ESL-additional notes and sol

Pu He

March. 2018

Abstract

This document is my notes on reading through the classic textbook Elements of Statistical Learning. Most of it was written while I was taking the class Applied Stats II at Columbia University. My code for generating some of the plots and simulation results are not included in this file. This note has two purpose: first, I try to explain something I found interesting and/or hard to understand in the text; second, I try to finish all exercises in the text.

1 Introduction

2 Overview of Supervised Learning

2.1 Equation (2.22) and changing threshold in two-class classifier

The Bayes classifier in a binary case reduces to picking k s.t. $p_k > 0.5$ if the loss matrix in (2.21), defined as $L_{kk'}$ being the loss/penalty of classifying class k into k', is given below

$$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

This is also called 0-1 loss.

If we modify the 0-1 Loss to be assigning different weights to false positive and false negative based on application requirement,

$$\begin{bmatrix} 0 & L_{FP} \\ L_{FN} & 0 \end{bmatrix}$$

then we would obtain a different threshold than 0.5 in the final Bayes classifier. To be precise,

$$\begin{aligned} k &= \arg\min_k \sum_l L_{lk} p_l \\ &= \arg\min_k \left(\begin{cases} L_{FN} p_1 & k = 0 \\ L_{FP} p_0 & k = 1 \end{cases} \right) \\ &= \arg\min_k \left(\begin{cases} L_{FN} p_1 & k = 0 \\ L_{FP} (1 - p_1) & k = 1 \end{cases} \right) \end{aligned}$$

Let $k = 1$, we get $L_{FP}(1 - p_1) < L_{FN} p_1$, i.e. $p_1 > \frac{L_{FP}}{L_{FN} + L_{FP}}$. This provides theoretical justification (Bayes) of modifying threshold in ML classification tasks when different mistakes should be treated differently.

Similar to the loss matrix discussion in Chapter 9 on page 311, if there are only two classes, and the ML algorithms we are using for classification can take observation weights (like decision tree), we can weight the contribution to the loss function of each training example (if that point is misclassified), i.e. $L_{kk'}$. We can do that only in 2-class since we know for sure the weight of each training example as $L_{kk'}$for class k data points. We estimate the proportion of class 1 in a node m with the weighting $L_{kk'}$, $p_{(m)}^{weighted}$ and make classification decision with threshold 0.5, which is what ML packages typically do when we submit them a weight for the two classes. This is equivalent to estimate the unweighted proportion of class 1 in node m, and classify into class 1 iff $p_{(m)}^{unweighted} > \frac{L_{FP}}{L_{FN} + L_{FP}}$. This is as pointed out in the last sentence in Loss Matrix discusion on p311-p312.

For example for xgboost, I found http://xgboost.readthedocs.io/en/latest/how_to/param_tuning.html as a guide to tune pos_scale_weight. Note that usually the reason for that reweighting pos classes is for handling unbalanced data set as opposed to giving different loss for FP and FN. The difference is that, even if we have a perfect balance data set (50% positive, 50% negative), based on parameter tuning guide, we should leave scale_pos_weight as default 1.0; however, if from business point of view FP is much more costly, we should still give class 0 points more weights, i.e., set scale_pos_weight = $\frac{L_{FN}}{L_{FP}}$. Note that this is true for both classification and regression and in xgboost classification, if we use binary-crossentropy loss, the tree-fitting problem is actually a regression tree problem.

Based on the discussion there http://xgboost.readthedocs.io/en/latest/how_to/param_tuning.html, I think there is a way to deal with situation where we have both extremely unbalanced classes and different loss metric for FP and FN. We could first use scale_pos_weight to balance pos and neg classes as indicated in http://xgboost.readthedocs.io/en/latest/how_to/param_tuning.html. It is important to note that, as pointed out there, after rebalancing, the probability estimate is not correct anymore but we could still tune our model using AUCROC if we do not care about getting correct probability estimate. After using validation set to get the best AUCROC, we next tune the threshold for classification T, i.e. $\hat{p}^{weighted} > T$ to be class 1. We tune T s.t. a combination of precision and recall (or counts of FP and FN) that reflects our different loss metrics w.r.t. FP and NP are maximized. Alternatively, we could tune T directly on some business determined metrics (say F_1 scores with more weight on precision) without validating on AUCROC first

since AUCROC can be viewed as an overall metric that balances precision and recall. This is to separate ways to deal with unbalanced classes (adjusting weights) and ways to deal with different panelty for FP and FN (tuning threshold to minimize the correct metric that takes into account our loss matrix). In some sense, if we do not do the second step modifying the threshold and keep default as 0.5, our weighting on different classes imply one loss matrix.

2.2 Equation (2.27)

First we are conditioning on a fixed test point x_0 and thus we can view x_0 as a constant hereafter.

The confusion here is that there are two independent distribution: one is to generate training set τ of size N containing $\{x_i, y_i\}_{i=1}^N$, the other distribution is to generate at test time, the value of $y_0 = \mathbb{E}[y|x = x_0] + \epsilon = f(x_0) + \epsilon_0$. Since we are conditioning on x_0, $f(x_0)$ is constant here and only ϵ_0 is random. So $\mathbb{E}_{y_0|x_0}$ can be viewed as expectation over ϵ_0. By assumption, this two distribution $y_0|x_0$ and τ are independent. We do the following expansion and square it to get the conclusion.

$$y_0 - \hat{y}_0 = (y_0 - f(x_0)) + (f(x_0) - \mathbb{E}_\tau[\hat{y}_0]) + (\mathbb{E}_\tau[\hat{y}_0] - \hat{y}_0)$$

Note that in this case, by assumption of equation (2.26), we know that the true $f(x_0) = x_0'\beta$ is linear. Also by assumption, ϵ is independent of everything else which leads to mean independence $\mathbb{E}[\epsilon_i|x_i] = 0$ or in matrix form $\mathbb{E}[\boldsymbol{\epsilon}|\boldsymbol{X}] = 0$. In econometrics this is called mean independence. Together with the assumption that true model is linear, this leads to OLS estimate to be unbiased.

Conditioning on x_0 and consider expectation over $\mathbb{E}_{y_0|x_0}$ and $\mathbb{E}_\tau$, the third term is only about $\mathbb{E}_\tau$ and the first term is only about $\mathbb{E}_{y_0|x_0}$ while the second term is the biased term, which are constant. After taking square, the cross-terms are all 0 under $\mathbb{E}_{y_0|x_0}\mathbb{E}_\tau$

For term $Var_\tau(\hat{y}_0) = \mathbb{E}_\tau\left[(\mathbb{E}_\tau[\hat{y}_0] - \hat{y}_0)^2\right]$, this is the variance in the usual econometric sense. Therefore

$$\begin{aligned}
Var(\hat{y}_0) &= Var\left(x_0'\hat{\beta}\right) \\
&= x_0' Var\left(\hat{\beta}\right) x_0 \\
&= x_0' \mathbb{E}_\tau\left(\left(\boldsymbol{X'X}\right)^{-1} \boldsymbol{X'\epsilon\epsilon'X} \left(\boldsymbol{X'X}\right)^{-1}\right) x_0 \\
&= x_0' \mathbb{E}_\tau\left(\left(\boldsymbol{X'X}\right)^{-1} \boldsymbol{X'} \mathbb{E}_{\boldsymbol{\tau}|\boldsymbol{X}}\left[\boldsymbol{\epsilon\epsilon'}|\boldsymbol{X}\right] \boldsymbol{X} \left(\boldsymbol{X'X}\right)^{-1}\right) x_0 \\
&= x_0' \mathbb{E}_\tau\left(\left(\boldsymbol{X'X}\right)^{-1}\right) x_0 \sigma^2
\end{aligned}$$

To see the connection of the above equation to Econometrics, since ϵ_i is independent of x_i, we have in econometrics, homoskedaciticity here and therefore the asymptotic covariance matrix of $\hat{\beta}$ is $(\mathbb{E}[xx'])^{-1}\sigma^2$ and as an approximation with N samples from distribution τ, we have that $Var\left(\hat{\beta}\right) \sim \left(\boldsymbol{X'X}\right)^{-1}\sigma^2$

2.3 Equation (2.28)

Note that since $\frac{X'X}{N} \to Cov(X)$ and the $Cov(X)$ in the book is the Covariance matrix of the $p-$length random vector x under distribution τ, which is a constant. Therefore we must have $Cov(X)^{-1} Cov(x_0) = I_p$

2.4 Exericise

Ex 2.2

If we do everything from the begining of the data generating process unconditionally, we should expect the boundary to be a diagonal line since everything is symmetric. Indeed, without distinguishing between probability density and probability mass function, the boundary is given by

$$\begin{aligned} P(red|X=x) = \frac{P(red)P(X=x|red)}{P(X=x)} &= \frac{P(blue)P(X=x|blue)}{P(X=x)} = P(blue|X=x) \\ P(X=x|red) &= P(X=x|blue) \end{aligned}$$

Note that the above is due to $P(red) = P(blue)$. We do the calculation and everything cancels out and we have,

$$\begin{aligned} &\frac{1}{10}\sum_k \int_{m_k} \frac{1}{\sqrt{2\pi \det\left(\frac{1}{5}I\right)}} e^{-\frac{5}{2}\left((x_1-m_{k,1})^2+(x_2-m_{k,2})^2\right)} \frac{1}{\sqrt{2\pi \det(I)}} e^{-\frac{1}{2}\left((m_{k,1}-1)^2+(m_{k,2})^2\right)} dm_{k,1}dm_{k,2} \quad = \\ &\frac{1}{10}\sum_k \int_{m_k} \frac{1}{\sqrt{2\pi \det\left(\frac{1}{5}I\right)}} e^{-\frac{5}{2}\left((x_1-m_{k,1})^2+(x_2-m_{k,2})^2\right)} \frac{1}{\sqrt{2\pi \det(I)}} e^{-\frac{1}{2}\left((m_{k,1})^2+(m_{k,2}-1)^2\right)} dm_{k,1}dm_{k,2} \end{aligned}$$

$$\begin{aligned} e^{-\frac{5}{2}x_1^2+\frac{(5x_1+1)^2}{12}-\frac{5}{2}x_2^2+\frac{(5x_2)^2}{12}} &= e^{-\frac{5}{2}x_2^2+\frac{(5x_2+1)^2}{12}-\frac{5}{2}x_1^2+\frac{(5x_1)^2}{12}} \\ x_1 &= x_2 \end{aligned}$$

A more reasonable way is that we take the center of blue clusters $m_1, m_2, , , m_{10}$ and the red clusters $m_{11}, m_{12}, , , m_{20}$ as fixed or conditioning on the positions of the centers, then similarly, the boundary is given by,

$$\begin{aligned} \frac{1}{10}\sum_{k=1}^{10} \frac{1}{\sqrt{2\pi \det\left(\frac{1}{5}I\right)}} e^{-\frac{5}{2}\left((x_1-m_{k,1})^2+(x_2-m_{k,2})^2\right)} &= \frac{1}{10}\sum_{k=11}^{20} \frac{1}{\sqrt{2\pi \det\left(\frac{1}{5}I\right)}} e^{-\frac{5}{2}\left((x_1-m_{k,1})^2+(x_2-m_{k,2})^2\right)} \\ \sum_{k=1}^{10} e^{-\frac{5}{2}\left((x_1-m_{k,1})^2+(x_2-m_{k,2})^2\right)} &= \sum_{k=11}^{20} e^{-\frac{5}{2}\left((x_1-m_{k,1})^2+(x_2-m_{k,2})^2\right)} \end{aligned}$$

Ex 2.4

The target point is drawn from a $N(0, I_p)$ and therefore has the expected squared distance from the origin is going to be p since $||x_0||^2$ is χ^2_p distributed. It remains to show that z_i has standard Normal distribution. We condition on $x_0$first and then since different components of x_i are independent we have that $a'x_i|x_0 \sim Normal$. Note that $E[a'x_i|x_0] = 0$ and $Var[a'x_i|x_0] = a'Ia = ||a||^2 = 1$. Therefore, conditioning on x_0, z_i has standard normal distribution, which obviously does NOT depend on x_0 at all. It follows that the unconditional distribution of z_i is also standard normal.

Ex 2.9

First of all note that $R_{te}\left(\hat{\beta}\right) = E\left[\tilde{y}_j - \tilde{x}_j'\hat{\beta}\right]$ for any j in the test set since for any $(\tilde{x}_i, \tilde{y}_i)$ is independent of $\hat{\beta}$. Therefore, WLOG we could assume $M = N$ and then we have $E\left[R_{te}\left(\hat{\beta}\right)\right] = \frac{1}{N}\sum_{i=1}^{N} E\left[\tilde{y}_i - \tilde{x}_i'\hat{\beta}\right]$. Denote $\tilde{\beta}$ as the least square solution $\tilde{\beta} = (\tilde{X}'\tilde{X})^{-1}\tilde{X}'\tilde{Y}$ to the test set of size N, which is denoted as $(\tilde{X}, \tilde{Y})$. Since $\hat{\beta}$ is the least square solution based on the realization of the training set (X, Y), which is an i.i.d copy of $(\tilde{X}, \tilde{Y})$, we have then that almost surely, the two random variables have the following relationship,

$$\frac{1}{N}\sum_{i=1}^{N}\left(\tilde{y}_i - \tilde{x}_i'\hat{\beta}\right) \geq \frac{1}{N}\sum_{i=1}^{N}\left(\tilde{y}_i - \tilde{x}_i'\tilde{\beta}\right)$$

, which leads to

$$\begin{aligned} E\left[R_{te}\left(\hat{\beta}\right)\right] = E\left[\frac{1}{N}\sum_{i=1}^{N}\left(\tilde{y}_i - \tilde{x}_i'\hat{\beta}\right)\right] &\geq E\left[\frac{1}{N}\sum_{i=1}^{N}\left(\tilde{y}_i - \tilde{x}_i'\tilde{\beta}\right)\right] \\ \text{(since they are identically distributed)} &= E\left[\frac{1}{N}\sum_{i=1}^{N}\left(y_i - x_i'\hat{\beta}\right)\right] \\ &= E\left[R_{tr}\left(\hat{\beta}\right)\right] \end{aligned}$$

3 Linear Methods for Regression

3.1 Exericise

Ex 3.6

$$\begin{aligned} f(\beta|y) &\propto f(\beta) f(y|\beta) \\ &= \exp\left(-\frac{1}{2\sigma^2}(y - X\beta)'(y - X\beta) - \frac{1}{2\tau}\beta'\beta\right) \\ &= \exp\left(-\frac{1}{2\sigma^2}(y - X\beta)'(y - X\beta) - \frac{1}{2\tau}\beta'\beta\right) \end{aligned}$$

,which can be easily seen to be a normal density, whose mode equal to mean. Then picking the mode of the posterior is equivalent to maximize the log of the density w.r.t. β,

$$\max_{\beta} \quad \log\left(f\left(\beta|y\right)\right) \quad = \min_{\beta} \quad \left(y - X\beta\right)'\left(y - X\beta\right) + \frac{\sigma^2}{\tau}\beta'\beta$$

Note that the objective on the RHS of the above expression is identical to (3.41) of ESL with $\lambda = \frac{\sigma^2}{\tau}$ except for that we also penalize the intercept term. Therefore, the ridge estimate of β is also the mode/mean of the posterior distribution of β given proper priors.

Ex 3.12

Using block matrix,

$$\left(\begin{bmatrix} X' & \sqrt{\lambda}I \end{bmatrix}\begin{bmatrix} X \\ \sqrt{\lambda}I \end{bmatrix}\right)^{-1}\begin{bmatrix} X' & \sqrt{\lambda}I \end{bmatrix}\begin{bmatrix} Y \\ 0 \end{bmatrix} = \left(X'X + \lambda I\right)^{-1}X'Y$$

,which is the ridge solution(3.44)

4 Linear methods for Classification

4.1 Logistic Regression P119

In Econometric terms, multinomail logit model assumes that individual i choose class k with probability $\frac{e^{x'_{ik}\beta_k}}{\sum_k e^{x'_{ik}\beta_k}}$ where x_{ik} could depends on both i and k, and β_k can be different for all k. In

The formulation in Equation (4.19) is the special case of multinomial logit model where covariates x_{ik} for individual i and class k does not depend on k, i.e. there are no product or class specific characteristics. In this case we can NOT fit one set of β_k for all K class since we can always choose one class (WLOG, class K) as the base class and divide $e^{\beta'_K x_i}$ to both numerator and denominator in $\mathbb{P}\left[G_k|X = x_i\right]$ for any i and any $k = 1, 2, ...K$. Therefore, the model with all K β_k's is equivalent to Equation (4.18) and (4.19) where the base class K is normalized to $e^0 = 1$. Most computer science application falls into this special case (say ImageNet classification). When there are only two classes, this is the usual logistic regression case. Note that in this case, say X has intercept, then we cannot include covariates that never changes in the N samples.

Another special case of multinomial logit is conditional logit model where $\beta_k = \beta$ are the same for all k. In this case we cannot have indivdual characteristic z_i component in x_{ik} since $z'_i\beta$ can be cancelled out. This won't be a problem in typical discrete choice models when there are a base case (outside option) whose utility is normalized to 0. Empirical IO models studying differentiated products falls into this case, where we usually have product characteristics (BLP) and potentially individual characteristics (micro BLP).

Equation (4.27) shows that MLE in the logit model presented in Equation (4.19) can be solved as IRLS.

4.2 Exercise

Ex 4.1

We can use the lagrangian multiplier to solve the problem,

$$\max_{a,\lambda} a'Ba - \lambda\left(a'Wa - 1\right)$$

FOC gives,

$$\begin{aligned} 2Ba &= 2\lambda Wa \\ W^{-1}Ba &= \lambda a \end{aligned}$$

and,

$$a'Wa = 1$$

Therefore, the solution of FOC is that λ is a eigenvalue of matrix $W^{-1}B$ and a is one corresponding eigenvector of λ with $a'Wa = 1$

Premultiply the first equation by a' yields

$$a'Ba = \lambda a'Wa = \lambda$$

Therefore λ must be the largest eigenvalue of $W^{-1}B$ otherwise we can do better by setting λ to be the largest eigenvalue. The problem reduces to finding an eigenvector a of matrix $W^{-1}B$ corresponding to the largest eigenvalue.

Ex 4.2

(a)

Following the formula (4.9) in ESL, let $\pi_2 = \frac{N_2}{N_1+N_2}$ and $\pi_1 = \frac{N_1}{N_1+N_2}$, we have that $\log \frac{P(class2|X=x)}{P(class1|X=x)} > 0$ yields the equation

$$x'\hat{\Sigma}^{-1}\left(\hat{\mu}_2 - \hat{\mu}_1\right) > \frac{1}{2}\left(\hat{\mu}_2 + \hat{\mu}_1\right)'\hat{\Sigma}^{-1}\left(\hat{\mu}_2 - \hat{\mu}_1\right) - \log\left(\frac{N_2}{N_1}\right)$$

(b)

The OLS gives

$$\begin{bmatrix} N & 1'X \\ X'1 & X'X \end{bmatrix}\begin{bmatrix} \beta_0 \\ \beta \end{bmatrix} = \begin{bmatrix} 1'Y \\ X'Y \end{bmatrix}$$

, where

$$\begin{aligned} X'Y &= -\frac{N}{N_1}\sum_{i\in Class1} x_i + \frac{N}{N_2}\sum_{j\in Class2} x_j \\ &= N\left(\hat{\mu}_2 - \hat{\mu}_1\right) \end{aligned}$$

$$1'Y = N - N = 0$$

Therefore the equation reduces to

$$\begin{aligned} N\beta_0 + 1'X\beta &= 0 \\ X'1\beta_0 + X'X\beta &= N\left(\hat{\mu}_2 - \hat{\mu}_1\right) \end{aligned}$$

Solve for β,

$$\left(X'X - \frac{X'11'X}{N}\right)\beta = N\left(\hat{\mu}_2 - \hat{\mu}_1\right)$$

Note that

$$\begin{aligned} X'X &= \sum_i x_i x_i' \\ &= (N-2)\frac{\sum_{i\in class1}\left(x_i - \hat{\mu}_1\right)\left(x_i - \hat{\mu}_1\right)' + \sum_{i\in class2}\left(x_i - \hat{\mu}_2\right)\left(x_i - \hat{\mu}_2\right)'}{N-2} + \\ &\quad \sum_{i\in class1}\left(\hat{\mu}_1 x_i' + x_i\hat{\mu}_1'\right) + \sum_{i\in class2}\left(\hat{\mu}_2 x_i' + x_i\hat{\mu}_2'\right) - N_1\hat{\mu}_1\hat{\mu}_1' - N_2\hat{\mu}_2\hat{\mu}_2' \\ &= (N-2)\hat{\Sigma} + N_1\hat{\mu}_1\hat{\mu}_1' + N_2\hat{\mu}_2\hat{\mu}_2' \end{aligned}$$

Therefore,

$$\begin{aligned} X'X - \frac{X'11'X}{N} &= (N-2)\hat{\Sigma} + N_1\hat{\mu}_1\hat{\mu}_1' + N_2\hat{\mu}_2\hat{\mu}_2' - \frac{1}{N}\left(\sum_i x_i\right)\left(\sum_i x_i\right)' \\ &= (N-2)\hat{\Sigma} + N_1\hat{\mu}_1\hat{\mu}_1' + N_2\hat{\mu}_2\hat{\mu}_2' - \frac{1}{N}\left(N_1\hat{\mu}_1 + N_2\hat{\mu}_2\right)\left(N_1\hat{\mu}_1 + N_2\hat{\mu}_2\right)' \\ &= (N-2)\hat{\Sigma} + \frac{N_1N_2}{N}\left(\frac{N}{N_2}\hat{\mu}_1\hat{\mu}_1' + \frac{N}{N_1}\hat{\mu}_2\hat{\mu}_2' - \frac{N_1}{N_2}\hat{\mu}_1\hat{\mu}_1' - \frac{N_2}{N_1}\hat{\mu}_2\hat{\mu}_2' - \hat{\mu}_1\hat{\mu}_2' - \hat{\mu}_2\hat{\mu}_1'\right) \\ &= (N-2)\hat{\Sigma} + N\hat{\Sigma}_B \end{aligned}$$

Q.E.D

(c)

For any β, $\hat{\Sigma}_B\beta = \frac{N_1N_2}{N^2}(\hat{\mu}_2 - \hat{\mu}_1)(\hat{\mu}_2 - \hat{\mu}_1)'\beta = \left[\frac{N_1N_2}{N^2}(\hat{\mu}_2 - \hat{\mu}_1)'\beta\right](\hat{\mu}_2 - \hat{\mu}_1)$ is in the direction of $\hat{\mu}_2 - \hat{\mu}_1$. Therefore, $\hat{\Sigma}\hat{\beta} \propto \hat{\mu}_2 - \hat{\mu}_1$ and $\hat{\beta} \propto \hat{\Sigma}^{-1}(\hat{\mu}_2 - \hat{\mu}_1)$

(d)

Let (y_1, y_2) be the coding of class 1 and class 2 respectively. Therefore the FOC equations for OLS is,

$$\begin{aligned} N\beta_0 + 1'X\beta &= N_1y_1 + N_2y_2 \\ X'1\beta_0 + X'X\beta &= N_2y_2\hat{\mu}_2 + N_1y_1\hat{\mu}_1 \end{aligned}$$

Solve for β as before,

$$\begin{aligned} \left(X'X - \frac{X'11'X}{N}\right)\beta &= N_2y_2\hat{\mu}_2 + N_1y_1\hat{\mu}_1 - X'1\frac{N_1y_1 + N_2y_2}{N} \\ &= N_2y_2\hat{\mu}_2 + N_1y_1\hat{\mu}_1 - (N_2\hat{\mu}_2 + N_1\hat{\mu}_1)\frac{N_1y_1 + N_2y_2}{N} \\ &= \frac{N_1N_2(y_1 - y_2)}{N}(\hat{\mu}_2 - \hat{\mu}_1) \end{aligned}$$

The rest follows exactly the same as in (b) and (c) and we can conclude that,

$$\hat{\beta} \propto \hat{\Sigma}^{-1}(\hat{\mu}_2 - \hat{\mu}_1)$$

(e)

Assume $\hat{\beta} = b\hat{\Sigma}(\hat{\mu}_2 - \hat{\mu}_1)$, then we can solve for β_0

$$\hat{\beta}_0 = -\frac{1'X}{N}b\hat{\Sigma}^{-1}(\hat{\mu}_2 - \hat{\mu}_1)$$

Then

$$\hat{f}(x) > 0 \iff x'\hat{\Sigma}^{-1}(\hat{\mu}_2 - \hat{\mu}_1) > \frac{1'X}{N}\hat{\Sigma}^{-1}(\hat{\mu}_2 - \hat{\mu}_1) = \left(\frac{N_2}{N}\hat{\mu}_2 + \frac{N_1}{N}\hat{\mu}_1\right)\hat{\Sigma}^{-1}(\hat{\mu}_2 - \hat{\mu}_1)$$

It is obvious that unless, $N_1 = N_2$, we cannot recover the LDA rule (4.9) or in (a)

Ex 4.3

For $x \in R^p$, the transformation is linear: $\hat{y} = \hat{B}'x$. Therefore it is easy to see, after the tranformation $\forall k$, $\hat{\mu}_k^* = \hat{B}'\hat{\mu}_k$ and,

$$\begin{aligned}
\hat{\Sigma}^* &= \frac{\sum_{k=1}^{K}\sum_{g_i=k}(\hat{y}_i - \hat{\mu}_k^*)(\hat{y}_i - \hat{\mu}_k^*)'}{N-K} \\
&= \frac{\sum_{k=1}^{K}\sum_{g_i=k}\hat{B}'(x_i - \hat{\mu}_k)(x_i - \hat{\mu}_k)'\hat{B}}{N-K} \\
&= \hat{B}'\hat{\Sigma}\hat{B}
\end{aligned}$$

Assume first that$\hat{\Sigma}$ is invertible. Note that the problem only makes sense when $\left(\hat{B}'\hat{\Sigma}\hat{B}\right)^{-1}$ exists. We thus assume that $\hat{B}'\hat{\Sigma}\hat{B}$ is invertible. Note that a sufficient condition for that is $rank\left(\hat{B}\right) = k$.

For K classes LDA, WLOG we only needs to show the LDA rule between arbitrary two classes is the same with the original data or transformed data. After transformation, for any two classes k and l, we choose class k if

$$\begin{aligned}
x'\hat{B}\hat{\Sigma}^{*-1}(\hat{\mu}_k^* - \hat{\mu}_l^*) &> \frac{1}{2}(\hat{\mu}_k^* + \hat{\mu}_l^*)'\hat{\Sigma}^{*-1}(\hat{\mu}_k^* - \hat{\mu}_l^*) - \log\left(\frac{N_k}{N_l}\right) \\
\iff x'\hat{B}\left(\hat{B}'\hat{\Sigma}\hat{B}\right)^{-1}\hat{B}'(\hat{\mu}_k - \hat{\mu}_l) &> \frac{1}{2}(\hat{\mu}_k + \hat{\mu}_l)'\hat{B}\left(\hat{B}'\hat{\Sigma}\hat{B}\right)^{-1}\hat{B}'(\hat{\mu}_k - \hat{\mu}_l) - \log\left(\frac{N_k}{N_l}\right)
\end{aligned}$$

We want to show the above ineq is equivalent to

$$x'\hat{\Sigma}^{-1}(\hat{\mu}_k - \hat{\mu}_l) > \frac{1}{2}(\hat{\mu}_k + \hat{\mu}_l)'\hat{\Sigma}^{-1}(\hat{\mu}_k - \hat{\mu}_l) - \log\left(\frac{N_k}{N_l}\right)$$

It suffices to show that for any class k,

$$\hat{\mu}_k = \hat{\Sigma}\hat{B}\left(\hat{B}'\hat{\Sigma}\hat{B}\right)^{-1}\hat{B}'\hat{\mu}_k$$

Define e_k to be a K by 1 vector having 1 at k^{th} position and 0 elsewhere. Let N_k to be the number of rows in X that belongs to class k. Then $\hat{\mu}_k = \frac{1}{N_k}X'Ye_k$. The above is thus equivalent to

$$X'Y = \hat{\Sigma}\hat{B}\left(\hat{B}'\hat{\Sigma}\hat{B}\right)^{-1}\hat{B}'X'Y$$

To prove this, we express the sample within class covariance matrix as,

$$\begin{aligned}
\hat{\Sigma} &= \frac{\sum_{k=1}^{K}\sum_{g_i=k}\left(x_i-\hat{\mu}_k\right)\left(x_i-\hat{\mu}_k\right)'}{N-K} \\
&= \frac{\sum_{k=1}^{K}\sum_{g_i=k}x_i x_i' - \sum_{k=1}^{K}N_k\hat{\mu}_k\hat{\mu}_k'}{N-K} \\
&= \frac{X'X - \sum_{k=1}^{K}N_k\frac{X'Ye_k}{N_k}\frac{\left(X'Ye_k\right)'}{N_k}}{N-K} \\
&= \frac{X'X - X'Y\left\{\sum_{k=1}^{K}\frac{\left(e_k e_k'\right)}{N_k}\right\}Y'X}{N-K} \\
&:= \frac{X'X - X'YDY'X}{N-K},
\end{aligned}$$

where $D := \sum_{k=1}^{K}\frac{\left(e_k e_k'\right)}{N_k}$. Define $A := Y'X\left(X'X\right)^{-1}X'Y$, which is a symmetric K by K matrix. We have that

$$\begin{aligned}
\left(\hat{B}'\hat{\Sigma}\hat{B}\right)^{-1} &= (N-K)\left(Y'X\left(X'X\right)^{-1}\left(X'X - X'YDY'X\right)\left(X'X\right)^{-1}X'Y\right)^{-1} \\
&= (N-K)\left(A-ADA\right)^{-1}
\end{aligned}$$

$$\hat{\Sigma}\hat{B} = \frac{\left(X'Y - X'YDA\right)}{N-K}$$

$$\begin{aligned}
\hat{B}'X'Y &= Y'X\left(X'X\right)^{-1}X'Y \\
&= A
\end{aligned}$$

Therefore noticing that A must be invertible as well since $A-ADA = A\left(I-DA\right)$ is invertible, we have

$$\begin{aligned}
\hat{\Sigma}\hat{B}\left(\hat{B}'\hat{\Sigma}\hat{B}\right)^{-1}\hat{B}'X'Y &= \frac{N-K}{N-K}\left(X'Y - X'YDA\right)\left(A-ADA\right)^{-1}A \\
&= X'Y\left(I-DA\right)\left(I-DA\right)^{-1}A^{-1}A \\
&= X'Y
\end{aligned}$$

Q.E.D

Ex 4.5

The objective function we are maximizing is

$$\max_{\beta_0,\beta} \sum_i y_i \log p_i + (1-y_i)\log(1-p_i)$$

,where $p_i = \frac{e^{\beta_0+\beta x_i}}{1+e^{\beta_0+\beta x_i}}$

If we have $x_0 \in R$ that perfectly separate the data. WLOG, assume $y_i = 1$ for all $x_i \geq x_0$ and $y_i = 0$ o.w., then if we always set $\beta_0 = -\beta x_0$ and only maximize over β, we have

$$0 \geq \max_{\beta_0,\beta} \sum_i y_i \log p_i + (1-y_i)\log(1-p_i) \geq \max_{\beta} \sum_{y_i=1} \left[\log \frac{e^{\beta(x_i-x_0)}}{1+e^{\beta(x_i-x_0)}}\right] + \sum_{y_i=0} \left[\log \frac{1}{1+e^{\beta(x_i-x_0)}}\right]$$

We can see that as $\beta \to \infty$, $\sum_{y_i=1} \left[\log \frac{e^{\beta(x_i-x_0)}}{1+e^{\beta(x_i-x_0)}}\right] \to 0$ since $x_i - x_0 \geq 0$ and $\sum_{y_i=0} \left[\log \frac{1}{1+e^{\beta(x_i-x_0)}}\right] \to 0$ since $x_i - x_0 \leq 0$. Therefore the original problem $\max_{\beta_0,\beta} \sum_i y_i \log p_i + (1-y_i)\log(1-p_i)$ will not have an optimal solution and the algorithm will send β to infinity and never converge. As for decision boundary, we would have $\hat{y}_i = 1_{x_i > x_0}$, i.e., the logistic regression found one perfect separation point.

To extend this to $x \in R^p$, now we need to assume there is a hyperplane $\beta^{*'}x + \beta_0^* = 0$ s.t. $\beta^{*'}x + \beta_0^* \geq 0$ for all $y_i = 1$ and $\beta^{*'}x + \beta_0^* \leq 0$ o.w.

$$\begin{aligned} 0 &\geq \max_{\beta_0,\beta} \sum_i y_i \log p_i + (1-y_i)\log(1-p_i) \\ &\geq \max_{\alpha} \sum_{y_i=1} \left[\log \frac{e^{\alpha\left(\beta^{*'}x_i+\beta_0^*\right)}}{1+e^{\alpha\left(\beta^{*'}x_i+\beta_0^*\right)}}\right] + \sum_{y_i=0} \left[\log \frac{1}{1+e^{\alpha\left(\beta^{*'}x_i+\beta_0^*\right)}}\right] \end{aligned}$$

Similarly, we would send α to $+\infty$ so that the maximum likelihood are going to 0. The decision boundary, when enough iterations have passed, would be close to $\beta^{*'}x + \beta_0^* = 0$, i.e., although the algorithm will not converge in usual sense, it will find a perfect separating hyperplane eventually.

To extend this to multi-class case, to be done.

Ex 4.6

(a)

Separability implies that we can find $\beta \in R^{p+1}$ s.t. for any (x_i, y_i) with $y_i = 1$, we have $\beta' x_i^* > 0$ and for any (x_i, y_i) with $y_i = -1$, $\beta' x_i^* < 0$. Since $||x_i||^* \neq 0$, for the same β we have, $\forall i$, $\beta' z_i > 0$ if $y_i = 1$ and $\beta' z_i < 0$ if $y_i = -1$. Since we only have finite number of points, we can scale β by $h := \min\left\{\min_{j;y_j=-1} |\beta' z_j|, \min_{j;y_j=1} \beta' z_j\right\} > 0$ s.t. $\forall i$, $\left(\frac{\beta}{h}\right)' z_i \geq 1$ if $y_i = 1$ and $\left(\frac{\beta}{h}\right)' z_i \leq 1$ if $y_i = -1$. Define $\beta_{sep} := \frac{\beta}{h}$, we have, $\forall i$, $y_i \beta'_{sep} z_i \geq 1$. Q.E.D.

(b)

Note that it is also obvious that if we can find a β_{sep} satisfiying the above condition in (a), it also separates the orginal data set (x_i, y_i). So the original problem is equivalent to finding a β_{sep} s.t. $\forall i$, $y_i\beta'_{sep}z_i \geq 1$. In each step, given a current β_{old}, the perceptron algorithm identifies a misclassified point z_i, i.e., $y_i\beta'_{old}x_i^* < 0 \iff y_i\beta'_{old}z_i < 0$, we have,

$$\begin{aligned} |\beta_{new} - \beta_{sep}|^2 &= |\beta_{old} - \beta_{sep} + y_i z_i|^2 \\ &= |\beta_{old} - \beta_{sep}|^2 + |y_i z_i|^2 + 2y_i \left(\beta_{old} - \beta_{sep}\right)' z_i \\ &= |\beta_{old} - \beta_{sep}|^2 + 1 + 2y_i\beta'_{old}z_i - 2y_i\beta'_{sep}z_i \\ &\leq |\beta_{old} - \beta_{sep}|^2 + 1 + 0 - 2 \\ &= |\beta_{old} - \beta_{sep}|^2 - 1 \end{aligned}$$

β_{start} will converge to β_{sep} in finite steps.
Q.E.D

Ex 4.7

For misclassified points, they are same as before. Now we are adding to the loss function for points that are already correctly classified: we want to maximize the margin/distance fomr those points to the decision boundary. This solves the optimal separating hyperplane problem in the sense that this gives us one optimal solution. However, the solution might not be optimal or even reasonably good for generalization purpose.

5 Basis Expansions and Regularizations

5.1 Piecewise Polynomials and Splines

In general, cubic splines have $4 * (K + 1) - 3K = K + 4$ variables/basis.

For natural cubic splines, we have $4 * (K + 1) - 3K - 2 * 2 = K$ basis

5.2 Piecewise Polynomials and Splines

5.3 P156

Why $df_\lambda \to 2$ and $S_\lambda \to H$ when $\lambda \to \infty$?

5.4 Equation (5.25)

5.5 Equation (5.66)

Discussion here, together with the second paragraph on page 170, indicates that the thin-place spline solution is one we do not penalize functions in space H_0 (linear function), but regularizes only components on H_1, which is generated by the RBF kernel.

5.6 Definition of RKHS

RKHS is defined as a Hilbert space of functions where evaluation functional is continuous and linear. RKHS has an associated kernel that represents all function in the space in the sense that evaluation at x for any function f is determined by taking inner product between f and a function determined by kernel K, i.e., $L_x(f) = f(x) =< K_x(.), f >_H$ for any $f \in H$ and any x. L_x is the evaluation functional for point x. The existence of K_x can be verified using Riesz representation theorem. We can view K_x as $K(,x)$ which is a function of its first argument and indexed by the second argument.

5.7 Computation of Smoothing Splines

B is lower 4-banded based on Ex 5.2 (a). Lower 4-banded means that for j^{th} columns of matrix B, we have $b_{ij} \neq 0$ only for $i = j, j+1, ..., j+4$. Thus $B'B$ is 4-banded. Next we check Ω_B. For i, j that differs by at least 4, $\int N_i''(t) N_j''(t)\, dt$=0 since their non-zero regions are not overlapped. Thus Ω_B is also 4-banded. The complexity of calculating $M = (B'B + \lambda\Omega_B)$ is also $O\left(p^2 n\right)$. If matrix L from Cholesky decomposition of M is also banded, then we are done. Since by forward substitution with $O(pn)$ we can get $L'\gamma$. And by another back-substitution we can solve for γ via $O(pn)$. Next we look at the complexity of Cholesky decomposition. Normally this is cube in the dimension of the matrix, i.e., $O\left(n^3\right)$. However, if the matrix is p-banded, it should take $O\left(p^2 n\right)$. Since M is 4-banded no matter how large n is, we can fit smoothing splines by $O(n)$, which is big improvement.

5.8 Exercise

Ex 5.1

Let $f(x) = \sum_{j=1}^{6} a_j h_j(x)$. Write out (5.3) for different regions,

$$f(x) = \begin{cases} \sum_{j=1}^{4} a_j x^{j-1} & \text{if } x \leq \xi_1 \\ \left(a_1 - a_5\xi_1^3\right) + \left(a_2 + 3a_5\xi_1^2\right) x + \left(a_2 - 3a_5\xi_1\right) x^2 + \left(a_2 + a_5\right) x^3 & \xi_1 \leq x \leq \xi_2 \\ \left(a_1 - a_5\xi_1^3 - a_6\xi_2^3\right) + \left(a_2 + 3a_5\xi_1^2 + 3a_6\xi_2^2\right) x + \left(a_2 - 3a_5\xi_1 - 3a_6\xi_1\right) x^2 + \left(a_2 + a_5 + a_6\right) x^3 & x \geq \xi_2 \end{cases}$$

We can see that it is a cubic polynomial at each region and since $\forall j, \quad h_j(x)$ is continuous and has continous second order derivatives, the linear combinations $f(x) = \sum_{j=1}^{6} a_j h_j(x)$ preserves those properties

and therefore in particular $f(x)$ has continuous second derivatives at the knots. Since it has 6 degress of freedom, the above formulation should be equivalent to the following naive basis representation of cubic spline:

$$g(x) = \begin{cases} \sum_{j=1}^{4} b_j x^{j-1} & \text{if } x \leq \xi_1 \\ \sum_{j=1}^{4} b_{4+j} x^{j-1} & \xi_1 \leq x \leq \xi_2 \\ \sum_{j=1}^{4} b_{8+j} x^{j-1} & x \geq \xi_2 \end{cases}$$

,where the constraints are

$$\begin{aligned} \sum_{j=1}^{4} b_j \xi_1^{j-1} &= \sum_{j=1}^{4} b_{4+j} \xi_1^{j-1} \\ b_2\xi_1 + 2b_3\xi_1 + 3b_4\xi_1^2 &= b_6\xi_1 + 2b_7\xi_1 + 3b_8\xi_1^2 \\ 2b_3 + 6b_4\xi_1 &= 2b_7 + 6b_8\xi_1 \\ \sum_{j=1}^{4} b_{8+j} \xi_2^{j-1} &= \sum_{j=1}^{4} b_{4+j} \xi_2^{j-1} \\ b_{10}\xi_2 + 2b_{11}\xi_2 + 3b_{12}\xi_2^2 &= b_6\xi_2 + 2b_7\xi_2 + 3b_8\xi_2^2 \\ 2b_{11} + 6b_{12}\xi_2 &= 2b_7 + 6b_8\xi_2 \end{aligned}$$

Ex 5.2

(a)

We use induction on m to prove the following statement: for any $m = 1, ...M$, we have that for any $i = 1, ..., K + 2M - m$, $B_{i,m}(x) = 0$ if $x \notin [\tau_i, \tau_{i+m}]$. The first step of $m = 1$ is trivial from (5.77)

By induction assumption, for any $i = 1, ..., K + 2M - (m-1)$, $B_{i,m-1}(x) = 0$ if $x \notin [\tau_i, \tau_{i+m-1}]$. Then for $i = 1, ..., K + 2M - m$, when $x \notin [\tau_i, \tau_{i+m}]$, we for sure have $x \notin [\tau_i, \tau_{i+m-1}]$ and $x \notin [\tau_{i+1}, \tau_{i+m}]$, therefore, $B_{i,m-1}(x) = B_{i+1,m-1}(x) = 0$ and thus $B_{i,m}(x) = 0$ from (5.78)
Then we set $m = M$. Q.E.D

Similarly, we can use induction to prove that for any $m = 1, ...M$, we have that for any $i = 1, ..., K + 2M - m$, $B_{i,m}(x) \geq 0$ when $x \in [\tau_i, \tau_{i+m}]$

(b)

Similarly, we use induction on m to prove the following statement: for any $m = 1, ...M$, we have that for any $i = 1, ..., K + 2M - m$, $B_{i,m}(x) > 0$ if $x \in (\tau_i, \tau_{i+m})$. The first step of $m = 1$ is again trivial from (5.77). We can also show that it holds for $m = 2$.

By induction assumption, the statement holds for $m - 1$, where $m \geq 3$, i.e. for any $i = 1, ..., K + 2M - (m-1)$, $B_{i,m-1}(x) > 0$ if $x \in (\tau_i, \tau_{i+m-1})$. Then for $i = 1, ..., K + 2M - m$, when $x \in (\tau_i, \tau_{i+m})$, we have either $x \in (\tau_i, \tau_{i+m-1})$ or $x \in (\tau_{i+1}, \tau_{i+m})$; if $x \in (\tau_i, \tau_{i+m-1})$, then $\frac{x-\tau_i}{\tau_{i+m-1}-\tau_{i,}} B_{i,m-1}(x) > 0$

and $\frac{\tau_{i+m}-x}{\tau_{i+m-1}-\tau_i,}B_{i+1,m-1}(x) \geq 0$ and therefore $B_{i,m}(x) > 0$ from (5.78). If $x \in (\tau_{i+1}, \tau_{i+m})$, then from induction assumption $\frac{\tau_{i+m}-x}{\tau_{i+m-1}-\tau_i,}B_{i+1,m-1}(x) > 0$ and $\frac{x-\tau_i}{\tau_{i+m-1}-\tau_i,}B_{i,m-1}(x) \geq 0$, therefore $B_{i,m}(x) > 0$ if $x \in (\tau_{i+1}, \tau_{i+m})$.
Then we set $m = M$. Q.E.D

(c)

For $m = 1$ it is trivial that $\sum_{i=1}^{K+2M-m} B_{i,m}(x) = 1$ for all $x \in [\tau_j, \tau_{j+1})$ where $j = 1, 2, ...K + 2M - 1$. Assume that for $m \geq 2$, the above holds for $m-1$, and we next prove the induction step.

Fix any $j = 1, 2, ..., K + 2M - 1$, and for any $x \in [\tau_j, \tau_{j+1})$, only keeping non-zero $B_{i,m-1}(x)$ and $B_{i+1,m-1}(x)$ in the RHS,

$$\begin{aligned}
\sum_{i=1}^{K+2M-m} B_{i,m}(x) &= \sum_{i=j-m+1}^{j+1} \frac{x-\tau_i}{\tau_{i+m-1}-\tau_i} B_{i,m-1}(x) + \sum_{i=j-m}^{j} \frac{\tau_{i+m}-x}{\tau_{i+m}-\tau_{i+1}} B_{i+1,m-1}(x) \\
&= \sum_{i=j-m+1}^{j+1} \frac{x-\tau_i}{\tau_{i+m-1}-\tau_i} B_{i,m-1}(x) + \sum_{i=j-m}^{j+1} \frac{\tau_{i+m-1}-x}{\tau_{i+m-1}-\tau_i} B_{i,m-1}(x) \\
&= \sum_{i=j-m+1}^{j+1} B_{i,m-1}(x) \\
&= \sum_{i=j-m+1}^{j+1} B_{i,m-1}(x) + \sum_{i=j+2}^{K+2M-m+1} B_{i,m-1}(x) + \sum_{i=1}^{j-m} B_{i,m-1}(x) \\
&= \sum_{i=1}^{K+2M-m+1} B_{i,m-1}(x) \\
&= 1
\end{aligned}$$

The last equality is from induction assumption.

Therefore we know that for $m = M$, we have $j = 1, 2, ..., K + 2M - 1$ and for $x \in [\tau_1, \tau_{K+2M}]$, $\sum_{i=1}^{K+M} B_{i,M}(x) = 1$

(d)

(e)

Ex 5.4

Natural cubic splines require that second and third derivatives outside $[\xi_1, \xi_K]$ to be 0, which in (5.70) means that,

$$\begin{aligned} \beta_2 &= \beta_3 = 0 \\ \beta_3 + \sum_{k=1}^{K} \theta_k &= 0 \\ \beta_2 - \sum_{k=1}^{K} 3\theta_k \xi_k &= 0 \end{aligned}$$

,which is equivalent to

$$\begin{aligned} \beta_2 &= 0 \\ \beta_3 &= 0 \\ \sum_{k=1}^{K} \theta_k &= 0 \\ \sum_{k=1}^{K} \theta_k \xi_k &= 0 \end{aligned}$$

, which is exactly the conditions in (5.71). Apply the first two to (5.70), and replace θ_K by $\theta_K = -\sum_{k=1}^{K-1} \theta_k$, we have,

$$\begin{aligned} f(X) &= \beta_0 + \beta_1 X + \sum_{k=1}^{K-1} \theta_k \left((X - \xi_k)_+^3 - (X - \xi_K)_+^3 \right) \\ &= \beta_0 + \beta_1 X + \sum_{k=1}^{K-1} \theta_k \left(\xi_K - \xi_k \right) \left(\frac{(X - \xi_k)_+^3 - (X - \xi_K)_+^3}{\xi_K - \xi_k} \right) \end{aligned}$$

Plug in $\theta_K = -\sum_{k=1}^{K-1} \theta_k$ to the last constraint then we have

$$\begin{aligned} \sum_{k=1}^{K-1} \left(\xi_K - \xi_k \right) \theta_k &= 0 \\ \left(\xi_K - \xi_{K-1} \right) \theta_{K-1} &= - \sum_{k=1}^{K-2} \left(\xi_K - \xi_k \right) \theta_k \end{aligned}$$

Replacing $(\xi_K - \xi_{K-1})\theta_{K-1}$ by $(\xi_K - \xi_{K-1})\theta_{K-1} = -\sum_{k=1}^{K-2}(\xi_K - \xi_k)\theta_k$ in $f(X)$ gives us

$$\begin{aligned} f(X) &= \beta_0 + \beta_1 X + \sum_{k=1}^{K-2} \theta_k (\xi_K - \xi_k) \left[\left(\frac{(X-\xi_k)_+^3 - (X-\xi_K)_+^3}{\xi_K - \xi_k} \right) - \left(\frac{(X-\xi_{K-1})_+^3 - (X-\xi_K)_+^3}{\xi_K - \xi_{K-1}} \right) \right] \\ &:= \beta_0 + \beta_1 X + \sum_{k=1}^{K-2} \theta_k (\xi_K - \xi_k) [d_k(X) - d_{K-1}(X)] \end{aligned}$$

, which is exactly the basis (5.4) and (5.5)

Ex 5.6

Modify the ξ_k's so that they are between $(0, T)$. And WLOG, we could preprocess the x coordinate of all the data points so that $\forall i$, $x_i \in [0, T]$. Then we could apply the truncated-power basis on region $[0, T]$ as before, i.e. fitting the data using M order truncated power basis

$$f(X) = \sum_{j=0}^{M-1} \beta_j X^j + \sum_{k=1}^{K} \theta_k (X - \xi_k)_+^{M-1}$$

Note that we may want to ensure our periodic function $f(X)$ is smooth and have continuous $(M-2)^{th}$ derivatives everywhere. For that we can simply add the following $(M-1)$ constraints when fitting the splines to ensure that at the beginning and the end of any period, the function has continuous $(M-2)$ derivatives,

$$\beta_0 = \sum_{j=0}^{M-1} \beta_j T^j + \sum_{k=1}^{K} \theta_k (T - \xi_k)^{M-1}$$

and for any $l = 1, 2, ..M-2$,

$$\beta_l = \sum_{j=l}^{M-1} \beta_j \left(\prod_{p=j-l+1}^{j} p \right) T^{j-l} + \left(\prod_{p=M-l}^{M-1} p \right) \sum_{k=1}^{K} \theta_k (T - \xi_k)^{M-l-1}$$

Ex 5.7

(a)

Since g is natural cubic spline, $g''(x) = g'''(x) = 0$ for $x \le x_1$ or $x \ge x_N$. Apply integration by parts,

$$\begin{aligned}\int_a^b g''(x)\,h''(x)\,dx &= h'(x)\,g''(x)\,|_a^b - \int_a^b h'(x)\,dg''(x)\\ &= 0-0-\int_a^b h'(x)\,g'''(x)\,dx\\ &= -\int_{x_1}^{x_N} h'(x)\,g'''(x)\,dx\end{aligned}$$

For $g'''(x)$, it is a piecewise linear (not necessarily defined on $x_j's$) function on (x_1, x_N) since it is a cubic spline. Therefore,

$$\begin{aligned}\int_a^b g''(x)\,h''(x)\,dx &= -\int_{x_1}^{x_N} h'(x)\,g'''(x)\,dx\\ &= -\sum_{j=1}^{N-1}\int_{x_j}^{x_{j+1}} h'(x)\,g'''\left(x_j^+\right)dx\\ &= -\sum_{j=1}^{N-1}\left(h(x_{j+1})-h(x_j)\right)g'''\left(x_j^+\right)\\ &= 0\end{aligned}$$

The last equality is due to $\forall j$, $\tilde{g}(x_j) = g(x_j) = z_j$

(b)

$$\begin{aligned}\int_a^b \tilde{g}''(t)^2\,dt &= \int_a^b \left(\tilde{g}''(t)-g''(t)+g''(t)\right)^2 dt\\ &= \int_a^b h''(t)^2\,dt+\int_a^b g''(t)^2\,dt+2\int_a^b h''(t)\,g''(t)\,dt\\ &= \int_a^b h''(t)^2\,dt+\int_a^b g''(t)^2\,dt \geq \int_a^b g''(t)^2\,dt\end{aligned}$$

The equality holds iff $\int_a^b h''(t)^2\,dt = 0$, i.e., $h''(x) = 0$ on $[a,b]$. Since $\forall j = 1,2,..,N$, $h(x_j) = 0$ by assumption, $\int_a^b h''(t)^2\,dt = 0 \iff h(x) = 0$ on $[a,b]$

(c)

Given a general twicely continously differentiable function f that is not a natural cubic spline with knots at $(x_j)_N$, let g again be a natural cubic spline with knots at $(x_j)_N$ but interpolating the $(x_j, f(x_j))_N$ exactly. Thus,

$$\sum_{j=1}^{N}\left(y_j - f(x_j)\right)^2 = \sum_{j=1}^{N}\left(y_j - g(x_j)\right)^2$$

Howeverm from (b), we know that $\int_a^b f''(t)^2\,dt > \int_a^b g''(t)^2\,dt$ since we do not have $f(x) = g(x)$ on $[a,b]$. Therefore we conclude that ,

$$\sum_{j=1}^{N}(y_j - f(x_j))^2 + \lambda\int_a^b f''(t)^2\,dt > \sum_{j=1}^{N}(y_j - g(x_j))^2 + \lambda\int_a^b g''(t)^2\,dt$$

Therefore, the minimizer must be a natural cubic spline with knots at $(x_j)_N$.

Ex 5.12

The minimizer of the following problem

$$\min_f \sum_{j=1}^{N} w_i(y_j - f(x_j))^2 + \lambda\int_a^b f''(t)^2\,dt$$

must be again a natural cubic spline with knots at each of the x_i because the arguments of Ex 5.7 still applies regardless of $w_i's$.
Therefore for the smoothing spline problem with ties in X we can cast the problem (5.9) equivalently as et

$$\min_f \sum_{j=1}^{N} w_j(\bar{y}_j - f(x_j))^2 + \lambda\int_a^b f''(t)^2\,dt$$

with weighted observations where $w_i = n_i$ and let n_i be the number of observations at point x_i, $i = 1,2,3...N$, $\bar{y}_j = \frac{\sum_{i=1}^{n_j} y_{i,j}}{n_j}$ as the mean of y_i at x_i and N as the unique $x_i's$ in the training data. The same solution charaterization applies. Note that

$$n_j\left(\frac{\sum_{i=1}^{n_j} y_{i,j}}{n_j} - f(x_j)\right)^2 - \sum_{i=1}^{n_j}(y_{i,j} - f(x_j))^2 = \frac{\left(\sum_{i=1}^{n_j} y_{i,j}\right)^2}{n_j} - \sum_{i=1}^{n_j} y_{i,j}^2$$

, which is independent of f

Ex 5.15

(a)

We start from the definitions (5.45), (5.46) and (5.47)

From (5.47) I guess the definition of inner product of the space $\mathcal{H}_K$ is as $\langle f(x), g(x)\rangle_{\mathcal{H}_K} := \sum_{i=0}^{\infty}\frac{c_i d_i}{\gamma_i}$ if $f(x) = \sum_{i=0}^{\infty} c_i\phi_i(x) \in \mathcal{H}_K$ and $g(x) = \sum_{i=0}^{\infty} d_i\phi_i(x) \in \mathcal{H}_K$. This definition is well-defined since due to (5.47), $\frac{c_i}{\sqrt{\gamma_i}}$ and $\frac{d_i}{\sqrt{\gamma_i}}$ are both l_2. This definition intuitively makes sense if we view $h_i(x) := \sqrt{\gamma_i}\phi_i(x)$ as the true basis and thus the "coordinates" of a function $f(x) = \sum_{i=0}^{\infty} c_i\phi_i(x) = \sum_{i=0}^{\infty}\frac{c_i}{\sqrt{\gamma_i}}h_i(x)$ is really the l_2

vector $\{\frac{c_i}{\sqrt{\gamma_i}}\}_i$. Then,

$$\begin{aligned}\langle K(x,x_i), f(x_i)\rangle_{\mathcal{H}_K} &= \sum_{i=0}^{\infty} \frac{\gamma_i \phi_i(x_i) c_i}{\gamma_i} \\ &= \sum_{i=0}^{\infty} c_i \phi_i(x_i) \\ &= f(x_i)\end{aligned}$$

(b)

$$\begin{aligned}\langle K(x,x_i), K(x,x_j)\rangle_{\mathcal{H}_K} &= \sum_{i=0}^{\infty} \frac{\gamma_i \phi_i(x_i) \gamma_i \phi_i(x_j)}{\gamma_i} \\ &= \sum_{i=0}^{\infty} \gamma_i \phi_i(x_j)\, \phi_i(x_i) \\ &= K(x_i, x_j)\end{aligned}$$

Or we can use part (a) to see this directly.

(c)

$$\begin{aligned}J(g(x)) &= \langle g(x), g(x)\rangle_{\mathcal{H}_K} \\ &= \sum_{i=0}^{N}\sum_{j=0}^{N} \alpha_i \alpha_j \langle K(x,x_i), K(x,x_j)\rangle_{\mathcal{H}_K} \\ &= \sum_{i=0}^{N}\sum_{j=0}^{N} \alpha_i \alpha_j K(x_i, x_j)\end{aligned}$$

(d)

From (a), we know that for any i, $\rho(x_i) = \langle K(x,x_i), \rho(x)\rangle_{\mathcal{H}_K} = 0$ and therefore $\forall i$,

$$\begin{aligned}\tilde{g}(x_i) &= \langle K(x,x_i), \tilde{g}(x_i)\rangle_{\mathcal{H}_K} \\ &= \langle K(x,x_i), g(x_i)\rangle_{\mathcal{H}_K} \\ &= g(x_i)\end{aligned}$$

Therefore $\sum_{i=1}^{N} L\left(y_i, \tilde{g}\left(x_i\right)\right) = \sum_{i=1}^{N} L\left(y_i, g\left(x_i\right)\right)$. Also note that $\left\langle \rho\left(x\right), g\left(x\right)\right\rangle_{\mathcal{H}_K} = \left\langle \rho\left(x\right), \sum_{j=1}^{N} \alpha_j K\left(x, x_j\right)\right\rangle_{\mathcal{H}_K} = \sum_{j=1}^{N} \alpha_j \left\langle \rho\left(x\right), K\left(x, x_j\right)\right\rangle_{\mathcal{H}_K} = 0$. Therefore, we have,

$$\begin{aligned} J\left(\tilde{g}\right) &= \left\langle g\left(x\right), g\left(x\right)\right\rangle_{\mathcal{H}_K} + \left\langle \rho\left(x\right), \rho\left(x\right)\right\rangle_{\mathcal{H}_K} + 2\left\langle \rho\left(x\right), g\left(x\right)\right\rangle_{\mathcal{H}_K} \\ &= J\left(g\right) + ||\rho||^2_{\mathcal{H}_K} \geq J\left(g\right) \end{aligned}$$

"=" holds iff $||\rho||^2_{\mathcal{H}_K} = 0 \iff \rho\left(x\right) = 0$ from the definition of inner product.

This shows that the optimal solution of (5.48) must lie in the linear span of $\{K\left(x, x_i\right)\}_{i=1}^{n}$, a closed subspace of $\mathcal{H}_K$

Ex 5.18

Since a general p^{th} order polynomial is orthogonal to ψ, it is orthogonal to subspace W_j for any j, then for it to be exists in the space $V_0 \oplus W_1 \oplus W_2 \cdots$, it must be in V_0, i.e., is represented exactly in V_0. Q.E.D

6 Kernel Smoothing Methods

6.1 6.1

(6.8) is from solving the optimization problem using matrix derivatives. The solution is exactly the same as in weighted least squares.

6.2 6.4 structural methods

Varing coefficient models in Equation (6.17) is similar to the IO model in micro-data where individual characteristics interacting with price elaciticity, where features for predicting whether customer i will purchase product j is $(X, Z) = (p_j, z_i)$. However in IO models, usually we just do a linear interaction, i.e., $u_{ij} = \beta_0 + (\beta_1 + z_i'\theta)p_j$ where here in varying coefficient models, we have a kernel regression form.

6.3 Exercise

Ex 6.2

Let $A := \sum_i K_\lambda(x_0, x_i)$, $C := \sum_i x_i K_\lambda(x_0, x_i)$ and $D := \sum_i x_i^2 K_\lambda(x_0, x_i)$. Then using the notation around (6.8), note that

$$\begin{aligned} b(x_0)' (B'W(x_0)B)^{-1} B'W(x_0)B &= b(x_0)' (B'W(x_0)B)^{-1} B'W(x_0) \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_{N-1} \\ 1 & x_N \end{bmatrix} \\ &= \begin{bmatrix} 1 & x_0 \end{bmatrix} \end{aligned}$$

Then it is trivial that

$$\begin{aligned} \sum_i l_i(x_0) &= b(x_0)' (B'W(x_0)B)^{-1} B'W(x_0)1 \\ &= 1 \end{aligned}$$

Also,

$$\begin{aligned} \sum_i l_i(x_0)(x_i - x_0) &= \sum_i l_i(x_0) x_i - x_0 \sum_i l_i(x_0) \\ &= x_0 - x_0 * 1 \\ &= 0 \end{aligned}$$

For polynomial regression, repeat what we have done,

$$b(x_0)' (B'W(x_0)B)^{-1} B'W(x_0)B = \begin{bmatrix} 1 & x_0 & \dots & x_0^k \end{bmatrix}$$

Then it is trivial that (including the case where B only has one column of $1's$),

$$\begin{aligned} b_0(x_0) = \sum_i l_i(x_0) &= b(x_0)' (B'W(x_0)B)^{-1} B'W(x_0)1 \\ &= 1 \end{aligned}$$

Also, for $j = 1, 2, ...k$

$$\begin{aligned} b_j(x_0) &= \sum_{i=1}^{N}\sum_{m=1}^{j} l_i(x_0)\binom{j}{m} x_i^m (-x_0)^{j-m} \\ &= \sum_{m=1}^{j}\binom{j}{m}(-x_0)^{j-m}\sum_{i=1}^{N} l_i(x_0)\, x_i^m \\ &= \sum_{m=1}^{j}\binom{j}{m}(-x_0)^{j-m} x_0^m \\ &= (x_0 - x_0)^j = 0 \end{aligned}$$

The implications for the bias is

$$\begin{aligned} \mathbb{E}\left[\hat{f}(x_0)\right] &= \sum_{i=1}^{N} l_i(x_0)\left(\sum_{m=0}^{k} f^{(m)}(x_0)(x_i - x_0)^m + R\right) \\ &= \sum_{m=0}^{k} f^{(m)}(x_0)\sum_{i=1}^{N} l_i(x_0)(x_i - x_0)^m + R \\ &= f(x_0) + \sum_{m=1}^{k} f^{(m)}(x_0) * 0 + R \\ &= f(x_0) + R \end{aligned}$$

Therefore the bias is only on order $k+1$ and above.

Ex 6.7

Define $B_{(j)} := \begin{bmatrix} 1 & x_1 & x_1^2 & \dots & x_1^d \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_{j-1} & x_{j-1}^2 & & x_{j-1}^d \\ 1 & x_{j+1} & x_{j+1}^2 & & x_{j+1}^d \\ \vdots & \vdots & \vdots & & \vdots \end{bmatrix}$, $W(x_j)$ is N by N diagonal matrix with i^{th} diagonal element $K_\lambda(x_j, x_i)$ and $W_{(j)}$ is $N-1$ by $N-1$ diagonal matrix which is equal to $W(x_j)$ without j^{th} row and column. Define matrix $S_\lambda(j,i) := l_i(x_j) = b(x_j)'\left(B'W(x_j)B\right)^{-1}B'W(x_j)$ so that $\hat{f} = S_\lambda y$. Define $w_j := K_\lambda(x_j, x_j)$

$$\begin{aligned}
\left(B'_{(j)}W_{(j)}B_{(j)}\right)^{-1} &= (B'W(x_j)B)^{-1} + \frac{(B'W(x_j)B)^{-1} b(x_j) b(x_j)' w_j (B'W(x_j)B)^{-1}}{1 - w_j b(x_j)' (B'W(x_j)B)^{-1} b(x_j)} \\
\left(B'_{(j)}W_{(j)}B_{(j)}\right)^{-1} b(x_j) &= (B'W(x_j)B)^{-1} b(x_j) + \frac{(B'W(x_j)B)^{-1} b(x_j) b(x_j)' w_j (B'W(x_j)B)^{-1} b(x_j)}{1 - w_j b(x_j)' (B'W(x_j)B)^{-1} b(x_j)} \\
&= \frac{(B'W(x_j)B)^{-1} b(x_j)}{1 - w_j b(x_j)' (B'W(x_j)B)^{-1} b(x_j)}
\end{aligned}$$

Therefore,

$$\begin{aligned}
b(x_j)' \left(B'_{(j)}W_{(j)}B_{(j)}\right)^{-1} B'_{(j)}W_{(j)}y_{(j)} - y_j &= \frac{b(x_j)' (B'W(x_j)B)^{-1}}{1 - w_j b(x_j)' (B'W(x_j)B)^{-1} b(x_j)} (B'W(x_j)y - w_j y_j b(x_j)) - y_j \\
&= \frac{b(x_j)' (B'W(x_j)B)^{-1} B'W(x_j)y - y_j}{1 - w_j b(x_j)' (B'W(x_j)B)^{-1} b(x_j)}
\end{aligned}$$

Define $D := diag\left(1 - w_j b(x_j)' (B'W(x_j)B)^{-1} b(x_j)\right)$

$$\begin{aligned}
loocvErr &= \sum_{j=1}^{N} \left(b(x_j)' \left(B'_{(j)}W_{(j)}B_{(j)}\right)^{-1} B'_{(j)}W_{(j)}y_{(j)} - y_j\right)^2 \\
&= ||D^{-1}(S_\lambda - I)y||^2 \\
&= y'(S'_\lambda - I)D^{-2}(S_\lambda - I)y
\end{aligned}$$

Ex 6.8

Note that the joint density estimation is

$$\hat{f}_{X,Y}(x,y) = \frac{1}{N}\sum_{i=1}^{N} \phi_\lambda(x - x_i)\phi_\lambda(y - y_i)$$

Therefore,

$$\begin{aligned}
\mathbb{E}[Y|X=x] &= \int_y \frac{\hat{f}_{X,Y}(x,y)\,y}{\int_y \hat{f}_{X,Y}(x,y)\,dy} dy \\
&= \frac{\frac{1}{N}\sum_{i=1}^N \phi_\lambda(x-x_i)\int_y \phi_\lambda(y-y_i)\,y dy}{\frac{1}{N}\sum_{i=1}^N \phi_\lambda(x-x_i)\int_y \phi_\lambda(y-y_i)\,dy} \\
&= \frac{\frac{1}{N}\sum_{i=1}^N \phi_\lambda(x-x_i)\,y_i}{\frac{1}{N}\sum_{i=1}^N \phi_\lambda(x-x_i)} \\
&= \frac{\sum_{i=1}^N \phi_\lambda(x-x_i)\,y_i}{\sum_{i=1}^N \phi_\lambda(x-x_i)}
\end{aligned}$$

, which is indeed a NW estimator

Ex 6.10

$$\begin{aligned}
PE(\lambda) &= \sigma^2 + \mathbb{E}\left[\frac{1}{N}\sum_{i=1}^N \left(f(x_i) - y_i + y_i - \hat{f}_\lambda(x_i)\right)^2\right] \\
&= \sigma^2 + \mathbb{E}\left[\frac{1}{N}\sum_{i=1}^N \left(-\epsilon_i + y_i - \hat{f}_\lambda(x_i)\right)^2\right] \\
&= \sigma^2 + \sigma^2 + 2\frac{1}{N}\mathbb{E}\left[\sum_{i=1}^N \epsilon_i\left(-y_i + \hat{f}_\lambda(x_i)\right)\right] + \mathbb{E}\left[\frac{1}{N}\sum_{i=1}^N \left(y_i - \hat{f}_\lambda(x_i)\right)^2\right] \\
&= 2\sigma^2 + \mathbb{E}[ASR(\lambda)] + \frac{2}{N} tr\left(Cov\left(\epsilon, (S_\lambda - I_N)\,\epsilon\right)\right) \\
&= 2\sigma^2 + \mathbb{E}[ASR(\lambda)] + \frac{2}{N} tr\left((S_\lambda - I_N)\,I_N\sigma^2\right) \\
&= 2\sigma^2 + \mathbb{E}[ASR(\lambda)] + \frac{2\sigma^2}{N}\left(tr(S_\lambda) - N\right) \\
&= \mathbb{E}[ASR(\lambda)] + \frac{2\sigma^2}{N} tr(S_\lambda)
\end{aligned}$$

Therefore, $ASR(\lambda) + \frac{2\sigma^2}{N} tr(S_\lambda)$ is unbiased for $PE(\lambda)$

7 Model Assesment and Selection

7.1 P224 Bias Variance

Think of $f(x) = \mathbb{E}[y|x]$ throughout, which is the function we want to approximate. In Equation (7.13), we can solve for β by taking partial derivative and pass it inside expectation: treating x as a d dimention

random vector, the only random object,

$$\begin{aligned} 2\mathbb{E}\left[x(f\left(x\right)-x'\beta^*)\right]&=0\\ \beta^*&=\mathbb{E}[xx']^{-1}\mathbb{E}[xf\left(x\right)] \end{aligned}$$

This is the classic Econometric result of linear projection, projecting $f\left(x\right)$ onto the linear space of x, and β^* is the coefficient in the population and under the assumption of exogeneity we would have $\hat{\beta}_{OLS}\rightarrow\beta^*$ as $n\rightarrow N$.

Equation (7.14) holds since x_0 follows the same distribution as x above in $\mathbb{E}\left[x(f\left(x\right)-x'\beta^*)\right]=0$, we look at the first row corresponding to the intercept and conclude that

$$\begin{aligned} \mathbb{E}\left[1(f\left(x\right)-x'\beta^*)\right]&=0\\ \mathbb{E}\left[f\left(x\right)-x'\beta^*\right]&=0 \end{aligned}$$

Therefore the cross term equals to 0 and (7.14) holds.

For the interpretation of Equation (7.14), if the truth is a linear model, i.e., $f\left(x\right)=x'\beta^{truth}$. We can see that $\beta^*=\beta^{truth}$ from the expression of β^* and the model bias is 0. Regardless of whether the true model is linear or not, if we fit a linear model using OLS, as pointed out in the text, we will have 0 estimation bias. To have 0 estimation bias, we need to have pointwise 0 for all possible value of x_0, i.e. we must have,

$$\mathbb{E}_\tau\left[\hat{\beta}_{OLS}\right]=\beta^*$$

,which means that OLS estimator is unbiased estimate for the linear projection. We know from Econometrics that this only happens when we treat regressors as non-random and $\mathbb{E}_\tau$ is just over y or ϵ's in the training set τ. Thus we have

$$\begin{aligned} \mathbb{E}_\tau\left[\hat{\beta}_{OLS}\right]&=\mathbb{E}_{y\in\tau}\left[\boldsymbol{\left(X'X\right)^{-1}XY}\right]\\ &=\boldsymbol{\left(X'X\right)^{-1}X}\mathbb{E}_{y\in\tau}\left[\boldsymbol{Y}\right]\\ &=\boldsymbol{\left(X'X\right)^{-1}X}f\left(\boldsymbol{X}\right) \end{aligned}$$

When the truth is linear, and we are using OLS, and predictors are treated as non-random, Model bias and estimation bias are both 0 (pointwise at all x_0), which corresponds to the case discussed in p26 Equation (2.27)

Following this line of analysis, in Figure 7.2 closest fit in population, rather than closest fit, should correspond to $x'\beta^*$

7.2 P239 VC dimension

VC dimension analysis starts with a particular fixed hypothesis and the randomness is from training set τ. We use some form of Hoeffdin Inequality to bound the deviation of in sample error from true out-of-sample error, viewed as deviation between sample average with N examples and true expectation. The bound holds with probability $1-\eta$ and the upper bound involves N. The probability is w.r.t. drawing training data set. However during training we pick the hypothesis adaptively from a larger hypothesis set $\mathcal{H}$. We could modify the inequality to the case of $\mathcal{H}$ so that the upper bound involves $|\mathcal{H}|$ but this only works with finite $|\mathcal{H}|$. VC dimension is the right quantity to replace $|\mathcal{H}|$ by if $|\mathcal{H}| = +\infty$. So the inequality's upper bounds gets weaker but most importantly it works for any final hypothesis we adaptively pick from the set $\mathcal{H}$. In this sense this result is stronger than C_p, AIC type estimates like in Equation (7.24). Over there, as pointed out in text, it is easier to analyze after take $\mathbb{E}_\tau$, where under some additional assupmtion the optimism can be related to $Cov\left(\hat{y}_i, y_i\right)$. Note that Cov is over the distribution drawing trainng set and here we have similar obsevations: if the model is more complex or $|\mathcal{H}|$ is more complex, $\hat{y}$ should be more correlated with y, which translates to a larger generalization error keeping the training error the same.

7.3 P254 First paragraph on Model Selection and Test Error Estimates

There are several issues here: one is to select the best model based on some criterion and the other is to estimate test error. The first issue only concerns the relative, instead of absolute measure. As pointed out by the text, the criterion can be biased while not affecting anything and as shown in Fig. 7.7 and Fig 7.13, all methods considered do pretty well at picking out the best model, either AIC, bootstrap or CV.

As for test error estimate, now we care about bias and it is pointed out in the text that sometimes CV or Bootstrap has smaller upward bias then AIC, where the bias is averaged over different training set τ (say simuluating 100 training set). The upward bias from CV or Bootstrap is due to the smaller sample used to estimate the model. There are another force of downward bias in the estimated error for the chosen model since we are essentially fitting some extra degree of freedom on validation set or using CV, which creates some optimistic error estimate for the final chosen model which is optimized over validation set. As pointed out in the text if we heavily optimize hyperparamters over validation set, this could dominate the upward bias so that CV error underestimates the true test error. In that case we need to use a more traditional hold-out test set.

Another issue is what CV estimated is Expected Prediction Error or Conditional Prediction error), i.e. $\mathbb{E}_\tau \mathbb{E}_{(x_0,y_0)|\tau}$ vs. $\mathbb{E}_{(x_0,y_0)|\tau}$. For AIC or C_p described in Section 7.5, they are proxy for expected prediction error since all derivation is from Equation (7.22). For CV though, Section (7.12) in the book says it is the expected prediction error as well.

Note that the latter can be well estimated from a hold-out test set.

7.4 Exercise

Ex 7.1

Assuming Equation 7.22 holds, (which is partially shown in Ex 7.4), we only need to show Equation 7.23.

Let H be the projection matrix $\hat{y} = Hy$, then similarly to Ex 7.5, we have

$$\begin{aligned}\sum_{i=1}^{N} Cov\left(y_i, \hat{y}_i\right) &= tr\left(Cov\left(y, Hy\right)\right) \\ &= \sigma^2 tr\left(H\right) \\ &= \sigma^2 d\end{aligned}$$

Ex 7.2

Ex 7.3

(a)

We state without proof a matrix equality. For square matrix A and column vector a, b, we have

$$(A + ab')^{-1} = A^{-1} - \frac{A^{-1}ab'A^{-1}}{1 + b'A^{-1}a}$$

Then in OLS case, let $b_{(-i)}$ denote the OLS estimated coef without i^{th} row and let b be the OLS coefficients with all data. Let $\hat{f}(x_i) := \hat{y}_i := x_i'b$. We have,

$$\begin{aligned}b_{(-i)} &= (X'_{(-i)}X_{(-i)})^{-1}X'_{(-i)}y_{(-i)} \\ &= (X'X - x_ix_i')^{-1}(X'y - x_iy_i) \\ &= [(X'X)^{-1} + \frac{(X'X)^{-1}x_ix_i'(X'X)^{-1}}{1 - h_{ii}}](X'y - x_iy_i) \\ &= b - (X'X)^{-1}x_iy_i + \frac{(X'X)^{-1}x_i\hat{y}_i - (X'X)^{-1}x_ih_{ii}y_i}{1 - h_{ii}} \\ &= b + (X'X)^{-1}x_i[\frac{-(1 - h_{ii})y_i}{1 - h_{ii}} + \frac{\hat{y}_i - h_{ii}y_i}{1 - h_{ii}}] \\ &= b - (X'X)^{-1}x_i\frac{y_i - \hat{y}_i}{1 - h_{ii}}\end{aligned}$$

Then $y_i - \hat{f}^{-i}\left(x_i\right) = y_i - x_i'b_{(-i)} = y_i - x_i'b + x_i'(X'X)^{-1}x_i\frac{y_i - \hat{y}_i}{1 - h_{ii}} = y_i - \hat{y}_i + h_{ii}\frac{y_i - \hat{y}_i}{1 - h_{ii}} = \frac{y_i - \hat{y}_i}{1 - h_{ii}}$. This shows that the formula works for OLS and non-adaptive basis case.

For cubic smoothing spline or regularized OLS case, we have,

(b)

(c)

Ex 7.4

We denote $\mathbb{E}_y$ as $\mathbb{E}$ below for simplicity, i.e. taking expectation over training set outcome values and the predictors in the training set are fixed.

$$\begin{aligned}
\mathbb{E}\left[Err_{in}\right] &= \mathbb{E}\left[\frac{1}{N}\sum_{i=1}^{N}\mathbb{E}_{Y^0}\left[\left(Y_i^0 - f\left(x_i\right) + f\left(x_i\right) - \hat{f}\left(x_i\right)\right)^2 |\tau\right]\right] \\
&= \mathbb{E}\left[\frac{1}{N}\sum_{i=1}^{N}\sigma^2 + \mathbb{E}_{Y^0}\left[\left(f\left(x_i\right) - \hat{f}\left(x_i\right)\right)^2 |\tau\right]\right] \\
&= \sigma^2 + \frac{1}{N}\sum_{i=1}^{N}\mathbb{E}\left[\left(f\left(x_i\right) - \hat{f}\left(x_i\right)\right)^2\right]
\end{aligned}$$

$$\begin{aligned}
\mathbb{E}\left[e\bar{r}r\right] &= \mathbb{E}\left[\frac{1}{N}\sum_{i=1}^{N}\left[\left(y_i - f\left(x_i\right) + f\left(x_i\right) - \mathbb{E}\left[\hat{f}\left(x_i\right)\right] + \mathbb{E}\left[\hat{f}\left(x_i\right)\right] - \hat{f}\left(x_i\right)\right)^2\right]\right] \\
&= \sigma^2 + \frac{1}{N}\sum_{i=1}^{N}\mathbb{E}\left[\left(f\left(x_i\right) - \hat{f}\left(x_i\right)\right)^2\right] + 2\mathbb{E}\left[\frac{1}{N}\sum_{i=1}^{N}\left(y_i - f\left(x_i\right)\right)\left(f\left(x_i\right) - \mathbb{E}\left[\hat{f}\left(x_i\right)\right]\right)\right] \\
&\quad +2\mathbb{E}\left[\frac{1}{N}\sum_{i=1}^{N}\left(y_i - f\left(x_i\right)\right)\left(\mathbb{E}\left[\hat{f}\left(x_i\right)\right] - \hat{f}\left(x_i\right)\right)\right] \\
&= \mathbb{E}\left[Err_{in}\right] + 0 - \frac{2}{N}\mathbb{E}\left[\sum_{i=1}^{N}\left(y_i - \mathbb{E}\left[y_i\right]\right)\left(\hat{f}\left(x_i\right) - \mathbb{E}\left[\hat{f}\left(x_i\right)\right]\right)\right] \\
&= \mathbb{E}\left[Err_{in}\right] - \frac{2}{N}\sum_{i=1}^{N}Cov\left(y_i, \hat{y}_i\right)
\end{aligned}$$

Therefore, $\mathbb{E}\left[op\right] = \mathbb{E}\left[Err_{in}\right] - \mathbb{E}\left[e\bar{r}r\right] = \frac{2}{N}\sum_{i=1}^{N}Cov\left(y_i, \hat{y}_i\right)$

Ex 7.5

Similar to Ex 6.10, we have

$$\begin{aligned}
\sum_{i=1}^{N}Cov\left(y_i, \hat{y}_i\right) &= tr\left(Cov\left(y, Sy\right)\right) \\
&= tr\left(SCov\left(y, y\right)\right) \\
&= \sigma_\epsilon^2 tr\left(S\right)
\end{aligned}$$

Ex 7.6

In an additive error model with squared loss, we show in Ex 7.4 that $\frac{2}{N}\sum_{i=1}^{N} Cov\left(y_i, \hat{y}_i\right)$ is the correct expected optimism. Define $i(l)$ so that $x_{i(l)}$ is the l^{th} closest point to x_i. Note that in this framework we are treating x as fixed so $\mathbb{E}_\tau$ is really only $\mathbb{E}_y$ for $y's$ in the training set. We have for knn algorithm,

$$\begin{aligned}
\frac{2}{N}\sum_{i=1}^{N} Cov\left(y_i, \hat{y}_i\right) &= \frac{2}{N}\sum_{i=1}^{N} Cov\left(y_i, \frac{1}{k}\sum_{l=1}^{k} y_{i(l)}\right) \\
&= \frac{2}{N}\sum_{i=1}^{N} Cov\left(y_i, \frac{1}{k}y_i + \frac{1}{k}\sum_{l=2}^{k} y_{i(l)}\right) \\
&= \frac{2}{N}\sum_{i=1}^{N} \frac{1}{k}\sigma_\epsilon^2 + 0 \\
&= \frac{2}{N}\left(\frac{N}{k}\right)\sigma_\epsilon^2
\end{aligned}$$

So $\frac{N}{k}$ is the effective degree of freedom, or the definition of effective degree of freedom as $\frac{\sum_{i=1}^{N} Cov(y_i, \hat{y}_i)}{\sigma_\epsilon^2}$ is applicable here in the sense that if we plug in $d = \frac{N}{k}$ into $\frac{2}{N}d\sigma_\epsilon^2$ we get back the correct expected optimism.

Ex 7.7

Under square loss,

$$\begin{aligned}
GCV\left(\hat{f}\right) &= \frac{1}{N}\sum_{i=1}^{N} \frac{\left(y_i - \hat{f}\left(x_i\right)\right)^2}{1 - \frac{tr(S)}{N}} \\
&\backsimeq \frac{1}{N}\sum_{i=1}^{N}\left(y_i - \hat{f}\left(x_i\right)\right)^2 + 2\frac{tr\left(S\right)}{N}\frac{1}{N}\sum_{i=1}^{N}\left(y_i - \hat{f}\left(x_i\right)\right)^2 \\
&= e\bar{r}r + 2\frac{d}{N}\tilde{\sigma}_\epsilon^2
\end{aligned}$$

The last equality uses that $tr\left(S\right)$ is a reasonable expression for number of parameters d in C_p and a different way to estimate σ^2, $\frac{1}{N}\sum_{i=1}^{N}\left(y_i - \hat{f}\left(x_i\right)\right)^2$

Ex 7.8

Let $\alpha = \pi\left(1 + \sum_{j=1}^{l} 10^j y_j^*\right)$, where $y_j^* = 0$ if $y_j = 1$ and $y_j^* = 1$ if $y_j = 0$ for any j. We now prove that for any l, any$\left\{10^{-i}, y_i\right\}_{i=1}^{l}$ and any $i = 1, 2, ...l$, using the above defined α, we have $sin\left(\alpha 10^{-i}\right) > 0$ if $y_i = 1$ and $sin\left(\alpha 10^{-i}\right) < 0$ if $y_i = 0$.

For any i,

$$
\begin{aligned}
sin\left(\alpha 10^{-i}\right) &= sin\left(\pi\left(10^{-i}+\sum_{j=1}^{l}10^{j-i}y_j^*\right)\right) \\
&= sin\left(\pi\left(10^{-i}+\sum_{j=1}^{i-1}10^{j-i}y_j^*+y_i^*+\sum_{j=i+1}^{l}10^{j-i}y_j^*\right)\right) \\
&= sin\left(\pi\left(10^{-i}+\sum_{j=1}^{i-1}10^{j-i}y_j^*+y_i^*\right)\right) \\
&= sin\left(\pi\left(10^{-i}+\sum_{j=1}^{i-1}10^{j-i}y_j^*+y_i^*\right)\right)
\end{aligned}
$$

The term $\left(10^{-i}+\sum_{j=1}^{i-1}10^{j-i}y_j^*+y_i^*\right)$ in the last line above is equal to $\left(y_i^*.y_{i-1}^*y_{i-2}^*\ldots y_1^*1\right)\pi$ which lies in $(0,\pi)$ if $y_i^*=0$ and lies in $(\pi,2\pi)$ if $y_i^*=1$, which leads to the results we want, $sin\left(\alpha 10^{-i}\right)>0$ if $y_i^*=0$ and $y_i=1$ while $sin\left(\alpha 10^{-i}\right)<0$ if $y_i^*=1$ and $y_i=0$

Ex 7.9

Ex 7.10

This comment is wrong and does not indicate that cross-validation is biased in this simple case. The search over all predictors is one kind of fitting and if we just search based on $\frac{4}{5}$ of the training data, the selected binary predictor will have expected error = 0.5 over the left over $\frac{1}{5}$ of data since the response is purely random.

8 Model Averaging and Inference

8.1 EM algorithm

For expression (8.40), assuming Δ_i are all known, we can solve for π,θ_1 and θ_2 separately. For π, FOC gives $\frac{\sum(1-\Delta_i)}{1-\pi}=\frac{\sum\Delta_i}{\pi}$. Thus $\pi=\frac{\sum\Delta_i}{n}$

In the general EM algorithm in section 8.5.2, the expectation step in (8.43) is w.r.t. missing random variable Z^m.

8.2 Exericise

Ex 8.1

log function is concave and thus we have

$$\begin{aligned}\mathbb{E}_q[\log \frac{r(Y)}{q(Y)}] &\leq \log \mathbb{E}_q[\frac{r(Y)}{q(Y)}] \\ &= \log \int \frac{r(y)}{q(y)} q(y)\, dy \\ &= \log 1 = 0\end{aligned}$$

"=" iff $\log \frac{r(Y)}{q(Y)}$ is degenerate, i.e. $r = q$ a.e.. Thus $\mathbb{E}_q[\log r(Y)] \leq \mathbb{E}_q[\log q(Y)]$

Ex 8.2

We ignore the positive constraints for now and hopefully this constraint is satisfied at the maximizer we arrive at. We use Lagrangian Multiplier onto the equality constraint and thus we need to maximize over all $p_{z_m} := \mathbb{P}[Z_m = z_m]$ where z_m is any potential value the vector Z_m can take. For simplicity we consider only the case when Z_m takes finite number of values. In Gaussian Mixture models, $Z_m = \{\Delta_i\}_{i=1}^n$ and thus z_m can take values from 2^n values and thus finite-dimensional.

$$\max_{p_{z_m}} \sum_{z_m} [p_{z_m} \log \mathbb{P}(Z, Z_m = z_m; \theta')] - \sum_{z_m} [p_{z_m} \log p_{z_m}] + \lambda \left(\sum_{z_m} p_{z_m} - 1 \right)$$

FOC gives,

$$\begin{aligned}\log \mathbb{P}(Z, Z_m = z_m; \theta') - 1 - \log p_{z_m} + \lambda = 0 \quad \forall z_m \\ \sum_{z_m} p_{z_m} - 1 = 0\end{aligned}$$

Then express p_{z_m} using λ: $p_{z_m} = \mathbb{P}(Z, Z_m = z_m; \theta')\, e^{\lambda - 1}$ and plug this into the sum to 1 constraint to solve for λ

$$\begin{aligned}e^{\lambda-1} \sum_{z_m} \mathbb{P}(Z, Z_m = z_m; \theta') = 1 \\ \lambda - 1 = \log \frac{1}{\sum_{z_m} \mathbb{P}(Z, Z_m = z_m; \theta')}\end{aligned}$$

Plug λ back into p_{z_m}, we have the solution as for all z_m,

$$\begin{aligned} p_{z_m} &= \frac{\mathbb{P}(Z, Z_m = z_m; \theta')}{\sum_{z_m} \mathbb{P}(Z, Z_m = z_m; \theta')} \\ &= \mathbb{P}(Z | Z_m = z_m, \theta') \end{aligned}$$

Ex 8.3

$f_{U_k}(u_k) = \mathbb{E}_{u_l, l \neq k}\left[f_{U_k|U_l, l \neq k}(u_k; u_l, l \neq k)\right]$ after we reach stationarity, we can view each t as one realization of $f_{U_k|U_l, l \neq k}(u_k; u_l, l \neq k)$. We just take sample analog of $\mathbb{E}_{u_l, l \neq k}\left[f_{U_k|U_l, l \neq k}(u_k; u_l, l \neq k)\right]$

Ex 8.4

Similarly to the discussion on page 264 around expression (8.7),

$$\begin{aligned} \hat{f}_{bag}(x) &= \frac{1}{B}\sum_{b=1} \hat{\mu}_b^*(x) \\ &= \frac{1}{B}\sum_{b=1} h(x)' (H'H)^{-1} H' y_b^* \\ &= h(x)' (H'H)^{-1} H' \left(\frac{1}{B}\sum_{b=1} y_b^*\right) \end{aligned}$$

$\left(\frac{1}{B}\sum_{b=1} y_b^*\right) \to H(H'H)^{-1} H'y = H\hat{\beta}$ a.s. as $B \to \infty$. Thus,

$$\begin{aligned} \hat{f}_{bag}(x) &\to h(x)' (H'H)^{-1} H'H\hat{\beta} \\ &= h(x)' \hat{\beta} = \hat{\mu}(x) \end{aligned}$$

Ex 8.5

For misclassification loss nothing needs to be modified. For squared, support vector, exponential and binomial deviance loss, we could simply extend loss function by using similar coding of (10.55) in the book and in Ex 10.5.

Ex 8.6

Ex 8.7

Let$g(\theta', \theta) := Q(\theta', \theta) + \log \Pr(Z|\theta) - Q(\theta, \theta)$ and $f(\theta') := \log \Pr(Z|\theta')$. It is obvious that $g(\theta', \theta') = \log \Pr(Z|\theta') = f(\theta')$. Also, from (8.47)

$$\begin{aligned} g(\theta', \theta) - f(\theta') &= Q(\theta', \theta) - Q(\theta, \theta) - (\log \Pr(Z|\theta') - \log \Pr(Z|\theta)) \\ &= R(\theta', \theta) - R(\theta, \theta) \leq 0 \end{aligned}$$

Therefore g is a minorization of $f(\theta')$.

9 General Additive Models

9.1 Gini index

9.2 Loss matrix

9.3 Missing data

The text pointed out that measures 1-3 to deal with missing data on page 333 rely on MCAR.

9.4 Computation Complexity

For GAM, we have to do initial sorting first since Cholesky decomposition of $M = (B'B + \lambda\Omega_B)$ can be done efficiently if we sort the observations by predictor values, based on the Section 5.8 in this note. This is the initial sorting of $pN \log N$. Note that each spline fit takes $O(N)$, therefore we have mpN since we need to do it for mp times.

For MARS, by the discussion on page 325, when considering a particular term in $\mathcal{M}$ interacting one predictor, it takes $O(N)$ to decide the best reflection point. Therefore for each step we need $O(pmN)$ where m is the number of terms already in $\mathcal{M}$, and p is the number of predictors. $O\left(Nm^2\right)$ corresponding to the fit of existing m terms in $\mathcal{M}$ when adding a $(m+1)^{th}$ term.

9.5 Exericise

Ex 9.1

For smoothing spline, the smoother is $S_\lambda = X(X'X + \lambda\Omega_N)^{-1}X'$ for original X or some basis X. Note that we have $S_\lambda \hat{y}_{OLS} = S_\lambda X\hat{\beta}_{OLS} = S_\lambda X(X'X)^{-1}X'y = S_\lambda y = \hat{y}_{smooth}$

For local linear regression, from equation (6.8) on page 195, we have $\hat{y}_i^{loc} = b_i'\left(B'W(x_i)B\right)^{-1}B'W(x_i)\,y = s_i'y$, where b_i' is the i^{th} row of matrix B and s_i' is the i^{th} row of the smoothing matrix S_{loc} for local linear regression. Then $s_i'\hat{y}_{OLS} = b_i'\left(B'W(x_i)B\right)^{-1}B'W(x_i)\,B(B'B)^{-1}B'y = b_i'\hat{\beta}_{OLS} = \hat{y}_i^{OLS}$ for each i. Therefore $S_{loc}\hat{y}_{OLS} = \hat{y}_{OLS}$

To show that smoothing spline fit to mean zero response has mean zero, note that

Ex 9.5

(a)

Assume that $X_1, ...X_p$ are non-random and the only randomness is ϵ. Also assume that the m terminal node, i.e., m regions are fixed a priori instead of fitted from the data. Let $I(i)$ denote the set of data points

lying in the region/node containing point i. Note that trivially, $i \in I(i)$. We have

$$\begin{aligned}\sum_{i=1}^{n} Cov\left(y_i, \hat{y}_i\right) &= \sum_{i=1}^{n} Cov\left(y_i, \frac{1}{|I(i)|} y_i\right) \\ &= \sigma^2 \sum_{i=1}^{n} \frac{1}{|I(i)|} \\ &= \sigma^2 \sum_{j=1}^{m} \frac{1}{|I(i)|} |I(i)| = m\sigma^2\end{aligned}$$

Therefore by the simplifing assumption we have the effective degree of freedom as m. Recall that in linear models when we select the best p predictors from a much larger set, the effective degree of freedom will be bigger than p. Similarly here, when the m regions are not prefixed and fitted by the data, the true degree of freedom can be much larger than m.

(e)

The sum of the diagonal matrix S is $\sum_{i=1}^{n} \frac{1}{|I(i)|} = m$, the same conclusion as in (a)

Ex 9.6

10 Boosting and Additive Trees

10.1 Gradient Boosting as Forward Stagewise Additive Modelling with Exponential Loss

Two notes on this link:

First, for algorithms like decision tree used as base classifier in boosting, we usually are not minimizing expression (10.11) directly; instead, we are minimizing a loss different than 0-1 loss or classification error. So as pointed out by the second paragraph from bottom on page 344, step 2(a) in Adaboost.M1 is at best an approximation for minimizing expression (10.11)

Secondly, According to Step 3 in Algorithm 10.1 on page 339, weights to G_m is α_m; on page 344, however, expression (10.12) gives the weights of each basis as $\beta_m = \frac{1}{2}\alpha_m$. Dispite this discrepency, we have that the two decision rules/ensemble classifiers are actually equivalent since $sign\left(\alpha_m G_m\right) = sign\left(\beta_m G_m\right)$

10.2 Deviance

Binomial Deviance loss here is just negative log-likelihood

If we model the probability as $p\left(x\right) = \frac{1}{1+e^{-2f(x)}}$ and denote the true probability as $\mathbb{P}[Y = 1|x]$. To see in

population, minimizing negative log-likelihood or deviance gives the same solution as exponential loss,

$$\begin{aligned}&\min_{f(x)} \mathbb{E}_{Y|x}\left[Y' \log\left(p\left(x\right)\right) + \left(1 - Y'\right) \log\left(1 - p\left(x\right)\right)\right] \\ =&\min_{f(x)} \mathbb{P}[Y = 1|x] \log\left(p\left(x\right)\right) + \left(1 - \mathbb{P}[Y = 1|x]\right) \log\left(1 - p\left(x\right)\right)\end{aligned}$$

Note that $\frac{\partial p(x)}{\partial f(x)} = 2p(x)\left(1 - p(x)\right)$. FOC gives,

$$\mathbb{P}[Y = 1|x] 2\left(1 - p\left(x\right)\right) - 2(1 - \mathbb{P}[Y = 1|x])p\left(x\right) = 0$$

,which gives $\mathbb{P}[Y = 1|x] = p^*\left(x\right) = \frac{1}{1+e^{-2f^*(x)}}$, i.e., $f^*\left(x\right) = \frac{1}{2}\log\left(\frac{\mathbb{P}[Y=1|x]}{\mathbb{P}[Y=-1|x]}\right)$

Although deviance and exponential loss yields the same solution in population, we can see that due to the log term (say in expression 10.18), deviance is much more insensitive to outliers.

10.3 Gradient Boosting

10.3.1 Loss function

For table 10.2, deviance loss function's derivatives w.r.t. $z_{ik} = f_k\left(x_i\right)$. Think of z_{ik} as the input of the last softmax layer in a $K - class$ classifier, where $p_k\left(z_i\right) = \frac{e^{z_{ik}}}{\sum_l e^{z_{il}}}$. Note that $\frac{\partial p_k(z_i)}{\partial z_{ik}} = p_k\left(z_i\right)\left(1 - p_k\left(z_i\right)\right)$ and for $l \neq k$, $\frac{\partial p_k(z_i)}{\partial z_{il}} = -p_k\left(z_i\right) p_l\left(z_i\right)$. And deviance, or cross-entropy loss, or negative of log-likelihood function is $L\left(y_i, z_i\right) := -\sum_{k \in K} 1_{y_i=k} \log p_k\left(z_i\right)$, which is (10.22) in the book. Then the negative gradient of the loss function, which is the gradient of the log-likelihood function evaluated at point i would be:

$$\begin{aligned}-\frac{\partial L\left(y_i, z_i\right)}{\partial z_{ik}} &= \sum_{l \in K} 1_{y_i=l} \frac{1}{p_l\left(z_i\right)} \frac{\partial p_l\left(z_i\right)}{\partial z_{ik}} \\ &= \sum_{l \neq k} 1_{y_i=l} \frac{1}{p_l\left(z_i\right)} - p_k\left(z_i\right) p_l\left(z_i\right) + 1_{y_i=k} \frac{1}{p_k\left(z_i\right)} p_k\left(z_i\right)\left(1 - p_k\left(z_i\right)\right) \\ &= -p_k\left(z_i\right) + 1_{y_i=k}\end{aligned}$$

,which is the same as in the table 10.2

For two class case, this reduces to $k = 0, 1$ but we only need z_{i1} for each i where z_{i0} is normalized to 0, i.e., $\mathbb{P}\left(i^{th} \text{ point is class } 1\right) = p_1\left(z_{i1}\right) = \frac{1}{1+e^{-z_{i1}}}$. Therefore

$$-\frac{\partial L\left(y_i, z_{i1}\right)}{\partial z_{i1}} = -p_1\left(z_{i1}\right) + 1_{y_i=1}$$

10.3.2 Analogy to steepest descent

Gradient Boosting is essentially taking expression (10.35) and use negative gradient as the target to train a new tree, which are step 2(a) and 2(b) in Algorithm 10.3. This is a trick to get around the problem of

solving Expression (10.29). Usually if we want to minimize a function, after calculating gradient at a point $f'(x_0)$ we will do a line search, like in expression (10.36), on ρ in $f(x - \rho f'(x_0))$ to determine the best step size. As pointed out in the text, step 3(c) in Algorithm 10.3 is like one step combining fitting the terminal nodes values and determining the step size for that terminal node.

One alternative to Algorithm 10.3 is that we could replace step 3(c) by just using γ_{jm} from the regression tree fit to the pseudo residuals r_{im} (say like $\frac{1}{N_{jm}} \sum_{i \in R_{jm}} r_{im}$ in the squared-loss case), and then using a pre-determined learning rate ν in the update step 4(d), i.e., $f_m(x) = f_{m-1}(x) + \nu \sum_{j=1}^{J_m} \gamma_{jm} I(x \in R_{jm})$. This learning rate ν is an hyperparameter just like the learning rate in gradient descent, and can be optimized or picked via validation. I believe xgboost uses a variant of this procedure. This is also mentioned in Section 10.12.1 of the book.

10.4 Difference between Adaboost vs. Gradient Boosting Tree

10.5 Partial dependency plots

On the analysis in the text between (10.48) and (10.51)
10.47-10.48 is that for a particular value of x_S of X_S, we use the same distribution, the marginal distribution of X_C to average $f(X)$. So for example in Fig 10.15, as the x-variable moves around, the distribution of all other variables used to get the average f does NOT vary as x changes. However, for the conditional distribution alternative (10.49), as x changes, the distribution $X_C|X_S$ also changes accordingly. The question is which one makes sense?

10.5.1 Expression (10.52)

Seems like this only holds when we set $\sum_{k=1}^{K} f_k = 0$

10.6 Exercise

Ex 10.1

We need to minimize over β only

$$\begin{aligned} &\min_{\beta} \sum_{i=1}^{N} w_i^{(m)} e^{-\beta y_i G_m(x_i)} \\ =&\min_{\beta} \sum_{i=1}^{N} w_i^{(m)} e^{-\beta} I(y_i = G_m(x_i)) + w_i^{(m)} e^{\beta} I(y_i \neq G_m(x_i)) \end{aligned}$$

FOC gives

$$\begin{aligned}
0 &= -\sum_{i=1}^{N} w_i^{(m)} e^{-\beta} I\left(y_i = G_m\left(x_i\right)\right) + \sum_{i=1}^{N} w_i^{(m)} e^{\beta} I\left(y_i \neq G_m\left(x_i\right)\right) \\
e^{2\beta} &= \frac{\sum_{i=1}^{N} w_i^{(m)} - \sum_{i=1}^{N} w_i^{(m)} I\left(y_i \neq G_m\left(x_i\right)\right)}{\sum_{i=1}^{N} w_i^{(m)} I\left(y_i \neq G_m\left(x_i\right)\right)} \\
&= \frac{1 - err_m}{err_m} \\
\beta &= \frac{1}{2} \log\left(\frac{1 - err_m}{err_m}\right)
\end{aligned}$$

Ex 10.2

By conditioning on x, essentially we treat it as a constant and try to find the best $f\left(x\right)$.

$$\min_{f(x)} \mathbb{E}_{Y|x}\left[\mathbb{P}\left[Y=1|x\right] e^{-f(x)} + \mathbb{P}\left[Y=-1|x\right] e^{f(x)}\right] = \min_{f(x)}\left[\mathbb{P}\left[Y=1|x\right] e^{-f(x)} + \mathbb{P}\left[Y=-1|x\right] e^{f(x)}\right]$$

FOC gives,

$$f\left(x\right) = \frac{1}{2}\log\left(\frac{\mathbb{P}\left[Y=1|x\right]}{\mathbb{P}\left[Y=0|x\right]}\right)$$

Ex 10.3

Ex 10.4

Ex 10.5

(a)

Let p_k be the true class probabilities of class $k = 1, 2, ...K$. The problem becomes,

$$\begin{aligned}
\min\nolimits_f \quad & \sum_{k=1}^{K} p_k e^{-\frac{1}{K} Y' f} \\
s.t. \quad & 1' f = 0
\end{aligned}$$

$$\begin{aligned}
\min\nolimits_f \quad & \sum_{k=1}^{K} p_k e^{-\frac{1}{K}\left(f_k - \frac{1}{K-1}\sum_{j \neq k} f_j\right)} \\
s.t. \quad & 1' f = 0
\end{aligned}$$

$$\begin{aligned} \min_f \quad & \sum_{k=1}^{K} p_k e^{-\frac{1}{K-1} f_k} \\ s.t. \quad & 1'f = 0 \end{aligned}$$

Then we apply LM,

$$\min_f \quad \textstyle\sum_{k=1}^{K} p_k e^{-\frac{1}{K-1} f_k} + \lambda 1'f$$

Solving FOC gives

$$\begin{aligned} \lambda^* &= \frac{\left(\prod_{k=1}^{K} p_k\right)^{\frac{1}{K}}}{K-1} \\ f_k^* &= (K-1)\ln \frac{p_k}{\left(\prod_{k=1}^{K} p_k\right)^{\frac{1}{K}}} \end{aligned}$$

(b)

Assume $f_m(x_i) = f_{m-1}(x_i) + \beta G(x_i)$ and f, G has the same coding as in 10.55. Note that

$$\frac{1}{K} Y_i' G(x_i) = \begin{cases} \frac{1}{K-1} & \text{if } Y_i = G(x_i) \\ -\frac{1}{(K-1)^2} & o.w. \end{cases}$$

At each iteration step we try to optimize, subpressing $f_{m-1}(x_i)$ term into $w_i's$

$$\min_{\beta, G} \quad \sum_{i=1}^{N} w_i e^{-\beta \frac{1}{K} Y_i' G(x_i)}$$

First fix β, optimize over G, note that,

$$\begin{aligned} \sum_{i=1}^{N} w_i e^{-\beta \frac{1}{K} Y_i' G(x_i)} &= \sum_{Y_i = G(x_i)} w_i e^{-\frac{\beta}{K-1}} + \sum_{Y_i \neq G(x_i)} w_i e^{\frac{\beta}{(K-1)^2}} \\ &= \sum_{i} w_i e^{-\frac{\beta}{K-1}} + \left(e^{\frac{\beta}{(K-1)^2}} - e^{-\frac{\beta}{K-1}}\right) \sum_{Y_i \neq G(x_i)} w_i \end{aligned}$$

So we pick $G = G_m$ to minimize $\sum_{Y_i \neq G(x_i)} w_i$. Then we only need to optimize over β by plugging to G_m:

$$\min_{\beta} \sum_{i=1}^{N} w_i e^{-\beta \frac{1}{K} Y_i' G_m(x_i)}$$

FOC gives,

$$\begin{aligned}
\sum_i \frac{1}{K} Y_i' G(x_i)\, w_i e^{-\beta \frac{1}{K} Y_i' G(x_i)} &= 0 \\
\sum_{Y_i = G(x_i)} \frac{1}{K-1} w_i e^{-\frac{\beta}{K-1}} - \sum_{Y_i \neq G(x_i)} \frac{1}{(K-1)^2} w_i e^{\frac{\beta}{(K-1)^2}} &= 0 \\
\ln \frac{(K-1)(1-err_m)}{err_m} &= \frac{K}{(K-1)^2}\beta
\end{aligned}$$

, where err_m is defined the same way as in (10.13). Then we have $\beta_m = \frac{(K-1)^2}{K} \ln \frac{(K-1)(1-err_m)}{err_m}$. The weight update for i is then,

$$w_i^{(m+1)} = w_i^{(m)} e^{-\beta_m \frac{1}{K} Y_i' G_m(x_i)}$$

Ex 10.6

Denote the numbers in contingency table on the bottom of page 385 as a matrix C. We want to test the relationship between GAM and GBM test error. Thus the null is $c_{21} + c_{22} = c_{12} + c_{22}$, i.e. $H_0 : c_{21} = c_{12}$. Since we are interested in two-sided test here, the alternative hypotheis would then be $H_1 : c_{21} \neq c_{12}$. Next we derive an exact test to test H_0.

Under H_0, the counts in c_{12} or c_{21} should have equal probability. Conditioning on the total number $c_{12} + c_{21}$ (in our case 51), $c_{12} \sim Binomial\,(0.5, c_{12} + c_{21})$. Thus the two-sided exact p-value should be

$$2 \sum_{i=33}^{51} \binom{51}{i} \left(\frac{1}{2}\right)^{51} = 0.0489$$

The number is different from given in the exercise but gives the same conclusion: reject the null.

Next we use another test where $\frac{c_{21} - c_{12}}{\sqrt{c_{21} + c_{12}}}$ is approximately $\mathcal{N}(0,1)$ in large samples, the two-sided p-value from this test is exactly 0.036. This is the Mcnerma Test.

Ex 10.7

Very similar to Ex 10.2, we have

$$\begin{aligned}
\min_{\gamma_{jm}} \sum_{x_i \in R_{jm}} L\left(y_i, f_{m-1}(x_i) + \gamma_{jm}\right) &= \min_{\gamma_{jm}} \sum_{x_i \in R_{jm}} w_i^{(m)} e^{-y_i \gamma_{jm}} \\
&= \min_{\gamma_{jm}} \sum_{x_i \in R_{jm}, y_i = 1} w_i^{(m)} e^{-\gamma_{jm}} + \sum_{x_i \in R_{jm}, y_i = -1} w_i^{(m)} e^{\gamma_{jm}}
\end{aligned}$$

FOC w.r.t. γ_{jm} gives,same test error

$$-\sum_{x_i \in R_{jm}, y_i=1} w_i^{(m)} e^{-\gamma_{jm}} + \sum_{x_i \in R_{jm}, y_i=-1} w_i^{(m)} e^{\gamma_{jm}} = 0$$

$$\frac{1}{2} \log \frac{\sum_{x_i \in R_{jm}, y_i=1} w_i^{(m)}}{\sum_{x_i \in R_{jm}, y_i=-1} w_i^{(m)}} = \gamma_{jm}$$

Ex 10.8

Ex 10.9

Ex 10.10

Assume we have two trees fitted,

$$p_1(x) = \frac{e^{f_1(x)}}{e^{f_1(x)} + e^{f_2(x)}}$$

Following the convention, we add the constraint $f_1(x) + f_2(x) = 0$, then we essentially have,

$$p_1(x) = \frac{e^{f_1(x)}}{e^{f_1(x)} + e^{-f_1(x)}} = \frac{1}{1 + e^{-2f_1(x)}}$$

That means only f_1 is needed and it determines $p_1(x)$, and therefore $p_2(x)$ too

11 Neuron Networks

11.1 Exercise

Ex 11.2

Taking derivatives w.r.t x_i,

$$\begin{aligned}
\frac{\partial R_i}{\partial x_l} &= -\sum_{k=1}^{K} 2\left(y_{ik} - (\beta_{0k} + \beta_k' z_i)\right) \sum_{m=1}^{M} \beta_{km} \frac{\partial z_{mi}}{\partial x_l} \\
&= -\sum_{k=1}^{K} 2\left(y_{ik} - (\beta_{0k} + \beta_k' z_i)\right) \sum_{m=1}^{M} \beta_{km} \sigma'\left(\alpha_{0m} + \alpha_m' x_i\right) \alpha_{ml} \\
&\approx -2 \sum_{k=1}^{K} \sum_{m=1}^{M} \beta_{km} \sigma'(0) \alpha_{ml} \left(y_{ik} - \left(\beta_{0k} + \sum_{m=1}^{M} \beta_{km} \sigma\left(\alpha_{0m} + \alpha_m' x_i\right) \right) \right)
\end{aligned}$$

$$
\begin{aligned}
f_k(x_i) &= \beta_{0k} + \sum_{m=1}^{M} \beta_{km}\sigma(\alpha_{0m} + \alpha'_m x_i) \\
&\approx \beta_{0k} + \sum_{m=1}^{M} \beta_{km}[\sigma(0) + \sigma'(0)(\alpha_{0m} + \alpha'_m x_i)]
\end{aligned}
$$

, which is linear in vector x_i
Q.E.D.

12 Support Vector Machines

12.1 Population Minimizer

In Table 12.1, population minimizer of SVM hinge loss is $sign\left(\mathbb{P}[Y=+1|x]-\frac{1}{2}\right)$. To see this, we do minimization pointwise for any value of x,

$$
\begin{aligned}
&\min_{f(x)} \mathbb{E}[[1-yf(x)]_+|x] \\
=&\min_{f(x)} \mathbb{P}[Y=+1|x][1-f(x)]_+ + (1-\mathbb{P}[Y=+1|x])[1+f(x)]_+
\end{aligned}
$$

The last line is a piecewise linear function of $f(x)$ and we can easily see that the minimal is obtained at either 1 or -1 depending on the value of $\mathbb{P}[Y=+1|x]$. Precisely, we have $f(x) = sign\left(\mathbb{P}[Y=+1|x]-\frac{1}{2}\right)$. From this we could see that SVM cannot give us a probability estimate of a point belonging to a certain class. However, as in the discussion around (12.31) in the text, we can swap the SVM hinge loss to binomial deviance and still use the kernel tricks to get an estimate of the probability.

12.2 Exercise

Ex 12.1

We start from problem (12.25), this problem is clearly convex in β and β_0. We introduce auxlliary variables $\xi'_i s$ and let $\xi_i = \max(0, 1-y_i f(x))$ by adding inequality constraints:

$$
\begin{aligned}
&\min C\sum \xi_i + \frac{1}{2}||\beta||^2 \\
s.t. &\xi_i \geq 0 \\
&\xi_i \geq 1 - y_i f(x)
\end{aligned}
$$

Ex 12.2

Starting from (12.25), $f(x) = h(x)' \beta + \beta_0$. Compared with Ex 5.15, we can let $h(x)$ be the basis in $\mathcal{H}_K$ which can have infinitely many components, i.e. $h_i(x) = \sqrt{\gamma_i}\phi_i(x)$. Note that $< h(x_1), h(x_2) >$in usual Euclidian space can be written as $\sum_i \sqrt{\gamma_i}\phi_i(x_1)\sqrt{\gamma_i}\phi_i(x_2) = K(x_1, x_2)$.

Then $||\beta||^2$ will be $||f||_{\mathcal{H}_K}$.Therefore, (12.25) will be the same problem with (12.27), (5.48) or (5.49), which have finite solution in the form of (12.28) even if $h(x)$ is infinite-dimensional. Then (12.25) will become equivalent to (12.29) with f as specified by (12.28). Note that we can view the constant term as a basis that is not penalized or not part of the basis at all, just a parellel shift added to the function in $\mathcal{H}_K$.

Alternatively, we can write f in (12.28) using basis of $h(x)$ in (12.25), where the j^{th} coefficients would be $\beta_j = \sum_{i=1}^N \alpha_i \sqrt{\gamma_j}\phi_j(x_i)$. Then we can see that $||\beta||^2 = \alpha' K \alpha$. The difference is that (12.29) is finite-dimensional while (12.25) can be infinite. The kernel property that the potentially infinite problem reduces to finite dimensional one is explained in Section 5.8 in the text and Ex 5.15.

Ex 12.10

Focusing on the equation (4.9) in ESL with 2 classes and in the transformed high-dimension space, first note that the term $\log \frac{\pi_1}{\pi_2}$ only requires knowledge of the labels.

We first show that the term$-\frac{1}{2}(\hat{\mu}_1 + \hat{\mu}_2)' \left(\hat{W}_h + \lambda I\right)^{-1} (\hat{\mu}_1 - \hat{\mu}_2)$ can be calculated only knowing the kernel function and not $h(x)$. Here

$$\hat{W}_h = \frac{1}{N-2} \sum_{k=1}^{2} \sum_{i \in G_k} \left[h(x_i) - \frac{\sum_{i \in G_k} h(x_i)}{N_k}\right] \left[h(x_i) - \frac{\sum_{i \in G_k} h(x_i)}{N_k}\right]',$$

where G_k is the set of index for class k. And N_k is the number of points in class k. Note that $\hat{\mu}_k = \frac{\sum_{i \in G_k} h(x_i)}{N_k}$.

Assume that $h(x)$ has dimension H, and we have N points in total. Define a H by N matrix U, whose first N_1 columns are $h(x_i) - \frac{\sum_{i \in G_1} h(x_i)}{N_1}$ for $i \in G_1$ and the next N_2 columns are $h(x_i) - \frac{\sum_{i \in G_2} h(x_i)}{N_2}$ for $i \in G_2$. Therefore we can write $\hat{W}_h = UU'$; then by Woodbury matrix identity,

$$\begin{aligned}
\left(\hat{W}_h + \lambda I_H\right)^{-1} &= \left(U \frac{I_N}{N-2} U' + \lambda I_H\right)^{-1} \\
&= \frac{I_H}{\lambda} - \frac{I_H}{\lambda} U \left((N-2) I_N + U' \frac{I_H}{\lambda} U\right)^{-1} U' \frac{I_H}{\lambda} \\
&= \frac{I_H}{\lambda} - \frac{1}{\lambda^2} U \left((N-2) I_N + \frac{U'U}{\lambda}\right)^{-1} U'
\end{aligned}$$

Therefore,

$$(\hat{\mu}_1 + \hat{\mu}_2)' \left(\hat{W}_h + \lambda I\right)^{-1} (\hat{\mu}_1 - \hat{\mu}_2) = \frac{(\hat{\mu}_1 + \hat{\mu}_2)'(\hat{\mu}_1 - \hat{\mu}_2)}{\lambda} - \frac{1}{\lambda^2} (\hat{\mu}_1 + \hat{\mu}_2)' U \left((N-2) I_N + \frac{U'U}{\lambda}\right)^{-1} U' (\hat{\mu}_1 - \hat{\mu}_2)$$

Note that all of $U'U$, $(\hat{\mu}_1+\hat{\mu}_2)'(\hat{\mu}_1-\hat{\mu}_2)$, $(\hat{\mu}_1+\hat{\mu}_2)'U$ and $U'(\hat{\mu}_1-\hat{\mu}_2)$ can be calculated by only know the kernel function $K(x_1,x_2) := h(x_1)'h(x_2)$.

It remains to show that the term $h(x)'\left(\hat{W}_h+\lambda I_H\right)^{-1}(\hat{\mu}_1-\hat{\mu}_2)$ can be calculated only using kernel function. But this is very similar:

$$h(x)'\left(\hat{W}_h+\lambda I_H\right)^{-1}(\hat{\mu}_1-\hat{\mu}_2) = \frac{h(x)'(\hat{\mu}_1-\hat{\mu}_2)}{\lambda} - \frac{1}{\lambda^2}h(x)'U\left((N-2)I_N+\frac{U'U}{\lambda}\right)^{-1}U'(\hat{\mu}_1-\hat{\mu}_2)$$

The result follows immediately after observing that all of $U'U$, $h(x)'(\hat{\mu}_1-\hat{\mu}_2)$, $h(x)'U$ and $U'(\hat{\mu}_1-\hat{\mu}_2)$ can be calculated by using only the kernel function.

13 Prototypes and Nearest Neighbours

13.1 Page 479

In the example in section 13.4.1, 50 prototypes per class is used to make it comparable to 5-nearest-neighbors. To make sense out of it, recall the knn algorithm has the following effective degree of freedom, assuming position of predictor x is fixed,

$$\frac{\sum_i Cov(\hat{y}_i,y_i)}{\sigma^2} = \frac{\sum_i Cov\left(\frac{1}{k}y_i,y_i\right)}{\sigma^2} = \frac{n}{k}$$

Thus, when we have 250 observations per class, $n=500$. And 5-nearest-neighbor has effective dof $d_{5nn}=100$.

Next we look at prototype methods, in particular LVQ. Based on exercise 9.15, making the simplifying assumption that if there are m regions used to classify points and the regions are pre-determined instead of fitted to the data, we can see that the effective dof is m. For prototype methods, each prototype represents a "dominate" region in the predictor space, in which x is closest to that focal prototype. Thus assigning the class of the cloest prototype to any point is just like classifying based on m regions where m equals total number of prototypes. In our case, if we want to have effective dof $d_{LVQ}=100$ we need 100 prototypes, i.e., 50 per class.

13.2 Exercise

Ex 13.1

$$\begin{aligned}
&\frac{\sum_{b=1}^B \sigma^2 + \sum_{b=1}^B \sum_{b'\neq b} Cov(x_b,x_{b'})}{B^2} \\
=&\frac{B\sigma^2 + B(B-1)\rho\sigma^2}{B^2} \\
=&\rho\sigma^2 + \frac{1-\rho}{B}\sigma^2
\end{aligned}$$

Ex 13.2

Since N points are independent and uniformly distributed in the unit cube centered at the origin, the probability of one point falls into the sphere with radius r centered at the origin is $\frac{v_p r^p}{1} = v_p r^p$. Therefore,

$$\mathbb{P}\left[R > r\right] = \left(1 - v_p r^p\right)^N$$

To solve for the median, we set $\left(1 - v_p r^p\right)^N = \frac{1}{2}$ and solve for r

$$Median\left(R\right) = v_p^{-\frac{1}{p}} \left(1 - \left(\frac{1}{2}\right)^{\frac{1}{N}}\right)^{\frac{1}{p}}$$

Ex 13.3

$$\begin{aligned}
\left(K-1\right)\sum_{k=1}^{K} p_k\left(x\right)\left(1-p_k\left(x\right)\right) &= \left(K-1\right)\left(1-p_{k^*}\left(x\right)^2\right) - \left(K-1\right)\sum_{k\neq k^*} p_k\left(x\right)^2 \\
&\leq \left(K-1\right)\left(1-p_{k^*}\left(x\right)^2\right) - \left(\sum_{k\neq k^*} p_k\left(x\right)\right)^2 \\
&= \left(K-1\right)\left(1-p_{k^*}\left(x\right)^2\right) - \left(1-p_{k^*}\left(x\right)\right)^2 \\
&= \left(1-p_{k^*}\left(x\right)\right)\left(\left(K-1\right) + \left(K-1\right)p_{k^*}\left(x\right) - 1 + p_{k^*}\left(x\right)\right) \\
&= \left(1-p_{k^*}\left(x\right)\right)\left(2K-2-K+Kp_{k^*}\left(x\right)\right) \\
&= \left(1-p_{k^*}\left(x\right)\right)\left(2\left(K-1\right) - K\left(1-p_{k^*}\left(x\right)\right)\right) \\
&= \left(K-1\right)E^*\left(2 - \frac{K}{K-1}E^*\right)
\end{aligned}$$

14 Unsupervised Learning

15 Random Forests

15.1 Bias-Variance Tradeoff in RF

Right panel in Figure 15.10 is very interesting. Unlike Boosting, we do not have bias variance tradeoff in the number of iterations, the more RF trees, the less the variance and we stop at some point where OOB loss flattens out. However, as this plot shows, we do have a bias-variance tradeoff in m, the number of sampled predictors at node-spliting. Higher m means more powerful individual model, thus leading to lower bias random forest ensemble. Smaller m results in smaller Ensemble variance $Var\left(\frac{1}{B}\sum_{b=1}^{B} T\left(x;\Theta_b\left(Z\right)\right)\right)$ driven by stronger de-correlation/smaller $\rho\left(x\right)$, as shown in Figure 15.9.

15.2 Random Forests vs. Adaptive Nearest Neighbors

In section 15.4.3, the text explains that random-forests voting mechanism is like nearest neighbors. We give some explanations on the similarity with Adaptive Nearest Neighbors. Recall Figure 13.13 on page 476. Since the vertical axis variable denoted as x_2 do not at all contribute to classifcation of red vs. green class, Random Forest is unlikely to pick x_2 as splitting but spliting on x_1 instead. As a results, fixing a query point, training data points closed with the query point measured in coordinate x_1 will more likely to lie in the same terminal node with the query point. And as a results will be given more weights in the random forest voting scheme, compared with training data points with equal Euclidean distance to the query point but far away from it in terms of x_1 coordinate. In this sense, RF is like a weighted version of Adaptive Nearest Neighbors.

15.3 Exercise

Ex 15.1

$$\begin{aligned}
&\frac{\sum_{b=1}^{B}\sigma^2+\sum_{b=1}^{B}\sum_{b'\neq b}Cov\left(x_b,x_{b'}\right)}{B^2}\\
=&\frac{B\sigma^2+B\left(B-1\right)\rho\sigma^2}{B^2}\\
=&\rho\sigma^2+\frac{1-\rho}{B}\sigma^2
\end{aligned}$$

At a first glance, this will give us negative variance when $\rho<0$. However, the $B\times B$ covariance matrix in this case is

$$V=\sigma^2\begin{bmatrix}1 & \rho & ... & \rho\\ \rho & ... & \rho & ...\\ ... & \rho & 1 & \rho\\ \rho & ... & \rho & 1\end{bmatrix}$$

, which must be positive semi-definite. That implies

$$\begin{aligned}
[\frac{1}{B},\frac{1}{B},...,\frac{1}{B}]V[\frac{1}{B},\frac{1}{B},...,\frac{1}{B}]^T&\geq 0\\
\rho\sigma^2+\frac{1-\rho}{B}\sigma^2&\geq 0\\
\rho&\geq -\frac{1}{B-1},\text{ for large } B
\end{aligned}$$

Let $B\to\infty$, we have $\rho\geq 0$. This means that for covariance matrix with this structure $V\left(\rho,B\right)$, as B gets larger the possible values for ρ ensuring $V\geq 0$ becomes narrower. If we require $V\left(\rho,B\right)\geq 0$ for any $B\in\mathcal{N}$, one necessary condition is $\rho\geq 0$.

Ex 15.2

Notice that if we conditioning on Z and $T(x, \Theta(Z))$'s bootstrap sample Z^*does NOT contain x, the distribution of a tree $T(\Theta(Z))|Z, x \notin Z^*$ is equal to the distribution $T(\Theta(Z^{-x}))|Z^{-x}$ where $Z^{-x} = Z - \{x\}$, i.e.

$$T(\Theta(Z))|Z, x \notin Z^* \sim T\left(\Theta\left(Z^{-x}\right)\right)|Z^{-x}$$

Define B^{-x} as the cardinality of set $\{1 \leq b \leq B : x \notin T(\Theta_b(Z))\}$. Then since $\mathbb{P}\{x \notin T(\Theta_b(Z))\} = \left(1 - \frac{1}{N}\right)^N$, $\mathbb{E}[B^{-x}] = B\left(1 - \frac{1}{N}\right)^N$. Thus, if $B \to \infty$, $B^{-x} \to \infty$ a.s.. We have that as $B \to \infty$, the OOB estimates at point x given by

$$\begin{aligned}\hat{f}^B_{rf;OOB}(x) &= \frac{1}{B^{-x}} \sum_{b \in B^{-x}} T(x; \Theta_b(Z)) \\ &\to \mathbb{E}_{\Theta|Z, x \notin Z^*}[T(x; \Theta(Z))] \text{ a.s.} \\ &= \mathbb{E}_{\Theta|Z^{-x}}\left[T\left(x; \Theta\left(Z^{-x}\right)\right)\right]\end{aligned}$$

It is obvious that if we want to do leave-one-out cross validation, for evaluation at x, we need to train the model using Z^{-x}, then what we get is $\hat{f}^{B,-x}_{rf}(x) = \frac{1}{B}\sum_b T(x; \Theta_b(Z^{-x})) \to \mathbb{E}_{\Theta|Z^{-x}}[T(x; \Theta(Z^{-x}))]$ a.s.. Therefore, $\hat{f}^B_{rf;OOB}(x)$ and $\hat{f}^{B,-x}_{rf}(x)$ converges a.s. to the same random variable.

Ex 15.3

The data is generated as follows, if the average of J uniform [0,1] random variables is greater than 0.5, i.e. $\frac{\sum_{j=1}^J X_j}{J} > 0.5$, then $\mathbb{P}[Y = 1] = 1 - q \geq \frac{1}{2}$, o.w. $\mathbb{P}[Y = 1] = q$. Bayes error rate is defined as $\mathbb{E}\left[1 - \max_{y \in \{0,1\}} \mathbb{P}[Y = y|X]\right]$. Then, we have

$$\mathbb{E}\left[\max_{y \in \{0,1\}} \mathbb{P}[Y = y|X]\right] = \mathbb{P}\left[\frac{\sum_{j=1}^J X_j}{J} > 0.5\right]\mathbb{E}[1 - q] + \mathbb{P}\left[\frac{\sum_{j=1}^J X_j}{J} \leq 0.5\right]\mathbb{E}[1 - q] = 1 - q$$

Bayes error rate is therefore q.

Ex 15.4

Note that $\bar{x}^*_1, \bar{x}^*_2$ are identically distributed.

$$Corr(\bar{x}^*_1, \bar{x}^*_2) = \frac{Cov(\bar{x}^*_1, \bar{x}^*_2)}{Var(\bar{x}^*_1)}$$

Note that $Cov(\bar{x}^*_1, \bar{x}^*_2) = \frac{1}{n^2}\left(\sum_{i,j} Cov\left(x^*_{1,i}, x^*_{2,j}\right)\right) = \frac{1}{n^2}\left(\sum_{i,j} Cov\left(x^*_{1,i}, x^*_{2,j}\right)\right)$ where $\bar{x}^*_1 = \frac{\sum_{i=1}^n x^*_{1,i}}{n}$, i.e., $x^*_{1,i}$ is the 1st bootstrap sample's i^{th} point; $x^*_{2,j}$ is similarly defined. We condition on the training sample

Z in the following discussion. Let z_i denote the i^{th} point in the training set Z. First note that, for any i

$$\begin{aligned}\mathbb{E}\left[x_{1,i}^*\right] &= \mathbb{E}\left[\mathbb{E}\left[x_{1,i}^*|Z\right]\right] \\ &= \mathbb{E}\left[\frac{1}{n}\sum_{k=1}^{n} z_k\right] \\ &= \mu\end{aligned}$$

$$\begin{aligned}Cov\left(x_{1,i}^*, x_{2,j}^*\right) &= \mathbb{E}\left[\mathbb{E}\left[x_{1,i}^* x_{2,j}^*|Z\right]\right] - \mathbb{E}\left[x_{1,i}^*\right]\mathbb{E}\left[x_{2,j}^*\right] \\ &= \mathbb{E}\left[\frac{1}{n^2}\sum_{k,l} z_k z_l\right] - \mu^2 \\ &= \frac{1}{n^2}\left(n\left(\sigma^2+\mu^2\right)+\left(n^2-n\right)\mu^2\right) - \mu^2 \\ &= \frac{\sigma^2}{n} \text{ for any } i,j\end{aligned}$$

Thus $Cov\left(\bar{x}_1^*, \bar{x}_2^*\right) = \frac{\sigma^2}{n}$. Note that for any i, $Var\left(x_{1,i}^*\right) = \mathbb{E}\left[\mathbb{E}\left[x_{1,i}^{*2}|Z\right]\right] - \mu^2 = \mathbb{E}\left[\frac{1}{n}\sum z_i^2\right] - \mu^2 = \sigma^2$. For $i \neq j$, $\mathbb{E}\left[x_{1,i}^* x_{1,j}^*\right] = \mathbb{E}\left[\mathbb{E}\left[x_{1,i}^* x_{1,j}^*|Z\right]\right] = \mathbb{E}\left[\frac{1}{n^2}\sum_{k,l} z_k z_l\right] = \frac{\sigma^2}{n} + \mu^2$

$$\begin{aligned}Var\left(\bar{x}_1^*\right) &= \frac{1}{n^2}\sum_{i,j} Cov\left(x_{1,i}^*, x_{1,j}^*\right) \\ &= \frac{1}{n^2}\left(nVar\left(x_{1,i}^*\right) + \sum_{i\neq j} Cov\left(x_{1,i}^*, x_{1,j}^*\right)\right) \\ &= \frac{1}{n^2}\left(nVar\left(x_{1,i}^*\right) + \sum_{i\neq j} Cov\left(x_{1,i}^*, x_{1,j}^*\right)\right) \\ &= \frac{1}{n}\sigma^2 + \frac{1}{n^2}\left(n^2-n\right)\left(\frac{\sigma^2}{n} + \mu^2 - \mu^2\right) \\ &= \sigma^2\left(\frac{1}{n} + \frac{n-1}{n^2}\right) = \frac{2n-1}{n^2}\sigma^2\end{aligned}$$

Plug everything back

$$Corr\left(\bar{x}_1^*, \bar{x}_2^*\right) = \frac{\frac{\sigma^2}{n}}{\frac{2n-1}{n^2}\sigma^2} = \frac{n}{2n-1}$$

Then for bootstrap aggregation of sample mean, $\bar{x}_{bag} := \frac{1}{B}\sum_b \bar{x}_b^*$, the variance is given as below, following

Ex 15.1

$$\begin{aligned} Var\left(\bar{x}_{bag}\right) &= \left(\frac{n}{2n-1} + \frac{1-\frac{n}{2n-1}}{B}\right)\frac{2n-1}{n^2}\sigma^2 \\ &= \frac{\sigma^2}{n} + \frac{\frac{1}{n}-\frac{1}{n^2}}{B}\sigma^2 \to \frac{\sigma^2}{n} \end{aligned}$$

However, sample mean's variance is just $\frac{\sigma^2}{n}$; boostrap version $\bar{x}_{bag}$ does not reduce the variance of the sample mean statistic. To contrast this case, in expression (15.5)-(15.7), $\sigma^2(x) := Var\left(T\left(x;\Theta\left(Z\right)\right)\right)$ gets reduce to $\rho(x)\sigma^2(x)$; however, $T\left(x;\Theta\left(Y\right)\right)$ is not a linear function of data Y that the tree is grown on.

Ex 15.5

We clarify notations first. $\Theta\left(Z\right)$ denotes both the bootstrap sampling from a particular training set Z and the fitting of one random forest tree, which includes the sampling of m predictors from all p predictors at each node-splitting. Evaluation at point x of the b^{th} random forest tree is then denoted as $T\left(x;\Theta_b\left(Z\right)\right)$ and when talking about generic random forest tree we could leave out the subscript b and just use $T\left(x;\Theta\left(Z\right)\right)$. The evaluation of random forest ensemble at x is thus $\frac{1}{B}\sum_{b=1}^{B}T\left(x;\Theta_b\left(Z\right)\right)$. As discussed on page 599, the randomness involved here can be decomposed as Z, sampling of training data set and $\Theta|Z$, the bootstrap sampling and predictor sampling conditioning on a particular training set Z.

It is a bit unclear what we need to prove here. I think what we need to show here is

$$corr\left[T\left(x;\Theta_1\left(Z\right)\right),T\left(x;\Theta_2\left(Z\right)\right)\right] = \frac{Var_Z\left(\mathbb{E}_{\Theta|Z}\left[T\left(x;\Theta\left(Z\right)\right)\right]\right)}{Var_Z\left(\mathbb{E}_{\Theta|Z}\left[T\left(x;\Theta\left(Z\right)\right)\right]\right)+\mathbb{E}_Z\left(Var_{\Theta|Z}\left[T\left(x;\Theta\left(Z\right)\right)\right]\right)} > 0$$

By definition we have $\hat{f}_{rf}\left(x\right) = \mathbb{E}_{\Theta|Z}\left[T\left(x;\Theta\left(Z\right)\right)\right]$. So the expression $Var\left(\hat{f}_{rf}\left(x\right)\right) = Var\left(\mathbb{E}_{\Theta|Z}\left[T\left(x;\Theta\left(Z\right)\right)\right]\right) = Var_Z\left(\mathbb{E}_{\Theta|Z}\left[T\left(x;\Theta\left(Z\right)\right)\right]\right)$. It is then easy to see that the results are trivial if (15.5) holds with $\rho(x)$ and $\sigma(x)$ given by (15.6) and (15.7). We only need to prove (15.5) then.

Consider Random Forest Ensemble's variance $Var\left(\frac{1}{B}\sum_{b=1}^{B}T\left(x;\Theta_b\left(Z\right)\right)\right)$. Directly applying (15.1), we have $Var\left(\frac{1}{B}\sum_{b=1}^{B}T\left(x;\Theta_b\left(Z\right)\right)\right) = \rho\left(x\right)\sigma^2\left(x\right)+\frac{1-\rho(x)}{B}\sigma^2\left(x\right)$ where ρ and σ is defined by (15.6) and (15.7). We then use the total variance formula:

$$\begin{aligned} \rho\left(x\right)\sigma^2\left(x\right)+\frac{1-\rho\left(x\right)}{B}\sigma^2\left(x\right) &= Var\left(\frac{1}{B}\sum_{b=1}^{B}T\left(x;\Theta_b\left(Z\right)\right)\right) \\ &= Var_Z\left(\mathbb{E}_{\Theta|Z}\left[\frac{1}{B}\sum_{b=1}^{B}T\left(x;\Theta_b\left(Z\right)\right)\right]\right)+\mathbb{E}_Z\left(Var_{\Theta|Z}\left[\frac{1}{B}\sum_{b=1}^{B}T\left(x;\Theta_b\left(Z\right)\right)\right]\right) \\ &= Var_Z\left(\frac{1}{B}\sum_{b=1}^{B}\mathbb{E}_{\Theta|Z}\left[T\left(x;\Theta_b\left(Z\right)\right)\right]\right)+\mathbb{E}_Z\left(Var_{\Theta|Z}\left[\frac{1}{B}\sum_{b=1}^{B}T\left(x;\Theta_b\left(Z\right)\right)\right]\right) \\ &= Var_Z\left(\mathbb{E}_{\Theta|Z}\left[T\left(x;\Theta_b\left(Z\right)\right)\right]\right)+\mathbb{E}_Z\left(Var_{\Theta|Z}\left[\frac{1}{B}\sum_{b=1}^{B}T\left(x;\Theta_b\left(Z\right)\right)\right]\right) \end{aligned}$$

Focusing on term $Var_{\Theta|Z}\left[\frac{1}{B}\sum_{b=1}^{B}T\left(x;\Theta_b\left(Z\right)\right)\right]$, note that conditioning on Z, $T\left(x;\Theta_b\left(Z\right)\right)$'s should be uncorrelated for a pair (b_1,b_2) because bootstrap and feature sampling is i.i.d., as discussed on page 599. Then we can simplify

$$Var_{\Theta|Z}\left[\frac{1}{B}\sum_{b=1}^{B}T\left(x;\Theta_b\left(Z\right)\right)\right]=\frac{Var_{\Theta|Z}\left[T\left(x;\Theta\left(Z\right)\right)\right]}{B}$$

Therefore we have

$$\rho\left(x\right)\sigma^2\left(x\right)+\frac{1-\rho\left(x\right)}{B}\sigma^2\left(x\right)=Var_Z\left(\mathbb{E}_{\Theta|Z}\left[T\left(x;\Theta_b\left(Z\right)\right)\right]\right)+\frac{\mathbb{E}_Z\left(Var_{\Theta|Z}\left[T\left(x;\Theta\left(Z\right)\right)\right]\right)}{B}$$

We then take limit $B\to\infty$,

$$Var\left(\mathbb{E}_{\Theta|Z}\left[T\left(x;\Theta\left(Z\right)\right)\right]\right)=Var\left(\hat{f}_{rf}\left(x\right)\right)=\rho\left(x\right)\sigma^2\left(x\right)$$

16 Ensemble Learning

17 Undirected Graphical Models

18 High Dimensional Problems