Elements of Statistical Learning

Junrui Di

Contents

Chapter 2. Overview of Supervised Learning

0. Notations

- Use upper case letters X, Y, G for generic variables
  - Input variable X with jth component denoted as X_j
  - Quantitative output Y
  - Qualitative output G
- Observed values in lowercase
  - ith observation of X is x_i (a scalar or a vector)
- Matrices are represented in bold uppercase letters
  - A set of N input p-vectors x_i, $i = 1...N$ will be $\mathbf{X} \in \mathbb{R}^{N \times p}$
  - p-vector of input x_i for the ith observation v.s. the N-vector $\mathbf{x}_j$ for all the observations on variable X_j
  - All vectors are assumed to be column vectors, the ith row of $\mathbf{X}$ is x_i^T.

1. Types of variables

- Qualitative variables, factors, categorical or discrete variables → **Classification**
- Quantitative measurements → **Regression**
- Ordered qualitative variables

2. Two simple approaches to prediction: least squares and nearst neighbors

2.1 Linear models and least squares

- Linear model
- Input: $X^T = (X_1, X_2, ..., X_p)$, Outcome: Y
- Model: $\hat{Y} = \hat{\beta}_0 + \sum_{j=1}^{p} X_j \hat{\beta}_j$ or $\hat{Y} = X^T \hat{\beta}$
- Least Square
- To minimize $\text{RSS}(\beta) = \sum_{i=1}^{N} (y_i - x_i^T \beta)^2$ or $\text{RSS}(\beta) = (\mathbf{y} - \mathbf{X}\beta)^T (\mathbf{y} - \mathbf{X}\beta)$
- Differentiate w.r.t. β gives $\mathbf{X}^T(\mathbf{y} - \mathbf{X}\beta) = 0$
- Solves to $\hat{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$
- Fitted value at the ith input x_i is $\hat{y}_i = x_i^T \hat{\beta}$

2.2 Nearest neighbor methods

The k-nearest neighbor fit for $\hat{Y}$: $\hat{Y}(x) = \frac{1}{k} \sum_{x_i \in N_k(x)} y_i$, where $N_k(x)$ is the neighborhood of x as the k closest points x_i in the training set.

For k-nearest neighbor fit, the error on the training data should be approximately an increasing function of k, and 0 for $k = 1$. We cannot use sum of squared errors as training criterion for picking k.

There is only one parameter in the fit, which is k. But the effective number of parameters is N/k, because there would be N/k neighbors and we need that many means for each of the neighborhood.

2.3 From least square to nearst neighbors

Least square: smooth linear decision boundary and stable to fit, but heavily rely on assumption of linear decision boundary. **Low variance but high bias**.

knn: no strong assumption and can adapt to any situation, but unstable (depend on a handful of input points and their positions). **High variance but low bias**.

3 Statistical decision theory

[1.] Quantitative output framework:

- Output $Y \in \mathbb{R}$, and input$X \in \mathbb{R}^p$
- Joint distribution $\Pr(X, Y)$
- Goal: find a function $f(X)$ to predict Y.
- Loss $L(Y, f(X))$, e.g. a squared error loss $L(Y, f(X))$

The expected squared prediction value is

$$\begin{aligned} EPE(f) &= E(Y - f(X))^2 \\ &= E_X E_{Y|X}([Y - f(X)]^2|X) \quad \text{conditioning on X} \end{aligned}$$

which can be minimized by $f(x) = \text{argmin}_c E_{Y|X}([Y-c]^2|X)$, which can be solved by $f(x) = E(Y|X=x)$, also known as the regression function. **The best prediction of Y at any point $X = x$ is the conditional mean when best is measured by average squared error**.

knn mimics this framework, by $\hat{f}(x) = \text{Ave}(y_i|x_i \in N_k(x))$ with two approximations

- expectation is approximated by averaging over sample data;
- conditioning at a point is relaxed to conditioning on some region close to the target point

Least square also mimics this framework, with the assumption that the regression function $f(x)$ is approximately linear in its argument, i.e. $f(x) \approx x^T\beta$. Therefore, β can be solved by $\beta = [E(XX^T)]^{-1}E(XY)$. That is not to condition on X, rather we used our knowlege of the functional relationship to pool over values of X. Least square estimates replace $E(\cdot)$ by averaging over the training data.

[2.] Qualitative output framework:

- Suppose there are K classes in G.
- Loss function can be represented by a $K \times K$ matrix $\mathbf{L}$, where each position $L(k,l)$ is the loss for misclassifying G_k as G_l. Most commonly we can use the zero_one loss, that is the set the the loss as 1.

The expected prediction error is

$$\begin{aligned}\text{EPE} &= E[L(G, \hat{G}(X))] \\ &= E_X \sum_{k=1}^{K} L[G_k, \hat{G}(X)] Pr(G_k|X)\end{aligned}$$

which can be minimized by

$$\begin{aligned}\hat{G}(x) &= \text{argmin}_{g\in G} \sum_{k=1}^{K} L[G_k, g] Pr(G_k|X=x) \\ &= \text{argmin}_{g\in G}[1 - Pr(g|X=x)] \\ &= \max Pr(g|X=x)\end{aligned}$$

This is known as the *Bayes Classifier*, such that we classify to the most probably class, using the conditional distribution $Pr(G|X)$.

knn directly approximates this solution using majority vote in a nearest neighborhood, except that conditional probability at a point is relaxed to conditional probability within a neighborhood of a point and probability are approximated by training sample proportions.

4. Function approximation

Data $\{x_i, y_i\}$ are considered to be from a $p+1$ dimensional Euclidean space. The function $f(x)$ has domain equal to a p-dimensional subspace. Data and function are related via the model $y_i = f(x_i) + \epsilon_i$. The goal for learning is to find an approximation to $f(x)$ in $\mathbb{R}^p$ given the representation in the domain of data which is $\mathbb{R}^{p+1}$

The question is to find a set of parameters θ for the function $f_\theta(x)$ with following criterion

[1.] Least square

To minimize the $\text{RSS}(\theta) = \sum_{i=1}^{N}(y_i - f_\theta(x_i))^2$

[2.] Maximum likelihood estimation

If we have a random sample y_i, $i = 1...N$ from a density $Pr_\theta(y)$. The log-probability of the observed sample is $L(\theta) = \sum_i \log Pr_\theta(y_i)$. The most reasonable values for θ are those for which the probability of the observed sample is the largest.

5. Restricted estimators

Minimizing the RSS leads to many solution, because any function $\hat{f}$ that passing through the training points is a solution. Therefore, we need to add complexity restrictions, that is, for all input points x sufficiently close to each other in some metirc, $\hat{f}$ exhibits osme specical structure such as nearly constant, linear, or low-order polynomial behavior.

5.1 Roughness penalty and Baysian methods

$$\text{PRSS}(f;\lambda) = \text{RSS}(f) + \lambda J(f)$$

with penalty $J(\cdot)$. E.g. cubic smoothing splines penalizes on large values of second order derivative.

5.2 Kernal methods and local regression

Kernel methods control the nature of the local neighborhood, using a kernel function $K_\lambda(x_0, x)$, which put weights to points x in a region near x_0 (λ controls the width of the neighborhood).

A local regression estimate of $f(x_0)$ as $f_{\hat{\theta}}(x_0)$ where $\hat{\theta}$ minimizes $\text{RSS}(f_\theta, 0) = \sum_i K_\lambda(x_0, x_i)(y_i - f_\theta(x_i))^2$.

5.3 Basis functions and dictionary methods

$$f_\theta(x) = \sum_{m=1}^{M} \theta_m h_m(x)$$

6. Model selection and the Bias-variance tradeoff

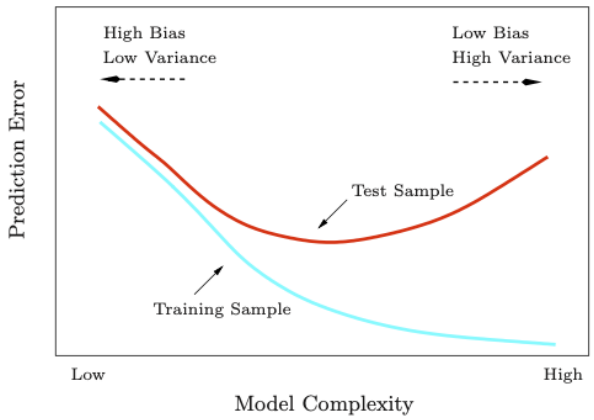

Figure 1: Model Complexity v.s. Prediction Errors

Data $\{x_i, y_i\}$, model $y = f(x) + \epsilon$, where $E(\epsilon) = 0$, $var(\epsilon) = \sigma^2$

$$E[(y - \hat{f}(x))^2] = (\text{Bias}[\hat{f}(x)])^2 + Var[\hat{f}(x)] + \sigma^2$$

* $\text{Bias}[\hat{f}(x)] = E[\hat{f}(x)] - f(x)$: error caused by simplifying assumptions build into the method

- $Var[\hat{f}(x)] = E[(E[\hat{f}(x)] - \hat{f}(x))^2]$: variance of the learning method
- irreducible error σ^2 due to the new test target.

Derivation

Derivation [edit]

The derivation of the bias–variance decomposition for squared error proceeds as follows.[9][10] For notational convenience, we abbreviate $f = f(x)$, $\hat{f} = \hat{f}(x; D)$ and we drop the D subscript on our expectation operator recall that, by definition, for any random variable X, we have

$$\operatorname{Var}[X] = \operatorname{E}[X^2] - \Big(\operatorname{E}[X]\Big)^2.$$

Rearranging, we get:

$$\operatorname{E}[X^2] = \operatorname{Var}[X] + \Big(\operatorname{E}[X]\Big)^2.$$

Since f is deterministic, i.e. independent of D,

$$\operatorname{E}[f] = f.$$

Thus, given $y = f + \varepsilon$ and $\operatorname{E}[\varepsilon] = 0$ (because ε is noise), implies $\operatorname{E}[y] = \operatorname{E}[f + \varepsilon] = \operatorname{E}[f] = f$.

Also, since $\operatorname{Var}[\varepsilon] = \sigma^2$,

$$\operatorname{Var}[y] = \operatorname{E}[(y - \operatorname{E}[y])^2] = \operatorname{E}[(y - f)^2] = \operatorname{E}[(f + \varepsilon - f)^2] = \operatorname{E}[\varepsilon^2] = \operatorname{Var}[\varepsilon] + \Big(\operatorname{E}[\varepsilon]\Big)^2 = \sigma^2 + 0^2 = \sigma^2.$$

Thus, since ε and $\hat{f}$ are independent, we can write

$$\begin{aligned}
\operatorname{E}\left[(y - \hat{f})^2\right] &= \operatorname{E}\left[(f + \varepsilon - \hat{f})^2\right] \\
&= \operatorname{E}\left[(f + \varepsilon - \hat{f} + \operatorname{E}[\hat{f}] - \operatorname{E}[\hat{f}])^2\right] \\
&= \operatorname{E}\left[(f - \operatorname{E}[\hat{f}])^2\right] + \operatorname{E}[\varepsilon^2] + \operatorname{E}\left[(\operatorname{E}[\hat{f}] - \hat{f})^2\right] + 2\operatorname{E}\left[(f - \operatorname{E}[\hat{f}])\varepsilon\right] + 2\operatorname{E}\left[\varepsilon(\operatorname{E}[\hat{f}] - \hat{f})\right] + 2\operatorname{E}\left[(\operatorname{E}[\hat{f}] - \hat{f})(f - \operatorname{E}[\hat{f}])\right] \\
&= (f - \operatorname{E}[\hat{f}])^2 + \operatorname{E}[\varepsilon^2] + \operatorname{E}\left[(\operatorname{E}[\hat{f}] - \hat{f})^2\right] + 2(f - \operatorname{E}[\hat{f}])\operatorname{E}[\varepsilon] + 2\operatorname{E}[\varepsilon]\operatorname{E}\left[\operatorname{E}[\hat{f}] - \hat{f}\right] + 2\operatorname{E}\left[\operatorname{E}[\hat{f}] - \hat{f}\right](f - \operatorname{E}[\hat{f}]) \\
&= (f - \operatorname{E}[\hat{f}])^2 + \operatorname{E}[\varepsilon^2] + \operatorname{E}\left[(\operatorname{E}[\hat{f}] - \hat{f})^2\right] \\
&= (f - \operatorname{E}[\hat{f}])^2 + \operatorname{Var}[\varepsilon] + \operatorname{Var}\left[\hat{f}\right] \\
&= \operatorname{Bias}[\hat{f}]^2 + \operatorname{Var}[\varepsilon] + \operatorname{Var}\left[\hat{f}\right] \\
&= \operatorname{Bias}[\hat{f}]^2 + \sigma^2 + \operatorname{Var}\left[\hat{f}\right].
\end{aligned}$$

Finally, MSE loss function (or negative log-likelihood) is obtained by taking the expectation value over $x \sim P$:

$$\operatorname{MSE} = \operatorname{E}_x\left\{\operatorname{Bias}_D[\hat{f}(x; D)]^2 + \operatorname{Var}_D\left[\hat{f}(x; D)\right]\right\} + \sigma^2.$$

Figure 2: Bias-Variance Tradeoff

Chapter 3: Linear Methods for Regression

1. Introduction

Linear regression assumes that the regression function $E(Y|X)$ is linear in the inputs $X_1, \ldots, X_p$.

2. Linear regression models and least square

[1.] *Linear regression from a least square point of view* (minimal assumption about the distribution)

- Form: $f(X) = \beta_0 + \sum_{j=1}^{p} X_j \beta_j$
- Data: $\{x_i, y_i\}$ $i = 1...N$, each $x_i = (x_{i1}...x_{ip})^T$ is a feature vector, with parameters $\beta = (\beta_0, \beta_1, ..., \beta_p)^T$
- Least square: To minimize $\text{RSS}(\beta) = \sum_{i=1}^{N}(y_i - f(x_i))^2 = \sum_{i=1}^{N}(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2$ or $\text{RSS}(\beta) = (\mathbf{y} - \mathbf{X}\beta)^T(\mathbf{y} - \mathbf{X}\beta)$ in matrix form. **LSE makes no assumptions about the validity of the model form**
- LSE: $\hat{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$
- Fitted value: $\hat{\mathbf{y}} = \mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y}$. $\mathbf{H} = \mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}$ is the projector of $\mathbf{y}$ onto the subspace spanned by column space of $\mathbf{X}$.
- Inference on parameters (assuming y_i's are uncorrelated and gave constant variance σ^2, and x_i are fixed)
  - $Var(\hat{\beta}) = (\mathbf{X}^T\mathbf{X})^{-1}\sigma^2$
  - $\hat{\sigma}^2 = \frac{1}{N-p-1}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$

[2.] *Linear regression with Gaussian error*

- Model Assumption: $Y = \beta_0 + \sum_{j=1}^{p} X_j\beta_j + \epsilon$, where $\epsilon \sim N(0, \sigma^2)$
- Distributional properties of model parameters
  - $\hat{\beta} \sim N(\beta, (\mathbf{X}^T\mathbf{X})^{-1}\sigma^2)$
  - $(N-p-1)\hat{\sigma}^2 \sim \sigma^2\chi^2_{N-p-1}$
  - $\hat{\beta}$ and $\hat{\sigma}^2$ are statistically independent.
- Inference on single parameter β_j

Under $H_o : \beta_j = 0$, $z_j = \frac{\hat{\beta}_k}{\hat{\sigma}\sqrt{(\mathbf{X}^T\mathbf{X})^{-1}_{ii}}} \sim t_{N-p-1}$, and β_j has a $1-2\alpha$ confidence interval of $(\hat{\beta}_j - z^{1-\alpha}\hat{\sigma}\sqrt{(\mathbf{X}^T\mathbf{X})^{-1}_{ii}}, \hat{\beta}_j + z^{1-\alpha}\hat{\sigma}\sqrt{(\mathbf{X}^T\mathbf{X})^{-1}_{ii}})$

- Nested Model Comparison (test whether the added variables are necessary to the model)

$$F = \frac{(\text{RSS}_0 - \text{RSS}_1)/(p_1 - p_0)}{\text{RSS}_1/(N - p_1 - 1)} \sim F_{p_1-p_0, N-p_1-1}, \quad \text{where } \text{RSS}_1 \text{ is for the larger model}$$

2.1 The Gauss-Markow Theorem

Least square estimates of β have the smallest variance among all linear unbiased estimates.

The least square estimator to estimate parameters $\theta = \alpha^T\beta$ is $\hat{\theta} = \alpha^T(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$. It is an unbiased estimator, i.e. $E(\alpha^T\hat{\beta}) = \alpha^T\beta$. Gauss-Markow theorem states that $Var(\alpha\hat{\beta})$ has the smallest variance for any unbiased estimator.

We may want to trade a little bias for larger reduction in variance.

2.2 Regression by succesive orthogonolization

Algorithm 3.1 *Regression by Successive Orthogonalization.*

1. Initialize $\mathbf{z}_0 = \mathbf{x}_0 = \mathbf{1}$.
2. For $j = 1, 2, \ldots, p$

   Regress $\mathbf{x}_j$ on $\mathbf{z}_0, \mathbf{z}_1, \ldots, \mathbf{z}_{j-1}$ to produce coefficients $\hat{\gamma}_{\ell j} = \langle \mathbf{z}_\ell, \mathbf{x}_j \rangle / \langle \mathbf{z}_\ell, \mathbf{z}_\ell \rangle$, $\ell = 0, \ldots, j-1$ and residual vector $\mathbf{z}_j = \mathbf{x}_j - \sum_{k=0}^{j-1} \hat{\gamma}_{kj}\mathbf{z}_k$.
3. Regress $\mathbf{y}$ on the residual $\mathbf{z}_p$ to give the estimate $\hat{\beta}_p$.

Figure 3: Gram-Schmidt procedure for multiple regression

2.3 Multiple outcomes

Data: $Y_1 ... Y_K$, with the model $Y_k = \beta_{0k} + \sum_{j=1}^{p} X_j\beta_{jk} + \epsilon_k$, with the matrix form $\mathbf{Y} = \mathbf{XB} + \mathbf{E}$, where $\mathbf{Y}$ is $N \times K$, $\mathbf{X}$ is $N \times p + 1$, and $\mathbf{B}$ us $(p+1) \times K$.

$\text{RSS}(\mathbf{B}) = \sum_k \sum_i (y_{ik} - f_k(x_i))^2 = \text{tr}(\mathbf{Y} - \mathbf{XB})^T(\mathbf{Y} - \mathbf{XB})$ is the RSs with LSE $\hat{\mathbf{B}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y}$

3. Subset selection

a. Best subset selection

b. Forward and backward selection

4. Shrinkage methods

4.1 Ridge regression

- RSS

$$\hat{\beta}^{\text{ridge}} = \text{argmin}_\beta \{ \sum_{i=1}^{N} (y_i - \beta_0 \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \}$$

or in the matrix form

$$\text{RSS}(\lambda) = (\mathbf{y} - \mathbf{X}\beta)^T(\mathbf{y} - \mathbf{X}\beta) + \lambda\beta^T\beta$$

with the solution

$$\hat{\beta}^{\text{ridge}} = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y}$$

Even if $\mathbf{X}^T\mathbf{X}$ is not of full rank, $(\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})$ is still nonsingular.

- Degree of freedom $\text{df}(\lambda) = \text{tr}[\mathbf{X}(\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T] = \sum_{j=1}^{p} \frac{d_j^2}{d_j^2+\lambda}$
- Ridge solutions are not equivariant under scaling of the inputs, and one normally standardizes the inputs before solving for estimation.

4.2 Lasso

- RSS

$$\hat{\beta}^{\text{lasso}} = \text{argmin}_\beta\{\sum_{i=1}^{N}(y_i - \beta_0\sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda\sum_{j=1}^{p}|\beta_j|\}$$

* Shrinkage $s = t/\sum_j |\hat{\beta}_j|$ where $\hat{\beta}_j$ is the least square estimation.

4.3 Subset selection,ridge, and lasso

[1.] *Orthonormal input matrix* $\mathbf{X}$

Estimator	Formula
Best subset (size M)	$\hat{\beta}_j \cdot I(\lvert\hat{\beta}_j\rvert \geq \lvert\hat{\beta}_{(M)}\rvert)$
Ridge	$\hat{\beta}_j/(1+\lambda)$
Lasso	$\text{sign}(\hat{\beta}_j)(\lvert\hat{\beta}_j\rvert - \lambda)_+$

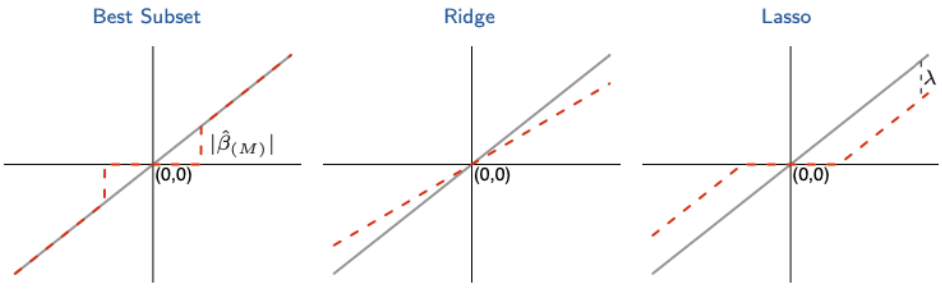

Figure 4: Gram-Schmidt procedure for multiple regression

- Ridge: proportional shrinkage
- LASSO: translate by a constant factor and truncating at zero, i.e soft thresholding
- Best subject: drops all the variables with coefficient smaller than the Mth largest, i.e. hard thresholding

[2.] *Nonorthogonal case*

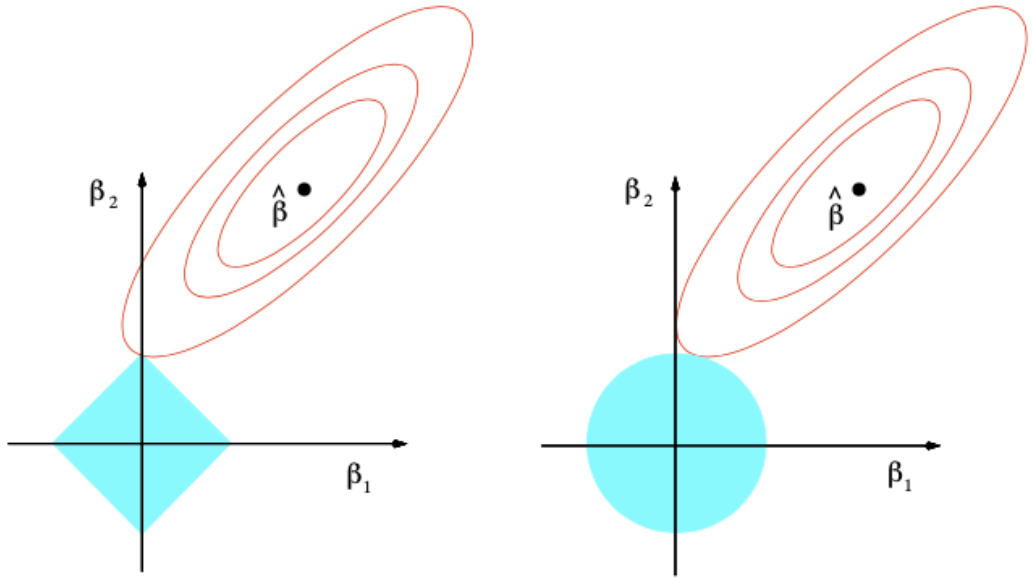

Figure 5: Gram-Schmidt procedure for multiple regression

Elastic net

$$\lambda \sum_{j=1}^{p} (\alpha \beta_j^2 + (1-\alpha)|\beta_j|)$$

4.4 Least angle regression

5. Methods using derived input directions

5.1 Principal components regression

PC regression forms the derived input columns $\mathbf{z}_m = \mathbf{X}v_m$ and then regresses $\mathbf{y}$ on $\mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_M$. Since they are orthogonal, each parameter is simply $\hat{\theta}_n = \frac{<\mathbf{z}_m, \mathbf{y}>}{<\mathbf{z}_m, \mathbf{z}_m>}$. It can be converted back to $\hat{\beta}_M^{pcr} = \sum_{m=1}^{M} \hat{\theta}_m v_m$.

The mth principal component direction v_m solveS:

$$\begin{aligned} &\max_\alpha \mathrm{Var}(\mathbf{X}\alpha) \\ &\text{subject to } ||\alpha| = 1, \alpha^T \mathbf{S} v_l = 0, \quad l = 1, ...n-1 \end{aligned}$$

where $\mathbf{S}$ is the sample covariance

Algorithm 3.3 *Partial Least Squares.*

1. Standardize each $\mathbf{x}_j$ to have mean zero and variance one. Set $\hat{\mathbf{y}}^{(0)} = \bar{y}\mathbf{1}$, and $\mathbf{x}_j^{(0)} = \mathbf{x}_j,\ j = 1, \ldots, p$.
2. For $m = 1, 2, \ldots, p$
   (a) $\mathbf{z}_m = \sum_{j=1}^{p} \hat{\varphi}_{mj}\mathbf{x}_j^{(m-1)}$, where $\hat{\varphi}_{mj} = \langle \mathbf{x}_j^{(m-1)}, \mathbf{y} \rangle$.
   (b) $\hat{\theta}_m = \langle \mathbf{z}_m, \mathbf{y} \rangle / \langle \mathbf{z}_m, \mathbf{z}_m \rangle$.
   (c) $\hat{\mathbf{y}}^{(m)} = \hat{\mathbf{y}}^{(m-1)} + \hat{\theta}_m \mathbf{z}_m$.
   (d) Orthogonalize each $\mathbf{x}_j^{(m-1)}$ with respect to $\mathbf{z}_m$: $\mathbf{x}_j^{(m)} = \mathbf{x}_j^{(m-1)} - [\langle \mathbf{z}_m, \mathbf{x}_j^{(m-1)} \rangle / \langle \mathbf{z}_m, \mathbf{z}_m \rangle]\mathbf{z}_m,\ j = 1, 2, \ldots, p$.
3. Output the sequence of fitted vectors $\{\hat{\mathbf{y}}^{(m)}\}_1^p$. Since the $\{\mathbf{z}_\ell\}_1^m$ are linear in the original $\mathbf{x}_j$, so is $\hat{\mathbf{y}}^{(m)} = \mathbf{X}\hat{\beta}^{\text{pls}}(m)$. These linear coefficients can be recovered from the sequence of PLS transformations.

Figure 6: Gram-Schmidt procedure for multiple regression

5.2 Partial least square

The mth PLS direction $\hat{\psi}_m$ solves:

$$\max_\alpha \text{Corr}^2(\mathbf{y}, \mathbf{X}\alpha)\text{Var}(\mathbf{X}\alpha)$$
$$\text{subject to } ||\alpha| = 1, \alpha^T \mathbf{S}\hat{\psi}_l = 0, \quad l = 1, ...n - 1$$

Chapter 4: Linear Methodsfor Classification

1. Linear discriminant analysis

Question set up for classification:

- Goal: To know the class posteriors $Pr(G|X)$ for optimal classification
- Parameters: $f_k(x)$ is the class conditional density of X in class $G = k$, and π_k is the probability of class k, with $\sum_{k=1}^{K} \pi_k = 1$.
- Bayes theorem gives that $Pr(G = k|X = x) = \frac{f_k(x)\pi_k}{\sum_{l=1}^{K} f_l(x)\pi_l}$

Suppose each class density is a multivariate Gaussian

$$f_k(x) = \frac{1}{(2\pi)^p|\mathbf{\Sigma}|^{1/2}} e^{-\frac{1}{2}(x-\mu_k)^T\mathbf{\Sigma}^{-1}(x-\mu_k)} \quad \text{assume equal variance across classes}$$

To compare the two classes k and l, we have

$$\log \frac{Pr(G = k|X = x)}{Pr(G = l|X = x)} = \log \frac{\pi_k}{\pi_l} - \frac{1}{2}(\mu_k + \mu_l)^T\mathbf{\Sigma}^{-1}(\mu_k - \mu_l) + x^T\mathbf{\Sigma}^{-}1(\mu_k - \mu_l)$$

which is linear in x.

- Linear discriminant function is to solve for $G(x) = \text{argmax}_k \delta_k(x)$

$$\delta_k(x) = x^T\mathbf{\Sigma}^{-1}\mu_k - \frac{1}{2}\mu_k^T\mathbf{\Sigma}^{-1}\mu_k + \log \pi_k$$

π_k, μ_k, $\mathbf{\Sigma}$ all needs to be estimated empirically.

- Quadratic discriminant function is where $\mathbf{\Sigma}_k$ are not all the same, then the function becomes

$$\delta_k(x) = -\frac{1}{2}\log|\mathbf{\Sigma}_k| + \log \pi_k - \frac{1}{2}(x - \mu_k)^T\mathbf{\Sigma}_k^{-1}(x - \mu_k)$$

1.1 Regularized disciminant analysis

Allow one to shrink the separate covariance of QDA toward a common covariance as in LDA

$$\hat{\mathbf{\Sigma}}_k(\alpha) = \alpha\hat{\mathbf{\Sigma}}_k + (1 - \alpha)\hat{\mathbf{\Sigma}}$$

2. Logistic regression

For multiple K class,

$$Pr(G = k|X = x) = \frac{\exp(\beta_{k0} + \beta_k^T x)}{1 + \sum_{l=1}^{K-1}\exp(\beta_{l0} + \beta_l^T x)} \quad k = 1, ..., K-1$$
$$Pr(G = l|X = x) = \frac{1}{1 + \sum_{l=1}^{K-1}\exp(\beta_{l0} + \beta_l^T x)}$$

2.1 Fitting a binary class logistic regression

- Log likelihood for multiclass

$$l(\theta) = \sum_{i=1}^{N} \log p_{g_i}(x_i; \theta)$$

where $p_k(x_i; \theta) = Pr(G = k | X = x_i; \theta)$.

- For a two class case, $p_1(x; \theta) = p(x; \theta)$ corresponding to $y_i = 1$, and $p_2(x; \theta) = 1 - p(x; \theta)$ corresponding to $y_i = 0$. Then the log likelihood becomes

$$\begin{aligned} l(\beta) &= \sum_{i=1}^{N} \{y_i \log p(x_i; \theta) + (1 - y_i \log(1 - p(x_i; \theta)))\} \\ &= \sum_{i} \{y_i \beta^T x_i - \log(1 + e^{\beta^T x_i})\} \end{aligned}$$

- First order derivative

$$\begin{aligned} \frac{\partial l(\beta)}{\partial \beta} &= \sum_{i} x_i (y_i - p(x_i; \beta)) = 0 \\ &= \mathbf{X}^T (\mathbf{y} - \mathbf{p}) \end{aligned}$$

- Second order derivative

$$\begin{aligned} \frac{\partial^2 l(\beta)}{\partial \beta \partial \beta^T} &= -\sum_{i} x_i x_i^T p(x_i; \beta)(1 - p(x_i; p)) \\ &= -\mathbf{X}^T \mathbf{W} \mathbf{X} \end{aligned}$$

- Newton Raphson

$$\begin{aligned} \beta^{\text{new}} &= \beta^{\text{old}} - (\frac{\partial^2 l(\beta)}{\partial \beta \partial \beta^T})^{-1} \frac{\partial l(\beta)}{\partial \beta} \\ &= \beta^{\text{old}} + (\mathbf{X}^T \mathbf{W} \mathbf{X})^{-1} \mathbf{X}^T (\mathbf{y} - \mathbf{p}) \\ &= (\mathbf{X}^T \mathbf{W} \mathbf{X})^{-1} \mathbf{X}^T \mathbf{W} (\mathbf{X} \beta_{old} + \mathbf{W}^{-1} (\mathbf{y} - \mathbf{p})) \\ &= (\mathbf{X}^T \mathbf{W} \mathbf{X})^{-1} \mathbf{X}^T \mathbf{W} \mathbf{z} \end{aligned}$$

The last step can be considered as iteratively reweighted least squares

$$\beta^{\text{new}} \rightarrow \text{argmin}_{\beta} (\mathbf{z} - \mathbf{X}\beta)^T \mathbf{W} (\mathbf{z} - \mathbf{X}\beta)$$

2.1 L1 regularized logistic regression

$$\max_{\beta_0, \beta_1} \{\sum_{i=1}^{N} [y_i (\beta_0 + \beta^T x_i) - \log(1 + e^{\beta_0 + \beta^T x_i})] - \lambda \sum_{j=1}^{p} |\beta_j|\}$$

3. Separating hyperplanes

Hyperplane (affine set) L defined by the equation $f(x) = \beta_0 + \beta^T x = 0$, in $\mathbb{R}^2$, is a line, with the properties

- For any two points in L, $\beta^T(x_1 - x_2) = 0$
- For any point x_0 in L, $\beta^T x_0 = -\beta_0$
- The signed distance of any point x to L is $\frac{1}{||\beta||}(\beta^T x + \beta_0) = \frac{1}{||f'(x)||} f(x)$

3.1 Perpcetron learning algorithm

For a two class problem $y_i \in \{-1, 1\}$

$$x_i^T \beta + \beta_0 < 0 \quad \text{if } y_i = 1 \text{ is misclassied}$$
$$x_i^T \beta + \beta_0 > 0 \quad \text{if } y_i = -1 \text{ is misclassied}$$

Therefore, the goal is to minimize

$$D(\beta, \beta_0) = -\sum_{i \in \mathcal{M}} y_x (x_i^T \beta + \beta_0)$$

where $\mathcal{M}$ is the set of misclassified points. This quantity is nonnegative and proportional to the distance of the misclassfied points to the decision boundary $\beta^T x + \beta_0 = 0$. The gradient is

$$\partial \frac{D(\beta, \beta_0)}{\partial \beta} = -\sum_{i \in \mathcal{M}} y_i x_i$$
$$\partial \frac{D(\beta, \beta_0)}{\partial \beta} = -\sum_{i \in \mathcal{M}} y_i$$

3.2 Optimal separating hyperplanes

Definition: OSH separates the two classes and maximizes the distance to the closest point from either class.

$$\max_{\beta, \beta_0, ||\beta||=1} M$$
$$\text{subject to } y_i(x_i^T \beta + \beta_0) \geq M \quad \forall i$$

Interpretation: all points are at least a signed distance M from the decision boundary defined by β and β_0, and seek the largest the M. $||\beta|| = 1$ can be removed by changing the condition to $y_i(x_i^T \beta + \beta_0) \geq M||\beta||$

If we arbitrarily set $||\beta|| = 1/M$, the question becomes

$$\min_{\beta, \beta_0} \frac{1}{2}||\beta||^2$$
$$\text{subject to } y_i(x_i^T \beta + \beta_0) \geq 1 \quad \forall i$$

The constraints define a margin around the linear decision boundary of thickness $1/||\beta||$.

The question is to mimimize the Lagrange function

$$L_p = \frac{1}{2}||\beta||^2 - \sum_i \alpha_i [y_i(x_i^T \beta + \beta_0) - 1]$$

Chapter 5: Basis Expansions and Regularization

1. Introduction

Core concept: To augment and replace the vector of inputs X with additional variables which are transformations of X and then use the linear models in this new space of derived input features.

$$f(X) = \sum_{m=1}^{M} \beta_m h_m(X)$$

where $h_m(X) : \mathbb{R}^p \to \mathbb{R}$ is a transformation of X. This is called a linear basis expansion in X.

2. Peicewise Polynomials and Splines (restricted model)

Dividing the domain of X into contiguous intervals, and representing f by a separate polynomial in each interval.

An order-M (degree of the polynomial plus 1, i.e. cubic spline has $M = 4$) spline with knots ξ_j, $j = 1 \dots K$ is a piecewise polynomial of order M, and has continuous derivatives up to order $M - 2$. The form of the truncate power basis is

$$h_j(X) = X^{j-1}, j = 1 \dots M$$
$$h_{M+l}(x) = (X - \xi_l)_+^{M-1}, l = 1...K$$

Cubic spline is typically good enough to depict continuity unless we need smooth derivatives.

One approach is to parametrize a family of spline by the number of basis functions or degree of freedom and have the observations x_i determine the positions of the knots

2.1 Natural Cubic Splines

An NCS adds additional constraints, namely that the function is linear bey9ond the boudary knots which frees up four degrees of freedom (two constraints each in both boundary regions)

An NCS with K knots is represented by K basis functions.

$$N_1(X) = 1, N_2(X) = X, N_{k+2}(X) = d_k(X) - d_{k-1}(X)$$
$$d_k(X) = \frac{(X - \xi_k)_+^3 - (X - \xi_K)_+^3}{\xi_K - \xi_k}$$

3. Filtering and Feature Extraction

4. Smoothing Spines

Smoothing splines aviods the knot selection porblem compeltely by using a maximal set of knots. The complexity of the fit is controlled by regularization.

$$RSS(f, \lambda) = \sum_{i=1}^{N} \{y_i - f(x_i)\}^3 + \lambda \int \{f''(t)\}^2 dt$$

The penalty term penalizes curvature in the function using the smoothing parameter λ. $\lambda = 0$: f is any function that interpolates the data. $\lambda = \infty$: simple least squares linear fit with no second derivative tolerated. The results has a unique minimizer which is a natural cubic spline with knots at the unique values of the x_i. The solution can be written as

$$f(x) = \sum_{j=1}^{N} N_j(x)\theta_j$$

where $N_j(x)$ are an N-dimensional set of basis functions for representing this family of natural splines.

The criterion thus reduces to

$$RSS(\theta, \lambda) = (\mathbf{y} - \mathbf{N}\theta)^T(\mathbf{y} - \mathbf{N}\theta) + \lambda\theta^T\mathbf{\Omega}_N\theta$$

where $\{\mathbf{N}\}_{ij} = N_j(x_i)$ and $\{\mathbf{\Omega}_N\}_{jk} = \int N_j^{''}(t)N_k^{''}(t)dt$, with the solution

$$\hat{\theta} = (\mathbf{N}^T\mathbf{B} + \lambda\mathbf{\Omega}_N)^{-1}\mathbf{N}^T\mathbf{y}$$

4.1 Degress of Freedom and Smoother Matrices

Fitted value can be written as

$$\hat{\mathbf{f}} = \mathbf{N}(\mathbf{N}^T\mathbf{B} + \lambda\mathbf{\Omega}_N)^{-1}\mathbf{N}^T\mathbf{y} = \mathbf{S}_\lambda y$$

$\mathbf{S}_\lambda$ is known as the smoother matrix, which depends only on x_i and λ.

$\mathbf{S}_\lambda$ is symmetric and positive semidefinite and has real eigendecompistion.

The effective degree of freedom of a smoothing spline is

$$\mathrm{df}_\lambda = \mathrm{trace}(\mathbf{S}_\lambda)$$

5. Automatic Selection of Smoothing Parameters

Fix the df to determine λ

```
smooth.spline(x,y,df = 6)
```

Covariance: $\mathrm{Cov}(\hat{\mathbf{f}}) = \mathbf{S}_\lambda\mathrm{Cov}(\mathbf{y})\mathbf{S}_\lambda^T = \mathbf{S}_\lambda\mathbf{S}_\lambda^T$ assuming iid normal error

Bias: $\mathrm{Bias}(\hat{\mathbf{f}}) = f - \mathbf{S}_\lambda f$

small df_λ: underfit and trims down the hills and fills in the valleys

large df_λ: overfit and wiggly

6. Nonparametric Logistic Regression

$$\log\frac{Pr(Y=1|X=x)}{Pr(Y=0|X=x)} = f(x)$$

where $f(x)$ is a smooth function. $Pr(Y=1|X=x)$ is therefore also smooth.

The log likelihood is

$$l(f;\lambda) = \sum_{i=1}^{N}[y_i f(x_i) - \log(1 + e^{f(x_i)})] - \frac{1}{2}\int \{f''(t)\}^2 dt$$

7. Multidimensional Splines

Generalized form

$$\min_f \sum_{i=1}^{N} B\{y_i - f(x_i)\}^2 + \lambda J[f]$$

where J is an appropriate penalty functional for stabilizing a function f in $\mathbb{R}^d$. E.g. when $d = 2$, $J[f] = \int\int_{\mathbb{R}}^2 [(\frac{\partial^2 f(x)}{\partial x_1^2})^2 + 2(\frac{\partial^2 f(x)}{\partial x_1 \partial x_2}) + (\frac{\partial^2 f(x)}{\partial x_2^2})^2] dx_1 dx_2$

8. Regularization and Reproducing Kernel Hilbert Spaces

9. Wavelet Smoothing

Chapter 6: Kernel Smoothing Method

0. Introduction

Class of regression techniques that achieve flexibility in estimating the regression function $f(X)$ over $\mathbb{R}^p$ by fitting different but simple model separately at each query point x_0 using only those observations close to the target point x_o to fit the simple model. This localization is achieved via a weighting function kernel $K_\lambda(x_0, x_i)$. which assigns weight to x_i based on its distance from x_0.

1. One-dimensional Kernel Smoothers

KNN uses unweighted average in the neighbourbood which caused the bumpy estimation. One can assign weights that die off smoothly within distance from the traget point.

Kernel function

$$K_\lambda(x_0, x) = D(\frac{|x - x_0|}{h_\lambda(x_0)})$$

Locally weighted regression (at each target point x_0):

$$\min_{\alpha(x_0),\beta(x_0)} = \sum_{i=1}^{N} K_\lambda(x_0, x_i)[y_i - \alpha(x_0) - \beta(x_0)x_i]^2$$

2. Selecting Width of the Kernel

- small window $\rightarrow$ variance and small bias
- wide window the other way around

3. Kernel Density Estimation and Classification

Kernel density: $\hat{f}_X(x_0) = \frac{1}{N\lambda}\sum_{i=1}^{N} K_\lambda(x_0, x_i)$. A popular choice of K_λ is the Gaussian kernel $\phi(|x - x_0|/\lambda)$.

Suppose for a J class problem ,w e fit nonparametric density estimates $\hat{f}_j(X), j = 1 \dots J$ in each class, and we know the class priors $\hat{\pi}_j$, then $\hat{\mathrm{P}}_\mathrm{r}(G = j|x_x0) = \frac{\hat{\pi}_j \hat{f}_j(x_0)}{\sum_k \hat{\pi}_k \hat{f}_k(x_0)}$

3.1 Naive Bayes

Assumption: Given a class $G = j$, the features X_k in the p diemsnonal space are independent, i.e. $f_j(X) = \Pi_{k=1}^{p} f_{jk}(X_k)$

- f_{jk} can be estimated using one dimensional kernel density estimates.

4. Radial Basis Functions and Kernels

Radial basis function treats kernel functions $K_\lambda(\xi, x)$ as basis functions for a smmothing spline estimation, whcih leads to

$$f(x) = \sum_{j=1}^{M} K_{\lambda_j}(\xi_j, x)\beta_j$$

Chapter 7: Model Assessment and Selection

1. Introduction

The performance of a learning method relates to its prediction capability on independent test data

2. Bias, Variance, and MOdel Complexity

2.1 Regression

- **Test error**: aka *generalization error*, is the prediction error over an independent test sample $\text{Err}_\tau = EL(Y, \hat{f}(x))|\tau$. τ represents a fixed training set.
- **Expected prediction error**: or *expected test error* $\text{Err} = E[L(Y, \hat{f}(X))] = E[\text{Err}_\tau]$. Expectation averages over everything that is random, including the randomness in the training set that produced $\hat{f}$.
- **Training error**: is the average loss over the training sample $\bar{\text{err}} = \frac{1}{N}\sum_{i=1}^{N} L(y_i, \hat{f}(x_i))$

2.2 Classification

Loss function can be 0-1 loss oir $-2\times$log likelihood (aka deviance.)

2.3 Split of data

Training (fit the model), validation (model selection), and test (model assessment).

3. Bia-Variance Decomposition

The expected prediction error of a regression fit $\hat{f}(x)$ at input $X = x_0$

$$\begin{aligned}\text{Err}(x_0) &= E[(Y - \hat{f}(x_0))^2|X = x_0] \\ &= \sigma_\epsilon^2 + [E\hat{f}(x_0) - f(x_0)]^2 + E[\hat{f}(x_0) - E\hat{f}(x_0)]^2 \\ &= \sigma_\epsilon^2 + \text{Bias}^2(\hat{f}(x_0)) + \text{Var}(\hat{f}(x_0))\end{aligned}$$

4. The Optimism of the Training Error Rate

5. Estimates of In-Sample Prediction Error

$C_p = \bar{\text{err}} + 2\frac{d}{N}\hat{\sigma}_\epsilon^2$, d is the number of inputs or basis functions

$AIC = -\frac{2}{N}\text{loglik} + 2\frac{d}{N}$

$AIC(\alpha) = \bar{\text{err}}(\alpha) + 2\frac{d(\alpha)}{N}\hat{\sigma}_\epsilon^2$ for models with tuning parameter α

6. Effective Number of Parameters

7. The Bayesian Approachj and BIC

General form $\text{BIC} = -2\text{loglik} + (\log N)d$

10. Cross Validation

Goal: To estimate prediction error. It estimates the expected extra-sample error Err $= E[L(Y, \hat{f}(X))]$

10.1 K Fold CV

Split the data into K parts with equal size. For the kth part, we fit the model to the other $K-1$ parts of the data and calculate the prediction error of the fitted model when predicting the kth part of the data. We do this for $k = 1, 2, ...K$ and combine the K estimates of the prediction errors.

$\| : \{1...N\} \to \{1...K\}$ represents the partition. $\hat{f}^{-k}(x)$ denoted the fitted value while excluding the set k. The cross validation estimate of the prediction error is

$$\text{CV}(\hat{f}) = \frac{1}{N}\sum_{i=1}^{N} L(y_i - \hat{f}^{-k}(x))$$

or

$$\text{CV}(\hat{f}) = \frac{1}{K}\sum_{k=1}^{K}[\frac{1}{n_k}\sum_{i\in\text{kth val set}} L(y_i - \hat{f}^{-k}(x))]$$

LOCV: Low bias but high variance, also high computational burden.

One Standard Error Rule Choose the most parsimonious model whose error is no more than one standard error above the error of the best model.

Generalized Cross Validation A convenient approximation to leave one out CV, for linear fitting under squared error loss.

$$GCV(\hat{f}) = \frac{1}{N}\sum_{i}[\frac{y_i - \hat{f}(x_i)}{1 - \text{trace}(\mathbf{S})/N}]^2$$

10.2 Tips for CV

DON'T subset the data before CV, because the subset was chosen on the basis of all of the samples. Leaving samples out after the subset does not correctly mimic the application of the classifier.

In general, with a multistep modeling procedure, cross validation must be applied to the entire sequence of modeling steps.

11. Bootstrap

Chapter 8: Model Inference and Averaging

1. Introduction

- minimize sum of square for regression
- minimize cross entropy for classification

2. The Bootstrape and Maximum Likelihood Methods

2.1 A smoothing example

Denote the training data by $\mathbf{Z} = \{z_1, z_2, \ldots, z_n\}$, with $z_i = (x_i, y_i)$, and $x_i \in \mathbb{R}^1$. If we use a B-spline basis expansion with 3 knots placed on the quartiles, we can represent the function by

$$E(Y|X = x) = \mu(x) = \sum_{j=1}^{7} \beta_j h_j(x)$$

MSE

The MSE can be represented by $\hat{\beta} = (\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{y}$, where $\mathbf{H} \in \mathbb{R}^{N\times 7}$ with ijth elements $h_j(x_i)$. The estimated covariance matrix of $\hat{\beta}$ is $\hat{\text{Var}}(\hat{\beta}) = (\mathbf{H}^T\mathbf{H})^{-1}\hat{\sigma}^2$, where $\hat{\sigma}^2 = \sum_{i=1}^{N}(y_i - \hat{\mu}(x_i))^2/N$. The standard error of a prediction $\hat{\mu}(x) = h(x)\hat{\beta}$ is $\hat{\text{SE}}[\hat{\mu}(x)] = [h(x)^T(\mathbf{H}^T\mathbf{H})^{-1}h(x)]^{1/2}\hat{\sigma}$. This can be used to construct the confidence interval.

Bootstrape Approach

Nonparametric bootstrap: Draw B datasets each of size $N = 50$ with replacement, where the sampling unit is $z_i = (x_i, y_i)$. Using each boostrapped dataset $\mathbf{Z}^*$, we fit a cubic spline $\hat{\mu}^*(x)$. With large enough resamles, we can use percentiles to get the confidence band.

Parametric bootstrap: simulate new responses by adding GAussian noise to the predicted values (assuming we assume a Gaussian error for the original model).

2.2 MLE

In general parametric bootstrap agree with MLE.

ADD MLE FURTHER

3. Baysian Methods

Baysian specifies a sampling model $\Pr(\mathbf{Z}|\theta)$, and a prior distribution for the parameter $\Pr(\theta)$ representing our knowledge about θ. The goal is to compute the posterior distribution

$$Pr(\theta|\mathbf{Z}) = \frac{Pr(\mathbf{Z}|\theta) \cdot Pr(\theta)}{\int Pr(\mathbf{Z}|\theta) \cdot Pr(\theta) d\theta}$$

.

A future observation can be predicted via the predictive distribution

$$Pr(z^{new}|\mathbf{Z}) = \int Pr(z^{new}|\theta) Pr(\theta|\mathbf{Z}) d\theta$$

Chapter 9: Additive Models, Trees, and Related Methods

1. Introduction

Chapter 10: Boosting and Addtive Trees

1. Introduction

Chapter 11: Neural Networks

1. Introduction

Chapter 12: Support Vector Machines and Flexible Discriminants

1. Introduction

Chapter 13: Prototype Methods and Nearest-Neighbors

1. Introduction

Chapter 14: Unsupervised Learning

1. Introduction

Chapter 15: Random Forests

1. Introduction

Chapter 16: Ensemble Learning

1. Introduction

Chapter 17: Undirected Graphical Models

1. Introduction

Chapter 18: High-Dimensional Problems: $p >> N$

1. Introduction