1 Multiple Linear Regression

Model Formula and assumptions: $Y_i = \sum_{j=1}^p \beta_j x_{ij} + \varepsilon_i \Leftrightarrow Y = X\beta + \varepsilon \Leftrightarrow Y_i = x_i^T\beta + \varepsilon_i \Leftrightarrow Y_i = \beta_1 + \sum_{j=2}^p X_{ij}\beta_j + \varepsilon_i$ where $\varepsilon_1, \dots, \varepsilon_n \overset{\text{iid}}{\sim} \mathcal{N}(0, \sigma^2)$ hence $\text{Cov}(\varepsilon) = \sigma^2 \mathbb{1}$. Assume $n \geq p$ and thus X has full rank.

Least Squares Model $\hat\beta = \text{argmin}\,\|Y - X\beta\|_2^2 = (X^TX)^{-1}X^TY$. Def: $\varepsilon_i \approx Y_i - x_i^T\beta := r_i \Rightarrow \hat\sigma^2 = \frac{1}{n-p}\sum_{i=1}^n r_i^2$.

Assumptions: 1) LinRegEq is correct ($\mathbb{E}[\varepsilon_i] = 0$), 2) x_i's are exact, 3) Var of Errors is const ($\text{Var}(\varepsilon_i) = \sigma^2$) ("Homoscedasticity"), 4) Errors are uncorrelated ($\text{Cov}(\varepsilon_i, \varepsilon_j) = 0$), 5) Errors ε_i are jointly normally distributed $\Rightarrow Y_i$'s are jointly normally distributed.

Geometric Interpretation $X\hat\beta$ is the orthogonal projection of Y onto $\mathcal{X} \equiv \text{span}(X)$ (columnwise span) $\Rightarrow \hat Y = X\hat\beta = \underbrace{X(X^TX)^{-1}X^T}_{:=P}Y$. $\mathcal{X}$ is a p-dim subspace of $\mathbb{R}^n$, $P = P^T = P^2$, $\text{tr}(P) = \text{tr}(\mathbb{1}_p) = p$. The residuals live in $\mathcal{X}^\perp$: $\vec r = (1 - P)\vec Y$.

1.1 Properties of LS Estimator

(i) $\mathbb{E}[\hat\beta] = \beta$ hence $\hat\beta$ is an unbiased estimator, (ii) $\mathbb{E}[\hat Y] = \mathbb{E}[Y] = X\beta$, $\mathbb{E}[r] = 0$, (iii) $\text{Cov}(\hat\beta) = \sigma^2(X^TX)^{-1}$, (iv) $\text{Cov}(\hat Y) = \sigma^2 P$, $\text{Cov}(r) = \sigma^2(1 - P)$, (v) $\text{Var}(r_i) = \sigma^2(1 - P_{ii})$ hence $\mathbb{E}[\sum_i r_i^2] = \sum_i \text{Var}(r_i) = \sigma^2(n - \text{tr}(P)) = \sigma^2(n-p)$ hence $\mathbb{E}[\hat\sigma] = \mathbb{E}\left[\frac{1}{n-p}\sum_i r_i^2\right] = \sigma$ hence an unbiased estimator, (vi) $\hat\beta \sim \mathcal{N}_p\left(\beta, \sigma^2(X^TX)^{-1}\right)$, $\hat Y \sim \mathcal{N}_n(X\beta, \sigma^2 P)$, $r \sim \mathcal{N}_n(0, \sigma^2(1-P))$, $\hat\sigma^2 \sim \frac{\sigma^2}{n-p}\chi^2_{n-p}$

1.2 Tests and Confidence Regions

$H_{0,j}:\ \beta_j = 0$, std.error: $\sqrt{\widehat{\text{Var}}(\hat\beta_j)}$. Thus: $\frac{\hat\beta_j}{\sqrt{\sigma^2(X^TX)^{-1}_{jj}}} \sim \mathcal{N}(0,1)$ under $H_{0,j}$ and $T_j = \frac{\hat\beta_j}{\sqrt{\hat\sigma^2(X^TX)^{-1}_{jj}}} \sim t_{n-p}$ under $H_{0,j}$. Conf Int: $\hat\beta_j \pm \sqrt{\hat\sigma^2(X^TX)^{-1}_{jj}} \cdot t_{n-p;1-\frac{\alpha}{2}}$

Test of global H_0 H_0: $\beta_2 = \dots = \beta_p = 0$, H_A: $\exists j \in \{2, \dots, p\} : \beta_j \neq 0$. $F = \frac{\|\hat Y - \bar Y\|_2^2/(p-1)}{\|Y - \hat Y\|_2^2/(n-p)} \sim F_{p-1,n-p}$ under H_0. $R^2 = \frac{\|\hat Y - \bar Y\|_2^2}{\|Y - \bar Y\|_2^2} = 1 - \frac{\|Y - \hat Y\|_2^2}{\|Y - \bar Y\|_2^2}$

Confidence Lvls We say a Test $T_n : \mathcal{X}^n \to \{0,1\}$ is (i) lvl α if: $\sup_{P \in H_0} \mathbb{P}_{P^n}(T_n = 1) \leq \alpha$, (ii) pointwise asymptotic lvl α if $\sup_{P \in H_0} \lim_{n\to\infty} \mathbb{P}_{P^n}(T_n = 1) \leq \alpha$, (iii) uniform asymptotic lvl α if $\lim_{n\to\infty} \sup_{P \in H_0} \mathbb{P}_{P^n}(T_n = 1) \leq \alpha$

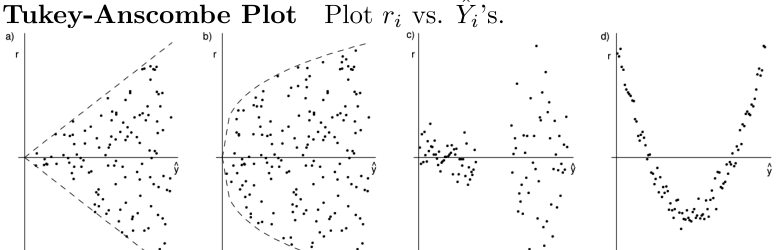
p-value Infimum over all significance lvl's α s.t. test rejects.

For new x_0**:** Confidence Interval: $\frac{x_0^T\hat\beta - x_0^T\beta}{\hat\sigma\sqrt{x_0^T(X^TX)^{-1}x_0}} \sim t_{n-p}$,

Prediction Interval: $\frac{y_0 - x_0^T\hat\beta}{\hat\sigma\sqrt{1 + x_0^T(X^TX)^{-1}x_0}} \sim t_{n-p}$

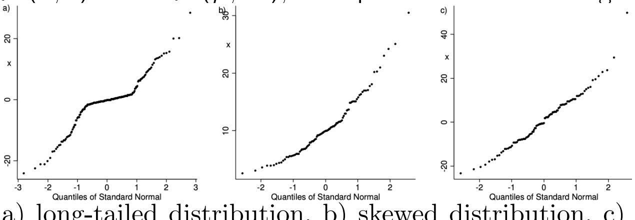
1.3 Analysis of Residuals & Checking of Model Assumptions

Tukey-Anscombe Plot Plot r_i vs. $\hat Y_i$'s.



a) Linear increase of sd (solution: log-transform), b) non-linear increase of sd, c) 2 groups with different variances, d) missing quadratic term in the model

QQ-Plot Plot empirical quantiles vs. theoretical quantiles of $\mathcal{N}(0,1)$. If $r \sim \mathcal{N}(\mu, \sigma^2)$, then plot would be a straight line.



a) long-tailed distribution, b) skewed distribution, c) dataset with outlier

1.4 Generalized LS and weighted Regression

$Y = X\beta + \varepsilon$, $\varepsilon \sim \mathcal{N}_n(0, \Sigma)$, Σ known and $\Sigma = CC^T$. Reformulate: $\tilde Y = \tilde X\beta + \tilde\varepsilon$ with $\tilde Y = C^{-1}Y$, $\tilde X = C^{-1}X$, $\tilde\varepsilon \sim \mathcal{N}_n(0, \mathbb{1})$ and obtain: $\hat\beta = \left(X^T\Sigma^{-1}X\right)^{-1}X^T\Sigma^{-1}Y$, $\text{Cov}(\hat\beta) = \left(X^T\Sigma^{-1}X\right)^{-1}$

Special Case: $\Sigma = \sigma^2\text{diag}(z_1, \dots, z_n) \Rightarrow$ weighted LS problem: $\min_\beta \sum_{i=1}^n w_i\left(Y_i - x_i^T\beta\right)$, $w_i = \frac{1}{z_i}$

1.5 Model Selection

Maybe some β_j's are irrelevant. If $p > n$, X^TX is not invertible anymore and $\hat\beta\text{argmin}_\beta\|Y - X\beta\|_2^2$ not unique anymore and $\mathbb{E}[X\hat\beta] \neq X\beta$. Approach: $Y_i \approx \sum_{r=1}^q \hat\beta_{j_r}x_{i,j_r} =: \hat m_{\{j_1,\dots,j_q\}}(\vec x_i)$, $j_l \in \{1, \dots, p\}$. $aMSE = \frac{1}{n}\sum_{i=1}^n \mathbb{E}\left[(m(\vec x_i) - \hat m_q(\vec x_i))^2\right] = \frac{1}{n}\sum_{i=1}^n(\mathbb{E}[\hat m_q(\vec x_i)] - m(\vec x_i))^2 + \frac{1}{n}\sum_{i=1}^n \text{Var}(\hat m_q(\vec x_i)) = \frac{1}{n}\sum_{i=1}^n(\mathbb{E}[\hat m_q(\vec x_i)] - m(\vec x_i))^2 + \frac{q}{n}\sigma^2$

Penelized Regression (Mellows C_p **Statistic)** $\hat\beta_\lambda = \text{argmin}_\beta\|Y - X\beta\|_2^2 + \lambda\|\beta\|_0$. λ large $\to$ fewer variables. $C_p = \frac{\|Y - X_{\mathcal{M}}\hat\beta_{\mathcal{M}}\|_2^2}{\sigma^2_{\mathcal{M}}} - n + 2(n - df_{residuals})$. Akaike's Information Criterion (AIC) $\Leftrightarrow$ Mellows C_p for Gaussian Models: $\lambda = 2\hat\sigma^2$. Bayese Information Criterion (BIC): $\lambda = \log(n)\hat\sigma^2$

Searching for best Model Forward Selection: Start with smallest model and iteratively add a predictor variable that reduces RSS the most. You obtain a seq of Models $\mathcal{M}_0 \subseteq \mathcal{M}_1 \subseteq \dots$. Chose $\mathcal{M}_i$ that minimizes RSS_λ. Backward Selection: Start with full model and iteratively exclude pred. vars that increase RSS the least ($\mathcal{M}_0 \supseteq \mathcal{M}_1 \supseteq \dots$) and chose $\mathcal{M}_i$ that minimizes RSS_λ.

2 High Dimensional Regression

2.1 Ridge Regression

Assume X and Y are centered. (If not, center them). Optim problem: $\hat\beta^\lambda = \text{argmin}_{\beta\in\mathbb{R}^p}\left\{\|Y - X\beta\|_2^2 + \lambda\|\beta\|_2^2\right\} = \left(X^TX + \lambda\mathbb{1}\right)^{-1}X^TY$.

Note: $\lambda = 0$: Bias$= 0$, Var $= \sum_{i=1}^d \frac{\sigma_\varepsilon^2}{\sigma_i^2}$, $\lambda \to \infty$: Bias$\to \|w^*\|_2^2$, Var $\to 0$. $\hat\beta^\lambda$ is biased, but has smaller variance than $\hat\beta^{LS}$. $\text{MSE}(\hat\beta^\lambda) < \text{MSE}(\hat\beta^{LS})$

X **only orthogonal pred** $\Rightarrow$ X^TX diagonal: $(X^TX)_{kk} = d_k^2$, $D^2 := X^TX$. Then: $\hat\beta_k^\lambda = \frac{1}{d_k^2 + \lambda}(X^Ty)_k = \frac{d_k^2}{d_k^2+\lambda}\hat\beta_k^{OLS}$

X **non-orthogonal pred** Use SVD: $X = UDV^T$. Rotate: $\tilde X = XV$ (orthogonal). Then: $\hat{\tilde\beta}^\lambda = \left(V^TX^TXV + \lambda\mathbb{1}\right)^{-1}V^TX^TY$

Kernel Ridge Regression $\beta = X^T\alpha$ and $K = XX^T$: $\hat\alpha = \text{argmin}_{\alpha\in\mathbb{R}^n}\|Y - K\alpha\|_2^2 + \lambda\alpha^TK\alpha \Rightarrow K\hat\alpha = K(K + \lambda\mathbb{1})^{-1}Y$

2.2 LASSO Regression

$\hat\beta^\lambda = \text{argmin}_{\beta\in\mathbb{R}^p}\left\{\|Y - X\beta\|_2^2 + \lambda\|\beta\|_1\right\}$, Biased estimator

X **orthogonal** X^TX diag: $\hat\beta_k^\lambda = \text{sign}(\hat\beta_k^{OLS}) \cdot \max\left\{0, \left|\hat\beta_k^{OLS}\right| - \frac{\lambda}{2}\right\}$

Elastic Lasso $\hat\beta^{\lambda_1,\lambda_2} = \text{argmin}_{\beta\in\mathbb{R}^p}\|Y - X\beta\|_2^2 + \lambda_2\|\beta\|_2^2 + \lambda_1\|\beta\|_1$ Equiv: $\hat\beta^{\lambda_1,\lambda_2} = \text{argmin}_{\beta\in\mathbb{R}^p}\|Y - X\beta\|_2^2$ s.t. $(1-\alpha)\|\beta\|_1 + \alpha\|\beta\|_2^2 \leq s$ with $\alpha = \frac{\lambda_2}{\lambda_2 + \lambda_1} \in [0,1]$

Group Lasso p pred. are grouped into L groups of size p_l. $\vec\beta = (\vec\beta_1, \dots, \vec\beta_L)^T$ and X made of L blocks of col's X_l: $X\beta = \sum_{l=1}^L X_l\beta_l$. Hence: $\hat\beta_\lambda^{GL} = \text{argmin}_{\beta\in\mathbb{R}^p}\left\|Y - \sum_{l=1}^L X_l\beta_l\right\|_2^2 + \lambda\sum_{l=1}^L\sqrt{p_l}\|\beta_l\|_2$

3 Non Parametric Density Estimation

Histogram Origin x_0 & class width h: $I_j = (x_0 + jh, x_0 + (j+1)h]$ $f_{x_0,h} = \sum_{j\in\mathbb{Z}}\hat g_j \mathbb{1}_{x\in I_j}$ where $\hat g_j = \frac{\#\{i\in\{1,\dots,n\}: x_i\in I_j\}}{n\cdot h}$

Kernel Estimator Def Kernel: $K : \mathbb{R} \to \mathbb{R}_{\geq 0}$ s.t. $\int_{-\infty}^\infty K(x)dx = 1$, K is bounded and $\forall x \in \mathbb{R} : K(-x) = K(x)$.
Fix K and h. Def: $\hat f_h(x) := \frac{1}{n\cdot h}\sum_{i=1}^n K\left(\frac{x - x_i}{h}\right)$

Role of Bandwidth h large: $\hat f_h(x)$ "smooth" and slowly varying, h small: $\hat f_h(x)$ more wiggly

K-Nearest Neighbors Variable bandwidth: $h = h(x)$.

Bias-Variance Trade-off As $h \nearrow$: $|Bias| \nearrow$ and Var $\searrow$. $\text{MSE}(x) = \mathbb{E}\left[\left(\hat f(x) - f(x)\right)^2\right] = \left(\mathbb{E}\left[\hat f(x)\right] - f(x)\right)^2 + \text{Var}\left(\hat f(x)\right)$ Goal: Chose h s.t. $IMSE = \int \text{MSE}(x)dx$ is minimized.
$h_{opt}(x) = n^{-1/5}\left(\frac{f(x)\int K(z)^2dz}{(f''(x)\int z^2K(z))^2}\right)^{1/5}$ and $h_{opt} = n^{-1/5}\left(\frac{R(K)}{\sigma_K^4 \cdot R(f'')}\right)^{1/5}$ where $R(g) = \int g^2(x)dx$ and $\sigma_K^2 = \int x^2K(x)dx$. Estimate h iter-

atively: 1) take h_{init}, 2) estimate f'' by $f''_{h_{init}}$, 3) calculate $\hat{h}$, 4) repeat. Conclusion: MSE and IMSE $\sim \mathcal{O}(n^{-4/5})$

Higher Dimensions Setup: $X_1, \dots, X_n \overset{iid}{\sim} f(x_1, \dots, x_d)$. Model: $\hat{f}(\vec{x}) = \frac{1}{nh^d} \sum_{i=1}^n K\left(\frac{\vec{X} - \vec{X}_i}{h}\right)$. Properties of K: $K(\vec{u}) \geq 0$, $\int_{\mathbb{R}^d} K(\vec{u}) d\vec{u} = 1$, $\int_{\mathbb{R}^d} \vec{u} K(\vec{u}) d\vec{u} = \vec{0}$, $\int_{\mathbb{R}^d} \vec{u}\vec{u}^T K(\vec{u}) d\vec{u} = \mathbb{1}_d$. Curse of Dimensionality: Best possible MSE Rate: $\mathcal{O}\left(n^{-\frac{4}{4+d}}\right)$

4 Non Parametric Regression

Model: $Y_i = m(x_i) + \varepsilon_i$ with $\mathbb{E}[\varepsilon_i] = 0$, $\mathrm{Var}(\varepsilon_i) = \sigma_\varepsilon^2$ and $m(x) = \mathbb{E}[Y|X = x]$

4.1 Kernel Regression Estimator

$\hat{f}_X(x) = \frac{1}{nh} \sum_{i=1}^n K\left(\frac{x - x_i}{h}\right)$, $\hat{f}_{X,Y}(x, y) = \frac{1}{nh^2} \sum_{i=1}^n K\left(\frac{x-x_i}{h}\right) K\left(\frac{y - Y_i}{h}\right)$.
Hence: $m(x) = \frac{\sum_{i=1}^n K\left(\frac{x-x_i}{h}\right) Y_i}{\sum_{i=1}^n K\left(\frac{x-x_i}{h}\right)} = \frac{\sum_{i=1}^n w_i(x) Y_i}{\sum_{i=1}^n w_i(x)}$ (**Nadaraya-Watson**)

Role of h $h \to \infty \Rightarrow m(x) \approx \text{const}$, $h \to 0 \Rightarrow m(x) \approx \delta_x$.
$h_{opt} = n^{-1/5} \left(\frac{\sigma_\varepsilon^2 \int K^2(z) dz}{(m''(x) \int z^2 K(z) dz)^2}\right)^{1/5}$

Inference for underlying Reg. Curve (Hat Matrix) Def: $S : \mathbb{R}^n \to \mathbb{R}^n$, $(Y_1, \dots, Y_n) \mapsto (\hat{m}(x_1), \dots, \hat{m}(x_n)) := \hat{\vec{m}}(\vec{x}) = \hat{Y}$. Hence $\hat{Y} = SY$. Note: $S_{r,s} = w_s(x_r)$ where $w_i(x) = \frac{K\left(\frac{x-x_i}{h}\right)}{\sum_{j=1}^n K\left(\frac{x-x_j}{h}\right)}$ and $m(x) = \sum_{i=1}^n w_i(x) Y_i$. Thus: $\mathrm{Cov}\left(\hat{\vec{m}}(\vec{x})\right) = \sigma_\varepsilon^2 SS^T$, Note: $\mathrm{tr}(S) = p$ = deg. of freedom. Estim of σ_ε^2: $\hat{\sigma}_\varepsilon^2 = \frac{1}{n - df} \sum_{i=1}^n (Y_i - \hat{m}(x_i))^2$.
Hence: $\widehat{s.e.}(\hat{m}(x_i)) = \sqrt{\widehat{\mathrm{Var}}(\hat{m}(x_i))} = \hat{\sigma}_\varepsilon \sqrt{(SS^T)_{ii}}$ resulting in: $\hat{m}(x_i) \sim \mathcal{N}\left(\mathbb{E}[\hat{m}(x_i)], \sigma_\varepsilon^2 SS^T\right)$. The Conf Int for $\hat{m}$ (not m) is given as: $I = \hat{m}(x_i) \pm 1.96 \cdot \widehat{s.e.}(\hat{m}(x_i))$

4.2 Local Polynomial (LOESS)

$\hat{\beta}(x) = \mathrm{argmin}_{\beta \in \mathbb{R}^p} \sum_{i=1}^n K\left(\frac{x - x_i}{h}\right) \left(Y_i - \beta_1 - \sum_{j=2}^p \beta_j (x - x_i)^{j-1}\right)^2$.
Weighted LS: $w = \alpha \left(1 - \left(\frac{dist}{maxdist}\right)^3\right)^3$, $\alpha < 1 \Rightarrow \hat{\beta}(x) = \left(X^T W X\right)^{-1} X^T W Y$

4.3 Smoothing Splines and Penalized Regression

Penalized RSS minimize $\sum_{i=1}^n (Y_i - m(x_i))^2 + \lambda \int m''(x)^2 dx$
$\lambda = 0$: m can be any fct interpolating $\mathcal{D}$, $\lambda = \infty$: linear regression.
For $0 < \lambda < \infty$: Cubic Spline Solution: Let $a \leq x_1 \leq \cdots \leq x_n \leq b$, $g : [a, b] \to \mathbb{R}$ is a cubic spline if: a) $\forall$ Intervals $(a, x_1), \dots, (x_n, b)$ g is a cubic polynomial, b) g has two continuous derivatives on $[a, b]$. g is called "natural" if $g''(a) = g''(b) = g'''(a) = g'''(b) = 0$
Smoothing Spline Solution $m_\lambda(x) = \sum_{j=1}^n \beta_j B_j(x)$ where $B_j(.)$ are basis fcts of natural splines. Estim β with penalized RSS. Def $B \in \mathbb{R}^{n \times n}$ s.t. j-th col: $B_{.,j} = (B_j(x_1), \dots, B_j(x_n))^T$. Def $\Omega_{j,k} = \int B_j''(z) B_k''(z) dz$. Then: $\hat{\beta} = \left(B^T B + \lambda \Omega\right)^{-1} B^T Y$ and $\hat{Y} = S_\lambda Y$ where $S_\lambda = B\left(B^T B + \lambda \Omega\right)^{-1} B^T$. Remark: $S_\lambda = S_\lambda^T$, hence: $\mathrm{Eig}(S_\lambda) \subset \mathbb{R}$

5 Cross Validation

5.1 Properties of different CV-Schemas

One rand. Split into $\mathcal{D}_{\text{train}}$ and $\mathcal{D}_{\text{test}}$: Depends on <u>one</u> split, proportion $\frac{\mathcal{D}_{\text{test}}}{\mathcal{D}_{\text{train}}}$ is arbitrary, bias <u>and</u> var is poor. ok in clear cut cases, fast **LOOCV:** approx. unbiased estim for GE. $n - 1$ instead of n training samples $\Rightarrow$ slight bias. High Var: Strong correlation of $m_{n-1}^{(-i)}(.)$ and $m_{n-1}^{(-j)}(.)$ **L-d-OCV** higher Bias than LOOCV lower Var than LOOCV **K-fold CV** larger bias than LOOCV, larger Var than LdOCV, unclear if Var better than LOOCV

5.2 Computational Shortcut

Setup: fitting cubic smoothing spline or least squares param estimat. $(\hat{m}(x_1), \dots, \hat{m}(x_n))^T = S\vec{Y}$, for LOOCV: $\frac{1}{n} \sum_{i=1}^n \left(Y_i - \hat{m}_{n-1}^{(-i)}\right)^2 = \frac{1}{n} \sum_{i=1}^n \left(\frac{Y_i - \hat{m}(x_i)}{1 - S_{ii}}\right)^2$. Can compute CV score by fitting $\hat{m}(.)$ <u>once on full set</u>! Computing S_{ii} $\forall i$ takes $\mathcal{O}(n)$ operations. Generalized CV: $\frac{1}{n} \sum_{i=1}^n (Y_i - \hat{m}(x_i))^2 / \left(1 - \frac{1}{n} \mathrm{tr}(S)\right)^2$

6 Bootstrap

6.1 Non-parametric BS

$\mathbb{E}^*\left[\hat{\theta}_n^*\right] \approx \frac{1}{B} \sum_{i=1}^B \hat{\theta}_n^{*(i)}$, $\mathrm{Var}^*(\hat{\theta}_n^*) \approx \frac{1}{B-1} \sum_{i=1}^B \left(\hat{\theta}_n^{*(i)} - \mathbb{E}^*\left[\hat{\theta}_n^*\right]\right)^2$

α-quantile of distr. of $\hat{\theta}_n^* \approx \alpha$-quantile of $\hat{\theta}_n^{*(1)}, \dots, \hat{\theta}_n^{*(B)}$
Bootstrap CI: $\left[2\hat{\theta}_n - q_{1-\alpha/2}^*, 2\hat{\theta}_n - q_{\alpha/2}^*\right]$, $q_\alpha^* = \alpha$-BS-quant. of $\hat{\theta}_n^*$

6.2 Double Bootstrap

Algorithm { Repeat M times: { Draw $Z_1^*, \dots, Z_n^* \sim P_n$, Compute $\hat{\theta}^*$, a) [Compute $I^{**}(1-\alpha)$]: Repeat B times: {
Generate $Z_1^{**}, \dots, Z_n^{**} \sim P^*$, Compute $\hat{\theta}^{**}$ }
$I^{**}(1 - \alpha) = \left[2\hat{\theta} - q_{1-\alpha/2}^{**}, 2\hat{\theta} - q_{\alpha/2}^{**}\right]$
b) [Check Coverage]: $\text{cover}^{*(m)}(1 - \alpha) := \mathbb{1}_{\left[\hat{\theta} \in I^{**}(1-\alpha)\right]} \in \{0, 1\}$ }
$p^*(\alpha) = \frac{1}{M} \sum_{m=1}^M \text{cover}^{*(m)}(1 - \alpha) \approx \mathbb{P}\left[\hat{\theta} \in I^{**}(1 - \alpha)\right]$ }
Vary α to find α'^* s.t. $p^*(\alpha'^*) = 1 - \alpha$

6.3 Model based BS

Instead of resampling the data like in non-parametric BS, estimate θ by $\hat{\theta}$ with LS or MLE and then sample $Z_1^*, \dots, Z_n^* \overset{iid}{\sim} \mathbb{P}_{\hat{\theta}}$. Construct CI like in non-param BS. if n small $\Rightarrow$ non-param-BS poor, sensitive to model-misclassification

7 Classification

Goal: $\pi_j(x) = \mathbb{P}[Y = j | X = x]$ for all classes $1, \dots, J$.
Bayese Classifier $\mathcal{C}_B(x) = \mathrm{argmax}_{0 \leq j \leq J-1} \pi_j(x)$. More generally: $\mathcal{C}_B(x) = \mathrm{argmin}_{0 \leq k \leq J-1} \sum_{j=0}^{J-1} \mathcal{L}(j, k) \pi_j(x)$ ($\mathcal{L}(j, k)$ is the loss/cost of predicting k but true class being j)

7.1 Linear Discriminant Analysis (LDA)

Model: $(X|Y = j) \sim \mathcal{N}_p(\mu_j, \Sigma)$, $\mathbb{P}[Y = j] = p_j$, $\sum_{j=0}^{J-1} p_j = 1$ Hence: $\mathbb{P}[Y = j | X = x] = \pi_j(x) = \frac{f_{X|Y=j}(x) \cdot p_j}{\sum_{k=0}^{J-1} f_{X|Y=k}(x) \cdot p_k}$ Estimating the parameters: $\hat{\mu}_j = \frac{\sum_{i=1}^n x_i \mathbb{1}_{[Y_i = j]}}{\sum_{i=1}^n \mathbb{1}_{[Y_i = j]}} = \frac{1}{n_j} \sum_{i=1}^n X_i \mathbb{1}_{[Y_i = j]}$, $\hat{p}_j = \frac{n_j}{n} = \frac{1}{n} \sum_{i=1}^n \mathbb{1}_{[Y_i = j]}$
$\hat{\Sigma} = \frac{1}{n - J} \sum_{j=0}^{J-1} \sum_{i=1}^n (x_i - \hat{\mu}_j)(x_i - \hat{\mu}_j)^T \mathbb{1}_{[Y_i = j]}$
$\hat{\Sigma}_j = \frac{1}{n_j - 1} \sum_{i=1}^n (x_i - \hat{\mu}_j)(x_i - \hat{\mu}_j)^T \mathbb{1}_{[Y_i = j]}$
$\Rightarrow \hat{\mathcal{C}}_{LDA}(x) = \mathrm{argmax}_{0 \leq j \leq J-1} \hat{\delta}_j(x)$ where
$\hat{\delta}_j(x) = x^T \hat{\Sigma}^{-1} \hat{\mu}_j - \hat{\mu}_j^T \hat{\Sigma}^{-1} \hat{\mu}_j / 2 + \log(\hat{p}_j) = (x - \hat{\mu}_j / 2)^T \hat{\Sigma}^{-1} \hat{\mu}_j + \log(\hat{p}_j)$
The decision boundary is linear. Nbr of parameters: For a predictors and b groups we have $a \cdot b$ mean params, $a(a+1)/2$ CovMat params, $b - 1$ priors hence $ab + a(a+1)/2 + b - 1$ params in total

7.2 Quadratic Discriminant Analysis (QDA)

Model: $(X|Y = j) \sim \mathcal{N}_p(\mu_j, \Sigma_j)$, $\mathbb{P}[Y = j] = p_j$, $\sum_{j=0}^{J-1} p_j = 1$
$\Rightarrow \hat{\delta}_j(x) = -\log\left(\det\left(\hat{\Sigma}_j\right)\right)/2 - (x - \hat{\mu}_j)^T \hat{\Sigma}_j^{-1} (x - \hat{\mu}_j)/2 + \log(\hat{p}_j)$
Nbr of params: ab mean params, $ba(a+1)/2$ CovMat params, $b - 1$ priors, hence $ab + ab(a+1)/2 + b - 1$ params in total

7.3 Logistic Regression

7.3.1 Binary Classification

$\pi(x) = \mathbb{P}[Y = 1 | X = x]$ and $\mathbb{P}[Y = 0 | X = x] = 1 - \pi(x)$. Model: $\log\left(\frac{\pi(x)}{1 - \pi(x)}\right) = g(x)$

Linear Logistic Regression $g(x) = \sum_{j=1}^p \beta_j x_j$. Use MLE to estimate params. Assume $Y_1, \dots, Y_n \overset{iid}{\sim}$ Bernoulli$(\pi(x))$: $\mathcal{L} = \prod_{i=1}^n \pi(x_i)^{Y_i} (1 - \pi(x_i))^{1 - Y_i} \Rightarrow -\log(\mathcal{L}) = -\sum_{i=1}^n \left[Y_i \sum_{j=1}^p \beta_j x_{ij} - \log\left(1 + \exp\left(\sum_{j=1}^p \beta_j x_{ij}\right)\right)\right]$

7.3.2 Multiclass Case

Encode multiclass problem into J binary class problems: $Y_i^{(j)} = \mathbb{1}_{[Y_i = j]}$. Now run log reg on each class: $\hat{\pi}_j(x) = \frac{\exp\left(\sum_{r=1}^p \hat{\beta}_r^{(j)} x_r\right)}{1 + \exp\left(\sum_{r=1}^p \hat{\beta}_r^{(j)} x_r\right)}$
$\tilde{\pi}_j(x) = \frac{\hat{\pi}_j(x)}{\sum_{k=0}^{J-1} \hat{\pi}_k(x)} \Rightarrow \hat{\mathcal{C}}(x) = \mathrm{argmax}_{0 \leq j \leq J-1} \tilde{\pi}_j(x)$

7.4 ROC Curve

Binary Cl.: $\hat{Y}(x) = 1$ iff $\hat{\pi}(x) = \hat{\mathbb{P}}[Y = 1 | X = x] > \theta$ for a given $\theta \in [0, 1]$. Def: Sensitivity(θ) = TPR = $\frac{\text{TP}}{\text{P}}$, Specificity$(\theta)$ = $\frac{\text{TN}}{\text{N}}$, 1-sp$(\theta)$ = FPR = $\frac{\text{FP}}{\text{N}}$. ROC Curve: Plot Sensitivity vs. 1-Specificity (TPR vs. FPR)

Precision	$\frac{\#TP}{\#\{\hat{y} = +1\}} = \mathbb{P}_n(y = 1 \mid \hat{y} = 1)$	FDR (1 - Precision)	$\frac{\#FP}{\#\{\hat{y} = +1\}} = \mathbb{P}_n(y = -1 \mid \hat{y} = +1)$
Recall (TPR, power)	$\frac{\#TP}{\#\{y = +1\}} = \mathbb{P}_n(\hat{y} = 1 \mid y = 1)$	FPR (type I error)	$\frac{\#FP}{\#\{y = -1\}} = \mathbb{P}_n(\hat{y} = +1 \mid y = -1)$

8 Flexible Regr. and Class. Methods

8.1 Additive Models

Model: $g_{add}(x) = \mu + \sum_{j=1}^{p} g_j(x_j)$ with $\mu \in \mathbb{R}$, $g_j : \mathbb{R} \to \mathbb{R}$, $\mathbb{E}[g_j(x_j)] = 0 \ \forall j = 1, \dots, p$. g_j's are non-parametric. Not affected by curse of dimensionality.

Backfitting Def: $S_j : (U_1, \dots, U_n)^T \mapsto (\hat{U}_1, \dots, \hat{U}_n)^T$. The index j means the smoothing is done against the j-th predictor/parameter.
Algorithm: 1) $\hat{\mu} := \frac{1}{n}\sum_{i=1}^{n} Y_i$ and $g_j(.) = 0 \ \forall j = 1, \dots, p$
2) Cycle through indices: $j = 1, \dots, p, 1, \dots, p, 1, \dots$ while computing: $\hat{g}_j = S_j\left(\vec{Y} - \hat{\mu}\mathbb{1} - \sum_{k \neq j} \hat{g}_k\right)$ where $\hat{g}_j = (\hat{g}_j(X_{1j}), \dots, \hat{g}_j(X_{nj}))^T$ and stop if $\frac{\|\hat{g}_{j,new} - \hat{g}_{j,old}\|_2}{\|\hat{g}_{j,old}\|_2} \le$ tol (e.g. tol=10^{-6})
3) Normalize: $\tilde{g}_j(.) = \hat{g}_j(.) - \frac{1}{n}\sum_{i=1}^{n} \hat{g}_j(X_{ij})$

8.2 Neural Networks

Activation Functions: 1) Softmax: $\hat{\pi}\left(Y = k | \vec{X} = \vec{x}\right) = \frac{\exp(g_k(\vec{x}))}{\sum_j \exp(g_j(\vec{x}))}$,
2) Sigmoid: $\phi(t) = \frac{e^t}{1+e^t} = \frac{1}{1+e^{-t}}$, 3) ReLU: $\phi(t) = \max\{0, t\}$

8.3 Classification & Regression Trees (CART)

Model: $g_{tree}(\vec{x}) = \sum_{r=1}^{M} \beta_r \mathbb{1}_{[\vec{x} \in \mathcal{R}_r]}$ is a piecewise constant fct. with $\mathcal{P} = \mathcal{R}_1 \sqcup \dots \sqcup \mathcal{R}_M$ is a partition of $\mathbb{R}^p$.

Parameter Estimation Regression and Binary Classification: $\hat{\beta}_r = \frac{\sum_{i=1}^{n} Y_i \mathbb{1}_{[\vec{x}_i \in \mathcal{R}_r]}}{\sum_{i=1}^{n} \mathbb{1}_{[\vec{x}_i \in \mathcal{R}_r]}}$, J Class Problem: $\hat{\beta}_r^j = \frac{\sum_{i=1}^{n} \mathbb{1}_{[Y_i = j]} \mathbb{1}_{[\vec{x}_i \in \mathcal{R}_r]}}{\sum_{i=1}^{n} \mathbb{1}_{[\vec{x}_i \in \mathcal{R}_r]}}$
Search Algorithm Restict partition $\mathcal{P}$ of $\mathbb{R}^p$ to axes parallel rectangles $\mathcal{R}_r$. 1) $M = 1$, $\mathcal{P} = \{\mathcal{R}\} = \{\mathbb{R}^p\}$, 2) Refine $\mathcal{R}$ into $\mathcal{R}_{left} \sqcup \mathcal{R}_{right}$ along one of the p dimensions s.t. $-\log(\mathcal{L})$ is maximally reduced. Update $\mathcal{P} = \{\mathcal{R}_1, \mathcal{R}_2\} = \{\mathcal{R}_{left}, \mathcal{R}_{right}\}$, 3) Refine $\mathcal{P}$ by finding $\mathcal{R}_k$ and its split s.t. $-\log(\mathcal{L})$ is maximally reduced and update again: $\mathcal{P} = \mathcal{P}_{old} \backslash \mathcal{R}_k \cup \{\mathcal{R}_{k,1}, \mathcal{R}_{k,2}\}$, 4) Iterate step 3 for large nbr M , 5) Prune the tree until reasonable size.
Tree representation Select "best" tree by applying "cost complexity (=cp) pruning". Penalized goodness of fit: $R_\alpha(\tau) = R(\tau) + \alpha \cdot \text{size}(\tau)$. Here, size$(\tau)$ is the number of leaves in the tree τ and $R(.)$ is quality of fit measure. The **best tree**: $\tau(\alpha) := \text{argmin}_{\tau \subset \tau_M} R_\alpha(\tau)$. $\{\tau(\alpha) | \alpha \in [0, \infty)\}$ is a nested set and the same as the subsets of $\tau_M \supseteq \dots \supset \tau_\varnothing$. For model selection we need to select best α or its normalization $cp = \frac{\alpha}{R(\tau_\varnothing)}$ using CV.

1SE Rule Take smallest tree s.t. its error is at msot one std error larger than the minimal one.
Pros and Cons Greedy-tree-type altorithm produces unstable splits ⇒ if one split is "wrong", everything below it will be "wrong".

8.4 Random Forests

Algorithm 1) Draw n_{tree} BS samples from original $\mathcal{D}$. 2) $\forall$ BS samples grow an unpruned classif./regr. tree (maybe use nodesize to lower bound the #datapoints per node) s.t. at each node randomly sample m_{try} of the pred. var. and chose best split from among these vars. 3) Pred. new data by aggregating predictions of the n_{tree} trees (majority vote for classif. and average for regr.)
Remark Bagging: $m_{try} = p$
Estimation of error 1) At each BS iteration predict on OOB data. 2) Aggregate OOB pred. ⇒ Calculate error rate

9 Reproducing Kernel Hilbert Spaces

Def (Kernel) Let $\mathcal{X} \subseteq \mathbb{R}^d$. We call $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ a kernel iff $\forall m \ \forall x_1, \dots, x_m \in \mathcal{X}$: $K \in \mathbb{R}^{m \times m}$ with $K_{ij} := k(x_i, x_j)$ is psd and symm. (psd: $\forall c \in \mathbb{R}^m : c^T K c \ge 0$, symm: $\forall x, y \in \mathcal{X} : k(x, y) = k(y, x)$)
Def (RKHS) Let $\mathcal{H}$ be a hilbert space of fcts $f : \mathcal{X} \to \mathbb{R}$. $\mathcal{H}$ is a Reproducing Kernel Hilbert Space (RKHS) iff $\exists k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ s.t. $\forall x \in \mathcal{X} : k(x, .) \in \mathcal{H}$ and $\forall f \in \mathcal{H} \forall x \in \mathcal{X} : \langle f(.), k(x, .) \rangle = f(x)$
Median Heuristic Gaussian kernel: $2\sigma^2 = \text{median}\left(\|x_i - x_j\|^2\right)_{i \neq j}$

9.1 Support Vector Machines (SVM)

Assume $\mathcal{D}$ is linearly seperable. Goal: separate $\mathcal{D}$ into two classes with a hyperplane: find $w \in \mathbb{R}^d$ and $b \in \mathbb{R}$ s.t. $\min_{i \in \{1,\dots,n\}} |\langle w, x_i \rangle + b| = 1$ (canonical form). The distance of the hyperplane to the closest x_i is called the **margin**. If in canonical form: margin= $\frac{1}{\|w\|_2}$.

Algorithm Soft/Hard SVM: $\min_{w,b} \frac{1}{2}\|w\|_2^2 + c \cdot \sum_{i=1}^{n} \xi_i$ s.t. $\forall i \in \{1, \dots, n\} \ y_i(\langle w, x_i \rangle + b) \ge 1 - \xi_i$.

10 Bagging & Boosting

10.1 Bagging

Consider a tree algorithm yielding $\hat{g}(.) : \mathbb{R}^p \to \mathbb{R}$.
Algorithm 1) Generate BS samples: $(X_1^*, Y_1^*), \dots, (X_n^*, Y_n^*)$ and compute $\hat{g}^*(.)$. 2) Repeat 1 B times: $\hat{g}^{*1}(.), \dots, \hat{g}^{*B}(.)$. 3) Aggregate: $\hat{g}_{Bag}(.) = \frac{1}{B}\sum_{i=1}^{B} \hat{g}^{*i}(.) \approx \mathbb{E}^*[\hat{g}]^*(.)$

Remark $\mathbb{E}\left[\left(Y - \hat{f}_{Bag}(X)\right)^2\right] \le \mathbb{E}\left[\left(Y - \hat{f}^*(X)\right)^2\right] = \mathbb{E}\left[\left(Y - \hat{f}_{Bag}(X)\right)^2\right] + \underbrace{\mathbb{E}\left[\left(\hat{f}_{Bag}(X) - f^*(X)\right)^2\right]}_{=Var(f^*(X))}$

Bagging for trees Setup: Regr.Trees s.t. exactly m training samples are in each leaf node. Model: $\hat{Y}(x) = \sum_{i=1}^{n} w_i Y_i$ where $w_i = \frac{1}{m}\mathbb{1}_{\{x_i \& x \text{ in same leaf}\}}$. Hence: $\hat{Y}_{Bag}(x) = \sum_{i=1}^{n}\left(\frac{1}{B}\sum_{b=1}^{B} w_i^{*b}\right) Y_i = \sum_{i=1}^{n} w_{bag,i} Y_i$ where $w_i^{*b} \in \left\{0, \frac{1}{m}\right\}$.
Remark $\text{Var}\left(\hat{Y}_{bag}(X)\right) = \sigma^2 \sum_{i=1}^{m} w_{bag,i}^2 \le \sigma^2 \sum_{i=1}^{m} w_i^2 = \text{Var}\left(\hat{Y}(x)\right)$ Hence Bagging is a Variance reducing technique.

10.2 Boosting

Boosting is a bias reducing technique.
AdaBoost $g : \mathbb{R}^p \to \{-1, 1\}$, $Y \in \{-1, 1\}$, Idea: upweight observations previous model got wrong.
1) Initialize obs. weights $w_i = \frac{1}{N} \ \forall i = 1, \dots, N$
2) For $m = 1, \dots, M$: a) Fit classifier $\hat{g}_m(.)$ using w_i, b) Compute weighted error: $err_m = \frac{\sum_{i=1}^{N} w_i \mathbb{1}_{\{Y_i \neq \hat{g}_m(x_i)\}}}{\sum_{i=1}^{N} w_i}$, c) Compute $\alpha_m = \log\left(\frac{1-err_m}{err_m}\right)$, d) Set weights: $w_i = w_i \cdot \exp\left(\alpha_m \cdot \mathbb{1}_{\{Y_i \neq \hat{g}_m(x_i)\}}\right)$
3) Final Model: $\hat{G}(x) = \text{sign}\left(\sum_{m=1}^{M} \alpha_m \hat{g}_m(x)\right)$

Gradient Boosting Goal: $G(x) = g_0(x) + \sum_{m=1}^{M} \gamma \cdot g_m(x)$
1) Initialize $G(x) = g_0(x)$
2) For $m = 1, \dots, M$: a) $\forall i = 1, \dots, N$: $r_{im} = -\frac{\partial \mathcal{L}(y_i, G(x_i))}{\partial G(x_i)}$, b) fit Model $g_m(x_i)$ to r_{im}, c) set $G(x) = G(X) + \gamma g_m(x)$ for $\gamma \in (0, 1]$

11 Random Additional Material

Cook's Distance: Measure of Influence of a datapoint. $D_i := \frac{\sum_{j \neq i}\left(\hat{Y}_j - \hat{Y}_j^{-i}\right)^2}{p \cdot \|Y - \hat{Y}\|^2 / (n-p)}$, $\hat{Y}_j^{-i}$ is the fitted value of j when disregarding point i during fitting. $D_i > 0$ means x_i is very influential.

12 R Code

Function	Description
`solve()`	invert Matrix
`t()`	transpose Matrix
`%*%`	matrix multiplication
`df[,-c(6)]`	remove column 6 from df
`seq(1,40,1)`	generate sequence of evenly spaced values
`rep(1,7)`	create a ones-vector of length 7
`rnorm(n)`	generate n random numbers based on normal distr
`rgamma()`	generate random numbers based on gamma distr
`factor()`	apply to a vector if it is categorical to be able to perform regression tasks
`which.max()`	returns index of maximal entry in a vector (/matrix)
`as.formula()`	takes a text ("y~.") as input and stores it as a formula
`scale(mat)`	center and scale columns of a matrix / df
`fitted(fit)`	returns fitted values of a model
`resid(fit)`	returns residuals of a model
`boxplot(a[])`	creates boxplot of vector a
`quantile(...)`	x=data, probs=0.75 (e.g.): computes 75% quantile of x
`predict(...)`	args: `fit` (fitted model), `type`: e.g. `"response"`
`gam()`	package `mgcv`; generalized additive model. Used for adaptive models: e.g. smoothing-spline for log reg
`nnet()`	fits a feedforward NN
`ppr()`	projection pursuit regression (extension of additive models)
`rpart()`	package `rpart`; used for fitting classification and regressino trees; `type="class"` if classification tree
`sd()`	compute standard deviation
`glm`	package `glmnet`; generalized linear models (e.g. linear logistic regression)

Function	Description
`optim()`	can maximize/minimize a function
`choose(n,k)`	Binom Coeff
`plotcp()`	choose optimal tree pruning parameters (also `plotcp()`)
`prune.rpart()`	do pruning on tree
`density()`	density distribution; use `bw="SJ"` to get iteratively found optimal bandwidth
`ksmooth()`	Nadaraya-Watson kernel regression estimate; Returns Vector
`smooth.spline()`	smooth spline estimator
`loess()`	local polynomial regression
`boot.ci()`	`boot.ci(boot.out=res.boot,conf=0.95, type=c("basic","norm","perc"))` computes the reversed, normal approx. and quantile-quantile of the bootstrap
`coefficients()`	get coefficients of a fit, (e.g. `reg <- lm(y x)` and then `coefficients(reg)[2]`)
`pairs()`	make a pair-plot
`$sigma`	e.g. `summary(fit5)$sigma`
`prp()`	plot a tree
`svm`	Kernel SVM
`med.heur <-`	`1/median(dist(iris[samp,1:4])∧2)`
`multinom`	(e.g. `multinom(Species .,data=Iris)`) Multinomial regression

Listing 1: Theoretical True distribution

```
X <- cbind(1, x)                    ## design matrix
XtXinv <- solve(crossprod(X))       ## theoretical s.d.
tsd <- sqrt(5^2*XtXinv[2,2])
```

Listing 2: Backward/Forward Selection

```
mortal.full <- lm(Mortality~.,data=mortality)
mortal.empty <- lm(Mortality~1,data=mortality)
mortal.bw <- step(mortal.full,dir="backward")
mortal.fw <- step(mortal.empty,dir="forward",scope=list(
    upper=mortal.full,lower=mortal.empty))
library(leaps)
mortal.alls <- regsubsets(Mortality~.,data=mortality)
p.regsubsets(mortal.alls,cex=0.8,cex.main=.8)
```

Listing 3: Non-parametric Regression

```
bmwloess <- loess(y~x)        # local polynomial
dgf <- bmwloess$trace.hat     # degrees of freedom
bmwss <- smooth.spline(x,y,df=dgf)  # smooth spline
ox <- order(x)                # k-smooth destroyes order
bmwks <- ksmooth(x,y,kernel="normal",bandwidth=h,x.
    points=x) # Nadaraye-Watson
bmwks$x <- bmwks$x[order(ox)]
bmwks$y <- bmwks$y[order(ox)]
plot(x,y)
lines(x_new,bmwks$y)
lines(x_new,predict(bmwloess,newdata=data.frame(x=x_new)
    ))
llines(x_new,predict(bmwss,x=x_new)$y)
```

Listing 4: Hat Matrix

```
Snw <- Slp <- Sss <- matrix(0, nrow = n, ncol = n)
## The j-th column is given by S_j = Snw[,j]
In <- diag(n)
for(j in 1:n) {
    Snw[,j] <- ksmooth(x, In[,j], kernel = "normal",
        bandwidth = 0.2,x.points = x)$y
}
df.NW <- sum(diag(Snw))
## Getting the span parameter for loess and spar
    parameter for smooth.spline such that the degrees of
     freedom are (approximately) the same with the ones
    for Nadaraya-Watson estimator
dflp <- function(span, val) {
    for(j in 1:n)
        Slp[,j] <- loess(In[,j] ~ x, span = span)$fitted
    sum(diag(Slp)) - val
}
span <- uniroot(dflp, c(0.2, 0.5), val = df.NW)$root
for(j in 1:n) {
    Slp[,j] <- predict(loess(In[,j] ~ x, span = span),
        newdata = x)
    Sss[,j] <- predict(smooth.spline(x, In[,j], df = df.
        NW), x = x)$y
}
spar <- smooth.spline(x, In[,1], df = df.NW)$spar
...
estnw[,i] <- ksmooth(x, y, kernel = "normal", bandwidth
    = 0.2, x.points = x)$y
sigmanw <- sum((y - estnw[,i])^2) / (length(y) - sum(
    diag(Snw)))
senw[,i] <- sqrt(sigmanw * diag(Snw %*% t(Snw)))
...
```

Listing 5: Coverage Function

```
coverage <- function(x,est,se) {
    pos <- x == 0.5
    pw <- sum(abs(est[pos,] - m(x)[pos]) <= 1.96*se[pos
        ,]) # pointwise coverate
    simult <- sum(apply(abs(est - m(x)) <= 1.96 * se, 2,
        all)) # simultaneous coverage }
```

Listing 6: Confidence Intervals

```
newcountry <- data.frame(l2tv=log2(50),l2dr=log2(3000))
predict(fit,newdata=newcountry,interval="confidence")
```

Listing 7: LOOCV

```
loocv <- function(reg.data, reg.fcn){
    loo.reg.value <- function(i, reg.data, reg.fcn)
        return(reg.fcn(reg.data$x[-i], reg.data$y[-i],
            reg.data$x[i]))
    n <- nrow(reg.data)
    loo.values <- sapply(1:n, loo.reg.value, reg.data,
        reg.fcn)
    mean((reg.data$y - loo.values)^2)
}
```

Listing 8: CV

```
h <- 4
reg.fcn.nw <- function(reg.x, reg.y, x)
    ksmooth(reg.x, reg.y, x.points = x, kernel = "normal
    ", bandwidth = h)$y
(cv.nw <- loocv(reg, reg.fcn.nw))
n <- nrow(reg)
Id <- diag(n)
S.nw <- matrix(0, n, n)
for (j in 1:n)
    S.nw[, j] <- reg.fcn.nw(reg$x, Id[, j], reg$x)
(df.nw <- sum(diag(S.nw)))
y.fit.nw <- reg.fcn.nw(reg$x, reg$y, reg$x)
(cv.nw.hat <- mean(((reg$y - y.fit.nw)/(1 - diag(S.nw)))
    ^2))
library(sfsmisc)
hatMat(reg$x,trace=TRUE,pred.sm=reg.fcn.nw,x=reg$x)
S.nw.hatMat <- hatMat(reg$x,trace=FALSE,pred.sm=reg.fcn.
    nw,x=reg$x)
(cv.nw.hatMat <- mean(((reg$y - y.fit.nw)/(1 - diag(S.nw
    .hatMat)))^2))
...
est.ssopt <- smooth.spline(reg$x, reg$y, cv = TRUE)
cv.ssopt <- est.ssopt$cv.crit
```

Listing 9: Bootstrapping

```
tIQR <- function(x, ind) IQR(x[ind])
require("boot")
res.boot <- boot(data = sample40, statistic = tIQR, R =
    10000) # sim="ordinary"
bci <- boot.ci(res.boot, conf = 0.95, type = c("basic",
    "norm", "perc"))
```

Listing 10: LDA, QDA, logistic regression and ROC Curve

```
class_lda <- lda(x = Iris[, c("Petal.Length", "Petal.
    Width")], grouping = Iris[, "Species"])
predplot(class_lda, Iris, main = "Cl. w/ LDA")
class_qda <- qda(x = Iris[ ,c("Petal.Length", "Petal.
    Width")],grouping = Iris[, "Species"])
predplot(class_qda, Iris, main = "Cl. w/ QDA")
## Use function multinom to fit data
class_multinom <- multinom(Species ~ . , data = Iris)

require(ROCR)
fit <- glm(Survival ~ ., data = d.baby, family = "
    binomial")
pred <- prediction(fit$fitted.values, d.baby$Survival)
perf <- performance( pred, "tpr", "fpr" )
plot(perf, main = "ROC")
perf.cost <- performance(pred, "cost")
plot(perf.cost, main = title)
```

Listing 11: GAMs

```
require(sfsmisc)
form5 <- wrapFormula(logupo3~., data = d.ozone.e,
    wrapString="poly(*,degree=5)")
fit5 <- lm(form5, data = d.ozone.e)
require(mgcv)
gamForm <- wrapFormula(logupo3~., data = d.ozone.e)
g1 <- gam(gamForm, data = d.ozone.e)
# Formula: logupo3 ~ s(vdht) + s(wdsp) + s(hmdt) + s(
    sbtp) + s(ibht) + s(dgpg) + s(ibtp) + s(vsty) + s(
    day)
```

Listing 12: Trees and Forests

```
rp.veh0 <- rpart(Class~.,data=d.vehicle,control=rpart.
    control(cp=0.0,minsplit=2))
preds = predict(rp.veh0,newdata=d.vehicle,type="class")
table(d.vehicle$Class, preds)
cp.opt <- rp.veh$cptable[7, "CP"]
rp.veh.pruned <- prune.rpart(rp.veh, cp = cp.opt)
library(randomForest)
rf.model1 <-randomForest(factor(Class)~.,data=d.vehicle)
```

Listing 13: Ridge and Lasso

```
require(glmnet)
f.ridge <- glmnet(X, Y, alpha=0)
f.lasso <- glmnet(X, Y, alpha=1)
cv.eln <- cv.glmnet(X, Y, alpha=0.5, nfolds=10)
lambda.min <- cv.eln$lambda.min
lambda.1se <- cv.eln$lambda.1se
plot(log(f.lasso$lambda),apply(coef(f.lasso), 2,
    function(x) sum(x != 0)), type = "l", ylab = "Nbr
    selected preds") # could also use f.lasso$beta for
    apply(...)
first.lam.ind = min(which(f.lasso$df >= 16))
# coef(f.lasso)[, first.lam.ind] gets coeffs for this
    lambda
names(which(coef(f.lasso)[, first.lam.ind] != 0))
```