Sam Lin
Stats 202
Hw. 3

Chapter 6, Exercise 3 (p. 260)
a) Training RSS will steadily decrease, since increasing s from 0 will restrict the B_j coefficients, which will increase coefficients to least squares estimates, which make the model more flexible.

b) Test RSS will decrease initially and then eventually start increasing in a U shape. When we increase s from 0, B_j coefficients become more restricted which makes the model more flexible, which make the an initial decrease and then increasing again.

c) Variance will Steadily increase, with similar reasoning as above the more flexible model will give a steadily increasing variance (amount of new function f would change if we estimated it using a different set of data)

d) Bias will steadily decrease, since the model becomes steadily more flexible the steadily decreases because the bias calculates the error for the particular model. (Ex: high-bias corresponds to model like linear regression)

e) Irreducible error remains constant, because it is independent of the model which means independent of s

Chapter 6, Exercise 4 (p. 260)
a) Training RSS will steadily increase, because increasing lambda from 0 will restrict the B_j coefficients causing the model to become less flexible which increases the training RSS.

b) Test RSS will decrease initially and eventually start increasing in a U shape, because the less flexible model

c) Variance will steadily decrease, since a less flexible model corresponds to a less change in function, then variance would steadily decrease.

d) Bias will steadily increase, because a less flexible model corresponds higher bias. See example from previous question part d)

e) Same as previous part e) where the irreducible error in independent of model

Chapter 6, Exercise 9 (p. 263). Don't do parts (e), (f), and (g)
a)

```
train = sample(1:dim(College)[1], dim(College)[1] / 2)
test <- -train
College.train <- College[train, ]
College.test <- College[test, ]
```

b)

Sam Lin
Stats 202
Hw. 3

```
> fit.lm <- lm(Apps ~ ., data = College.train)
> pred.lm <- predict(fit.lm, College.test)
> mean((pred.lm - College.test$Apps)^2)
[1] 1156314
```

c)

```
> train.mat <- model.matrix(Apps ~ ., data = College.train)
> test.mat <- model.matrix(Apps ~ ., data = College.test)
> grid <- 10 ^ seq(4, -2, length = 100)
> fit.ridge <- glmnet(train.mat, College.train$Apps, alpha = 0, lambda = grid, thresh = 1e-12)
> cv.ridge <- cv.glmnet(train.mat, College.train$Apps, alpha = 0, lambda = grid, thresh = 1e-12)
> bestlam.ridge <- cv.ridge$lambda.min
> bestlam.ridge
[1] 18.73817
> pred.ridge <- predict(fit.ridge, s = bestlam.ridge, newx = test.mat)
> mean((pred.ridge - College.test$Apps)^2)
[1] 1608859
```

The test MSE is higher for ridge regression than for least squares.

d)

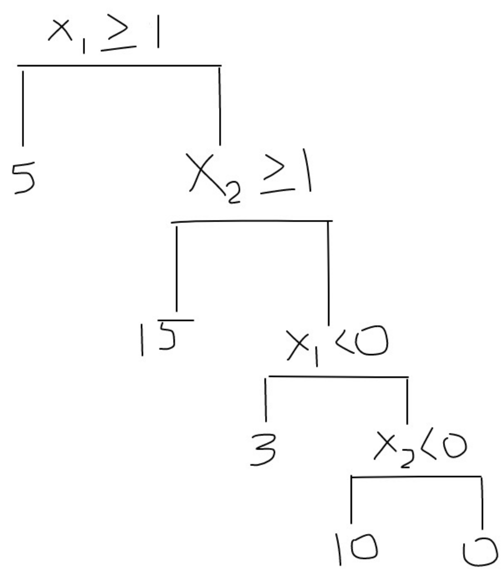
```
> fit.lasso <- glmnet(train.mat, College.train$Apps, alpha = 1, lambda = grid, thresh = 1e-12)
> cv.lasso <- cv.glmnet(train.mat, College.train$Apps, alpha = 1, lambda = grid, thresh = 1e-12)
> bestlam.lasso <- cv.lasso$lambda.min
> bestlam.lasso
[1] 21.54435
> pred.lasso <- predict(fit.lasso, s = bestlam.lasso, newx = test.mat)
> mean((pred.lasso - College.test$Apps)^2)
[1] 1635280
```

Test MSE is higher for ridge regression than for least squares.

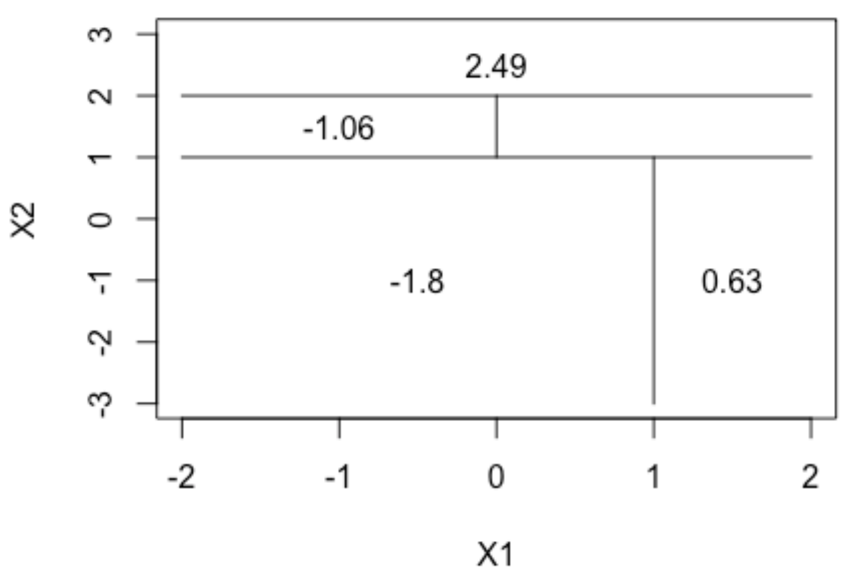
```
> predict(fit.lasso, s = bestlam.lasso, type = "coefficients")
19 x 1 sparse Matrix of class "dgCMatrix"
                        1
(Intercept) -836.50402310
(Intercept)    .
PrivateYes  -385.73749394
Accept         1.17935134
Enroll         .
Top10perc     22.70211938
Top25perc      .
F.Undergrad    0.07062149
P.Undergrad    0.01366763
Outstate      -0.03424677
Room.Board     0.01281659
Books         -0.02167770
Personal       .
PhD           -1.46396964
Terminal      -5.17281004
S.F.Ratio      5.70969524
perc.alumni   -9.95007567
Expend         0.14852541
Grad.Rate      5.79789861
```

The above are the non-zero coefficient estimates

Sam Lin
Stats 202
Hw. 3
Chapter 8, Exercise 4 (p. 332)
a)



b)

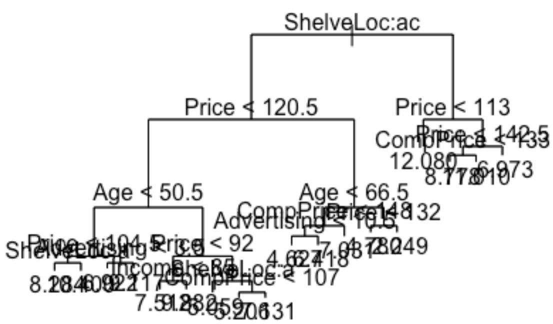


Chapter 8, Exercise 8 (p. 333)
a)

```
> train <- sample(1:nrow(Carseats), nrow(Carseats) / 2)
> Carseats.train <- Carseats[train, ]
> Carseats.test <- Carseats[-train, ]
```

Sam Lin
Stats 202
Hw. 3
b)

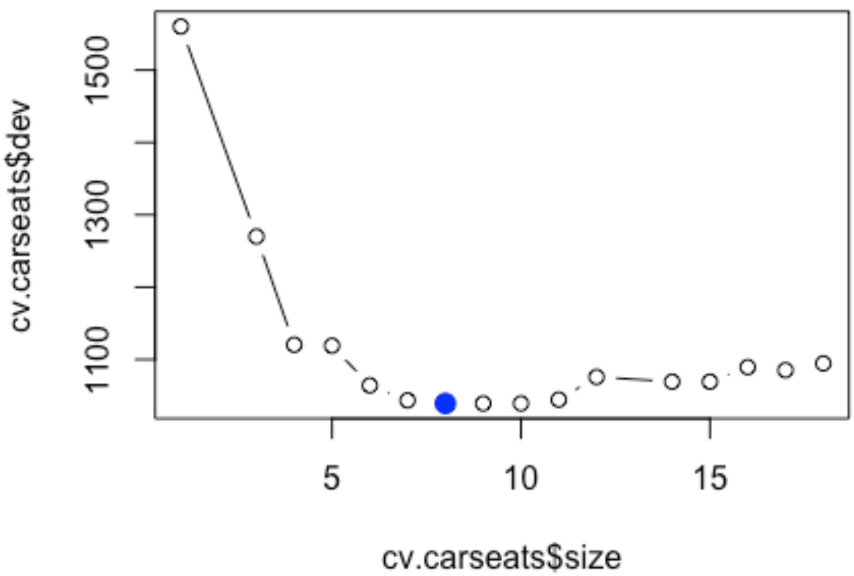
```
> tree.carseats <- tree(Sales ~ ., data = Carseats.train)
> summary(tree.carseats)

Regression tree:
tree(formula = Sales ~ ., data = Carseats.train)
Variables actually used in tree construction:
[1] "ShelveLoc"   "Price"       "Age"         "Advertising" "Income"      "CompPrice"
Number of terminal nodes:  18
Residual mean deviance:  2.36 = 429.5 / 182
Distribution of residuals:
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
-4.2570 -1.0360  0.1024  0.0000  0.9301  3.9130
>
```



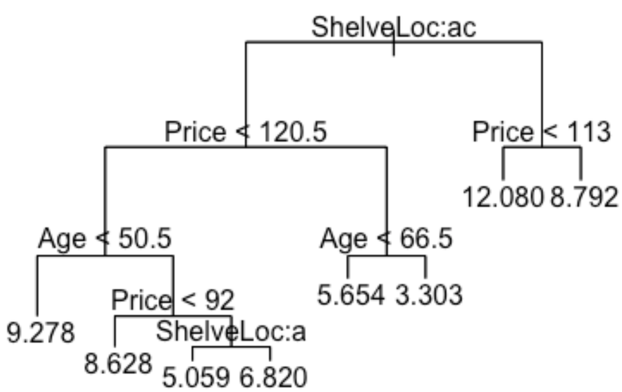

```
> yhat <- predict(tree.carseats, newdata = Carseats.test)
> mean((yhat - Carseats.test$Sales)^2)
[1] 4.148897
```

Test error rate is approximately 4.15.
c)

Sam Lin
Stats 202
Hw. 3



Tree of size 8 used for cross-validation. Prune the tree to get 8-node tree.




```
> yhat <- predict(prune.carseats, newdata = Carseats.test)
> mean((yhat - Carseats.test$Sales)^2)
[1] 5.09085
```

Pruning the tree increases the MSE from 4.15 to 5.1

Sam Lin
Stats 202
Hw. 3
d)

```
> bag.carseats <- randomForest(Sales ~ ., data = Carseats.train, mtry = 10, ntree = 500, importance = TRUE)
> yhat.bag <- predict(bag.carseats, newdata = Carseats.test)
> mean((yhat.bag - Carseats.test$Sales)^2)
[1] 2.608497
```

Bagging decreases Test MSE to 2.6

```
> importance(bag.carseats)
              %IncMSE IncNodePurity
CompPrice   14.4604993    134.271455
Income       5.1630635     79.226841
Advertising 15.1596943    126.693875
Population  -0.5090127     61.871536
Price       54.0987066    512.663116
ShelveLoc   44.1614233    313.709331
Age         22.7634097    186.366024
Education    1.9349058     42.697069
Urban       -3.5084333      9.406872
US           5.7346214     13.805919
```

e)

```
> rf.carseats <- randomForest(Sales ~ ., data = Carseats.train, mtry = 3, ntree = 500, importance = TRUE)
> yhat.rf <- predict(rf.carseats, newdata = Carseats.test)
> mean((yhat.rf - Carseats.test$Sales)^2)
[1] 3.329819
```

Test MSE of 3.3

```
> importance(rf.carseats)
              %IncMSE IncNodePurity
CompPrice    8.0047894     128.82565
Income       4.7334488     126.66774
Advertising 11.4329207     133.37395
Population  -1.9254797     102.87323
Price       37.9087112     386.42580
ShelveLoc   29.4161401     239.51932
Age         15.8838551     197.73564
Education   -0.4601408      69.98180
Urban       -1.6965485      14.78663
US           6.6421646      32.24750
```