DSO 530 - Homework 3

Xueyan Gu

3/27/2019

ISLR Chapter 5

5.4 Exercises

1.Answer:

What we have is:

$$Var(\alpha X + (1-\alpha)Y) = \alpha^2\sigma_X^2 + (1-\alpha)^2\sigma_Y^2 + 2\alpha(1-\alpha)\sigma_{XY}$$

Then, we take the first derivative of $Var(\alpha X + (1-\alpha)Y)$ and we can get:

$$\frac{\partial}{\partial\alpha}Var(\alpha X + (1-\alpha)Y) = 2\alpha\sigma_X^2 - 2\sigma_Y^2 + 2\alpha\sigma_Y^2 + 2\sigma_{XY} - 4\alpha\sigma_{XY}$$

Next, we can make the right equation to be 0:

$$2\alpha\sigma_X^2 - 2\sigma_Y^2 + 2\alpha\sigma_Y^2 + 2\sigma_{XY} - 4\alpha\sigma_{XY} = 0$$

We can have:

$$\alpha = \frac{\sigma_Y^2 - \sigma_X Y}{\sigma_X^2 + \sigma_Y^2 - 2\sigma_{XY}}$$

In order to prove that it is a minimum, we can take the second derivative and prove that it is positive. So we can have:

$$\frac{\partial^2}{\partial\alpha^2}Var(\alpha X + (1-\alpha)Y) = 2\sigma_X^2 + 2\sigma_Y^2 - 4\sigma_{XY} = 2Var(X-Y) \geq 0$$

3.Answer:

(a) K-fold Cross-Validation involves randomly dividing the set of observations into k non-overlapping groups of approximately equal size. The first fold is treated as a validation set, and the remaining folds acts as a training set. The test error is computed by averaging the k resulting MSE (mean squared error) estimates.

(b)

i. The validation set approach has two drawbacks compared to k-fold cross-validation.

(1) Firstly, the validation estimate of the test error rate can be highly variable, depending on precisely which observations are included in the training set and which observations are included in the validation set.

(2) Secondly, in the validation approach, only a subset of the observations - those that are included in the training set rather than in the validation set - are used to fit the model. This suggests that the validation set error rate may tend to overestimate the test error rate for the model fit on the entire data set.

ii. The LOOCV approach is a special case of k-fold cross-validation in which k = n. It has two drawbacks compared to k-fold cross-validation.

(1) Firstly, the LOOCV requires fitting the potentially expensive model n times compared to k-fold cross-validation which requires the model to be fitted only k times.

(2) Secondly, the LOOCV approachtypically doesn't shake up the data enough. The estimates from each fold are highly correlated and hence their average can have higher variance than k-fold cross-validation. It is better to use k = 5 or k = 10 yielding test error rate.

5.Answer:

(b)

i.

```
# Split the data into training set and validation set

library(ISLR)
attach(Default)
set.seed(1)
training = sample(dim(Default)[1], dim(Default)[1] / 2)
```

ii.

```
# Fit a multiple logistic regression model

fit.glm = glm(default ~ income + balance, data = Default, family = "binomial", subset = training)
summary(fit.glm)
```

```
##
## Call:
## glm(formula = default ~ income + balance, family = "binomial",
##     data = Default, subset = training)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.3583  -0.1268  -0.0475  -0.0165   3.8116
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.208e+01  6.658e-01 -18.148   <2e-16 ***
## income       1.858e-05  7.573e-06   2.454   0.0141 *
## balance      6.053e-03  3.467e-04  17.457   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1457.0  on 4999  degrees of freedom
## Residual deviance:  734.4  on 4997  degrees of freedom
## AIC: 740.4
##
## Number of Fisher Scoring iterations: 8
```

iii.

```
# Obtain a prediction of default status for each individual in the validation set

prob = predict(fit.glm, newdata = Default[-training, ], type = "response")
pred.glm = rep("No", length(prob))
pred.glm[prob > 0.5] = "Yes"
```

iv.

Compute the validation set error

```
mean(pred.glm != Default[-training,]$default)
```

```
## [1] 0.0286
```

Based on the result above, the test error rate is 2.86% using the validation set approach.

(c)

```
training = sample(dim(Default)[1], dim(Default)[1] / 2)
fit.glm = glm(default ~ income + balance, data = Default, family = "binomial", subset = training)
prob = predict(fit.glm, newdata = Default[-training, ], type = "response")
pred.glm = rep("No", length(prob))
pred.glm[prob > 0.5] = "Yes"
mean(pred.glm != Default[-training,]$default)
```

```
## [1] 0.0236
```

```
training = sample(dim(Default)[1], dim(Default)[1] / 2)
fit.glm = glm(default ~ income + balance, data = Default, family = "binomial", subset = training)
prob = predict(fit.glm, newdata = Default[-training, ], type = "response")
pred.glm = rep("No", length(prob))
pred.glm[prob > 0.5] = "Yes"
mean(pred.glm != Default[-training,]$default)
```

```
## [1] 0.028
```

```
training = sample(dim(Default)[1], dim(Default)[1] / 2)
fit.glm = glm(default ~ income + balance, data = Default, family = "binomial", subset = training)
prob = predict(fit.glm, newdata = Default[-training, ], type = "response")
pred.glm = rep("No", length(prob))
pred.glm[prob > 0.5] = "Yes"
mean(pred.glm != Default[-training,]$default)
```

```
## [1] 0.0268
```

Based on the results above, when we repeat the process in (b), the validation estimates of test error vary, depending on precisely which observations are included in the training set and which observations are included in the validation set.

ISLR Chapter 8

8.4 Exercises

5.Answer:

(1) If we use the majority vote approach, we can get that 6 predictions are Red and 4 predictions are Green. Therefore, we will classify X as Red because it is the most common class among the 10 predictions.
(2) If we use the average probability approach, the average of the 10 probabilities is $(0.1 + 0.15 + 0.2 + 0.2 + 0.55 + 0.6 + 0.6 + 0.65 + 0.7 + 0.75)/10 = 0.45$. Therefore, we will classify X as Red.
(3) The final classification of X under each of these two approaches is Red.

8.Answer:

(a)

```r
# Split the data set into a training set and a test set

library(ISLR)
set.seed(2)

training = sample(1:nrow(Carseats), 200)
Carseats_training = Carseats[training, ]
Carseats_test = Carseats[-training, ]
```

(b)

```r
# Fit a regression tree to the training set

library(tree)
tree_carseats = tree(Sales ~ ., data = Carseats_training)
summary(tree_carseats)
```

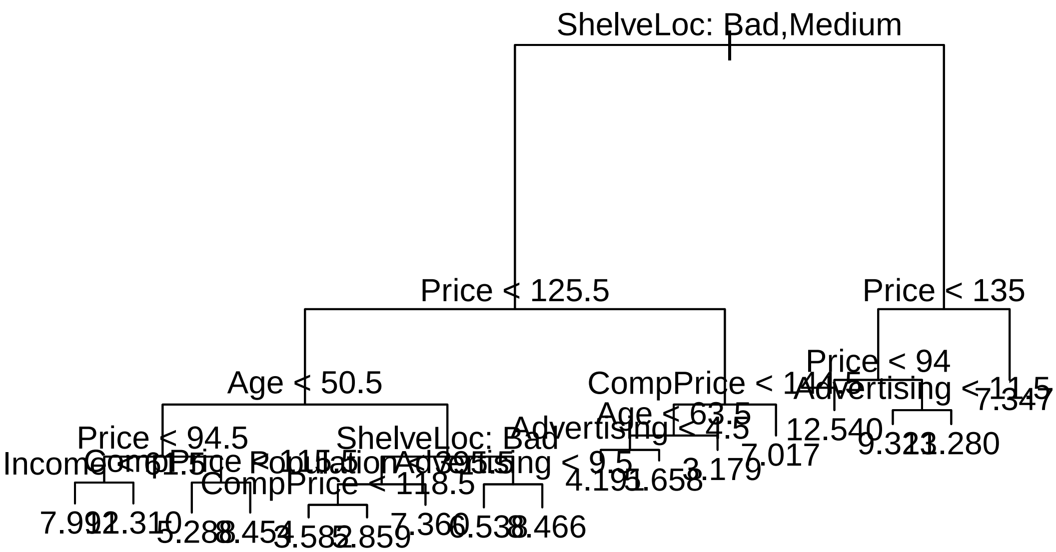
```
##
## Regression tree:
## tree(formula = Sales ~ ., data = Carseats_training)
## Variables actually used in tree construction:
## [1] "ShelveLoc"   "Price"       "Age"         "Income"      "CompPrice"
## [6] "Population"  "Advertising"
## Number of terminal nodes:  17
## Residual mean deviance:  2.341 = 428.4 / 183
## Distribution of residuals:
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -3.76700 -1.00900 -0.01558  0.00000  0.94900  3.58600
```

```r
library(maptree)
```

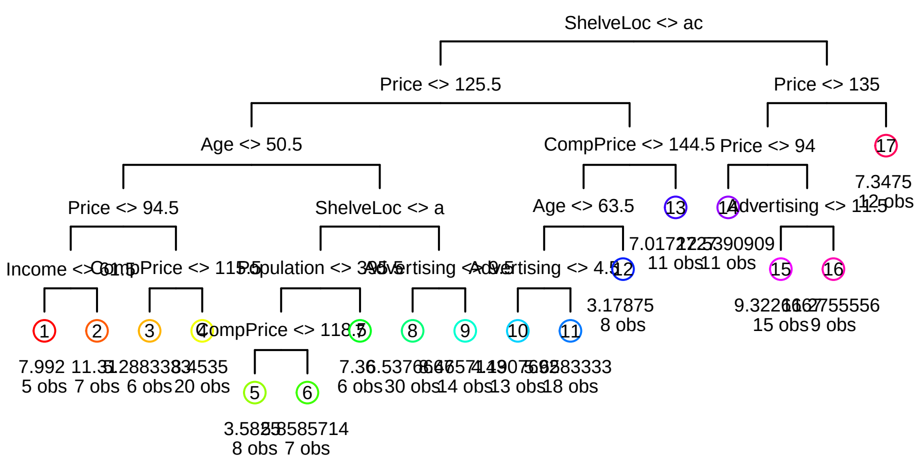
```
## Loading required package: cluster
```

```
## Loading required package: rpart
```

```r
plot(tree_carseats)
text(tree_carseats, pretty = 0)
```




```r
draw.tree(tree_carseats,cex=0.6)
```



As we can see from the graph above, we can interpret the results. For example, if ShelveLoc is [bad, medium], and price is smaller than 125.5, and age is smaller than 50.0, and price is smaller than 94.5, and income is smaller than 6, then the predicted outcome is the mean value of sales in that node, which is 7.9925 obs.

```r
pred = predict(tree_carseats, newdata = Carseats_test)
mean((pred - Carseats_test$Sales)^2)
```

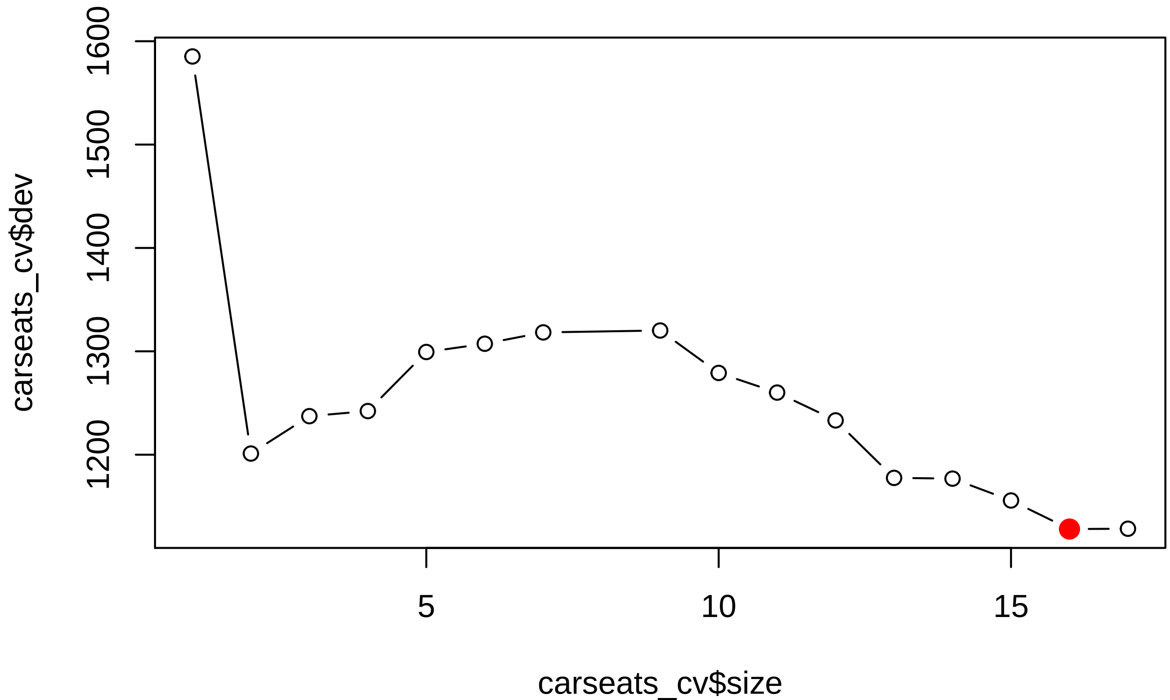
```
## [1] 4.844991
```

Based on the result above, we can get that the Test MSE is 4.844991.

(c)

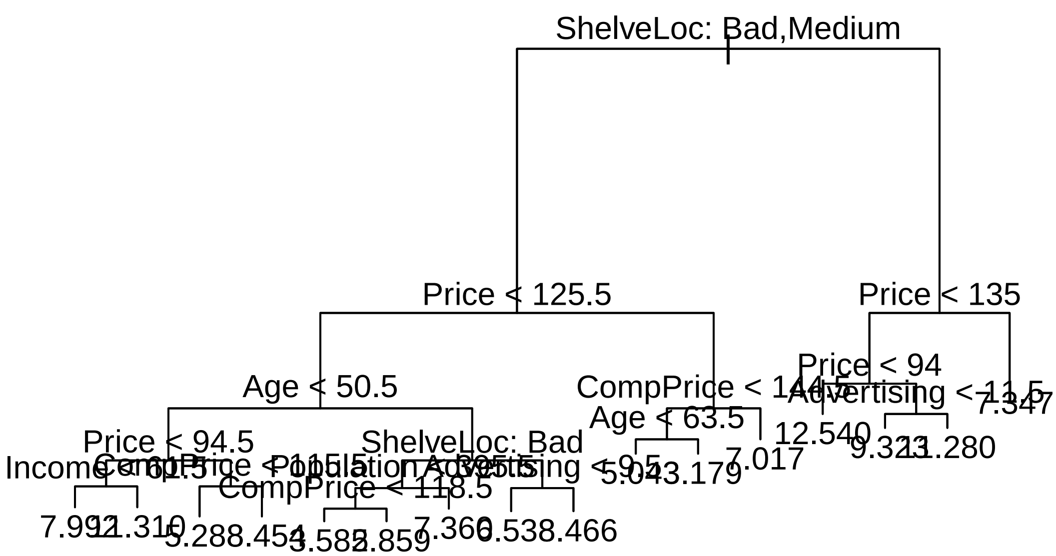
```r
# Use cross-validation

carseats_cv = cv.tree(tree_carseats)
plot(carseats_cv$size, carseats_cv$dev, type = "b")
tree_min = which.min(carseats_cv$dev)
points(carseats_cv$size[tree_min], carseats_cv$dev[tree_min], col = "red", cex = 2, pch = 20)
```

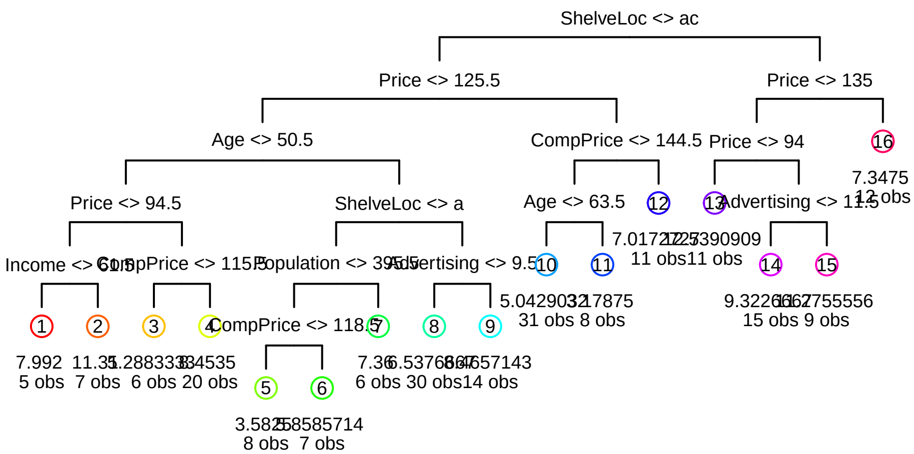


In this case, the tree of size 16 is selected by cross-validation. We now prune the tree to obtain the 16-node tree.

```r
prune_carseats = prune.tree(tree_carseats, best = 16)
plot(prune_carseats)
text(prune_carseats, pretty = 0)
```




```r
draw.tree(prune_carseats,cex=0.6)
```




```r
pred2 = predict(prune_carseats, newdata = Carseats_test)
mean((pred2 - Carseats_test$Sales)^2)
```

```
## [1] 4.893985
```

Based on the result, we can see that pruning the tree increases the Test MSE to 4.893985.

(d)

```r
# Use the bagging approach

library(randomForest)
```

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

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

```r
bag_carseats = randomForest(Sales ~ ., data = Carseats_training, mtry = 10, ntree = 500, importance = T
pred3 = predict(bag_carseats, newdata = Carseats_test)
mean((pred3 - Carseats_test$Sales)^2)
```

```
## [1] 2.369187
```

Based on the result, we can see that bagging decreases the Test MSE to 2.369187.

```r
importance(bag_carseats)
```

```
##                %IncMSE IncNodePurity
## CompPrice   26.8209582    166.979714
## Income       2.5178689     70.424671
## Advertising 12.7943382     95.674806
## Population   1.5809962     66.767407
## Price       57.3318051    477.292357
## ShelveLoc   50.8691964    475.187526
## Age         12.9786136    126.420511
## Education   -1.8091675     37.001724
## Urban       -3.5410771      5.936702
## US          -0.8447167      6.800383
```

Based on the results above, we can see that "Price" and "ShelveLoc" are the two most important variables.

(e)

```r
# Use random forests

```

```r
rf_carseats = randomForest(Sales ~ ., data = Carseats_training, mtry = 3, ntree = 500, importance = TRUI
pred4 = predict(rf_carseats, newdata = Carseats_test)
mean((pred4 - Carseats_test$Sales)^2)
```

```
## [1] 2.961581
```

In this case, with $m = \sqrt{p}$, we can see that Test MSE is 2.961581.

```r
importance(rf_carseats)
```

```
##                %IncMSE IncNodePurity
## CompPrice   13.9082873     143.70588
## Income      -1.1881180      95.40114
## Advertising  8.1765225     116.01780
## Population  -1.4440692     112.32290
## Price       37.5120708     386.90793
## ShelveLoc   39.9627269     350.46676
## Age         11.8981617     158.21652
## Education   -0.5830359      67.83125
## Urban       -0.7235525      14.13967
## US           0.5890730      16.34130
```

Based on the results above, we can see that "Price" and "ShelveLoc" are the two most important variables.