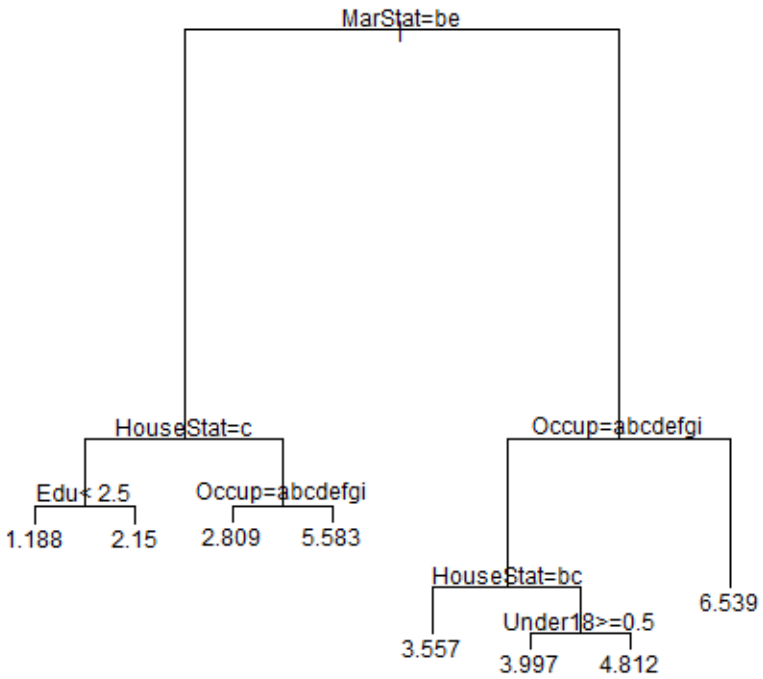
STATS 315B: Homework 1 Solutions

Spring 2018

1 Problem 1

The first split is on marital status for married/divorced/widowed and living together/single, which is reasonable since those who are not married are likely younger than those who have been married before. The next split is on occupation for those married before between not retired and retired, since retired people tend to be older. The next split is on householder status for those not married before between live with family and own/rent, since people living with their families tend to be younger. There are further splits on education and number of persons under 18.



a: Surrogate splits are used when the variable for the primary split is missing. They are calculated to have the optimal agreement with the primary split. From the tree, one example is for the householder status split between own/rent and live with family, a surrogate is number of persons under 18 less than 0.5.

b: For me the prediction is 2.9, which corresponds to class 25 thru 34.

```
library(rpart)
x <- read.csv("age_stats315B.csv")
```

```
factors <- c("Occup", "TypeHome", "sex", "MarStat",
  "DualInc", "HouseStat", "Ethnic", "Lang")
nums <- c("Edu", "Income", "LiveBA", "Persons", "Under18")
age <- x[,"age"]
xfac <- x[,factors]
xnum <- x[,nums]
xfac <- apply(xfac, 2, function(x) as.factor(x))
xcom <- cbind(xnum, xfac)
mod <- rpart(age ~ ., data = xcom, method = "anova", control = rpart.control(cp=0))
index <- which.min(mod$cptable[,"xerror"])
cutoff <- mod$cptable[index, "xerror"] + mod$cptable[index, "xstd"]
cpopt <- mod$cptable[min(which(mod$cptable[,"xerror"] < cutoff)), "CP"]
mod <- prune(mod, cp = cpopt)
mod_plot <- prune(mod, cp = 0.01)
summary(mod_plot)
png("tree.png")
plot(mod_plot)
text(mod_plot)
dev.off()
xcom_new <- data.frame(Edu = 5, Income = 1, LiveBA = 1,
  Persons = 0, Under18 = 0, Occup = as.factor(6),
  TypeHome = as.factor(3), sex = as.factor(2), MarStat = as.factor(5),
  DualInc = as.factor(1), HouseStat = as.factor(2), Ethnic = as.factor(2),
  Lang = as.factor(1))
predict(mod, xcom_new)
```

Problem 2

The aim of this problem is to fit a classification tree to the housing data using the R package *rpart*. We thus begin by loading the data, and indicating which variables are categorical (note that some of them are ordinal variables, and we should treat them as such!)

```
library(rpart)
info=c("Y","sex","marital_status","age","education","occupation","annual_income","length",
  "dual_incomes","size_household","nb_minors","householder_status","ethnic","language")
X=read.table("Housetype_Data.txt",sep=",",header=F)
Y=X[,1]
X=data.frame(X)
colnames(X)=info
X$Y<-as.factor(X$Y)
X$sex<-as.factor(X$sex)
X$marital_status<-as.factor(X$marital_status)
X$occupation<-as.factor(X$occupation)
X$dual_incomes<-as.factor(X$dual_incomes)
X$ethnic<-as.factor(X$ethnic)
X$householder_status<-as.factor(X$householder_status)
X$language<-as.factor(X$language)
#### The goal is too infer Y from X by fitting an  optimal classification tree
```

We can begin by analyzing the data: nb of missing values, spread, etc (just to get the feel of the data)

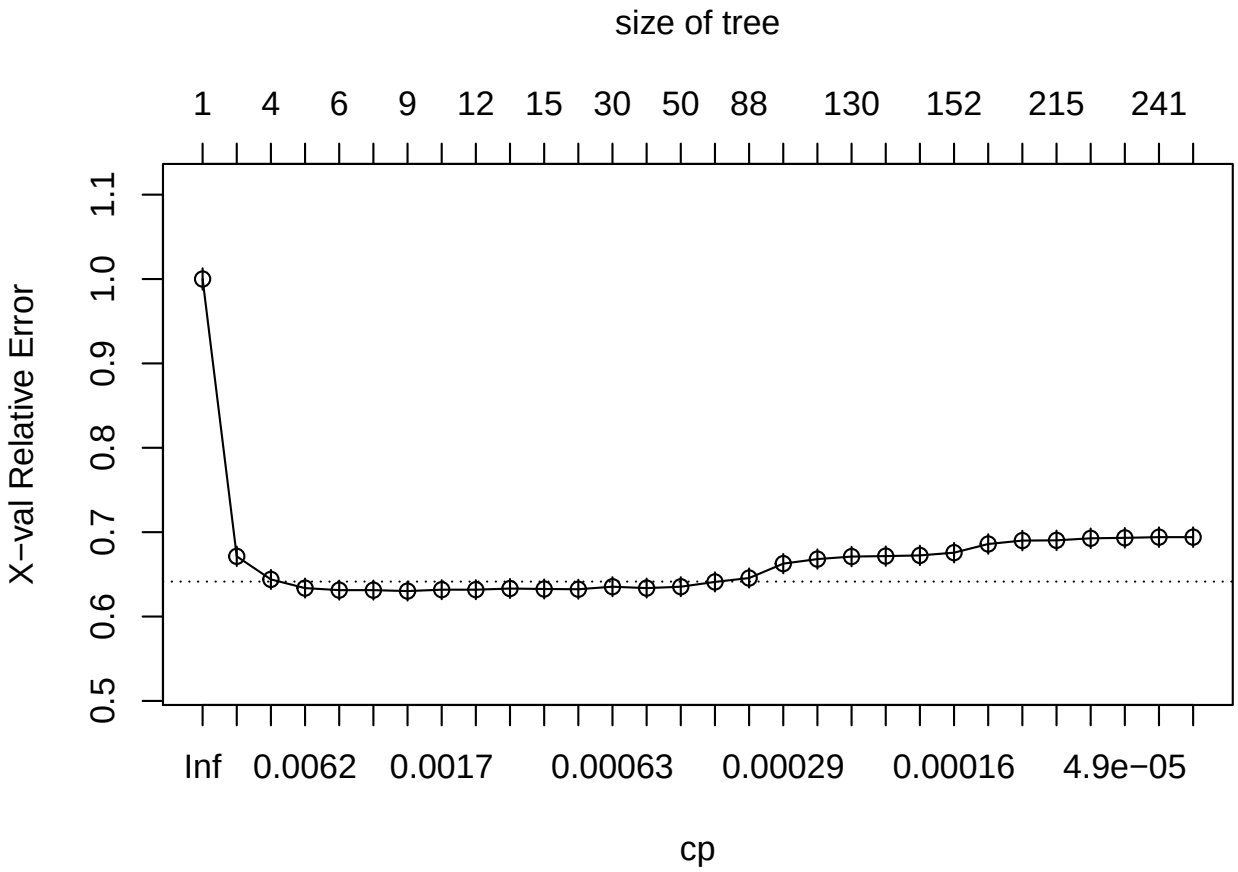
```
##### We begin by analyzing the data: nb of missing values, spread, etc
K= 5 ## number of possible classes for the housing
summary(X)
```

```
##  Y        sex      marital_status      age          education
##  1:5319   1:4043   1   :3351      Min.   :1.000   Min.   :1.000
##  2: 674   2:4970   2   : 667      1st Qu.:2.000   1st Qu.:3.000
##  3:2454            3   : 856      Median :3.000   Median :4.000
##  4: 162            4   : 312      Mean   :3.423   Mean   :3.831
##  5: 404            5   :3669      3rd Qu.:4.000   3rd Qu.:5.000
##                    NA's: 158      Max.   :7.000   Max.   :6.000
##                                                   NA's   :93
##    occupation   annual_income       length    dual_incomes size_household
##  1      :2798   Min.   :1.000   Min.   :1.0   1:5439       Min.   :1.000
##  6      :1475   1st Qu.:2.000   1st Qu.:4.0   2:2217       1st Qu.:2.000
##  4      :1049   Median :5.000   Median :5.0   3:1357       Median :3.000
##  2      : 786   Mean   :4.929   Mean   :4.2                Mean   :2.874
##  3      : 755   3rd Qu.:7.000   3rd Qu.:5.0                3rd Qu.:4.000
##  (Other):1994   Max.   :9.000   Max.   :5.0                Max.   :9.000
##  NA's   : 156   NA's   :377     NA's   :942                NA's   :331
##    nb_minors      householder_status     ethnic     language
##  Min.   :0.0000   1   :3304          7      :5845   1   :7803
##  1st Qu.:0.0000   2   :3664          5      :1235   2   : 591
##  Median :0.0000   3   :1851          3      : 893   3   : 260
##  Mean   :0.6767   NA's: 194          2      : 471   NA's: 359
##  3rd Qu.:1.0000                      8      : 238
##  Max.   :9.0000                      (Other): 270
##                                      NA's   :  61
```

```
###
```

We now try to grow the tree: we begin to set the Complexity paramter as low as possible to grow as "maximum tree". We will then prune the tree choosing the adequate Cp value using the 1MSE rule:we choose the most parsimonious model whose error is no more than one standard error above the error of the best mode.

```
# grow tree
fit <- rpart(Y~ .,method="class", data=X,cp=0) ### Y vs X (the rest of the variables)
### This  method does exactly what we want: it gives back a classification tree according to the fo
### The complexity paramter
plotcp(fit) # visualize cross-validation results
```




```
### Prune the tree: select the CP that is 1MSE above the cp-value that minimizes the misclassificat
# get index of CP with lowest xerror
opt_cp <- which.min(fit$cptable[,'xerror'])
candidate_cp<- which(fit$cptable[,'xerror']>(fit$cptable[opt_cp,'xerror']+fit$cptable[opt_cp,'xstd'
cp_mse_ind<- min(which(fit$cptable[,'xerror']<(fit$cptable[opt_cp,'xerror']+fit$cptable[opt_cp,'xst
cp_mse=fit$cptable[cp_mse_ind,'CP']
#get its value
#prune tree
pruned_model <- prune(fit,cp_mse)
### as input
printcp(pruned_model) # display the results in terms of complexity paramter
```

```
##
## Classification tree:
## rpart(formula = Y ~ ., data = X, method = "class", cp = 0)
##
## Variables actually used in tree construction:
```

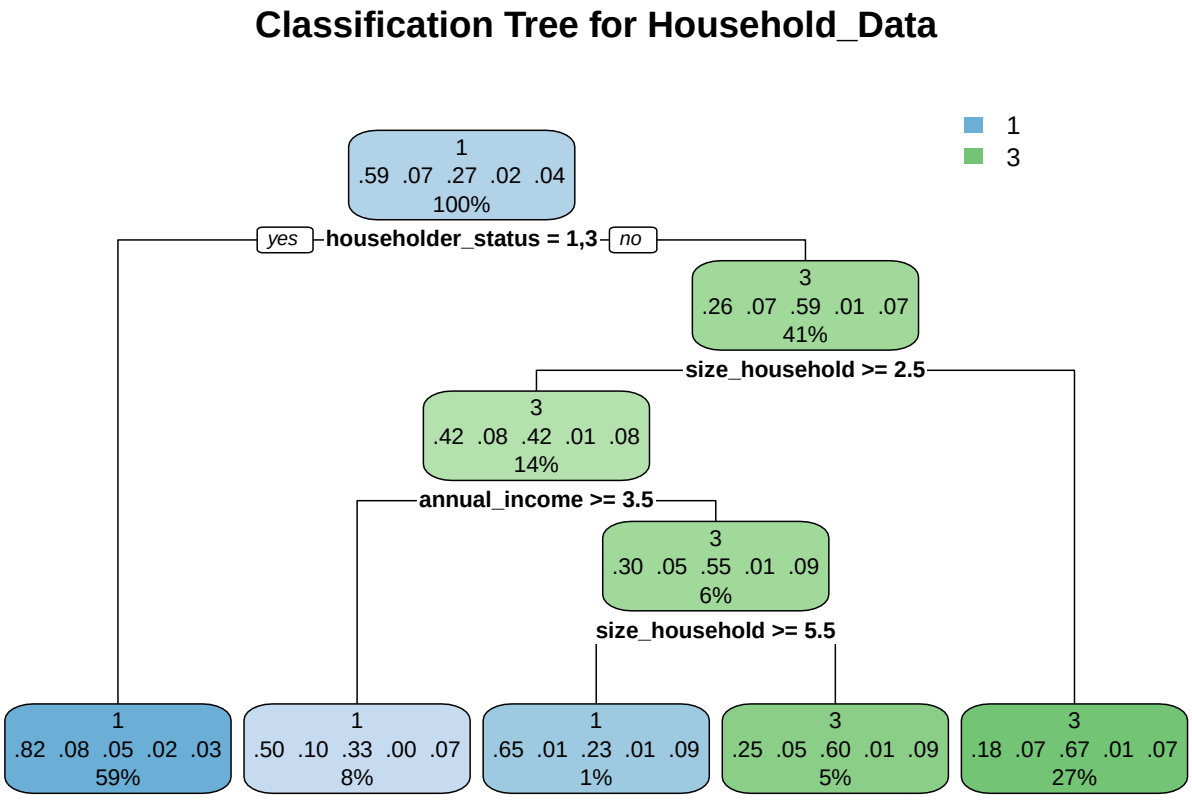
5

```
## [1] annual_income      householder_status size_household
##
## Root node error: 3694/9013 = 0.40985
##
## n= 9013
##
##          CP nsplit rel error  xerror     xstd
## 1 0.3286410      0   1.00000 1.00000 0.012640
## 2 0.0167840      1   0.67136 0.67136 0.011478
## 3 0.0078506      3   0.63779 0.64402 0.011328
## 4 0.0048728      4   0.62994 0.63373 0.011269
```

```
#plot tree
library(rpart.plot)
```

```
## Warning: package 'rpart.plot' was built under R version 3.3.2
```

```
rpart.plot(pruned_model, uniform=TRUE,main="Classification Tree for Household_Data")
```



We note that:

- class 2 (Condominium) and 4 (mobile home) are not predicted by our classification tree. This is due to the fact that these are relatively rare classes, which account for only respectively 7.5 and 1.8 % of the observations.
- The most important variable in our case if the householder status (own, rent, live with parent)
- The misclassification cross-validated error rate is roughly $0.409 \times 0.636 = 0.2601$

2 Problem 3

Overfitting: by fitting a very complex model, we might obtain a fitted model that explains the training data very well but does not generalize.

Concept drift: the relationship between the predictors and the response may evolve over time. That means that, even if we approximately recover the target function at the time the training data was collected, that function might produce poor predictions at later times.

3 Problem 4

If we search among all functions, the solution will not be unique. In general, there are infinitely many functions that interpolate between the training points and achieve zero training error. More importantly, this approach will lead to substantial overfitting. Even if we fix which solution to choose, it will have high variance across different training sets. In some cases, computing this solution may also be intractable.

4 Problem 5

The target is the prediction function that minimizes the risk under the true joint distribution of the predictors and the response. The target function offers the best predictions given the features available. However, if those features are only weakly associated with the response, the resulting predictions may still be inaccurate.

5 Problem 6

The population risk associated with F is

$$R(F) = \mathbb{E}_{XY}[L(Y, F(X))].$$

Since the joint distribution of (X, Y) is unknown, we evaluate it instead on the training data set $\{(X_i, Y_i)\}_{i=1,\cdots,n} \sim (X, Y)$ through the empirical risk

$$\hat{R}(F) = \frac{1}{n}\sum_{i=1}^{n} L(Y_i, F(X_i)).$$

The empirical risk will usually be optimistic(underestimated) for the actual risk, because it is optimized during the training process.

However, the higher the signal to noise ratio is, and the higher the number of training samples, the better surrogate it will be for the population risk.

6 Problem 7

The formula for the misclassification risk is

$$\text{Risk} = \sum_{k=1}^{K}\sum_{\ell=1}^{K} \Pr[y = c_\ell, \hat{y} = c_k] L_{kl}.$$

According to the problem, we can assume that $L_{kl} = r$ for some constant $r > 0$ for all $k \neq \ell$. Furthermore, we know that $L_{kk} = 0$ because the loss is zero for correct classification. Using these two facts, we get

$$\text{Risk} = r\sum_{k=1}^{K}\sum_{\ell=1}^{K} \Pr[y = c_\ell, \hat{y} = c_k] 1(k \neq l) = r\Pr[y \neq \hat{y}],$$

which is a constant times the misclassification error rate.

In the general case, the Bayes prediction rule for a new observation x is to predict $\hat{y} = c_k$, where

$$k = \text{argmin}_k \sum_{\ell=1}^{K} \Pr(y = c_\ell | x) L_{kl}.$$

However, if $L_{kl} = 1(k \neq \ell)$, then since

$$\sum_{\ell=1}^{K} \Pr(y = c_\ell | x) = 1,$$

the above simplifies to

$$k = \text{argmax}_k \Pr(y = c_k | x),$$

or equivalently,

$$k = \text{argmax}_k p(x | y = c_k)\pi(y).$$

In other words, the Bayes prediction rule chooses the class with the highest posterior probability in the special case of zero-one loss.

7 Problem 8

No. The Bayes rule will choose to classify observations into whichever class has the highest probability, so the accuracy only depends on how well we are identifying classes of high probability. There are many probabilities of the classes that correspond to the same Bayes rule, but the probabilities can be very different.

As a concrete example, consider two classes $k \in \{0, 1\}$ with true probabilities $\mathbb{P}[Y = c_1|X = x] = 0.40$ and $\mathbb{P}[Y = c_2|X = x] = 0.60$. However, the estimated probabilities $\hat{\mathbb{P}}[Y = c_1|X = x] = 0.10$ and $\hat{\mathbb{P}}[Y = c_2|X = x] = 0.90$ would also lead to the same prediction. Therefore, we see that the estimates do not necessarily have to be close to the true probabilities to result in the same predictions and low error.

8 Problem 9

In general, the bias-variance trade-off refers to the phenomenon that, in statistical prediction problems, (the absolute value of) bias decreases/increases and variance increases/decreases as the model complexity increases/decreases, respectively. Suppose we have a small function class, and we expand it to make it bigger and bigger. Then, the distance from the prediction to the target function decreases, which means that the bias decreases. On the other hand, it becomes increasingly harder to find the function in the class closest to the target function because there are larger number of functions that we can choose from. This means that the variance increases. Since bias and variance are two sources of error, it is important to choose the parameter carefully (for example, by cross validation) to balance the trade-off between them.

9 Problem 10

Consider the following situation where you make a split based on variable V such that $V < c$ defines the left region and $V > c$ the right region. In such a case, the splits down the tree from the left node are going to try to give the best answers conditioned on $V < c$ and similarly, on the right side. In training we want to assign the data point with missing V to the side that best reflects this conditioning. Therefore, we would want to use a variable that can predict V rather than the final answer. If we simply use a variable that can best predict y, then, we are going to diffuse the purity of the two splits. In prediction, we would like to use the conditioning on each variable. The sub-splits down the tree have been optimized given that conditioning and using a variable that is not good at predicting the conditioning would potentially send the point down the wrong branch.

10 Problem 11

For a given data set $\{(x_i, y_i)\}_{i=1,\dots,N}$, the squared-error risk is given by

$$g(c_1, \dots, c_m) := \sum_{i=1}^{N} [y_i - F(x_i)]^2 = \sum_{m=1}^{M} \sum_{i=1}^{N} I(x_i \in R_m)(y_i - c_m)^2,$$

because $\{R_m\}_{m=1,\cdots,M}$ are disjoint. Its partial derivative with respect to c_m, $m = 1, \dots, M$, is

$$\frac{\partial g}{\partial c_m} = 2 \sum_{i=1}^{N} I(x_i \in R_m)(c_m - y_i) = 2 \left(\sum_{i=1}^{N} I(x_i \in R_m) c_m - \sum_{i=1}^{N} I(x_i \in R_m) y_i \right).$$

Assuming there exists at least one data point whose predictor part is contained in R_m, i.e. $\sum_{i=1}^{N} I(x_i \in R_m) > 0$ (which is the case we are interested in), from the definition of $\hat{c}_m$ given in the problem we get

$$\frac{\partial g}{\partial c_m} = 2 \sum_{i=1}^{N} I(x_i \in R_m)(c_m - \hat{c}_m).$$

Hence, g is strictly increasing (decreasing) when $c_m > \hat{c}_m$ ($c_m < \hat{c}_m$), where $\hat{c}_m$ is as given in the problem. This shows that $\hat{c}_m$ is the minimizer of the squared-error risk. □

11 Problem 12

Let m denote the index of the region which we split, and let $\tilde{F}$ denote the new prediction function. There is no change in any region except for R_m, which is split into R_m^l and R_m^r. Since by the previous question, $F(x_i) = \sum_{j=1}^{M} \hat{c}_j 1_{x_i \in R_j}$ where $\hat{c}_j = \frac{\sum_{i:x_i \in R_j} y_i}{n_j}$, we have:

$$\forall i \notin R_m, F(x_i) = \tilde{F}(x_i)$$

That is,the regression surface is unchanged outside of R_m, and within this region, the surface has been updated to be the average within each of R_m^l and R_m^r, rather than the average of R_m overall. This means that the improvement I is only the sum of the improvement in the left and right region $I = I_l + I_r$. The change in the left region is

$$I_l = \sum_{i \in R_m^l} \left[(y_i - \bar{y})^2 - \left(y_i - \bar{y}^l\right)^2 \right],$$

where $\bar{y}$ denotes the average over all y's in R_m and $\bar{y}^l$ and $\bar{y}^r$ are averages over R_m^l and R_m^r, respectively. The first squared term is the approximation error before the split, while the second is the error afterwards. An analogous expression holds for R_m^r.

To begin simplifying this expression, expand the squares and use the fact that $\sum_{i=1}^{n} x_i = n\bar{x}$,

$$\begin{aligned} I_l = \sum_{i \in R_m^l} \left[(y_i - \bar{y})^2 - \left(y_i - \bar{y}^l\right)^2 \right] &= \sum_{i \in R_m^l} \left[y_i^2 - 2y_i\bar{y} + \bar{y}^2 - y_i^2 + 2y_i\bar{y}^l - (\bar{y}^l)^2 \right] \\ &= -2n_l\bar{y}^l\bar{y} + n_l\bar{y}^2 + n_l(\bar{y}^l)^2 \\ &= n_l\left(\bar{y}^l - \bar{y}\right)^2 \end{aligned}$$

Therefore, the overall change across both R_m^l and R_m^r is

$$I = n_l\left(\bar{y}^l - \bar{y}\right)^2 + n_r\left(\bar{y}^r - \bar{y}\right)^2. \tag{1}$$

To get rid of the $\bar{y}$ appearing in this expression, note that

$$\bar{y} = \frac{1}{n}\left(n_r\bar{y}^r + n_l\bar{y}^l\right),$$

and hence

$$\begin{aligned} \bar{y}^l - \bar{y} = \bar{y}^l - \frac{1}{n}\left(n_r\bar{y}^r + n_l\bar{y}^l\right) \\ = \frac{n_r}{n}\left(\bar{y}^l - \bar{y}^r\right), \end{aligned}$$

and

$$\bar{y}^r - \bar{y} = \frac{n_l}{n}\left(\bar{y}^r - \bar{y}^l\right).$$

Substituting into equation 1 yields

$$\begin{aligned} I = n_l \left(\frac{n_r}{n}\right)^2 \left(\bar{y}^l - \bar{y}^r\right)^2 + n_r \left(\frac{n_l}{n}\right)^2 \left(\bar{y}^r - \bar{y}^l\right)^2 &= \frac{n_l n_r^2 + n_r n_l^2}{n^2} \left(\bar{y}^r - \bar{y}^l\right)^2 \\ &= \frac{n_l n_r \left(n_r + n_l\right)}{n^2} \left(\bar{y}^r - \bar{y}^l\right)^2 \\ &= \frac{n_l n_r}{n} \left(\bar{y}^r - \bar{y}^l\right)^2, \end{aligned}$$

as needed.

Grading Notes: This problem asked for the derivation of the improvement yielded by splitting a region into two. One of the key elements was to argue that all regions –except for the one under going a split– remained unchanged, which simplifies the computations tremendously.

12 Problem 13

Suppose the part of response data that arrives at a particular node is $\{y_i\}_{i=1}^n$. The part that goes to the left child is $\{y_i\}_{i\in\mathcal{L}}$ and right child is $\{y_i\}_{i\in\mathcal{R}}$. Denote the corresponding means as $\bar{y}, \bar{y}_{\mathcal{L}}$ and $\bar{y}_{\mathcal{R}}$, the original improvement as $\hat{I}_m$ and that after modification as $\hat{I}'_m$. Then,

$$\begin{aligned} N\hat{I}_m &= \sum_{i=1}^{n}(y_i-\bar{y})^2 - \sum_{i\in\mathcal{L}}(y_i-\bar{y}_{\mathcal{L}})^2 - \sum_{i\in\mathcal{R}}(y_i-\bar{y}_{\mathcal{R}})^2 \\ &= \left(\sum_{i=1}^{n} y_i^2 - n(\bar{y})^2\right) - \left(\sum_{i\in\mathcal{L}} y_i^2 - n_L(\bar{y}_{\mathcal{L}})^2\right) - \left(\sum_{i\in\mathcal{R}} y_i^2 - n_R(\bar{y}_{\mathcal{R}})^2\right) \\ &= n_L(\bar{y}_{\mathcal{L}})^2 + n_R(\bar{y}_{\mathcal{R}})^2 - n(\bar{y})^2. \end{aligned}$$

Without loss of generality, we assume one point y_0 is moved from the left child to the right child. After modification, $\bar{y}' = \bar{y}$,

$$\bar{y}'_L = \frac{n_L\bar{y}_L - y_0}{n_L - 1}, \bar{y}'_R = \frac{n_R\bar{y}_R + y_0}{n_R + 1}.$$

Then we have,

$$\begin{aligned} N\hat{I}'_m &= (n_L-1)(\bar{y}'_{\mathcal{L}})^2 + (n_R+1)(\bar{y}'_{\mathcal{R}})^2 - n(\bar{y})^2 \\ &= \frac{(n_L\bar{y}_L - y_0)^2}{n_L-1} + \frac{(n_R\bar{y}_R + y_0)^2}{n_R+1} - n(\bar{y})^2. \end{aligned}$$

Thus the change in improvement is

$$\begin{aligned} N(\hat{I}'_m - \hat{I}_m) &= \frac{(n_L\bar{y}_L - y_0)^2}{n_L-1} + \frac{(n_R\bar{y}_R + y_0)^2}{n_R+1} - n_L(\bar{y}_{\mathcal{L}})^2 - n_R(\bar{y}_{\mathcal{R}})^2 \\ &= \frac{(n_L\bar{y}_L - y_0)^2 - (n_L-1)n_L(\bar{y}_{\mathcal{L}})^2}{n_L-1} + \frac{(n_R\bar{y}_R + y_0)^2 - (n_R+1)n_R(\bar{y}_{\mathcal{R}})^2}{n_R+1} \\ &= \frac{n_L(\bar{y}_L)^2 - y_0(2n_L\bar{y}_L - y_0)}{n_L-1} + \frac{-n_R(\bar{y}_R)^2 + y_0(2n_R\bar{y}_R - y_0)}{n_R+1} \\ &= y_0^2\left(\frac{1}{n_L-1} + \frac{1}{n_R+1}\right) + y_0\left(-\frac{2n_L\bar{y}_L}{n_L-1} + \frac{2n_R\bar{y}_R}{n_R+1}\right) + \left(\frac{n_L}{n_L-1}(\bar{y}_L)^2 - \frac{n_R}{n_R+1}(\bar{y}_R)^2\right). \end{aligned}$$

Thus the change of improvement in the risk is

$$\hat{I}'_m - \hat{I}_m = \frac{1}{N}\left(y_0^2\left(\frac{1}{n_L-1} + \frac{1}{n_R+1}\right) + y_0\left(-\frac{2n_L\bar{y}_L}{n_L-1} + \frac{2n_R\bar{y}_R}{n_R+1}\right) + \left(\frac{n_L}{n_L-1}(\bar{y}_L)^2 - \frac{n_R}{n_R+1}(\bar{y}_R)^2\right)\right).$$

13 Problem 14

The choice of whether to enlarge or restrict the class of F with which to define the target function is related to the bias-variance trade-off. A larger class F allows for fits with lower bias, but in the absence of a sufficient amount of training data, the estimates $\hat{f}$ may be highly variable. Further, in the case that the underlying response function belongs to a restricted class, estimation using that restricted class will be more statistically efficient.

It is important to evaluate the quality of the fit $\hat{f}$ using test data – improvement in mse on the training data alone is not enough. Whether or not increasing or reducing the size of F is a good idea depends on the presence of enough data to fit and evaluate $\hat{f}$ reliably (more data justifies the use of a richer class F) as well as any *apriori* knowledge about the complexity of the underlying response function (if it is simple, it is better to use a smaller class F containing plausible response functions).

14 Problem 15

The advantages of multi-way splitting are: (i) it can approximate more complicated function not well approximted by two-way splitting, while still retaining the possibility of mimicking binary splits (it might seem we can achieve any three-way split by doing two binary splits, but because of the greedy nature of the algorithm, some good three-way splits will likely be missed by the binary splits); (ii) ease of interpretability, since we don't need to multiply split on the same variable; (iii) we can predict faster, since the tree will likely be shallower.

The disadvantage are: (i) there is extra computational work in splitting; (ii) it can split the data too quickly, leaving insufficient data at the next level down.

15 Problem 16

The advantages of such a model include: (i) it can approximate more complicated functions; (ii) invariance to linear transformations of the input; (iii) it generalizes binary splitting (by using $a_j = 1$ and $a_i = 0$ for $i \neq j$).

The disadvantages are: (i) it increases the order of the computation in finding the best split; (ii) it increases the class of functions we can approximate well, likely making variance an issue; (iii) more difficult to interpret than regular trees; (iv) it makes it more difficult to deal with missing variables, since we need a new surrogate strategy.

Fun fact: these are known as "oblique decision trees", or decision tees with linear decision rules. They were included in the book that introduced CART (Breiman, L., Friedman, J., Stone, C. J., Olshen, R. A., 1984. *Classification and regression trees*. CRC press), and have been revisited periodically in the machine learning literature since then. They do not appear to be widely used in applications, but see the Hierarchical Mixture of Experts procedure for a better way to incorporate linear combination splits.