MA 751: HW10

Benjamin Draves

4/9/2019

Exercise 9.5

Part a

Suppose that we have a data $\{(x_i, y_i)\}_{i=1}^N$ such that $\mathbb{E}[y_i|x_i] = f(x_i)$ and $\text{Var}[y_i|x_i] = \sigma^2$. Suppose further that have a fitting operation that yields $\hat{\mathbf{y}}$. We choose define the degrees of freedom of this fitting operation by

$$\text{df} = \sum_{i=1}^{n} \frac{1}{\sigma^2}\text{Cov}(y_i, \hat{y}_i)$$

We look to give a rough estimate of this value in terms of the number of terminal nodes m. If our fitting procedure is given by a regression tree, T, then

$$\hat{y_i} = \frac{1}{N_{R(i)}} \sum_{j \in R(i)} y_j$$

where $R(i)$ is the terminal node that y_i resides and $N_{R(i)}$ is the number of observations in that region. With this, and using the linearity of covariance, we see that

$$\text{df} = \frac{1}{\sigma^2}\sum_{i=1}^{n} \frac{1}{N_{R(i)}} \sum_{j \in R(i)} \text{Cov}(y_i, y_j)$$

Now, if we assume (a) the y_i are conditionally uncorrelated so that

$$\text{Cov}(y_i, y_j) = \begin{cases} \sigma^2 & i = j \\ 0 & i \neq j \end{cases}$$

(b) the data are evenly split into the regions such that $N_{R(i)} = N/m$ we arrive at the following approximation

$$df \approx \frac{1}{\sigma^2}\sum_{i=1}^{n} \frac{\sigma^2 m}{N} = m$$

Therefore, we see the number of terminal nodes gives a rough approximation of the degrees of freedom of the regression tree model.

Part b

```
set.seed(123)
X <- matrix(rnorm(10*100), ncol = 10)
Y <- matrix(rnorm(10*100), ncol = 10)
```

Part c

```
library(tree)
tree.size <- c(1, 5,10)
Yhat <- array(NA, dim = c(3, 100, 10))

for(i in 1:10){
  #fit regression tree
  df <- data.frame(response = Y[,i], X)
  fit <- tree(response~., data = df)

  #loop over terminal node size
  for(j in 1:3){
    if(j == 1){
      #if one terminal node fit mean
      Yhat[j,,i] <- rep(mean(df$response), 100)
    }else{
      #else prune the tree as close to 5/10 as possible
      pruned.tree <- prune.tree(fit, best = tree.size[j])
      Yhat[j,,i] <- predict(pruned.tree, df)
    }
  }
}

#Calculate covariances for each m = 1, 5, 10
Cov1 <- apply(cbind(Y, Yhat[1,,]), 1, function(x) cov(x[1:10], x[11:20]))
Cov2 <- apply(cbind(Y, Yhat[2,,]), 1, function(x) cov(x[1:10], x[11:20]))
Cov3 <- apply(cbind(Y, Yhat[3,,]), 1, function(x) cov(x[1:10], x[11:20]))
```

Part d

Here we look to compare the estimates of the degrees of freedom from the simulation and the approximation we gave in part (a). Consider the table below that compares these estimates.

```
df <- data.frame(Approximation = c(1,5,10),
                 Simulation = c(sum(Cov1), sum(Cov2), sum(Cov3)))
knitr::kable(df)
```

Approximation	Simulation
1	0.3190207
5	30.4001031
10	50.5295877

As evident by the table, we see that our approximation is far too conservation. That is, this tree model demostrates much more degrees of freedom than what was estimated by our approximation in (a). In this approximation, we assumed that (i) the number of nodes in each terminal node were identical and (ii) the responses in each bin were uncorrelated. Clearly, the tree in (c) was constructing breaking both of these assumptions. Indeed, the tree model has freedom in placing more or less observations in certain nodes that will affect the scaling factor $N_{R(i)}$ and also the variance of the estimates given in (c). Moreover, as we construct the tree as observations have similiar x_i values, we could expect that the covariance $\text{Cov}(y_i, y_j) \neq 0$ even if this data was generated indepdently. As the tree model has problems with overfitting, we should expect that these observations will be artifically correlated. Both of these reasons could explain the higher degrees of freedom the model has in practice than the approximation given in (a).

Part e

Assumming the tree was a linear operator, we could write $\hat{y} = Sy$ where S is a smoothing matrix associated with the tree. Recall, as the tree method simply fits local averages of the data over the regions $R(i)$, we could write

$$S_{ij} = \begin{cases} \frac{1}{N_{R(i)}} & y_j \in R(i) \\ 0 & y_j \notin R(i) \end{cases}$$

In such, for a fixed i, we see that

$$\hat{y}_i = S_{i\cdot}^T y = \frac{1}{N_{R(i)}} \sum_{j \in R(i)} y_i$$

as desired. Notice that we can think of this scheme exactly as we did kNN regerssion instead that we cluster our points in some abstract feature space determined by the regions $\{R(i) : i \in [n]\}$

With this definition, we see that $S_{ii} = \frac{1}{N_{R(i)}}$ as $y_i \in R(i)$ for all $i \in [n]$. From here we can calculate the trace as follow

$$\text{tr}(S) = \sum_{i=1}^{n} \frac{1}{N_{R(i)}} = \sum_{j=1}^{m} \sum_{i: y_i \in R(j)} \frac{1}{N_{R_j}} = \sum_{j=1}^{m} 1 = m$$

Under the assumption that we can write this tree method as

Assumming the tree was a linear operator, we could write $\hat{y} = Sy$ where S is a smoothing matrix associated with the tree. Recall, as the tree method simply fits local averages of the data over the regions $R(i)$, we could write

$$S_{ij} = \begin{cases} \frac{1}{N_{R(i)}} & y_j \in R(i) \\ 0 & y_j \notin R(i) \end{cases}$$

In such, for a fixed i, we see that

$$\hat{y}_i = S_{i\cdot}^T y = \frac{1}{N_{R(i)}} \sum_{j \in R(i)} y_i$$

as desired. Notice that we can think of this scheme exactly as we did kNN regerssion instead that we cluster our points in some abstract feature space determined by the regions $\{R(i) : i \in [n]\}$

With this definition, we see that $S_{ii} = \frac{1}{N_{R(i)}}$ as $y_i \in R(i)$ for all $i \in [n]$. From here we can calculate the trace as follow

$$\text{tr}(S) = \sum_{i=1}^{n} \frac{1}{N_{R(i)}} = \sum_{j=1}^{m} \sum_{i:y_i \in R(j)} \frac{1}{N_{R_j}} = \sum_{j=1}^{m} 1 = m$$

Under the assumption that we can write this tree method as a linear method $\hat{y} = Sy$. If our goal is to estimate S, we can adopt a method similar to that from part (c). Recall from part (*c*) we generated two matries $\hat{Y}, Y \in \mathbb{R}^{100\times 10}$ where each column is a different replication of y_i or $\hat{y}_i$. Therefore, we look to fit the following models

$$\begin{bmatrix} \hat{y}_1 \\ \hat{y}_2 \\ \vdots \\ \hat{y}_{100} \end{bmatrix} = \begin{bmatrix} s_1^T y \\ s_2^T y \\ \vdots \\ s_{100}^T y \end{bmatrix}$$

Now recall that we replicated this proceduces each ten times in part (*c*) we have data $\{(\hat{y}_i^{(k)}, y_i^{(k)})\}_{i\in[n],k\in[10]}$. Therefore, for a fixed row j we can fit the model

$$\begin{bmatrix} \hat{y}_j^{(1)} \\ \hat{y}_j^{(2)} \\ \vdots \\ \hat{y}_j^{(10)} \end{bmatrix} = \begin{bmatrix} (y^{(1)})^T \\ (y^{(2)})^T \\ \vdots \\ (y^{(10)})^T \end{bmatrix} s_j^T$$

Notice this is exactly the standard OLS regression framework. Therfore, we can find the estimates of the rows of S by

$$\hat{s_j}^T = (YY^T)^{-1} Y \hat{Y}_j.$$

Moreover, notice that as we only are interested in the main diagonal of S, we need only consider the univariate regression model

$$\begin{bmatrix} \hat{y}_j^{(1)} \\ \hat{y}_j^{(2)} \\ \vdots \\ \hat{y}_j^{(10)} \end{bmatrix} = (y^{(j)})^T s_{jj}$$

with no itercept. Therefore, we will loop over the different tree sizes to get estimate $\{\hat{s}_{ii}\}_{i=1}^{n}$. We fit these models below.

```r
dgfree <- numeric(3)
#loop over tree size
for(i in 1:3){
  #loop over each row of Yhat and Y
  for(j in 1:100){
    #fit model and update the trace operatre
    dgfree[i] <- dgfree[i] + coef(lm(Yhat[i,j,]~Y[j,]-1))
    #notice this -1 ensures we fit withouth the intercept
  }
}
```

Therefore, we look to compare $\hat{df} = \text{tr}(\hat{\mathbf{S}})$ to the estimates given in (a) and (c). Consider the table below

```r
df <- round(data.frame(Approximation = c(1,5,10),
                 Part_C = c(sum(Cov1), sum(Cov2), sum(Cov3)),
                 Part_E = dgfree), 3)
knitr::kable(df)
```

Approximation	Part_C	Part_E
1	0.319	0.431
5	30.400	29.284
10	50.530	50.682

Notice that this estimate is much closer to the simualtion in (c) than that of the approximation given in (a). Therefore, we see that regarding this tree method as a linear smoother may, with respect to be degrees of freedom, may be appropriate.

Exercise 10.1

To derive equation (10.12) notice that (10.11) the optimization problem can be written as

$$(e^{\beta} - e^{-\beta}) \sum_{i=1}^{n} w_i^{(m)} I(y_i \neq G(x_i)) + e^{-\beta} \sum_{i=1}^{n} w_i^{(m)}$$

As we look to minimize this function with respect to β, we take a derivative and set equal to zero as follows

$$\overset{\frac{d}{d\beta}}{\Longrightarrow} (e^{\beta}+e^{-\beta})\sum_{i=1}^{n} w_i^{(m)} I(y_i \neq G(x_i)) - e^{-\beta}\sum_{i=1}^{n} w_i^{(m)} \overset{set}{=} 0$$

$$\begin{aligned}
(e^{\beta}+e^{-\beta})\sum_{i=1}^{n} w_i^{(m)} I(y_i \neq G(x_i)) &= e^{-\beta}\sum_{i=1}^{n} w_i^{(m)} \\
(e^{2\beta}+1)\sum_{i=1}^{n} w_i^{(m)} I(y_i \neq G(x_i)) &= \sum_{i=1}^{n} w_i^{(m)} \\
e^{2\beta}+1 &= \frac{\sum_{i=1}^{n} w_i^{(m)}}{\sum_{i=1}^{n} w_i^{(m)} I(y_i \neq G(x_i))} \\
2\beta &= \log\left(\frac{\sum_{i=1}^{n} w_i^{(m)}}{\sum_{i=1}^{n} w_i^{(m)} I(y_i \neq G(x_i))} - 1\right) \\
\beta &= \frac{1}{2}\log\left(\frac{\sum_{i=1}^{n} w_i^{(m)} - \sum_{i=1}^{n} w_i^{(m)} I(y_i \neq G(x_i))}{\sum_{i=1}^{n} w_i^{(m)} I(y_i \neq G(x_i))}\right) \\
\beta &= \frac{1}{2}\log\left(\frac{1 - \sum_{i=1}^{n} w_i^{(m)} I(y_i \neq G(x_i))/\sum_{i=1}^{n} w_i^{(m)}}{\sum_{i=1}^{n} w_i^{(m)} I(y_i \neq G(x_i))/\sum_{i=1}^{n} w_i^{(m)}}\right) \\
\beta &= \frac{1}{2}\log\left(\frac{1-\mathrm{err}_m}{\mathrm{err}_m}\right)
\end{aligned}$$

Therefore, this establishes equation 10.12.

Exercise 10.2

In this exercise, we look to establish the identify given in 10.16

$$\arg\min_{f(x)} \mathbb{E}_{Y|x}(e^{-Yf(x)}) = \frac{1}{2}\log\frac{P(Y=1|x)}{P(Y=-1|x)}$$

As we assume that $Y = \{\pm 1\}$ we can write for a fixed x value

$$\mathbb{E}_{Y|x}(e^{-Yf(x)}) = e^{-f(x)}\mathbb{P}(Y=1|x) + e^{f(x)}\mathbb{P}(Y=-1|x)$$

Therefore, we can minimize this expectation with respect to $f(x)$ by taking the derivatie with respect to $f(x)$, keeping in mind that x is fixed, and setting this to zero.

$$\overset{\frac{d}{df(x)}}{\Longrightarrow} -e^{-f(x)}\mathbb{P}(Y=1|x) + e^{f(x)}\mathbb{P}(Y=-1|x) \overset{set}{=} 0$$

$$\begin{aligned}
e^{f(x)}\mathbb{P}(Y=-1|x) &= e^{-f(x)}\mathbb{P}(Y=1|x) \\
e^{2f(x)} &= \frac{\mathbb{P}(Y=1|x)}{\mathbb{P}(Y=-1|x)} \\
f(x) &= \frac{1}{2}\log\frac{\mathbb{P}(Y=1|x)}{\mathbb{P}(Y=-1|x)}
\end{aligned}$$

Therefore, we see that the population miximizer is given by 1/2 the conditional log-odds.

Exercise 10.7

In this exercise we look to show that the optimal constants γ_{jm} in each region is given by the weighted log-odds in the two class - classification setting using exponential loss. This loss is given by $L(y, f(x)) = \exp\{-yf(x)\}$ so we look to solve

$$\begin{aligned}
\hat{\gamma}_{jm} &= \arg\min_{\gamma_{jm}} \sum_{x_i \in R_{jm}} \exp\left(-y_i(f_{m-1}(x_i) + \gamma_{jm})\right) \\
&= \arg\min_{\gamma_{jm}} \sum_{x_i \in R_{jm}} w_i^{(m)} \exp\left(-y_i\gamma_{jm}\right)
\end{aligned}$$

where $w_i^{(m)} = \exp(-y_i f_{m-1}(x_i))$. Notice this is identical the problem stated in (10.9). Therefore, we look to solve this problem in a similar way. Taking a derivative with respect to γ_{jm} we see that

$$\overset{\frac{d}{d\gamma_{jm}}}{\Longrightarrow} \sum_{x_i \in R_{jm}} -w_i^{(m)} \exp(-\gamma_{jm}) I(y_i = 1) + \sum_{x_i \in R_{jm}} w_i^{(m)} \exp(\gamma_{jm}) I(y_i = -1) \overset{set}{=} 0$$

Now notice that γ_{jm} is fixed with respect to x_i. Therefore, rearranging this expression, we have

$$\begin{aligned}
\exp(\gamma_{jm}) \sum_{x_i \in R_{jm}} w_i^{(m)} I(y_i = -1) &= \exp(-\gamma_{jm}) \sum_{x_i \in R_{jm}} w_i^{(m)} I(y_i = 1) \\
\exp(2\gamma_{jm}) &= \frac{\sum_{x_i \in R_{jm}} w_i^{(m)} I(y_i = 1)}{\sum_{x_i \in R_{jm}} w_i^{(m)} I(y_i = -1)} \\
\hat{\gamma}_{jm} &= \frac{1}{2} \log \frac{\sum_{x_i \in R_{jm}} w_i^{(m)} I(y_i = 1)}{\sum_{x_i \in R_{jm}} w_i^{(m)} I(y_i = -1)}
\end{aligned}$$

Therefore, we see that the optimal weight updates is given by hte weighted log-odds.