE for this fitted model using a test/holdout sample?

```
#Fit the tree model to our training data
tree_fit <- tree(y~x, df3_train)
summary(tree_fit)
```

```
##
## Regression tree:
## tree(formula = y ~ x, data = df3_train)
## Number of terminal nodes:  9
## Residual mean deviance:  0.01492 = 8.819 / 591
## Distribution of residuals:
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## -0.386100 -0.076710 -0.003729  0.000000  0.081700  0.354800
```

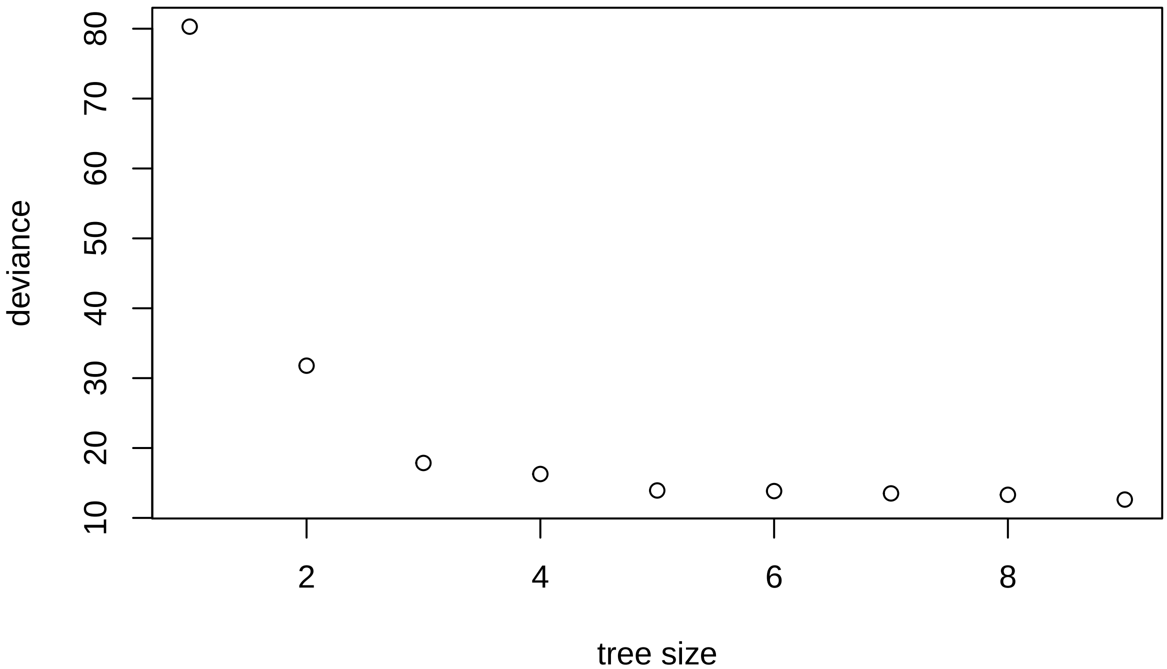
```
#Plot the tree model
plot(tree_fit)
text(tree_fit, pretty=0)
```




```
#Prune the tree, suggests 9 nodes based on 10 fold cross validation
#Tree with 9 terminal nodes has the lowest c.v total error rate
cv.tree_fit<- cv.tree(tree_fit, FUN=prune.tree)
cv.tree_fit
```

```
## $size
## [1] 9 8 7 6 5 4 3 2 1
##
## $dev
## [1] 12.62501 13.30797 13.50384 13.83890 13.92655 16.28115 17.85734 31.78149
## [9] 80.29168
##
## $k
## [1]       -Inf  0.8258021  0.8327021  0.8536502  0.8722689  2.7833930  5.5608773
## [8]  9.5550185 50.0113134
##
## $method
## [1] "deviance"
##
## attr(,"class")
## [1] "prune"         "tree.sequence"
```

```r
#Plot deviance vs. size of pruned tree at each node amount of 1 to 9
plot(cv.tree_fit$size,cv.tree_fit$dev,xlab='tree size',ylab='deviance')
```




```r
#########Find RMSE of each amount of terminal tree nodes from 1-9 for both training and test data
nT<- length(cv.tree_fit$size)
RMSE<- tibble()
for(j in 1:nT){
  treesize<-cv.tree_fit$size[j]
  if(cv.tree_fit$size[j] > 1){
        prunedTree = prune.tree(tree_fit,best=cv.tree_fit$size[j])
        yhtrain = predict(prunedTree,newdata=df3_train)
        yhtest  = predict(prunedTree,newdata=df3_test)

  } else{
    yhtrain=rep(prunedTree$frame$yval, nrow(df3_train))
    yhtest=rep(prunedTree$frame$yval, nrow(df3_test))
  }

  errors<-tibble(trainRMSE = sqrt(mean((df3_train$y-yhtrain)^2)),
  testRMSE= sqrt(mean((df3_test$y-yhtest)^2)),
  cvDev= cv.tree_fit$dev[j]/nrow(df3_train),
  treesize=treesize)

  RMSE<- RMSE%>%
    bind_rows(errors)%>%
    arrange(treesize)

}
RMSE
```

```
## # A tibble: 9 x 4
```

```
##   trainRMSE testRMSE  cvDev treesize
##       <dbl>    <dbl>  <dbl>    <int>
## 1     0.465    0.466 0.134         1
## 2     0.224    0.228 0.0530        2
## 3     0.185    0.190 0.0298        3
## 4     0.158    0.170 0.0271        4
## 5     0.143    0.159 0.0232        5
## 6     0.137    0.155 0.0231        6
## 7     0.132    0.152 0.0225        7
## 8     0.127    0.149 0.0222        8
## 9     0.121    0.142 0.0210        9
```

```r
#######End

#Using our RMSE from our function, we find the lowest test RMSE at a value of node=9
RMSE%>%
  filter(testRMSE==min(testRMSE))
```

```
## # A tibble: 1 x 4
##   trainRMSE testRMSE  cvDev treesize
##       <dbl>    <dbl>  <dbl>    <int>
## 1     0.121    0.142 0.0210        9
```

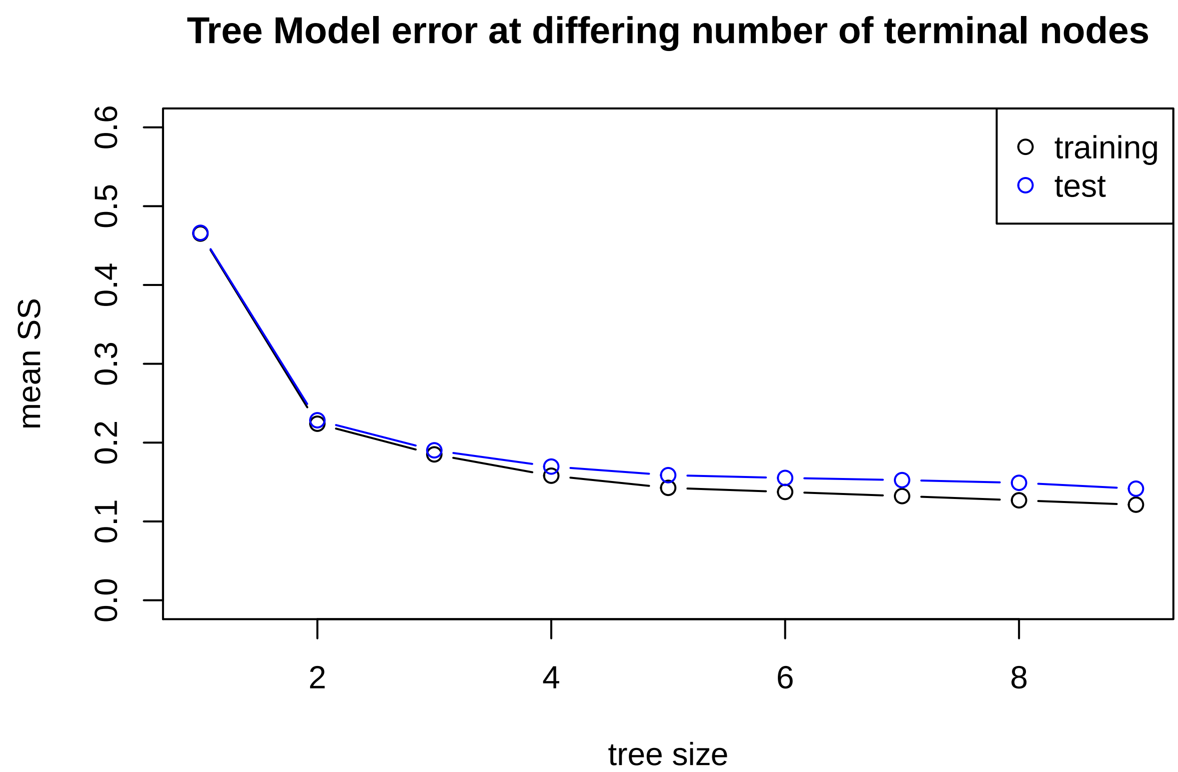
```r
#Using our RMSE from our function, we find the lowest train RMSE at a value of node=9
RMSE%>%
  filter(trainRMSE==min(trainRMSE))
```

```
## # A tibble: 1 x 4
##   trainRMSE testRMSE  cvDev treesize
##       <dbl>    <dbl>  <dbl>    <int>
## 1     0.121    0.142 0.0210        9
```

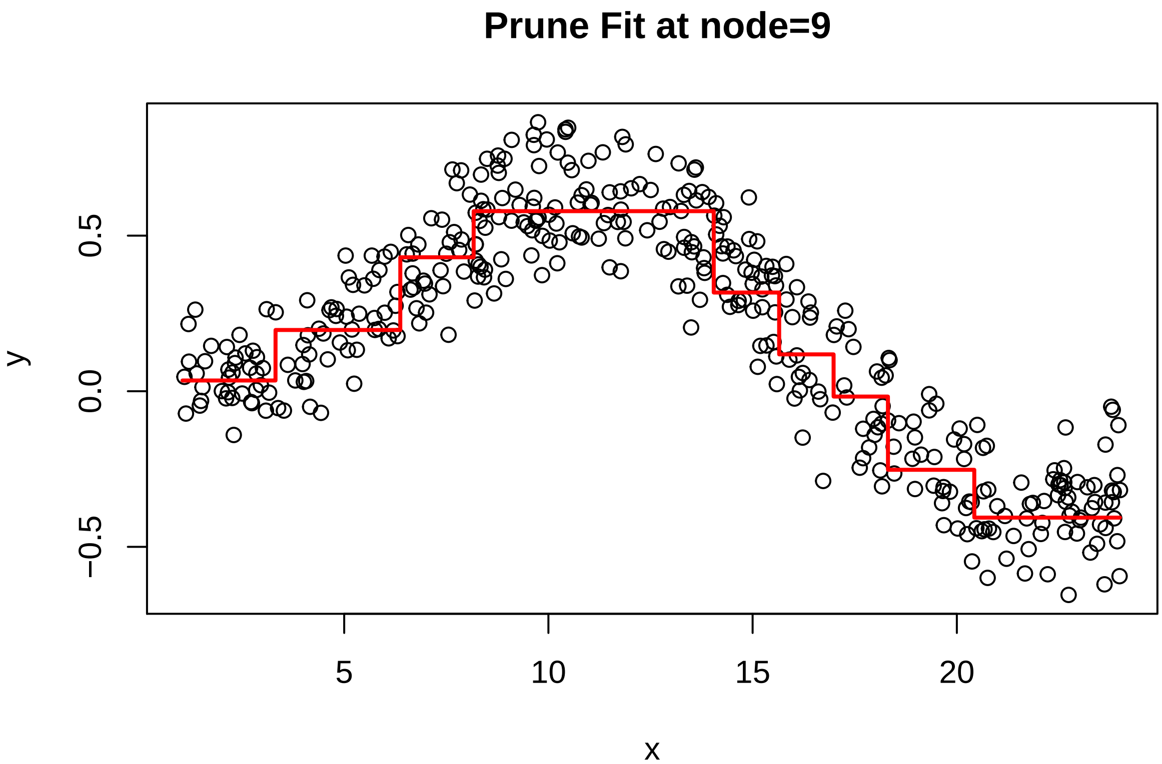
```r
#Using our RMSE from our function, we find the lowest cv mean error at a value of node=9
RMSE%>%
  filter(cvDev==min(cvDev))
```

```
## # A tibble: 1 x 4
##   trainRMSE testRMSE  cvDev treesize
##       <dbl>    <dbl>  <dbl>    <int>
## 1     0.121    0.142 0.0210        9
```

```r
#Plot test RMSE, train RMSE, and c.v error rate summed across all components, divided by number of rows
plot(RMSE$treesize,RMSE$trainRMSE,type='b',col='black',xlab='tree size',ylab='mean SS', ylim=c(0,.6))
points(RMSE$treesize,RMSE$testRMSE,type='b',col='blue')
title("Tree Model error at differing number of terminal nodes")
legend('topright',legend=c('training','test'),col=c('black','blue'),
       pch=1)
```




```
###############
#Plot values from df3 and associated fit at treesize=9
prune.fit<- prune.tree(tree_fit,best=9)

prune.fit.pred<- predict(prune.fit, df3_test)

#Make some predictons for the sake of plotting
pred_df<-data.frame(x=seq(min(df3$x), max(df3$x),by=.001))

#Predict outcome of predicted x values above using bag fit
pred.prune.df<- predict(prune.fit, newdata=pred_df)

#Plot Prune fit with 9 nodes
plot(df3_test)
lines(pred_df$x, pred.prune.df, col="red",lwd=2)
title (" Prune Fit at node=9")
```



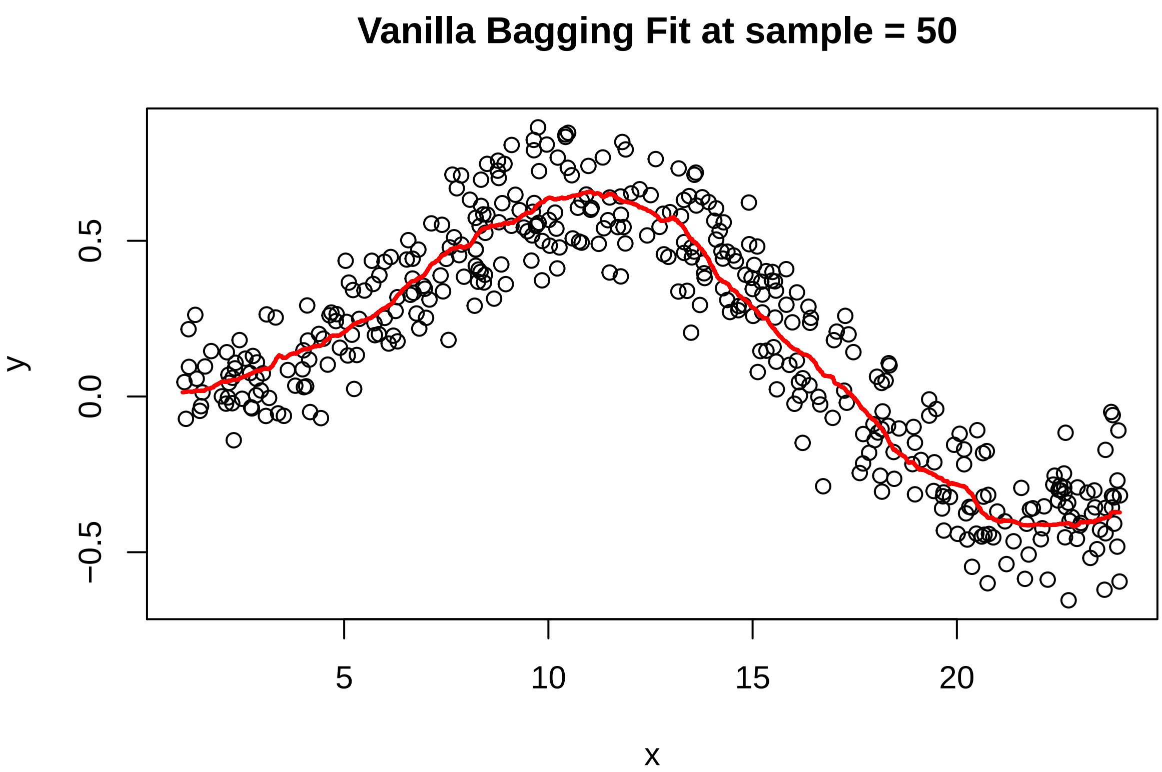

```
#Calculate RMSE of Pruned Tree at node = 9
sqrt(mean((prune.fit.pred-df3_test$y)^2))
```

```
## [1] 0.1415482
```

We find in our sample (subject to change based on cv.tree sampling) a minimum test RMSE, minimum training RMSE, and suggested cv node at node=9. This leads to a test RMSE of .1415.

b. Fit a bagged tree model using cross validation. Plot the data and fitted model. What is estimated RMSE for this fitted model using a test/holdout sample?

```
#Set bag model with m=1, as there is only one predictor
bag_fit=randomForest(y ~ x,data=df3_train,
                     ntree=1*500, sampsize=50)

#Predict y response of df3_test
pred.bag = predict(bag_fit, newdata=df3_test)

#Make some predictons for the sake of plotting
pred_df<-data.frame(x=seq(min(df3$x), max(df3$x),by=.001))

#Predict outcome of predicted x values above using bag fit
pred.bag.df<- predict(bag_fit, newdata=pred_df)


#Plot df3_test data and bagged fit on predicted values of all x's
plot(df3_test)
lines(pred_df$x, pred.bag.df, col="red",lwd=2)
title (" Vanilla Bagging Fit at sample = 50")
```




```
########Write Function to assess RMSE at different sample sizes

errors<-tibble()
for (i in 1:(length(df3_train$x)/10)){
  #Set sample size to i
  sampsize<-i
  #Fit bagged model with sampel size = i
  bag_fit=randomForest(y ~ x,data=df3_train,
                       ntree=1*500, sampsize=i)
  #Predict df3_test output using mode
  pred.bag<-predict(bag_fit, newdata=df3_test)

  #Calculate RMSE of actual test value to predicted test value
  RMSE<-sqrt(mean((df3_test$y-pred.bag)^2))

  #Store in tibble
  error.i<-tibble(testRMSE=RMSE, sampsize)

  #Add to errors
  errors<-errors%>%
    bind_rows(error.i)
}

errors
```

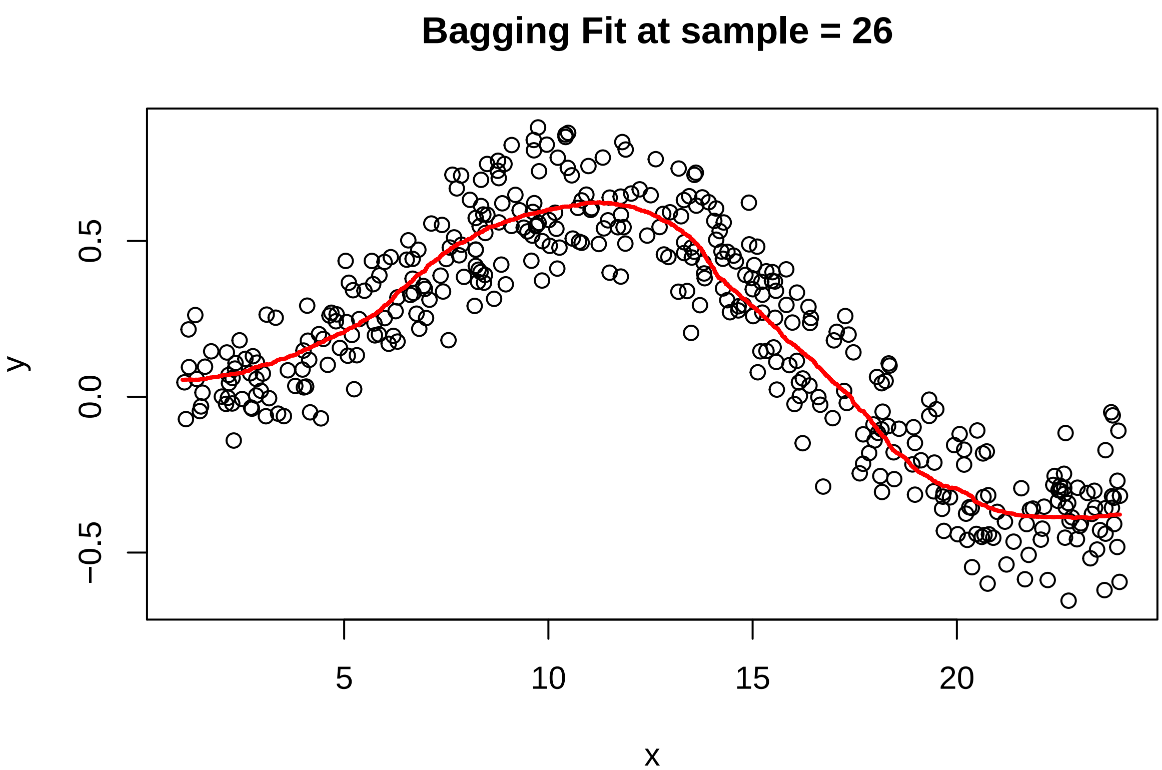
```
## # A tibble: 60 x 2
##    testRMSE sampsize
##       <dbl>    <int>
##  1    0.371        1
```

```
##   2    0.275        2
##   3    0.244        3
##   4    0.227        4
##   5    0.219        5
##   6    0.213        6
##   7    0.194        7
##   8    0.178        8
##   9    0.162        9
## 10    0.155       10
## # ... with 50 more rows
```

```r
########End

#Return samp size with minimum test RMSE
errors%>%
  filter(testRMSE==min(testRMSE))
```

```
## # A tibble: 1 x 2
##   testRMSE sampsize
##      <dbl>    <int>
## 1    0.128       26
```

```r
#Set bag model with m=1, as there is only one predictor, and sample size = 26
bag_fit=randomForest(y ~ x,data=df3_train,
                        ntree=1*500, sampsize=26)

#Predict y response of df3_test
pred.bag = predict(bag_fit, newdata=df3_test)

#Make some predictons for the sake of plotting
pred_df<-data.frame(x=seq(min(df3$x), max(df3$x),by=.001))

#Predict outcome of predicted x values above using bag fit
pred.bag.df<- predict(bag_fit, newdata=pred_df)


#Plot df3_test data and bagged fit on predicted values of all x's
plot(df3_test)
lines(pred_df$x, pred.bag.df, col="red",lwd=2)
title (" Bagging Fit at sample = 26")
```



The test RMSE of the bagged model with sample size of 26 is .1285. This is lower than the test RMSE of the tree model with 9 nodes, with a test RMSE = .1415. Note that the optimum sample size for bagging model is subject to change based on randomness in the boosttrap aggregation function.

c. Fit a boosted tree model using cross validation. Plot the data and fitted model. What is estimated RMSE for this fitted model using a test/holdout sample?

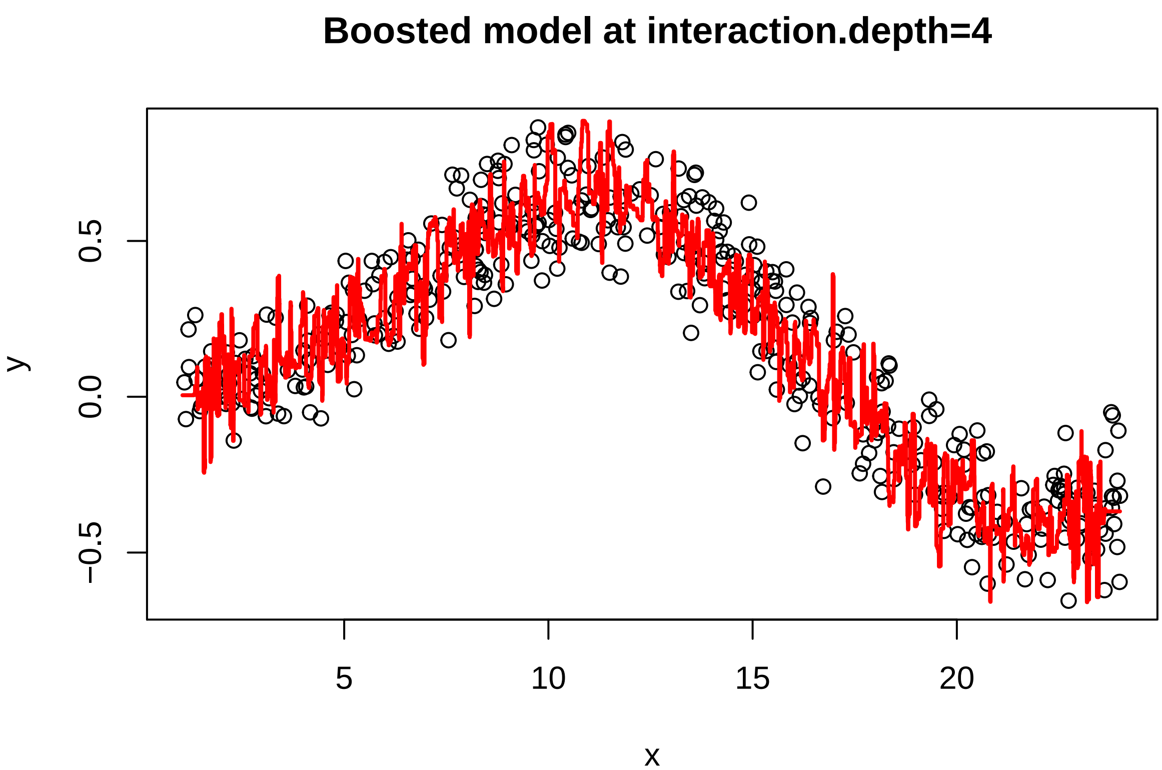
```
#Set boosted model
boost.fit<- gbm(y ~ x,data=df3_train,distribution="gaussian",
                 n.trees=5000, interaction.depth=4)

#Predict boosted values of test data
boost.pred<- predict(boost.fit, df3_test)
```

```
## Using 5000 trees...
```

```
#Make some predictons for the sake of plotting
pred_df<-data.frame(x=seq(min(df3$x), max(df3$x),by=.001))

#Predict outcome of predicted x values above using bag fit
pred.boost.df<- predict(boost.fit, newdata=pred_df)
```

```
## Using 5000 trees...
```

```
#Plot data
plot(df3_test)
lines(pred_df$x, pred.boost.df, col="red",lwd=2)
title (" Boosted model at interaction.depth=4")
```




```
#Calculate RMSE
sqrt(mean((boost.pred-df3_test$y)^2))
```

```
## [1] 0.1623079
```

```
########Write Function to assess RMSE at different Shrinkage Rates

errors<-tibble()
for (i in 1:10){
  #Set sample size to i
  shrinkage<- i/1000
  #Fit bagged model with sampel size = i
  boost.fit<- gbm(y ~ x,data=df3_train,distribution="gaussian",
                  n.trees=5000, interaction.depth=4, shrinkage=i/1000)
  #Predict df3_test output using mode
  pred.boost<-predict(boost.fit, newdata=df3_test)

  #Calculate RMSE of actual test value to predicted test value
  RMSE<-sqrt(mean((df3_test$y-pred.boost)^2))

  #Store in tibble
  error.i<-tibble(testRMSE=RMSE, shrinkage)

  #Add to errors
  errors<-errors%>%
    bind_rows(error.i)
}
```

```
## Using 5000 trees...
```

```
##
## Using 5000 trees...
##
## Using 5000 trees...
##
## Using 5000 trees...
##
## Using 5000 trees...
##
## Using 5000 trees...
##
## Using 5000 trees...
##
## Using 5000 trees...
##
## Using 5000 trees...
##
## Using 5000 trees...
```

```r
errors
```

```
## # A tibble: 10 x 2
##    testRMSE shrinkage
##       <dbl>     <dbl>
##  1    0.130     0.001
##  2    0.131     0.002
##  3    0.132     0.003
##  4    0.132     0.004
##  5    0.133     0.005
##  6    0.133     0.006
##  7    0.134     0.007
##  8    0.135     0.008
##  9    0.135     0.009
## 10    0.136     0.01
```

```r
##################

#We find a minimum test RMSE with a shrinkage of .001
errors%>%
  filter(testRMSE==min(testRMSE))
```

```
## # A tibble: 1 x 2
##   testRMSE shrinkage
##      <dbl>     <dbl>
## 1    0.130     0.001
```

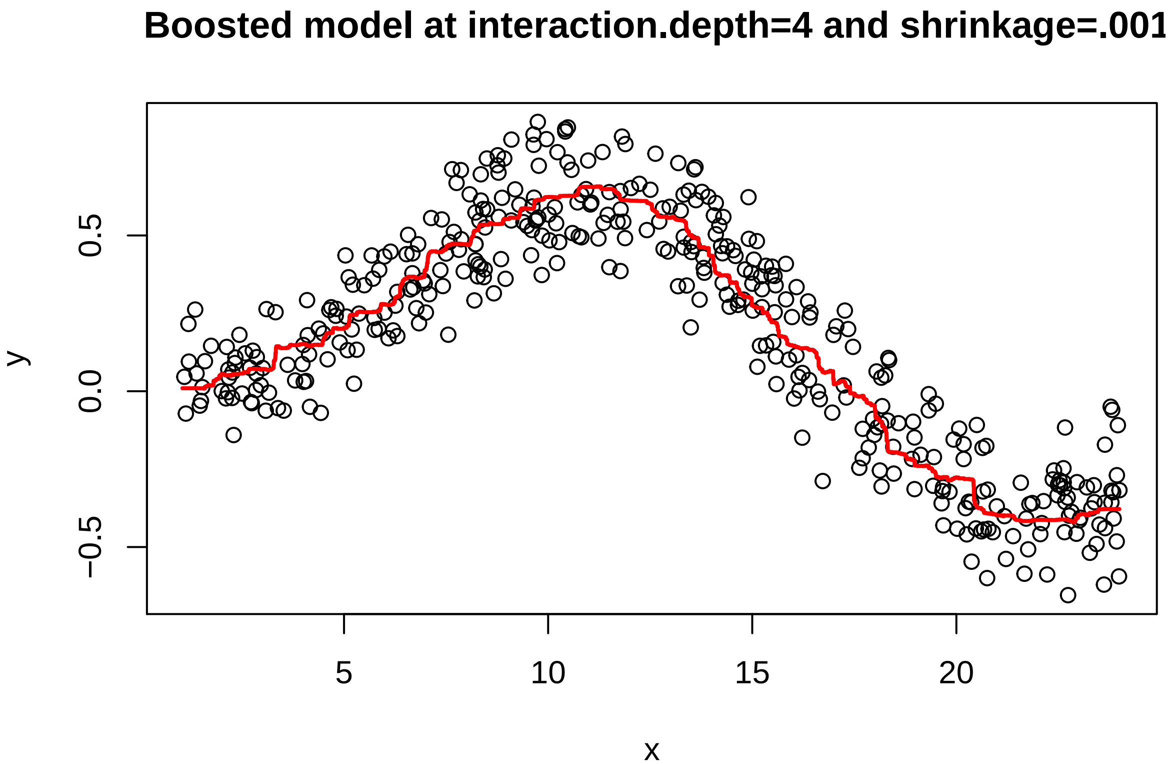
```r
#Set boosted model with shrinkage = .001
boost.fit<- gbm(y ~ x,data=df3_train,distribution="gaussian",
                 n.trees=5000, interaction.depth=4, shrinkage=.001)

#Predict boosted values of test data
boost.pred<- predict(boost.fit, df3_test)
```

```
## Using 5000 trees...
```

```r
#Make some predictons for the sake of plotting
pred_df<-data.frame(x=seq(min(df3$x), max(df3$x),by=.001))

#Predict outcome of predicted x values above using bag fit
pred.boost.df<- predict(boost.fit, newdata=pred_df)
```

```
## Using 5000 trees...
```

```r
#Plot data
plot(df3_test)
lines(pred_df$x, pred.boost.df, col="red",lwd=2)
title (" Boosted model at interaction.depth=4 and shrinkage=.001")
```



We find a test RMSE of .1304 for our boosted tree model at shrinkage=.001. This is slightly higher than our bagged model at sample size 26 (test RMSE=.1285) and lower than our tree model at node=9 (test RMSE = .1415).

Based on the cross validation results, our preferred model is the bagged model of sample size 26 (test RMSE = .1285).

2. Do problem 8.10a, b, c, and d from ISLR.

Use boosting to predict Salary in the hitters data set.

a. Remove observations for whom salary data is unknown then log-transform the salaries.

```
Hitters.naomit<-na.omit(Hitters)
Hitters.naomit$Salary<- log(Hitters.naomit$Salary)
```

b. Create a training set of the first 200 observations, and a test set consisting of the remaining.

```
train<-seq(1:200)

Hitters.train<-Hitters.naomit[train,]
Hitters.test<-Hitters.naomit[-train,]
```

c. Perform boosting on the training set with 1000 trees for a range of values of shrinkage parameter. Produce a plot with different shrinkage values and corresponding training MSE.

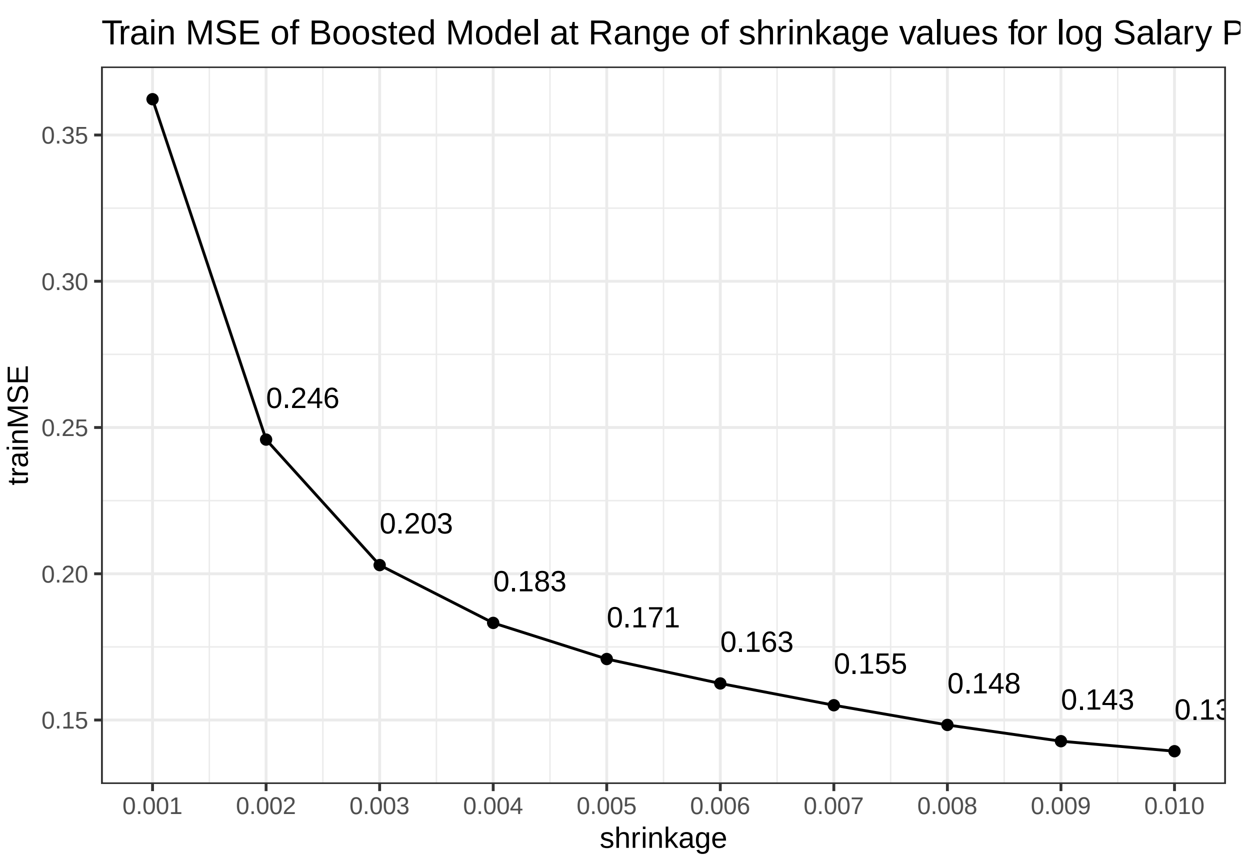
```
errors<-tibble()
for (i in 1:10){
  #Set sample size to i
  shrinkage<- i/1000
  #Fit bagged model with sample size = i
  boost.fit<- gbm(Salary ~ .,data=Hitters.train,distribution="gaussian",
                  n.trees=1000, shrinkage=i/1000)
  #Predict df3_test output using mode
  pred.boost<-predict(boost.fit)

  #Calculate RMSE of actual test value to predicted test value
  MSE<-mean((Hitters.train$Salary-pred.boost)^2)

  #Store in tibble
  error.i<-tibble(trainMSE=MSE, shrinkage)

  #Add to errors
  errors<-errors%>%
    bind_rows(error.i)
}
```

```
## Using 1000 trees...
##
## Using 1000 trees...
##
## Using 1000 trees...
##
## Using 1000 trees...
##
## Using 1000 trees...
##
## Using 1000 trees...
##
## Using 1000 trees...
##
## Using 1000 trees...
##
## Using 1000 trees...
##
## Using 1000 trees...
```

```r
errors
```

```
## # A tibble: 10 x 2
##    trainMSE shrinkage
##       <dbl>     <dbl>
##  1    0.362     0.001
##  2    0.246     0.002
##  3    0.203     0.003
##  4    0.183     0.004
##  5    0.171     0.005
##  6    0.163     0.006
##  7    0.155     0.007
##  8    0.148     0.008
##  9    0.143     0.009
## 10    0.139     0.01
```

```r
ggplot(errors, aes(x=shrinkage, y=trainMSE))+
  geom_point()+
  geom_line()+
  theme_bw()+
  labs(title= "Train MSE of Boosted Model at Range of shrinkage values for log Salary Prediction")+
  geom_text(aes(label=round(trainMSE,3)),hjust=0, vjust=-1.5)+
  scale_x_continuous(breaks=seq(.001, .01, by=.001))
```



d. Produce a plot with different shrinkage values and corresponding test MSE.

```r
errors<-tibble()
for (i in 1:10){
  #Set sample size to i
  shrinkage<- i/1000
  #Fit bagged model with sample size = i
  boost.fit<- gbm(Salary ~ .,data=Hitters.train,distribution="gaussian",
                  n.trees=1000, shrinkage=i/1000)
  #Predict df3_test output using mode
  pred.boost<-predict(boost.fit, newdata=Hitters.test)

  #Calculate RMSE of actual test value to predicted test value
  MSE<-mean((Hitters.test$Salary-pred.boost)^2)

  #Store in tibble
  error.i<-tibble(testMSE=MSE, shrinkage)

  #Add to errors
  errors<-errors%>%
    bind_rows(error.i)
}
```

```
## Using 1000 trees...
##
## Using 1000 trees...
##
## Using 1000 trees...
##
## Using 1000 trees...
##
## Using 1000 trees...
##
## Using 1000 trees...
##
## Using 1000 trees...
##
## Using 1000 trees...
##
## Using 1000 trees...
##
## Using 1000 trees...
```

```r
errors
```

```
## # A tibble: 10 x 2
##    testMSE shrinkage
##      <dbl>     <dbl>
##  1   0.337     0.001
##  2   0.298     0.002
##  3   0.293     0.003
##  4   0.290     0.004
##  5   0.294     0.005
##  6   0.288     0.006
##  7   0.282     0.007
```

```
## 8   0.288     0.008
## 9   0.282     0.009
## 10  0.282     0.01
```

```
ggplot(errors, aes(x=shrinkage, y=testMSE))+
  geom_point()+
  geom_line()+
  theme_bw()+
  labs(title= "Test MSE of Boosted Model at Range of shrinkage values for log Salary Prediction")+
  geom_text(aes(label=round(testMSE,3)),hjust=0, vjust=-1.5)+
  scale_x_continuous(breaks=seq(.001, .01, by=.001))
```