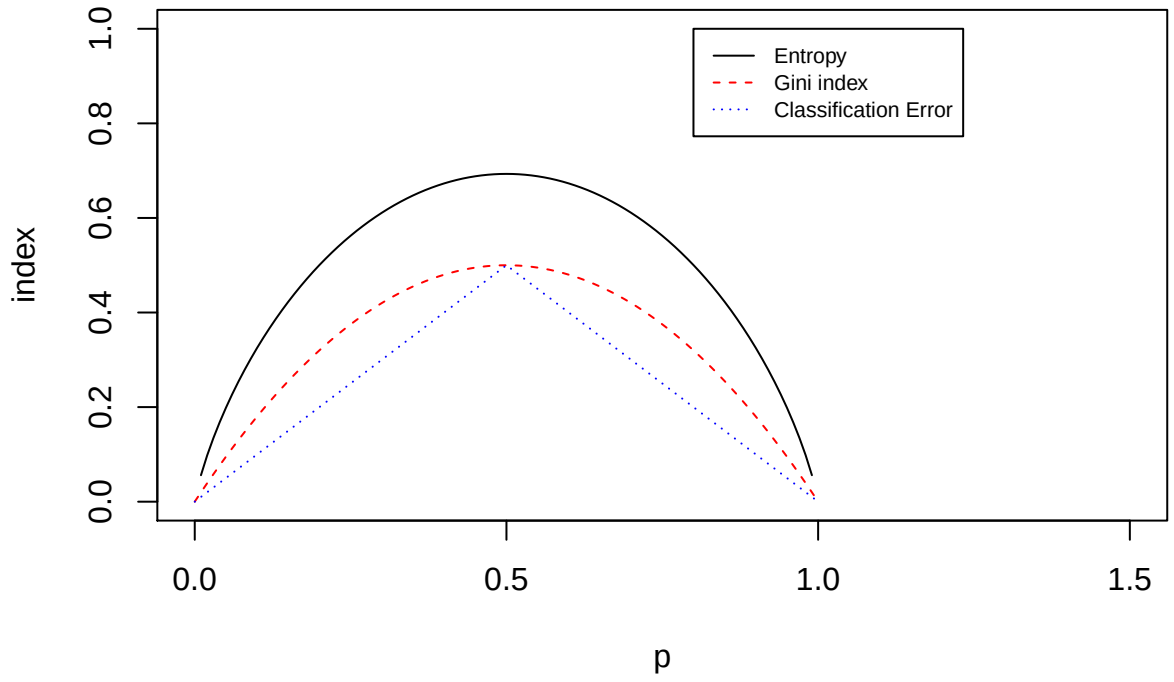
STAT 8330 FALL 2015 ASSIGNMENT 5

Peng Shao

October 8, 2015

▶ **Exercises 8.3. Solution.**

```
p <- seq(0, 1, 0.01)
error <- 1-apply(cbind(p, 1 - p), 1, max)
gini <- 2*p*(1 - p)
entropy <- -p * log(p) - (1 - p) * log(1 - p)
plot(c(0, 1.5), c(0, 1),
     type = "n",
     xlab = "p",
     ylab = "index")
lines(p, entropy, lty = 1)
lines(p, gini, col = "red", lty = 2)
lines(p, error, col = "blue", lty = 3)
legend(0.8, 1,
       c("Entropy", "Gini index", "Classification Error"),
       lty=c(1, 2, 3),
       col = c("black", "red", "blue"),
       cex = .7)
```



▶ **Exercises 8.5. Solution.**

```
probs <- c(0.1, 0.15, 0.2, 0.2, 0.55, 0.6, 0.6, 0.65, 0.7, 0.75)
vote <- ifelse(sum(probs > 0.5) > length(probs)/2, "RED", "NOT RED")
average <- ifelse(mean(probs) > 0.5,"RED", "NOT RED")
```

For the majority vote approach, the result is

```
vote
```

```
## [1] "RED"
```

And for average probability approach, the result is

```
average
```

```
## [1] "NOT RED"
```

▶ **Exercises 3. Solution.** I set the seed(1) at the beginning of problem 3, and I will not reset it again within this problem.

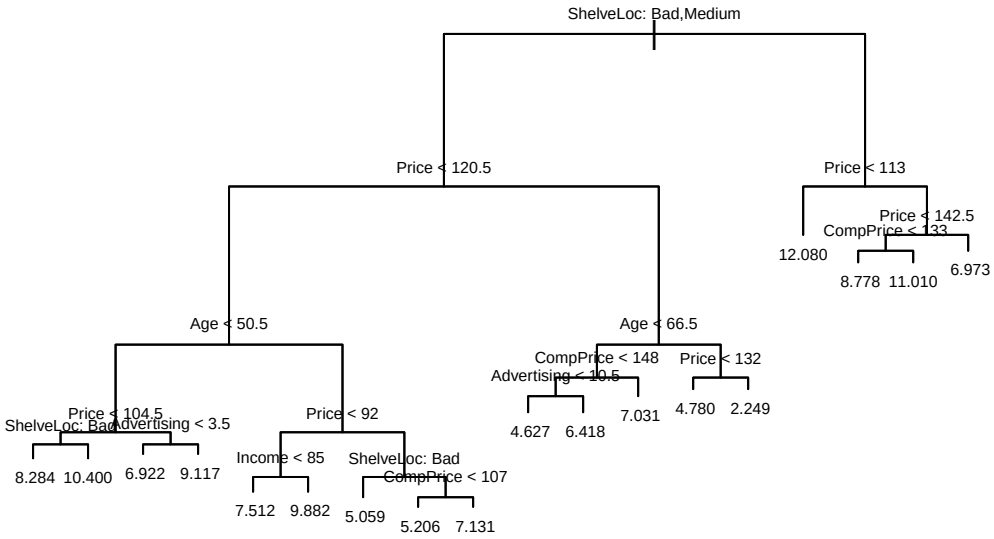
(a). This dataset has 10 candidate predictors, but the regression tree only use six of them, which are

```
as.character(summary(tree_carseats)$used)
```

```
## [1] "ShelveLoc"   "Price"       "Age"         "Advertising" "Income"
## [6] "CompPrice"
```

The regression tree is

```
plot(tree_carseats)
text(tree_carseats,
     pretty = 0,
     cex = 0.5)
```



We can get some knowledge fron the tree: 1) good quality of the shelving location for the car seats will result in high sales; 2) lower price and higher competitor price will result in higher sales; 3) the site with younger local population will have higher sales; 4) the more advertisement, the more sales; 5) the richer people are, the more they will buy child car seats. So, the good quality with very low price (less than 113) has the highest sales. Actually, these findings seem not so surprising and they are very reasonable.

The test MSE of this tree is 4.1488975.

(b). Using 10 fold cross-validation, the optimal level of tree complexity is

```
summary(prune_carseats)$size
```

```
## [1] 10
```

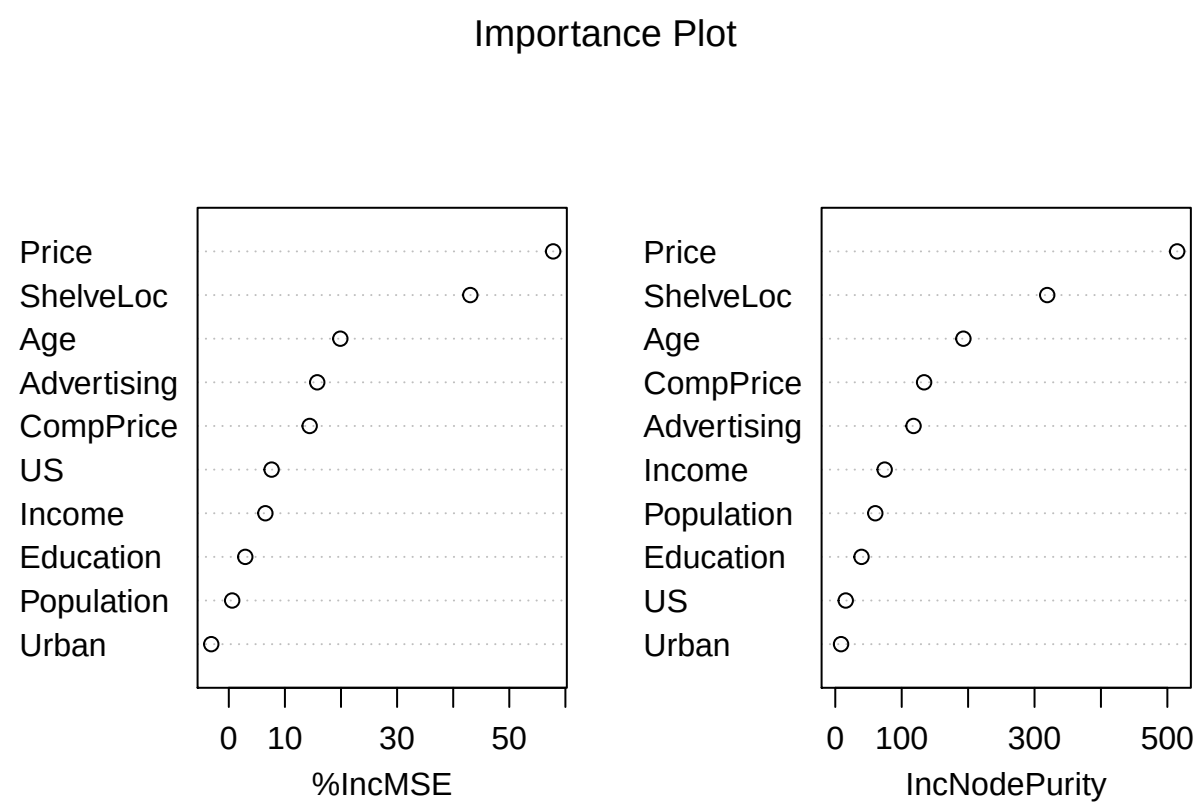
And the test MSE of pruned tree is 4.8197081. We can see it does not improve the test MSE.

(c). The test MSE of bagging is 2.6043694. The importances are shown below

```
importance(bag_carseats)
```

```
##                %IncMSE IncNodePurity
## CompPrice   14.4124562    133.731797
## Income       6.5147532     74.346961
## Advertising 15.7607104    117.822651
## Population   0.6031237     60.227867
## Price       57.8206926    514.802084
## ShelveLoc   43.0486065    319.117972
## Age         19.8789659    192.880596
## Education    2.9319161     39.490093
## Urban       -3.1300102      8.695529
## US           7.6298722     15.723975
```

```
varImpPlot(bag_carseats, main = "Importance Plot")
```



Thus, the most important variables are Price, ShelveLoc.

(d). Since this is a regression tree problem, so the distribution options should be `distribution = "gaussian"`, and I use 10 fold cross-validaiton to choose the best the parameteres from the candidate combinations:

- $B \in \{1000, 2000, 3000, 4000, 5000\}$;
- $d \in \{1, 2, 3, 4\}$;
- $\lambda \in \{0.01, 0.1\}$.

Then the best result from the program is

```
Best_B
```

```
## [1] 3000
```

```
Best_d
```

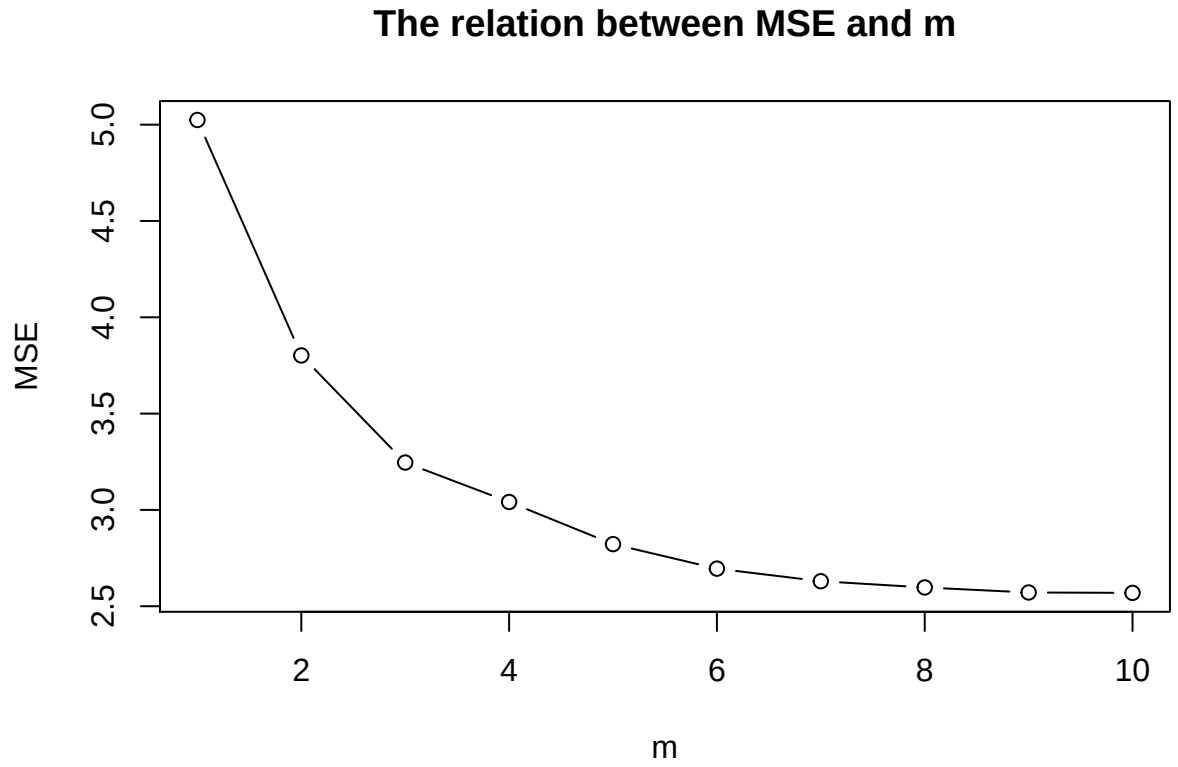
```
## [1] 1
```

```
Best_lambda
```

```
## [1] 0.01
```

That is the best "n.trees" is 3000, and the best "interaction.depth" is 1. The associated MSE is 1.4756.

(e) Usually, we have some suggestions like $m = \sqrt{(p)}, \frac{\sqrt{p}}{2}, 2\sqrt{p}$ or $\frac{p}{3}$ when we have large number features, to reduce the dimensionality. Here we only have 10 features so we can try every possible value of m. The plot of MSE v.s the value of m is



So we can see that the best forest with lowerest MSE – which is 2.5693 – uses all features instead of choosing some of them, which may probabily indicates that in such low dimensional problem, we do not need to do some pruning work to void overfitting. Another good result we can get is that, no matter what m is, the most important varibles are always "Price" and "ShelveLoc".

(f). Using the function `earth()` in the library `earth`, we can select the best model from 10-fold GCV based cross-validation by the following code,

```
mars_carseats <- earth(Sales ~ .,
                       data = carseats_train,
                       pmethod = "cv",
                       nfold = 10)
```

And the best model is

```
summary(mars_carseats)
```

```
## Call: earth(formula=Sales~., data=carseats_train, pmethod="cv", nfold=10)
##
##                  coefficients
```

```
## (Intercept)        -1.0410646
## ShelveLocGood       4.8629493
## ShelveLocMedium     1.8894338
## h(119-CompPrice)   -0.0729304
## h(CompPrice-119)    0.0939897
## h(Income-100)       0.0744332
## h(Advertising-2)    0.1286920
## h(156-Price)        0.0969771
## h(Price-156)       -0.1178460
## h(76-Age)           0.0482479
##
## Selected 10 of 19 terms, and 7 of 11 predictors using pmethod="cv"
## Termination condition: Reached nk 23
## Importance: Price, ShelveLocGood, CompPrice, Advertising, Age, ...
## Number of terms at each degree of interaction: 1 9 (additive model)
## GRSq 0.8253297  RSq 0.8554993  mean.oof.RSq 0.8181181 (sd 0.0559)
##
## pmethod="backward" would have selected:
##     12 terms 8 preds,  GRSq 0.8287965  RSq 0.8645581  mean.oof.RSq 0.8110143
```

with GCV = 1.3465885 and test MSE = 1.141406.

Here, the basis functions are

```
dimnames(mars_carseats$cuts)[[1]][mars_carseats$selected.terms]
```

```
##  [1] "(Intercept)"      "h(Price-156)"     "h(156-Price)"
##  [4] "ShelveLocGood"    "h(CompPrice-119)" "h(119-CompPrice)"
##  [7] "h(Advertising-2)" "ShelveLocMedium"  "h(76-Age)"
## [10] "h(Income-100)"
```

Comparing these to the predictors we got in part (a), we can easily find that they are identical, while the cut-off of each predictor is slightly different. To be noticed, the MARS which I use here is additive model by set the function option `degree = 1`, which controls the maximum degree of interaction. If we allow the model include interaction terms, we can get of the test MSE of the best model with second order interactions, which is 1.4052, and the that with with order interactions, which is 1.38. We can see that both of them is greater than the test MSE of additive model, so the interactions may result in overfitting.

▶ Exercises 4. Solution.

Firsly, this is a classification tree proble, so the cost function should be the error rate. Then the information in the analysis are:

- random seed is 1;
- folds of cross-validation is 10;
- for simple classification, the result is the best model after pruning;
- for bagging, B (which is the number of trees) is 500, the default value of funciton `randomForest()`;
- for boosting, the parameters are selected by cross-validation from cadidate combinations: $B \in \{1000, 3000, 5000\}$, $d \in \{1, 2, 3\}$ and $\lambda \in \{0.01, 0.1\}$;
- for random forest, m is selected from all possible value: 1~7;
- for MARS, the model is additive model, i.e., the maximum degree of intercation term is 1 (no intercation).

The code and the analysis method are similarly to the problem 3, so we will not include the details here. We just scan the error rates of these five models,

```
all_error_rate
```

```
##             Simple Tree       Tree with bagging       Tree with boostin
##               0.2560241               0.2861446               0.2349398
```

```
## Tree with random forest                   MARS
##               0.2228916              0.2379518
```

Then, only for the condition listed above, the best model is "Tree with random forest" with error rate = 0.2228916. For this model, the number of variables considered at each split is only one, and the number of trees in the forest is 500.