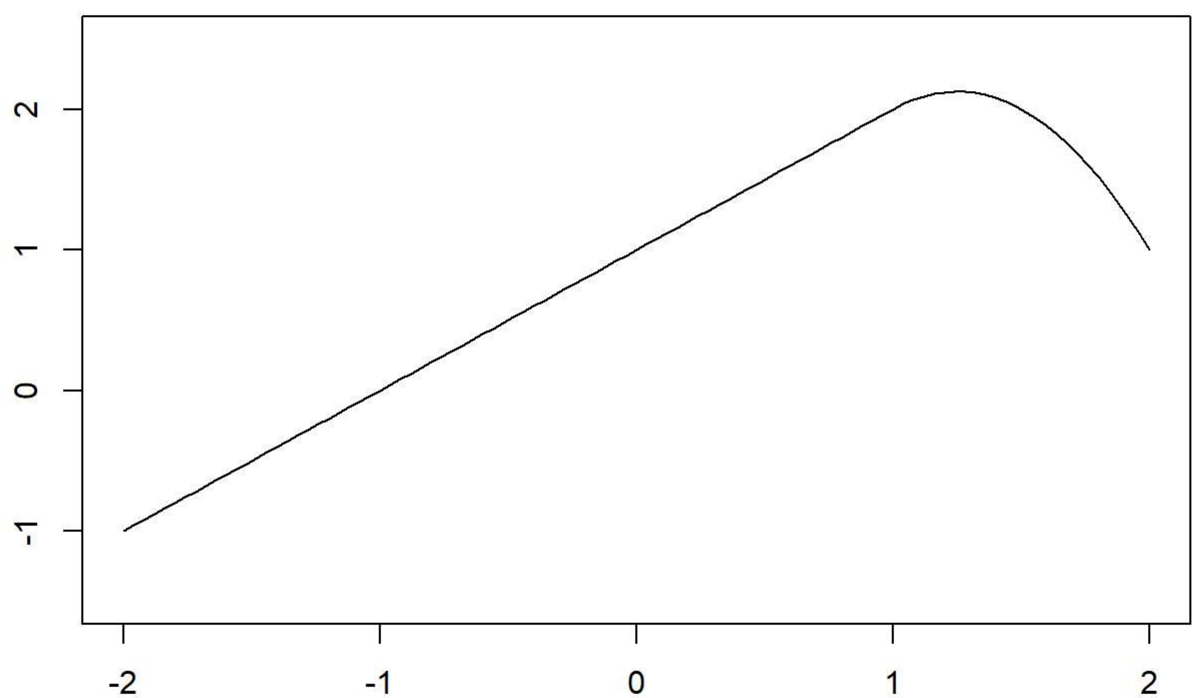
6.1 a) Best subset selection has the smallest training RSS because it considers all possible subsets and chooses the one with the least amount of RSS. It is impossible that a stepwise method provides a smaller RSS though it may equal. b) Best subset selection may have the smallest test RSS because it considers more models. However, one of the stepwise methods may fit the data better because of overfitting by best subset selection. c) i) True: Forward stepsize with (k+1) variables simply adds a variable to the k variable case. ii) True: Backward stepsize with k variables simply removes a variable from the (k+1) variable case iii)False: The variables used in backward and forward stepsize are not required to be related. iv) False: The variables used in backward and forward stepsize are not required to be related v) False: Best subset does not add variables incrementally.

6.2 a) The lasso method is less flexible and hence will give improved prediction accu-racy when its increase in bias is less than its decrease in variance. b) The ridge regression method is less flexible and hence will give improved prediction accu-racy when its increase in bias is less than its decrease in variance. c) Nonlinear methods are more flexible and hence will give improved prediction accu-racy when its increase in variance is less than its decrease in bias.

7.3

```
x = -50:50 / 25
y = 1 + x + -2 * (x-1)^2 * I(x>1)
plot(1, type="n", xlab="", ylab="", xlim=c(-2, 2), ylim=c(-1.5, 2.5))
lines(x, y)
```



Y-

intercept is 1. Slope is 1 within the range x= -2 to x= 1.

7.9

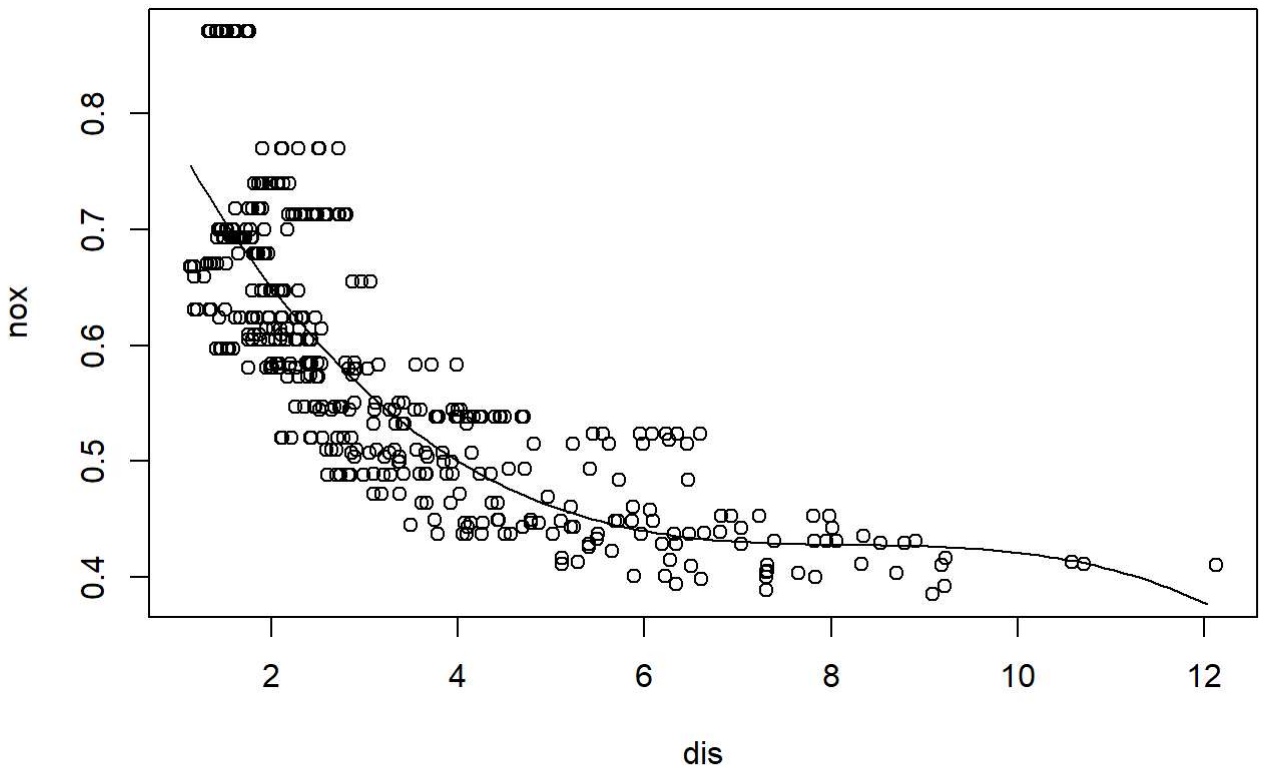
```r
library(MASS)
attach(Boston)
```

a.

```r
lm.fit = lm(nox ~ poly(dis, 3), data = Boston)
summary(lm.fit)
```

```
##
## Call:
## lm(formula = nox ~ poly(dis, 3), data = Boston)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.121130 -0.040619 -0.009738  0.023385  0.194904
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    0.554695   0.002759 201.021  < 2e-16 ***
## poly(dis, 3)1 -2.003096   0.062071 -32.271  < 2e-16 ***
## poly(dis, 3)2  0.856330   0.062071  13.796  < 2e-16 ***
## poly(dis, 3)3 -0.318049   0.062071  -5.124 4.27e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.06207 on 502 degrees of freedom
## Multiple R-squared:  0.7148, Adjusted R-squared:  0.7131
## F-statistic: 419.3 on 3 and 502 DF,  p-value: < 2.2e-16
```

```r
range = range(dis)
xvals = seq(from = range[1], to = range[2], by = 0.1)
yvals = predict(lm.fit, list(dis = xvals))
plot(nox ~ dis, data = Boston)
lines(xvals, yvals)
```



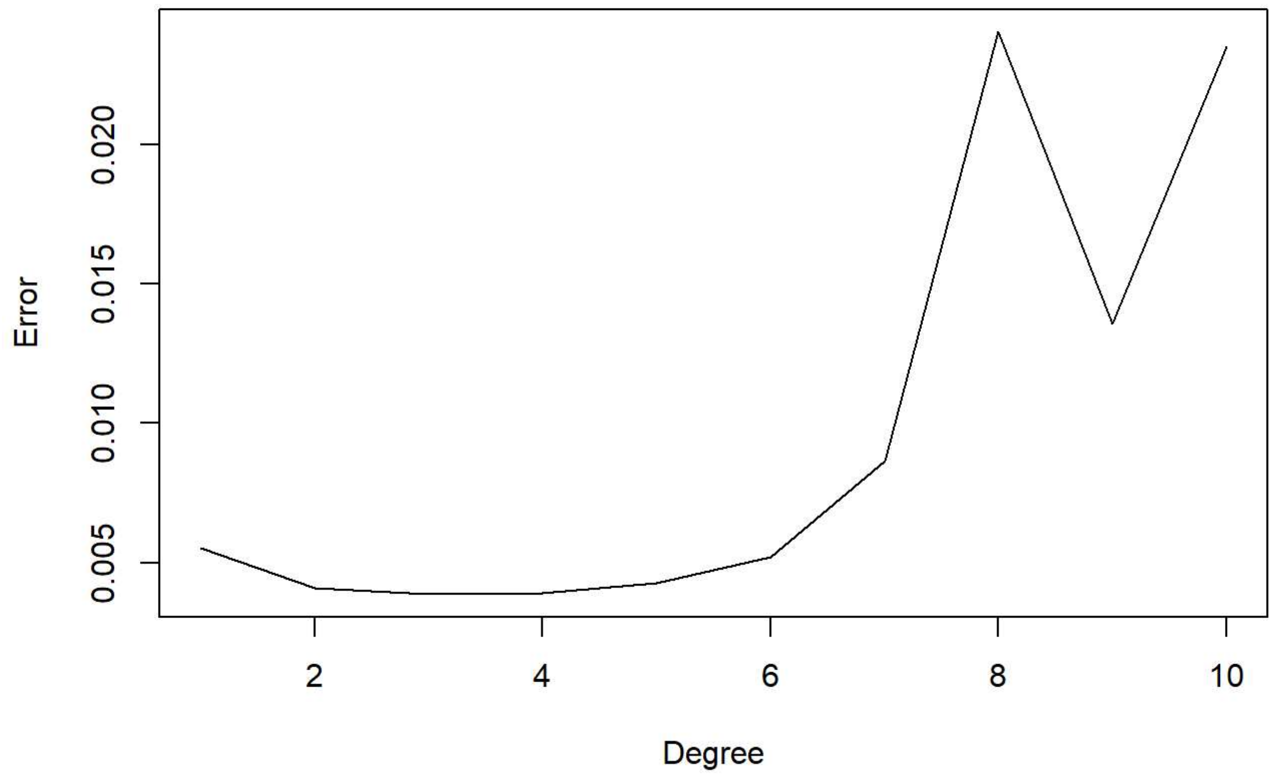
b.

```
rss = numeric(10)
for (i in 1:10) {
    lm.fit = lm(nox ~ poly(dis, i), data = Boston)
    rss[i] = sum(lm.fit$residuals^2)
}
rss
```

```
##  [1] 2.768563 2.035262 1.934107 1.932981 1.915290 1.878257 1.849484
##  [8] 1.835630 1.833331 1.832171
```

c. We will use 5-fold cross validation.

```
library(boot)
error = numeric(10)
for (i in 1:10) {
    glm.fit = glm(nox ~ poly(dis, i), data = Boston)
    error[i] = cv.glm(Boston, glm.fit, K = 10)$delta[2]
}
plot(1:10, error, xlab = "Degree", ylab = "Error", type = "l")
```



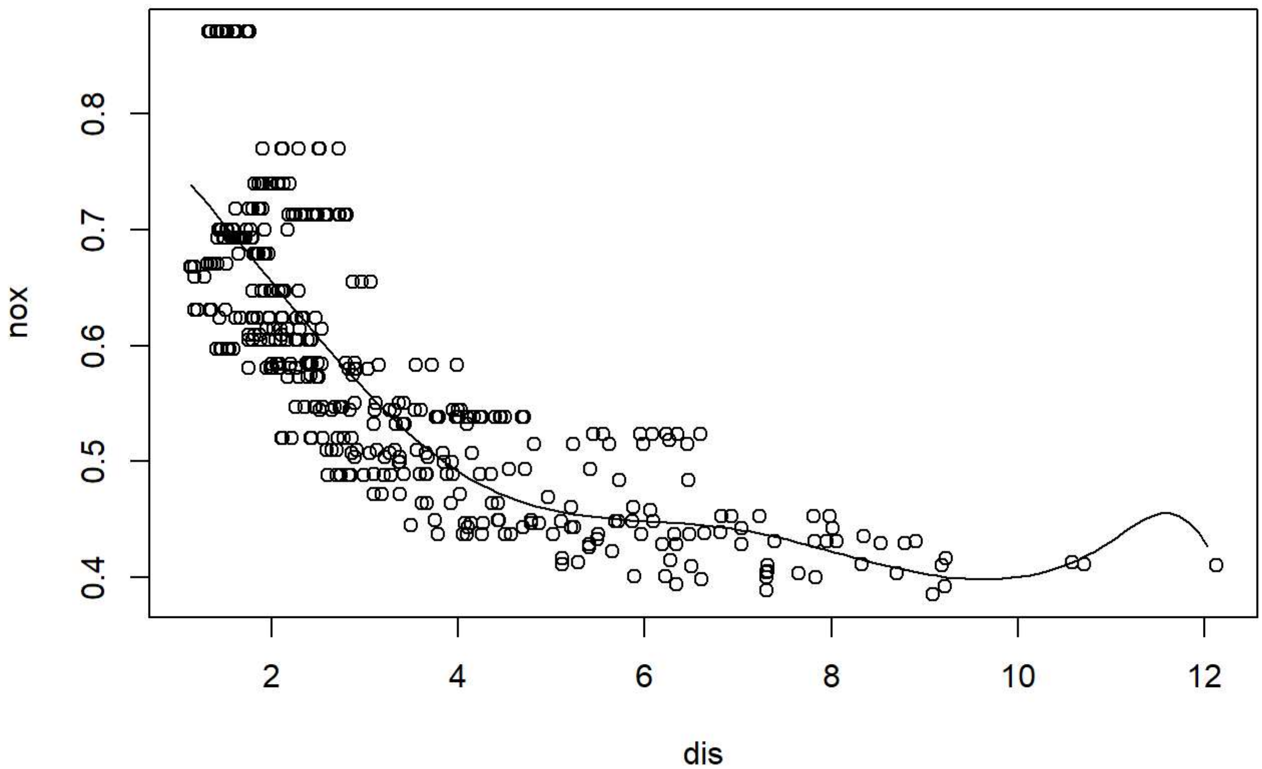
The cross validation shows us that the cross validated error decreases for 1 to 3, stays constant for 3 to 5 and then starts increasing. We will choose a degree of 3.

d.

```
library(splines)
spl.fit = lm(nox ~ bs(dis, df = 4, knots = c(4, 7, 11)), data = Boston)
summary(spl.fit)
```

```
##
## Call:
## lm(formula = nox ~ bs(dis, df = 4, knots = c(4, 7, 11)), data = Boston)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.124567 -0.040355 -0.008702  0.024740  0.192920
##
## Coefficients:
##                                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)                            0.73926    0.01331  55.537  < 2e-16
## bs(dis, df = 4, knots = c(4, 7, 11))1 -0.08861    0.02504  -3.539  0.00044
## bs(dis, df = 4, knots = c(4, 7, 11))2 -0.31341    0.01680 -18.658  < 2e-16
## bs(dis, df = 4, knots = c(4, 7, 11))3 -0.26618    0.03147  -8.459 3.00e-16
## bs(dis, df = 4, knots = c(4, 7, 11))4 -0.39802    0.04647  -8.565  < 2e-16
## bs(dis, df = 4, knots = c(4, 7, 11))5 -0.25681    0.09001  -2.853  0.00451
## bs(dis, df = 4, knots = c(4, 7, 11))6 -0.32926    0.06327  -5.204 2.85e-07
##
## (Intercept)                           ***
## bs(dis, df = 4, knots = c(4, 7, 11))1 ***
## bs(dis, df = 4, knots = c(4, 7, 11))2 ***
## bs(dis, df = 4, knots = c(4, 7, 11))3 ***
## bs(dis, df = 4, knots = c(4, 7, 11))4 ***
## bs(dis, df = 4, knots = c(4, 7, 11))5 **
## bs(dis, df = 4, knots = c(4, 7, 11))6 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.06185 on 499 degrees of freedom
## Multiple R-squared:  0.7185, Adjusted R-squared:  0.7151
## F-statistic: 212.3 on 6 and 499 DF,  p-value: < 2.2e-16
```

```
yvals = predict(spl.fit, list(dis = xvals))
plot(nox ~ dis, data = Boston)
lines(xvals, yvals)
```



The summary shows that all the spline terms are significant. The splines fit well except at the upper extreme.

e.

```
error = numeric(20)
for (i in 3:20) {
    lm.fit = lm(nox ~ bs(dis, df = i), data = Boston)
    error[i] = sum(lm.fit$residuals^2)
}
error
```

```
##  [1] 0.000000 0.000000 1.934107 1.922775 1.840173 1.833966 1.829884
##  [8] 1.816995 1.825653 1.792535 1.796992 1.788999 1.782350 1.781838
## [15] 1.782798 1.783546 1.779789 1.775838 1.774487 1.776727
```

Training RSS decreases until 14 degrees of freedom.

f. We will use 5-fold cross validation.

```
error = numeric(20)
for (i in 3:20) {
    lm.fit = glm(nox ~ bs(dis, df = i), data = Boston)
    error[i] = cv.glm(Boston, lm.fit, K = 5)$delta[2]
}
```

```
## Warning in bs(dis, degree = 3L, knots = numeric(0), Boundary.knots =
## c(1.1296, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = numeric(0), Boundary.knots =
## c(1.1296, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases
```

```
## Warning in bs(dis, degree = 3L, knots = numeric(0), Boundary.knots =
## c(1.137, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = numeric(0), Boundary.knots =
## c(1.137, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases
```

```
## Warning in bs(dis, degree = 3L, knots = c(`50%` = 3.3175), Boundary.knots =
## c(1.1296, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = c(`50%` = 3.3175), Boundary.knots =
## c(1.1296, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases
```

```
## Warning in bs(dis, degree = 3L, knots = c(`50%` = 3.1992), Boundary.knots =
## c(1.137, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = c(`50%` = 3.1992), Boundary.knots =
## c(1.137, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases
```

```
## Warning in bs(dis, degree = 3L, knots = c(`33.33333%` = 2.3682, `66.66667%`
## = 4.2673: some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = c(`33.33333%` = 2.3682, `66.66667%`
## = 4.2673: some 'x' values beyond boundary knots may cause ill-conditioned
## bases
```

```
## Warning in bs(dis, degree = 3L, knots = c(`33.33333%` = 2.39256666666667, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`33.33333%` = 2.39256666666667, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases
```

```
## Warning in bs(dis, degree = 3L, knots = c(`25%` = 2.1329, `50%` = 3.3175, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`25%` = 2.1329, `50%` = 3.3175, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases
```

```
## Warning in bs(dis, degree = 3L, knots = c(`25%` = 2.090175, `50%` =
## 3.19095, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = c(`25%` = 2.090175, `50%` =
## 3.19095, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases
```

```
## Warning in bs(dis, degree = 3L, knots = c(`20%` = 1.92866, `40%` =
## 2.5698, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = c(`20%` = 1.92866, `40%` =
## 2.5698, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases
```

```
## Warning in bs(dis, degree = 3L, knots = c(`16.66667%` = 1.82005,
## `33.33333%` = 2.346, : some 'x' values beyond boundary knots may cause ill-
## conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`16.66667%` = 1.82005,
## `33.33333%` = 2.346, : some 'x' values beyond boundary knots may cause ill-
## conditioned bases
```

```
## Warning in bs(dis, degree = 3L, knots = c(`16.66667%` = 1.86345,
## `33.33333%` = 2.4233, : some 'x' values beyond boundary knots may cause
## ill-conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`16.66667%` = 1.86345,
## `33.33333%` = 2.4233, : some 'x' values beyond boundary knots may cause
## ill-conditioned bases
```

```
## Warning in bs(dis, degree = 3L, knots = c(`14.28571%` = 1.78741428571429, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`14.28571%` = 1.78741428571429, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases
```

```
## Warning in bs(dis, degree = 3L, knots = c(`14.28571%` = 1.798, `28.57143%`
## = 2.16278571428571, : some 'x' values beyond boundary knots may cause ill-
## conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`14.28571%` = 1.798, `28.57143%`
## = 2.16278571428571, : some 'x' values beyond boundary knots may cause ill-
## conditioned bases
```

```
## Warning in bs(dis, degree = 3L, knots = c(`12.5%` = 1.75085, `25%` =
## 2.1185, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = c(`12.5%` = 1.75085, `25%` =
## 2.1185, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases
```

```
## Warning in bs(dis, degree = 3L, knots = c(`11.11111%` = 1.69522222222222, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`11.11111%` = 1.69522222222222, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases
```

```
## Warning in bs(dis, degree = 3L, knots = c(`11.11111%` = 1.6334, `22.22222%`
## = 2.0026, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = c(`11.11111%` = 1.6334, `22.22222%`
## = 2.0026, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases
```

```
## Warning in bs(dis, degree = 3L, knots = c(`10%` = 1.6424, `20%` =
## 1.96376, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = c(`10%` = 1.6424, `20%` =
## 1.96376, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases
```

```
## Warning in bs(dis, degree = 3L, knots = c(`10%` = 1.62262, `20%` =
## 1.92158, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = c(`10%` = 1.62262, `20%` =
## 1.92158, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases
```

```
## Warning in bs(dis, degree = 3L, knots = c(`9.090909%` = 1.64440909090909, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`9.090909%` = 1.64440909090909, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases
```

```
## Warning in bs(dis, degree = 3L, knots = c(`9.090909%` = 1.61284545454545, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`9.090909%` = 1.61284545454545, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases
```

```
## Warning in bs(dis, degree = 3L, knots = c(`8.333333%` = 1.5909, `16.66667%`
## = 1.85283333333333, : some 'x' values beyond boundary knots may cause ill-
## conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`8.333333%` = 1.5909, `16.66667%`
## = 1.85283333333333, : some 'x' values beyond boundary knots may cause ill-
## conditioned bases
```

```
## Warning in bs(dis, degree = 3L, knots = c(`8.333333%` = 1.56736666666667, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`8.333333%` = 1.56736666666667, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases
```

```
## Warning in bs(dis, degree = 3L, knots = c(`7.692308%` = 1.5895, `15.38462%`
## = 1.8498, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = c(`7.692308%` = 1.5895, `15.38462%`
## = 1.8498, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases
```

```
## Warning in bs(dis, degree = 3L, knots = c(`7.692308%` = 1.5331, `15.38462%`
## = 1.77, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = c(`7.692308%` = 1.5331, `15.38462%`
## = 1.77, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases
```

```
## Warning in bs(dis, degree = 3L, knots = c(`7.142857%` = 1.51814285714286, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`7.142857%` = 1.51814285714286, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases
```

```
## Warning in bs(dis, degree = 3L, knots = c(`6.666667%` = 1.55251333333333, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`6.666667%` = 1.55251333333333, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases
```

```
## Warning in bs(dis, degree = 3L, knots = c(`6.666667%` = 1.51914666666667, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`6.666667%` = 1.51914666666667, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases
```

```
## Warning in bs(dis, degree = 3L, knots = c(`6.25%` = 1.521275, `12.5%` =
## 1.74745, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = c(`6.25%` = 1.521275, `12.5%` =
## 1.74745, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases
```

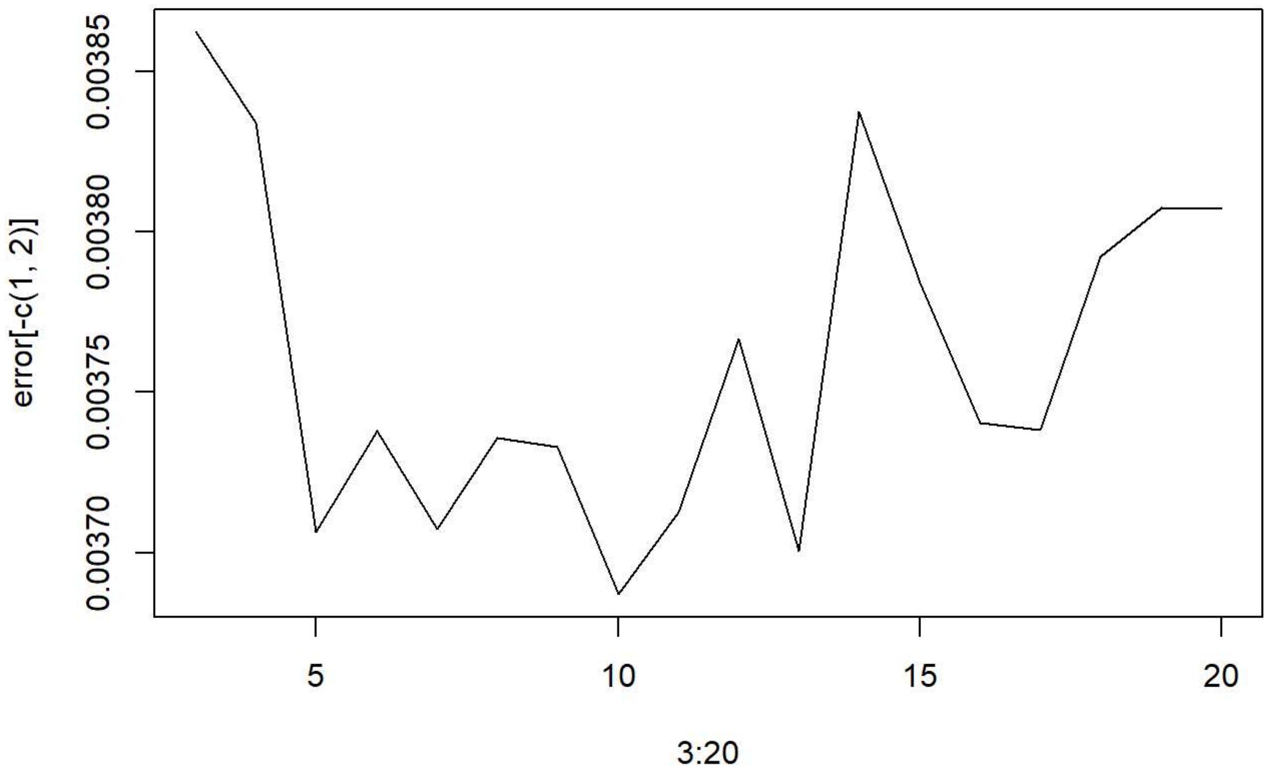
```
## Warning in bs(dis, degree = 3L, knots = c(`6.25%` = 1.512475, `12.5%` =
## 1.738975, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = c(`6.25%` = 1.512475, `12.5%` =
## 1.738975, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases
```

```
## Warning in bs(dis, degree = 3L, knots = c(`5.882353%` = 1.52554705882353, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`5.882353%` = 1.52554705882353, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases
```

```
## Warning in bs(dis, degree = 3L, knots = c(`5.882353%` = 1.50565882352941, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`5.882353%` = 1.50565882352941, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases
```

```
## Warning in bs(dis, degree = 3L, knots = c(`5.555556%` = 1.49893333333333, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(`5.555556%` = 1.49893333333333, :
## some 'x' values beyond boundary knots may cause ill-conditioned bases
```

```
plot(3:20, error[-c(1,2)], type = "l")
```



The

cross validated error decreases until 10 degrees of freedom.

8.5

```
x = c(0.1, 0.15, 0.2, 0.2, 0.55, 0.6, 0.6, 0.65, 0.7, 0.75)
# Majority Method
sum(x >= 0.5) > sum(x < 0.5)
```

```
## [1] TRUE
```

```
# Averaging Method
mean(x)
```

```
## [1] 0.45
```

For the majority method, the number of red predictions is greater than the number of green predictions, giving red as the result. For the averaging method, the average probability is 0.45, giving a green result.

8.8 a)

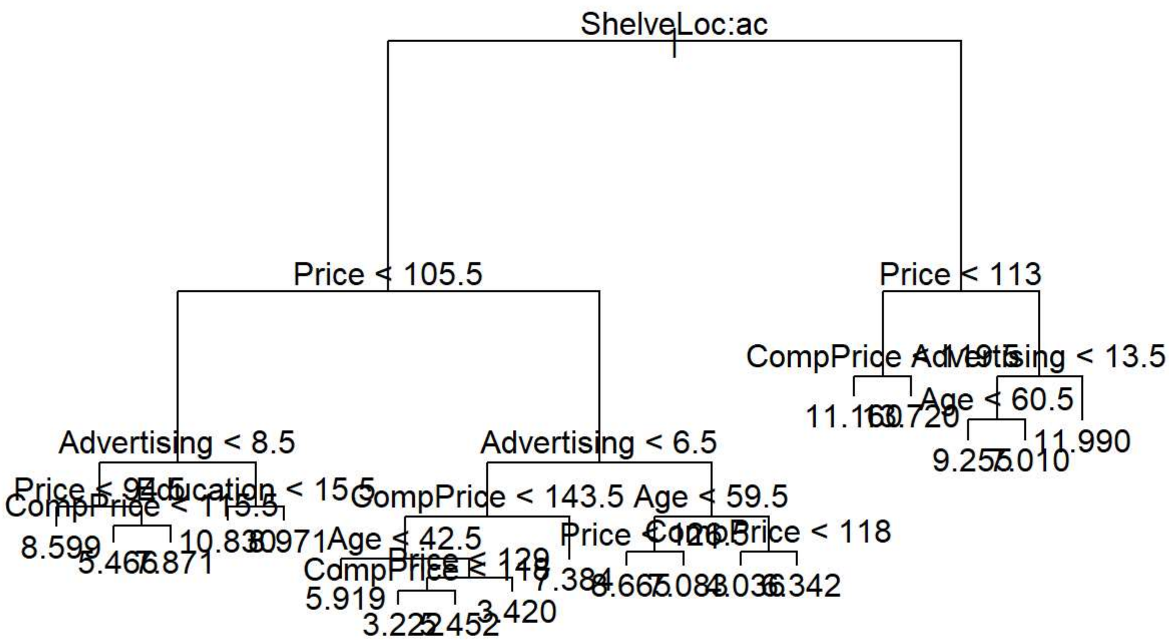
```r
library(ISLR)
attach(Carseats)

samp = sample(dim(Carseats)[1], dim(Carseats)[1]/2)
train = Carseats[samp, ]
test = Carseats[-samp, ]
```

b.

```r
library(tree)
tree = tree(Sales ~ ., data = train)
summary(tree)
```

```
##
## Regression tree:
## tree(formula = Sales ~ ., data = train)
## Variables actually used in tree construction:
## [1] "ShelveLoc"   "Price"       "Advertising" "CompPrice"   "Education"
## [6] "Age"
## Number of terminal nodes:  19
## Residual mean deviance:  2.066 = 373.9 / 181
## Distribution of residuals:
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -3.43000 -1.00700 -0.04369  0.00000  0.85740  3.05100
```

```r
plot(tree)
text(tree)
```



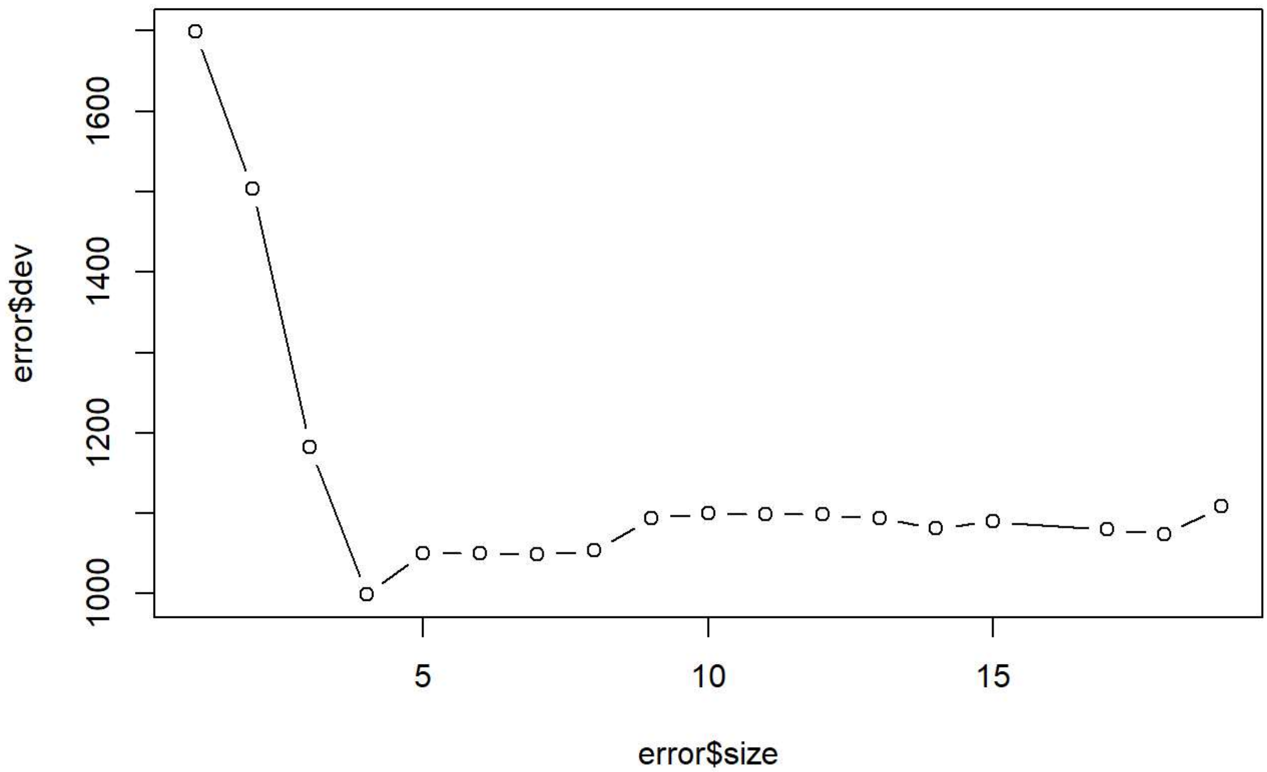

```
preds = predict(tree, test)
mean((test$Sales - preds)^2)
```

```
## [1] 5.428867
```

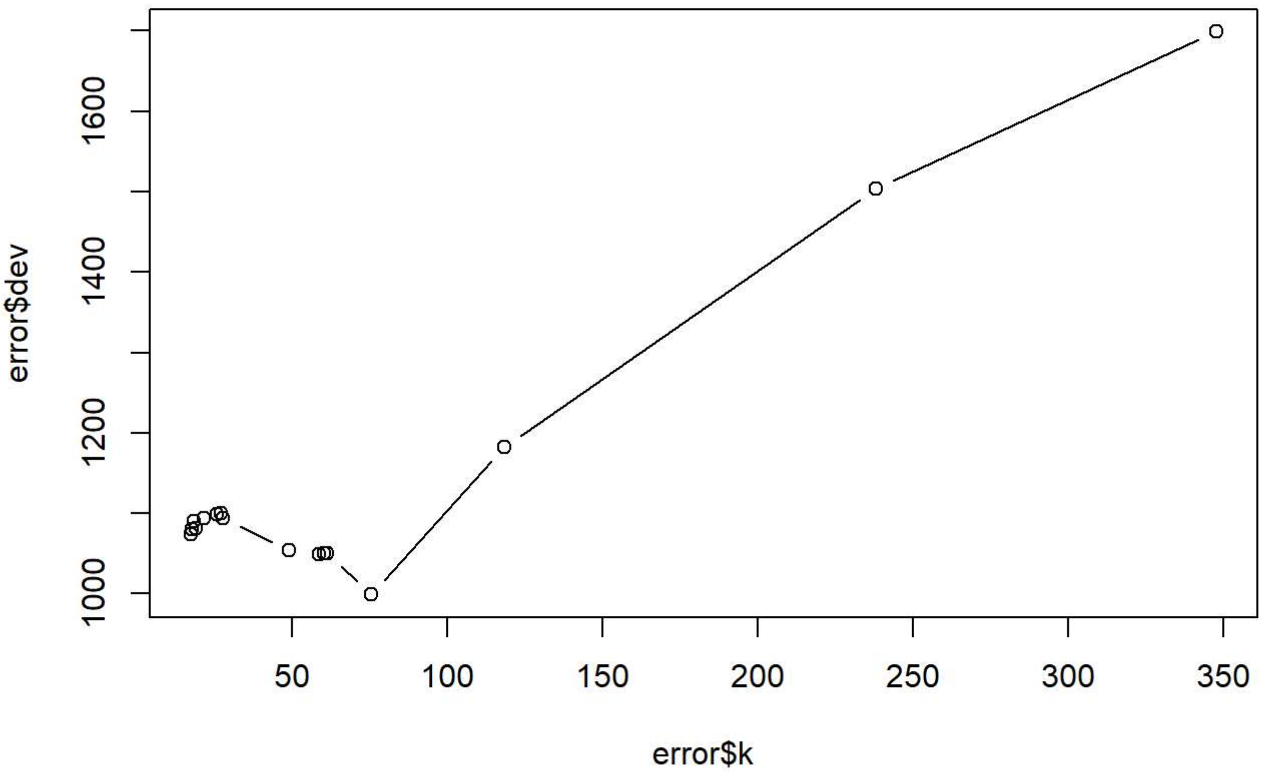
The RSS is 5.57.

c.

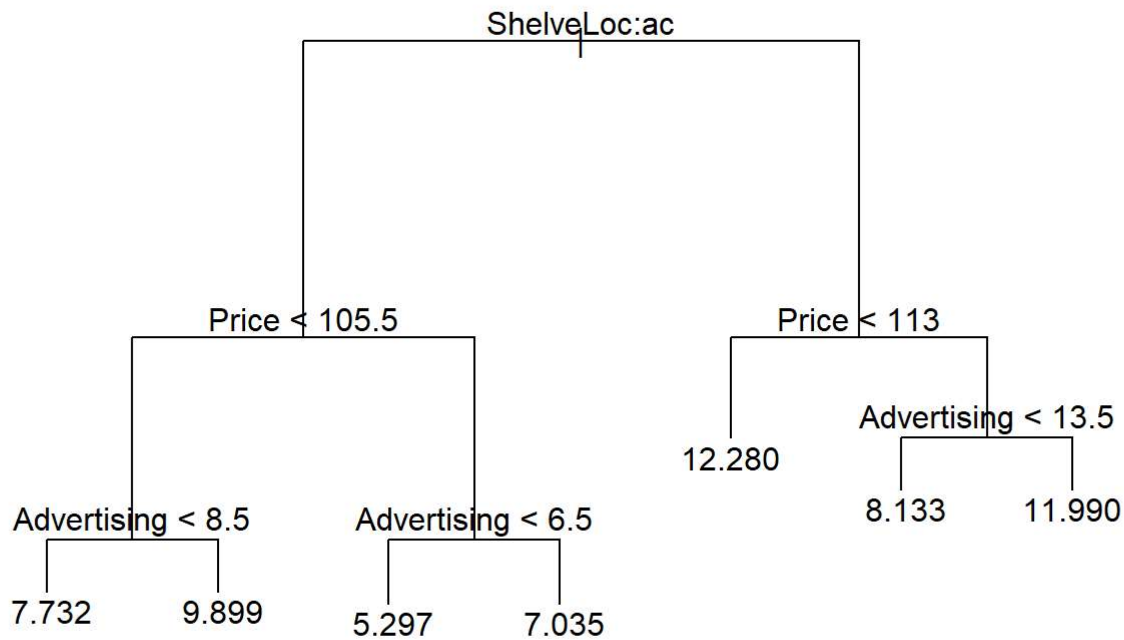
```
error = cv.tree(tree, FUN = prune.tree)
plot(error$size, error$dev, type = "b")
```




```
plot(error$k, error$dev, type = "b")
```




```
# We will use size of 7.
pruned = prune.tree(tree, best = 7)
plot(pruned)
text(pruned)
```




```
pruned_preds = predict(pruned, test)
mean((test$Sales - pruned_preds)^2)
```

```
## [1] 5.496529
```

Pruning the tree increases the RSS to 5.807

d.

```
library(randomForest)
```

```
## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
bag = randomForest(Sales ~ ., data = train, mtry = 10, ntree = 500, importance = T)
bag_preds = predict(bag, test)
mean((test$Sales - bag_preds)^2)
```

```
## [1] 2.819815
```

```
importance(bag)
```

```
##                %IncMSE IncNodePurity
## CompPrice   21.2092538    149.162721
## Income       2.3215667     67.656749
## Advertising 21.2220150    186.059129
## Population  -0.6317910     56.495913
## Price       58.2382344    501.937027
## ShelveLoc   48.4278584    378.759193
## Age         20.7951378    152.911879
## Education   -0.7444260     38.387545
## Urban        0.7313579      5.986053
## US           3.0473889     12.570747
```

Bagging improves test MSE up to 3.41. Price, ShelveLoc and Advertising are three most important predictors of Sale.

e.

```
rf = randomForest(Sales ~ ., data = train, mtry = 5, ntree = 500,
    importance = T)
rf_preds = predict(rf, test)
mean((test$Sales - rf_preds)^2)
```

```
## [1] 2.960797
```

```
importance(rf)
```

```
##                %IncMSE IncNodePurity
## CompPrice   14.0896124    142.313202
## Income       0.5347156     84.270096
## Advertising 18.5597501    182.766695
## Population   2.1944042     82.709218
## Price       46.9097371    460.873983
## ShelveLoc   41.9075703    331.553221
## Age         12.8285278    156.067798
## Education    1.2774812     54.208948
## Urban       -0.9294751      8.686444
## US           4.1599170     30.188641
```

Random forest increases the test MSE to 3.56. Changes in m vary test MSE between 3 to 4. We again see that Price, ShelveLoc and Advertising are the best predictors.