

Differentiation of vectors and matrices

- Def.: Derivative of a (scalar) function with respect to an $N \times 1$ vector is the $N \times 1$ vector of derivatives with resp. to each element.
- Def.: Derivative of a (scalar) function with respect to an $1 \times N$ vector is the $1 \times N$ vector of derivatives with resp. to each element.

$$x = (x_1, \dots, x_N) \implies \frac{\partial f}{\partial x} = \boxed{\quad}$$

- ⇒ Derivative of scalar product: a and x both are $N \times 1$:

$$\frac{\partial}{\partial x} (a^T \cdot x) = \boxed{\quad} \quad (\text{where } {}^T \text{ denotes transpose}).$$

(Generalizes well known scalar result, $\partial(b \cdot y)/\partial y = b$.)

- Def.: Derivative of $M \times 1$ vector w.r.t. $N \times 1$ vector is $M \times N$ matrix of derivatives.

Differentiation of vectors and matrices

- Def.: Derivative of a (scalar) function with respect to an $N \times 1$ vector is the $N \times 1$ vector of derivatives with resp. to each element.
- Def.: Derivative of a (scalar) function with respect to an $1 \times N$ vector is the $1 \times N$ vector of derivatives with resp. to each element.

$$x = (x_1, \dots, x_N) \implies \frac{\partial f}{\partial x} = \left(\frac{\partial f}{\partial x_1}, \dots, \frac{\partial f}{\partial x_N} \right).$$

- ⇒ Derivative of scalar product: a and x both are $N \times 1$:

$$\frac{\partial}{\partial x} (a^T \cdot x) = a^T \quad (\text{where } {}^T \text{ denotes transpose}).$$

(Generalizes well known scalar result, $\partial(b \cdot y)/\partial y = b$.)

- Def.: Derivative of $M \times 1$ vector w.r.t. $N \times 1$ vector is $M \times N$ matrix of derivatives.

Differentiation of vectors and matrices, contd.

- \Rightarrow Derivative of matrix-vector product: A is $M \times N$, x is $N \times 1$:

$$\frac{\partial}{\partial x}(Ax) = \boxed{\quad}$$

- \Rightarrow Derivative of quadratic form: A is $N \times N$, x is $N \times 1$:

$$\frac{\partial}{\partial x}(x^T Ax) = \boxed{\quad}$$

(Generalizes scalar $\partial(b \cdot y^2)/\partial y = 2b \cdot y$.)

- \Rightarrow Symmetric version of same: $A = A^T$ is $N \times N$ symmetric:

$$\frac{\partial}{\partial x}(x^T Ax) = \boxed{\quad}$$

- Def.: Boldface **1** denotes vector of ones: $\mathbf{1} = (1, \dots, 1)^T$.

Differentiation of vectors and matrices, contd.

- \Rightarrow Derivative of matrix-vector product: A is $M \times N$, x is $N \times 1$:

$$\frac{\partial}{\partial x}(Ax) = A.$$

- \Rightarrow Derivative of quadratic form: A is $N \times N$, x is $N \times 1$:

$$\frac{\partial}{\partial x}(x^T Ax) = x^T(A + A^T).$$

(Generalizes scalar $\partial(b \cdot y^2)/\partial y = 2b \cdot y$.)

- \Rightarrow Symmetric version of same: $A = A^T$ is $N \times N$ symmetric:

$$\frac{\partial}{\partial x}(x^T Ax) = 2x^T A.$$

- Def.: Boldface **1** denotes vector of ones: $\mathbf{1} = (1, \dots, 1)^T$.

Introduction: the high dimensional issue

Major issue: $X^T X$ is **not invertible**, infinitely many solutions!

Some possible directions:

-
-



Introduction: the high dimensional issue

Major issue: $X^T X$ is **not invertible**, infinitely many solutions!

Some possible directions:

- **dimension reduction** (reducing p to be smaller than n),
 - ▶ remove variables having low correlation with response;
 - ▶ more formal subset selections;
 - ▶ select a few “best” linear combinations of variables;
- **shrinkage methods** (adding constraint to β),
 - ▶ ridge regression;
 - ▶ lasso (least absolute shrinkage and selection operator)
 - ▶ elastic net.

Two simple approaches to prediction: linear regression model

The linear regression model

$$\begin{aligned} Y &= \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p + \varepsilon \\ &= X\beta + \varepsilon, \end{aligned} \quad \text{where } X = (\mathbf{1}, x_1, \dots, x_p),$$

can be used to predict the outcome y given the values x_1, x_2, \dots, x_p , namely

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x_1 + \hat{\beta}_2 x_2 + \cdots + \hat{\beta}_p x_p$$

Properties:

-
-
-
-
-

Two simple approaches to prediction: linear regression model

The linear regression model

$$\begin{aligned} Y &= \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p + \varepsilon \\ &= X\beta + \varepsilon, \end{aligned} \quad \text{where } X = (\mathbf{1}, x_1, \dots, x_p),$$

can be used to predict the outcome y given the values x_1, x_2, \dots, x_p , namely

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x_1 + \hat{\beta}_2 x_2 + \cdots + \hat{\beta}_p x_p$$

Properties:

- easy interpretation;
- easy computations involved;
- theoretical properties available;
- it works well in many situations.

Two simple approaches to prediction: least square

How do we **fit** the linear regression model to a training dataset?

- Most popular method:
- estimate β by minimizi

$$(y - X\beta)$$

where X is a $(N \times p)$ matrix and y a N -dimensional vector.

$$\hat{\beta} =$$

Two simple approaches to prediction: least square

How do we **fit** the linear regression model to a training dataset?

- Most popular method: **least square**;
- estimate β by minimizing the **residual sum of squares**

$$\text{RSS}(\beta) = \sum_{i=1}^N (y_i - x_i^T \beta)^2 = (y - X\beta)^T (y - X\beta)$$

where X is a $(N \times p)$ matrix and y a N -dimensional vector.

Differentiating w.r.t. β , we obtain the **estimating equation**

$$X^T(y - X\beta) = 0,$$

from which, when $(X^T X)$ is non-singular, we obtain

$$\hat{\beta} = (X^T X)^{-1} X^T y$$

Two simple approaches to prediction: least square for binary response

Simulated data with two variables and two classes:

$$Y = \begin{cases} 1 & \text{orange} \\ 0 & \text{blue} \end{cases}$$

If $Y \in \{0, 1\}$ is treated as a numerical response

$$\hat{Y} = \hat{\beta}_0 + \hat{\beta}_1 x_1 + \hat{\beta}_2 x_2,$$

a prediction rule

$$\hat{G} = \begin{cases} 1 \text{ (orange)} & \text{if } \hat{Y} > 0.5 \\ 0 \text{ (blue)} & \text{otherwise} \end{cases}$$

gives linear decision boundary $\{x^T \hat{\beta} = 0.5\}$

- optimal under [redacted]
- is it better with nonlinear decision boundary?

Two simple approaches to prediction: least square for binary response

Simulated data with two variables and two classes:

$$Y = \begin{cases} 1 & \text{orange} \\ 0 & \text{blue} \end{cases}$$

If $Y \in \{0, 1\}$ is treated as a numerical response

$$\hat{Y} = \hat{\beta}_0 + \hat{\beta}_1 x_1 + \hat{\beta}_2 x_2,$$

a prediction rule

$$\hat{G} = \begin{cases} 1 \text{ (orange)} & \text{if } \hat{Y} > 0.5 \\ 0 \text{ (blue)} & \text{otherwise} \end{cases}$$

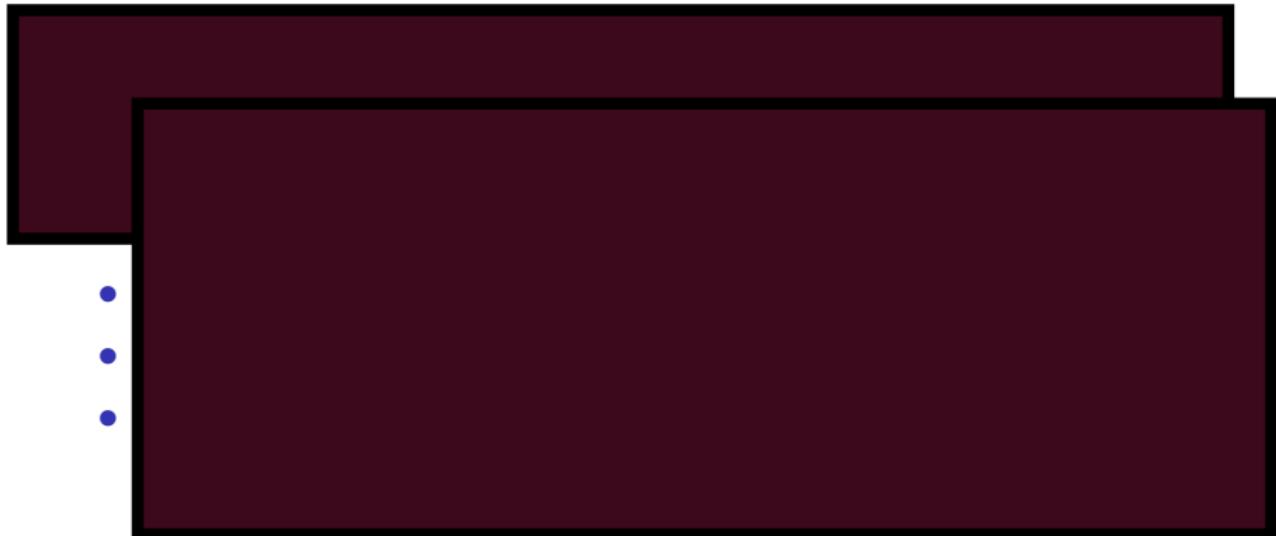
gives linear decision boundary $\{x^T \hat{\beta} = 0.5\}$

- optimal under Gaussian assumptions;
- is it better with nonlinear decision boundary?

Two simple approaches to prediction: Nearest neighbor methods

A different approach consists in looking at the **closest** (in the input space) **observations** to x and, based on their output, form $\hat{Y}(x)$.

The **k nearest neighbors prediction** of x is the mean



Two simple approaches to prediction: Nearest neighbor methods

A different approach consists in looking at the **closest** (in the input space) **observations** to x and, based on their output, form $\hat{Y}(x)$.

The **k nearest neighbors prediction** of x is the mean

$$\hat{Y}(x) = \frac{1}{k} \sum_{i:x_i \in N_k(x)} y_i,$$

where $N_k(x)$ contains the k closest points to x .

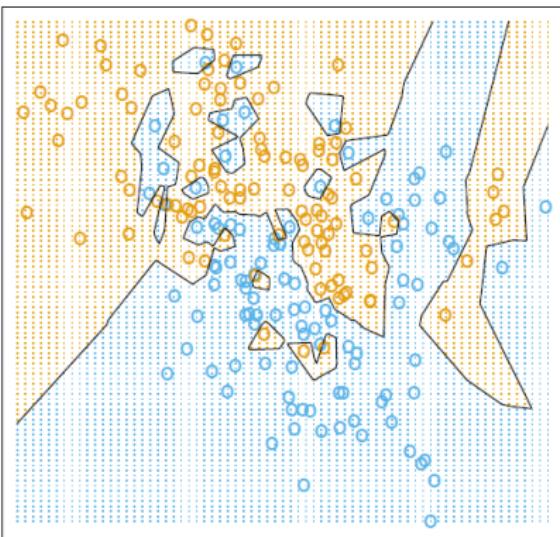
- **less assumptions** on $f(x)$;
- we need to decide k ;
- we need to define a metric (for now, consider the Euclidean distance).

Two simple approaches to prediction: nearest neighbor methods

Using the same data as before: Note:

$$Y = \begin{cases} 1 & \text{orange} \\ 0 & \text{blue} \end{cases}$$

- same approach, with $k = 1$;
- no training observations are missclassified!!!
- Is this a good solution?

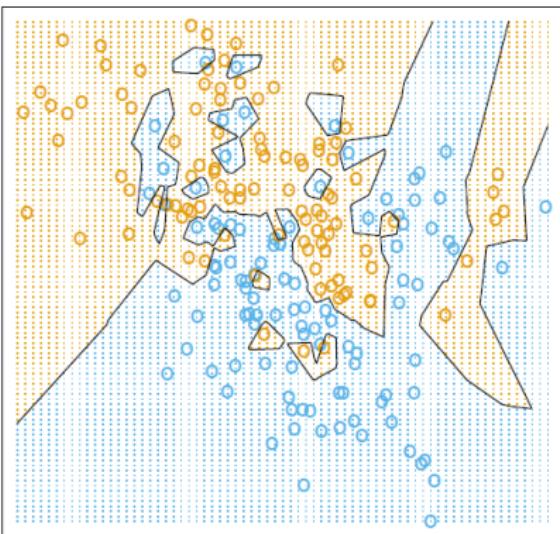


- bias-

Two simple approaches to prediction: nearest neighbor methods

Using the same data as before: Note:

$$Y = \begin{cases} 1 & \text{orange} \\ 0 & \text{blue} \end{cases}$$



- same approach, with $k = 1$;
- no training observations are missclassified!!!
- Is this a good solution?
 - ▶ the learner works greatly on the training set, but what is its prediction ability? (remember this term: **overfitting**);
 - ▶ It would be preferable to evaluate the performance of the methods in an independent set of observations (**test set**);
- bias-variance trade-off.

Statistical decision theory: theoretical framework

Statistical decision theory gives a mathematical framework for finding the optimal learner.

Let:

-
-
-

Our goal is to f

- we need a $f(X)$ where
- ▶ example

Statistical decision theory: theoretical framework

Statistical decision theory gives a mathematical framework for finding the optimal learner.

Let:

- $X \in \mathbb{R}^p$ be a p-dimensional random vector of inputs;
- $Y \in \mathbb{R}$ be a real value random response variable;
- $p(X, Y)$ be their joint distribution;

Our goal is to find a function $f(X)$ for predicting Y given X :

- we need a loss function $L(Y, f(X))$ for penalizing errors in $f(X)$ when the truth is Y ,
 - ▶ example: squared error loss, $L(Y, f(X)) = (Y - f(X))^2$.

Statistical decision theory: expected prediction error

Given $p(X, Y)$, it is possible to derive the **expected prediction error** of $f(X)$:



we have now a criterion for choosing a learner: find f which



is by far the most common and convenient loss function. Let us focus on it!

Statistical decision theory: expected prediction error

Given $p(X, Y)$, it is possible to derive the **expected prediction error** of $f(X)$:

$$\text{EPE}(f) = E [L(Y, f(X))] = \int_{x,y} L(y, f(x))p(x, y)dxdy;$$

we have now a criterion for choosing a learner: find f which **minimizes** $\text{EPE}(f)$.

The aforementioned squared error loss,

$$L(Y, f(X)) = (Y - f(X))^2,$$

is by far the most common and convenient loss function. Let us focus on it!

Statistical decision theory: squared error loss

If $L(Y, f(X)) = (Y - f(X))^2$, then

$$\begin{aligned} \text{EPE}(f) &= \mathbb{E}_{\pi}[(Y - f(X))^2] \\ &= \int_{\mathcal{X}} \int_{\mathcal{Y}} (y - f(x))^2 \pi(y|x) dy dx \end{aligned}$$

It is th

which

i.e., th

Thus,
point

$\mathbb{E}_\pi[Y | X=x]$ is the conditional mean.

Statistical decision theory: squared error loss

If $L(Y, f(X)) = (Y - f(X))^2$, then

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

It is then sufficient to minimize $E_{Y|X}[(Y - f(X))^2 | X]$ for each X :

$$f(x) = \operatorname{argmin}_c E_{Y|X}[(Y - c)^2 | X = x],$$

which leads to

$$f(x) = E[Y | X = x],$$

i.e., the conditional expectation, also known as regression function.

Thus, by average squared error, the best prediction of Y at any point $X = x$ is the conditional mean.

Statistical decision theory: estimation of optimal f

In practice, $f(x)$ must be estimated.

Linear regression:

- assumes a function linear in its arguments, $f(x) \approx x^T \beta$;
- $\operatorname{argmin}_{\beta} E[Y - X^T \beta]^2 \rightarrow \beta =$ [redacted]
- replacing the [redacted]
leads to $\hat{\beta}$. [redacted]
- Note:
 - ▶ no conditioning on [redacted]
 - ▶ we have used our k [redacted] pool over all values [redacted]
 - ▶ less rigid functional relationship may be considered, e.g. [redacted]

$$f(x) \approx \sum_{j=1}^p f(x_j).$$

Statistical decision theory: estimation of optimal f

In practice, $f(x)$ must be estimated.

Linear regression:

- assumes a function linear in its arguments, $f(x) \approx x^T \beta$;
- $\operatorname{argmin}_{\beta} E[Y - X^T \beta]^2 \rightarrow \beta = E[XX^T]^{-1}E[XY]$;
- replacing the expectations by averages over the training data leads to $\hat{\beta}$.
- Note:
 - ▶ no conditioning on X ;
 - ▶ we have used our knowledge on the functional relationship to pool over all values of X (model-based approach);
 - ▶ less rigid functional relationship may be considered, e.g.

$$f(x) \approx \sum_{j=1}^p f(x_j).$$

Statistical decision theory: estimation of optimal f

K nearest neighbors:

- uses **directly** $f(x) = E[Y|X = x]$:
- $\hat{f}(x_i) = \text{Ave}(y_i)$ for observed x_i 's;
- normally there is **at most** one observation for each point x_i ;
- uses points in the **neighborhood**,

$$\hat{f}(x) =$$



- there are two approximations:

- ▶
- ▶



Statistical decision theory: estimation of optimal f

K nearest neighbors:

- uses **directly** $f(x) = E[Y|X = x]$:
- $\hat{f}(x_i) = \text{Ave}(y_i)$ for observed x_i 's;
- normally there is **at most** one observation for each point x_i ;
- uses points in the **neighborhood**,

$$\hat{f}(x) = \text{Ave}(y_i|x_i \in N_k(x))$$

- there are two approximations:
 - ▶ **expectation** is approximated by **averaging** over sample data;
 - ▶ conditioning on a **point** is relaxed to conditioning on a **neighborhood**.

Statistical decision theory: estimation of optimal f

- assumption of k nearest neighbors: $f(x)$ can be approximated by a $\boxed{\text{[redacted]}}$
- for $N \rightarrow \infty$, all $\boxed{\text{[redacted]}}$
- for $k \rightarrow \infty$, $\hat{f}(x) \boxed{\text{[redacted]}}$
- under mild regularity condition on $p(X, Y)$,

$$\hat{f}(x) \rightarrow E[Y|X = x] \text{ for } N, k \rightarrow \infty \text{ s.t. } k/N \rightarrow 0$$

- is this an universal solution?

- ▶ $\boxed{\text{[redacted]}}$
- ▶ $\boxed{\text{[redacted]}}$

Statistical decision theory: estimation of optimal f

- assumption of k nearest neighbors: $f(x)$ can be approximated by a **locally constant function**;
- for $N \rightarrow \infty$, all $x_i \in N_k(x) \approx x$;
- for $k \rightarrow \infty$, $\hat{f}(x)$ is getting more stable;
- under mild regularity condition on $p(X, Y)$,

$$\hat{f}(x) \rightarrow E[Y|X = x] \text{ for } N, k \rightarrow \infty \text{ s.t. } k/N \rightarrow 0$$

- is this an universal solution?
 - ▶ small sample size;
 - ▶ curse of dimensionality (see later)

Statistical decision theory: other loss function

- It is **not necessary** to implement the squared error loss function (L_2 loss function);
- a **valid alternative** is the L_1 loss function:
 - ▶ the solution is the **conditional** [REDACTED]

$$\hat{f}(x) =$$
 [REDACTED]

[REDACTED] **estimates** than those obtained with the

conditional mean;

- ▶ the L_1 loss function has [REDACTED] → numerical difficulties.

Statistical decision theory: other loss function

- It is **not necessary** to implement the squared error loss function (L_2 loss function);
- a **valid alternative** is the L_1 loss function:
 - ▶ the solution is the **conditional median**

$$\hat{f}(x) = \text{median}(Y|X = x)$$

- ▶ **more robust estimates** than those obtained with the conditional mean;
- ▶ the L_1 loss function has **discontinuities in its derivatives** → numerical difficulties.

Statistical decision theory: other loss functions

What happens with a **categorical outcome** G ?

- similar concept, different [redacted]
- $G \in \mathcal{G} = \{1, \dots, K\} \rightarrow \hat{G} \in \mathcal{G} = \{1, \dots, K\}$;
- $L(G, \hat{G}) = L_{G, \hat{G}}$ a **$K \times K$ matrix**, where $K = |G|$;
- each element of the matrix l_{ij} is [redacted]
 - ▶ all elements on the diagonal are [redacted]
 - ▶ often non-diagonal elements are [redacted]

Statistical decision theory: other loss functions

What happens with a **categorical outcome** G ?

- similar concept, different loss function;
- $G \in \mathcal{G} = \{1, \dots, K\} \rightarrow \hat{G} \in \mathcal{G} = \{1, \dots, K\}$;
- $L(G, \hat{G}) = L_{G, \hat{G}}$ a **$K \times K$ matrix**, where $K = |G|$;
- each element of the matrix l_{ij} is the **price to pay to missallocate** category g_i as g_j
 - ▶ all elements on the diagonal are 0;
 - ▶ often non-diagonal elements are 1 (zero-one loss function).

Statistical decision theory: other loss functions

Mathematically:

$$EPE =$$

$$=$$

$$=$$

which is sufficient to

$$\hat{G} = \text{argmin}_{g \in G}$$

When using the 0-1

$$\hat{G} = \text{argmin}_{g \in G}$$

$$= \text{argmin}_{g \in G}$$

$$= \text{argmax}_{g \in G}$$

Statistical decision theory: other loss functions

Mathematically:

$$\begin{aligned} EPE &= E_{G,X}[L(G, \hat{G}(X))] \\ &= E_X \left[E_{G|X}[L(G, \hat{G}(X))] \right] \\ &= E_X \left[\sum_{k=1}^K L(g_k, \hat{G}(X)) \Pr(G = g_k | X) \right] \end{aligned}$$

which is sufficient to be minimized pointwise, i.e,

$$\hat{G} = \operatorname{argmin}_{g \in \mathcal{G}} L(g_k, g) \Pr(G = g_k | X = x).$$

When using the 0-1 loss function

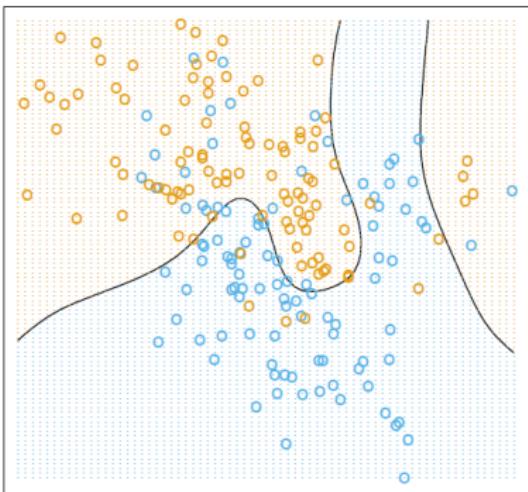
$$\begin{aligned} \hat{G} &= \operatorname{argmin}_{g \in \mathcal{G}} \sum_{k=1}^K \{1 - I(G = g_k)\} \Pr(G = g_k | X = x) \\ &= \operatorname{argmin}_{g \in \mathcal{G}} \{1 - \Pr(G = g_k | X = x)\} \\ &= \operatorname{argmax}_{g \in \mathcal{G}} \Pr(G = g_k | X = x) \end{aligned}$$

Statistical decision theory: other loss functions

Alternatively,

$$\hat{G}(x) = g_k \text{ if } P(G = g_k | X = x) = \max_{g \in \mathcal{G}} \Pr(G = g | X = x),$$

also known as



- k nearest neighbor:

- ▶ $\hat{G}(x) =$
frequency
- ▶ approximat

- regression:

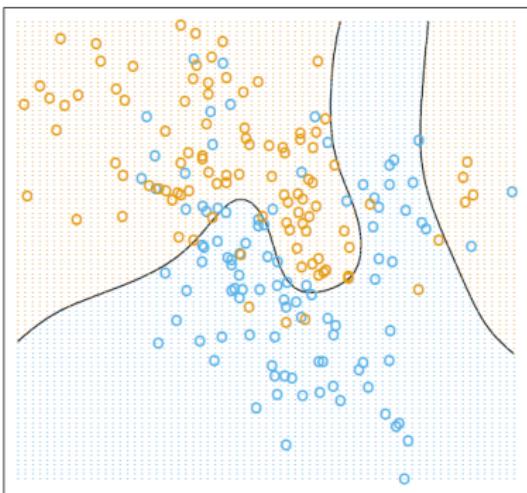
- ▶
- ▶

Statistical decision theory: other loss functions

Alternatively,

$$\hat{G}(x) = g_k \text{ if } P(G = g_k | X = x) = \max_{g \in \mathcal{G}} \Pr(G = g | X = x),$$

also known as **Bayes classifier**.



- k nearest neighbor:
 - ▶ $\hat{G}(x) = \text{category with largest frequency in } k \text{ nearest samples};$
 - ▶ approximation of this solution.
- regression:
 - ▶ $E[Y_k | X] = \Pr(G = g_k | X);$
 - ▶ also approximates the Bayes classifier.

Local methods in high dimensions

The two (extreme) methods seen so far:

- linear model, less
- k -nearest neighbor, less

For large set of training data:

- always possible to use k nearest neighbors?
- Breaks down in less

(Bellman, 1961).

Local methods in high dimensions

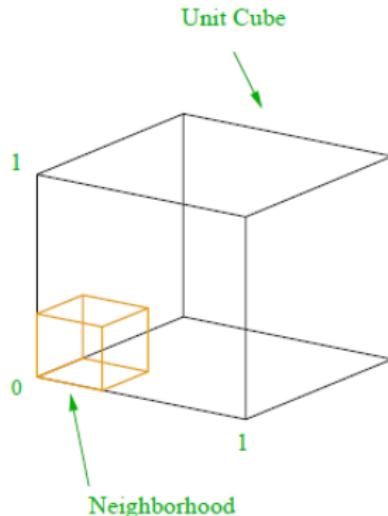
The two (extreme) methods seen so far:

- linear model, **stable but biased**;
- k -nearest neighbor, **less biased but less stable**.

For large set of training data:

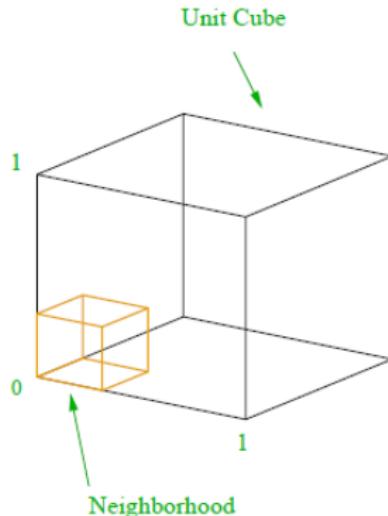
- always possible to use k nearest neighbors?
- Breaks down in high dimensions → **curse of dimensionality** (Bellman, 1961).

Local methods in high dimensions: curse of dimensionality



- Assume $X \sim \text{Unif}[0, 1]^p$;
- define e_p the **expected length size** of a hypercube containing a fraction r of input points;
- $e_p(r) =$ XXXXXXXXXX

Local methods in high dimensions: curse of dimensionality



- Assume $X \sim \text{Unif}[0, 1]^p$;
- define e_p the **expected length size** of a hypercube containing a fraction r of input points;
- $e_p(r) = r^{1/p}$ ($e^p = r \Leftrightarrow e = r^{1/p}$);

Local methods in high dimensions: curse of dimensionality

Assume $Y = f(X) = e^{-8||X||^2}$

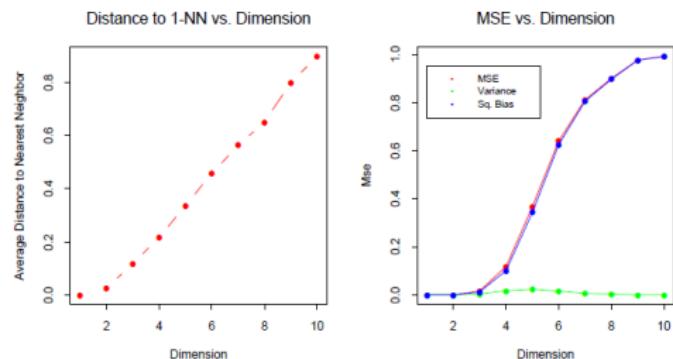
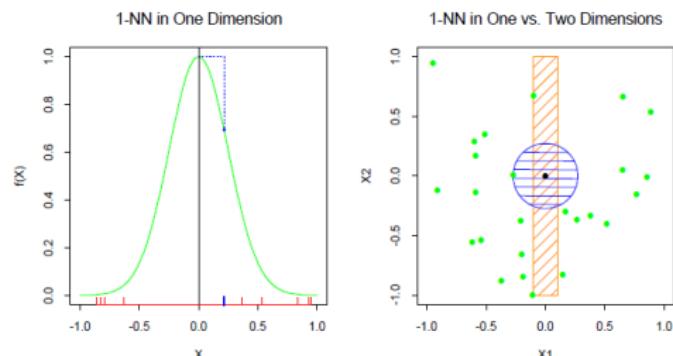
and use the 1-nearest neighbor to predict y_0 at $x_0 = 0$, i.e.

$\hat{y}_0 = y_i$ s.t. x_i nearest observed

$$\text{MSE}(x_0) =$$



NB: we will see often this bias-variance decomposition!



Local methods in high dimensions: curse of dimensionality

Assume $Y = f(X) = e^{-8||X||^2}$

and use the 1-nearest neighbor to predict y_0 at $x_0 = 0$, i.e.

$\hat{y}_0 = y_i$ s.t. x_i nearest observed

$$\text{MSE}(x_0) =$$

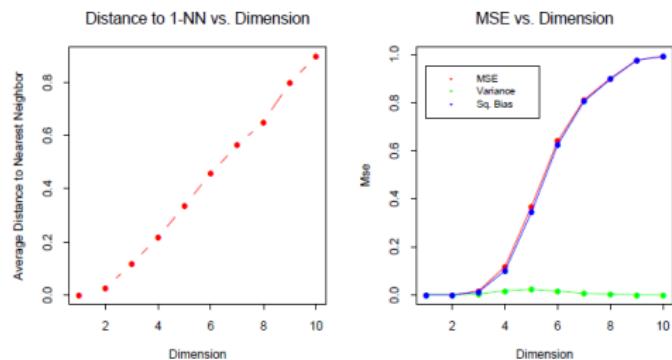
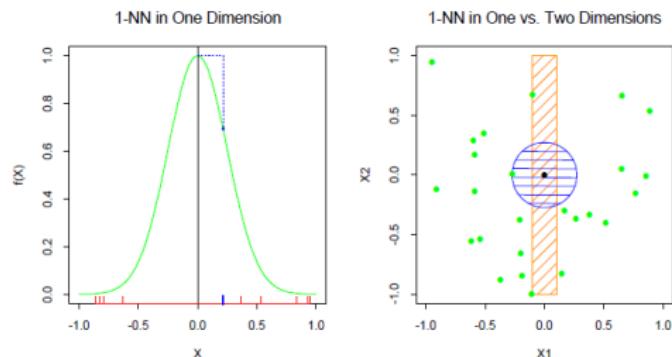
$$= E_{\mathcal{T}}[\hat{y}_0 - f(x_0)]^2$$

$$= E_{\mathcal{T}}[\hat{y}_0 - E_{\mathcal{T}}(\hat{y}_0)]^2$$

$$+ [E_{\mathcal{T}}(\hat{y}_0) - f(x_0)]^2$$

$$= \text{Var}(\hat{y}_0) + \text{Bias}^2(\hat{y}_0)$$

NB: we will see often this bias-variance decomposition!



Local methods in high dimensions: EPE in the linear model

- Assume now $Y = X^T \beta + \varepsilon$
- we want to predict $y_0 = x_0^T \beta + \varepsilon_0$ with x_0 fixed
- $\hat{y}_0 = x_0^T \hat{\beta}$ where $\hat{\beta} = (X^T X)^{-1} X^T y$



True and assumed linear model

- Bias=0
- $\text{Var}(\hat{y}_0) = x_0^T E(X^T X)^{-1} x_0 \sigma^2$
- $\text{EPE}(x_0) = \sigma^2 + x_0^T E(X^T X)^{-1} x_0 \sigma^2$

Local methods in high dimensions: EPE in the linear model

- Assume now $Y = X^T \beta + \varepsilon$
- we want to predict $y_0 = x_0^T \beta + \varepsilon_0$ with x_0 fixed
- $\hat{y}_0 = x_0^T \hat{\beta}$ where $\hat{\beta} = (X^T X)^{-1} X^T y$

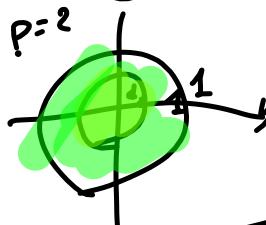
$$\begin{aligned}\text{EPE}(x_0) &= E(y_0 - \hat{y}_0)^2 \\ &= E \left[(y_0 - E[y_0|x_0] + E[y_0|x_0] - E[\hat{y}_0|x_0] + E[\hat{y}_0|x_0] - \hat{y}_0)^2 \right] \\ &= E(y_0 - E[y_0|x_0])^2 + (E[y_0|x_0] - E[\hat{y}_0|x_0])^2 \\ &\quad + E(\hat{y}_0 - E[\hat{y}_0|x_0])^2 \\ &= \text{Var}(y_0|x_0) + \text{Bias}^2(\hat{y}_0) + \text{Var}(\hat{y}_0)\end{aligned}$$

True and assumed linear model

- Bias=0
- $\text{Var}(\hat{y}_0) = x_0^T E(X^T X)^{-1} x_0 \sigma^2$
- $\text{EPE}(x_0) = \sigma^2 + x_0^T E(X^T X)^{-1} x_0 \sigma^2$

$$\Pr[\text{all } N \text{ points have distance } \geq d] = \boxed{\quad}$$

$t_i := \text{distance between } x_i \text{ and origin}$

$$\Pr[t_i \geq d] = 1 - \Pr[t_i < d]$$


$$= 1 - \frac{\text{Volume of ball radius } d}{\text{Volume of ball radius } 1}$$

Volume of a ball with radius $r = \frac{\pi^{P/2}}{(P/2)!} r^P$

$$= 1 - \frac{\frac{\pi^{P/2}}{(P/2)!} d^P}{\frac{\pi^{P/2}}{(P/2)!} 1^P} = 1 - d^P$$

$$\Pr[\forall i, t_i \geq d] = \prod_{i=1}^N \Pr[t_i \geq d]$$

" 1/k "

$$= (1 - d^P)^N$$

$$(1 - d^P)^\omega = \frac{1}{2}$$

$$1 - d^P = \left(\frac{1}{2}\right)^{1/\omega}$$

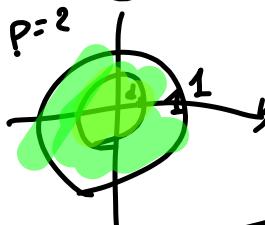
$$d^P = 1 - \left(\frac{1}{2}\right)^{1/\omega}$$

$$d = \left(1 - \left(\frac{1}{2}\right)^{1/\omega}\right)^{1/P}$$

qed

$$\Pr[\text{all } N \text{ points have distance } \geq d] = \frac{1}{2}$$

$t_i := \text{distance between } x_i \text{ and origin}$

$$\Pr[t_i \geq d] = 1 - \Pr[t_i < d]$$


$$= 1 - \frac{\text{Volume of ball radius } d}{\text{Volume of ball radius } 1}$$

Volume of a ball with radius $r = \frac{\pi^{P/2}}{(P/2)!} r^P$

$$= 1 - \frac{\frac{\pi^{P/2}}{(P/2)!} d^P}{\frac{\pi^{P/2}}{(P/2)!} 1^P} = 1 - d^P$$

$$\Pr[\forall i, t_i \geq d] = \prod_{i=1}^N \Pr[t_i \geq d]$$

" 1/k "

$$= (1 - d^P)^N$$

$$(1 - d^P)^N = \frac{1}{2}$$

$$1 - d^P = \left(\frac{1}{2}\right)^{1/N}$$

$$d^P = 1 - \left(\frac{1}{2}\right)^{1/N}$$

$$d = \left(1 - \left(\frac{1}{2}\right)^{1/N}\right)^{1/P}$$

qed

find $\hat{f}(x)$ as a useful approximation of $f(x)$

last week: $L(y, f(x)) = (y - f(x))^2$ squared loss function

leads to $f(x) = \underline{\mathbb{E}[Y|X=x]}$

k-nearest neighbour

$$\mathbb{E}[Y|X=x]$$

↑ average → neighbours

issues

- curse of dimensionality

- e.g.
of

- balance between bias-variance

Statistics vs Machine Learning

Statistical approach:

Starting from the model

$$Y = f(x) + \varepsilon, \quad \mathbb{E}[\varepsilon] = 0, \quad \varepsilon \perp\!\!\!\perp X$$

additive model → approximation of the truth

statist. $\hat{f}(x)$ approx $f(x)$

- we do not suppose $T = f(x)$ (deterministic)

BUT

we add an error term which captures:

- measurement errors;

- effects of non-measured variables;

- ...

Often $\varepsilon \sim \text{iid}$, $\varepsilon \sim \underline{\mathcal{N}}(0, \sigma^2)$

most natural approach, least square

find $\hat{f}(x)$ as a useful approximation of $f(x)$

last week: $L(y, f(x)) = (y - f(x))^2$ squared loss function

leads to $f(x) = \underline{\mathbb{E}[Y|X=x]}$

k-nearest neighbour

$$\mathbb{E}[Y|X=x]$$

↑ average → neighbours

issues

- curse of dimensionality
 - e.g. nearest point gets farer to the point of interest w increasing
- balance between bias-variance

Statistics vs Machine Learning

Statistical approach:

Starting from the model

$$Y = f(x) + \varepsilon, \quad \mathbb{E}[\varepsilon] = 0, \quad \varepsilon \perp\!\!\!\perp X$$

additive model → approximation of the truth

statist. $\hat{f}(x)$ approx $f(x)$

- we do not suppose $T = f(x)$ (deterministic)

BUT

we add an error term which captures:

- measurement errors;
- effects of non-measured variables;
- ...

Often $\varepsilon \sim \text{iid}$, $\varepsilon \sim \underline{\mathcal{N}}(0, \sigma^2)$

most natural approach, least square

Parametric approach

$$\mathcal{F} = \left\{ p(y, x; \underline{\theta}) , \underline{\theta} \in \Theta \subseteq \mathbb{R}^P \right\}$$

$$Y = f(x; \underline{\theta}) + \varepsilon \quad f_{\underline{\theta}}(x) = f(x; \theta)$$

e.g.:

- linear model: $f_{\underline{\theta}}(x) = x^T \underline{\beta}$ $\underline{\theta} = \underline{\beta}$

- linear basis expansion: $f_{\underline{\theta}}(x) = \sum_{k=1}^K h_k(x) \theta_k$

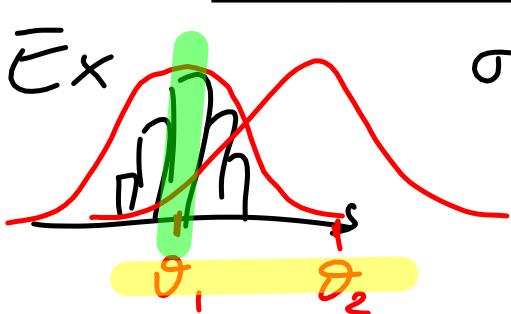
$$h_1(x) = x,$$

$$h_2(x) = x^2$$

$$RSS(\underline{\theta}) = \sum_{i=1}^n (y_i - f_{\underline{\theta}}(x_i))^2$$

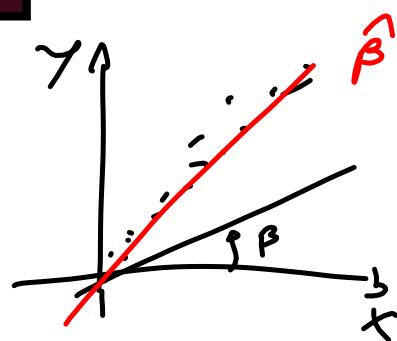
a function of $\underline{\theta}$

$$\hat{\underline{\theta}} =$$

$$\sigma^2 = 1$$

$$\hat{\theta} = \theta,$$



Simplest cases

- least squares

more complicated frameworks

- maximum likelihood estimation

Parametric approach

$$\mathcal{F} = \left\{ p(y, x; \underline{\theta}) , \underline{\theta} \in \Theta \subseteq \mathbb{R}^P \right\}$$

$$Y = f(x; \underline{\theta}) + \varepsilon \quad f_{\underline{\theta}}(x) = f(x; \theta)$$

e.g.:

- linear model: $f_{\underline{\theta}}(x) = x^T \underline{\beta}$ $\underline{\theta} = \underline{\beta}$

- linear basis expansion: $f_{\underline{\theta}}(x) = \sum_{k=1}^K h_k(x) \theta_k$

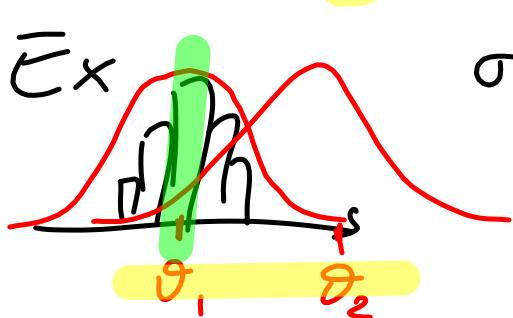
$$h_1(x) = x,$$

$$h_2(x) = x^2$$

$$RSS(\underline{\theta}) = \sum_{i=1}^n (y_i - f_{\underline{\theta}}(x_i))^2$$

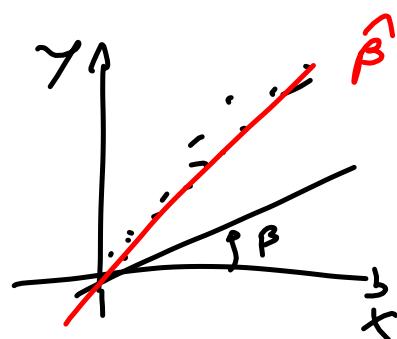
a function of $\underline{\theta}$

$$\hat{\underline{\theta}} = \underset{\underline{\theta}}{\text{arg min}} RSS(\underline{\theta})$$



$$\sigma^2 = 1$$

$$\hat{\underline{\theta}} = \underline{\theta},$$



Simplest cases

- least squares

more complicated frameworks

- maximum likelihood estimation

Likelihood estimators

$$Y_i \sim P$$

$p(y)$ is indexed by θ

Y_i i.i.d.

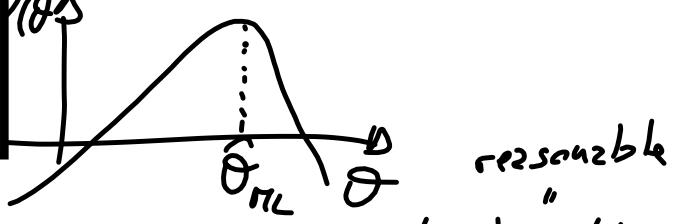
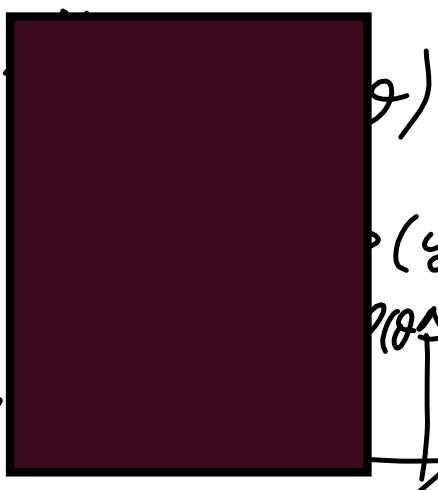
$$p(y; \theta)$$

e.g. Gaussian distributions $P(y; \underbrace{\mu, \sigma^2}_{\theta})$

$$L(\theta) =$$

$$\ell(\theta) =$$

$$\hat{\theta}_{ML} = \arg$$



most plausible
value for θ

Note: when

$$\hat{\theta}_{ML} = \hat{\theta}_{LS}$$

Restricted estimators

instead of estimating θ as

$$\hat{\theta} = \underset{\theta}{\operatorname{argmin}} \text{RSS}(\theta)$$

We look for

$$\hat{\theta} =$$

$$\text{where: PRSS}(\theta) = \text{RSS}(\theta) +$$

LASSO

$$J(\theta) = \sum_{j=1}^p$$

RIDGE

$$J(\theta) = \sum_{j=1}^p$$

Likelihood estimators

$Y_i \sim P$ $p(y)$ is indexed by θ

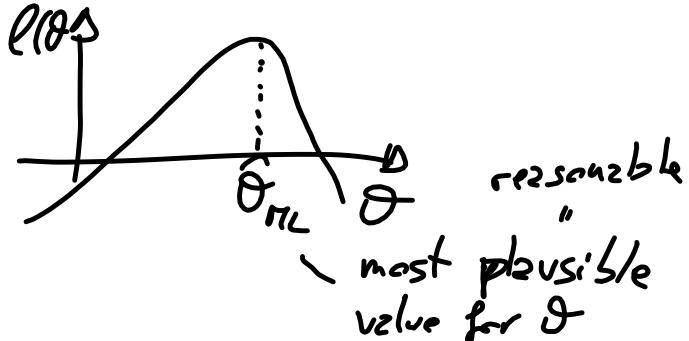
Y_i i.i.d. $p(y; \theta)$

e.g. Gaussian distributions $P(y; \underbrace{\mu, \sigma^2}_{\theta})$

$$L(\theta) = \prod_{i=1}^N p(y_i; \theta)$$

$$\ell(\theta) = \sum_{i=1}^n \log p(y_i; \theta)$$

$$\hat{\theta}_{ML} = \underset{\theta}{\operatorname{argmax}} \ell(\theta)$$



Note: when $\varepsilon \sim \mathcal{N}(0, \sigma^2)$

$$\hat{\theta}_{ML} = \hat{\theta}_{LS}$$

Restricted estimators

instead of estimating θ as

$$\hat{\theta} = \underset{\theta}{\operatorname{argmin}} RSS(\theta)$$

we look for

$$\hat{\theta} = \underset{\theta}{\operatorname{argmin}} PRSS(\theta)$$

where: $PRSS(\theta) = RSS(\theta) + \lambda J(\theta)$

$$\text{LASSO} \quad J(\theta) = \sum_{j=1}^p |\theta_j|$$

$$\text{RIDGE} \quad J(\theta) = \sum_{j=1}^p \theta_j^2$$

Linear methods for regression

$E[Y|X]$ is linear in inputs

$$E[Y|X] = X^T \beta$$

→ simple

→ often adequate

→ easy to interpret

→ often outperforms fancier methods

↓
simplicity!

→ sample size is small

→ sparse data

→ low



linear methods

regression
(ch 3)

classification
(ch 4)

Regression

- consider continuous Y

- linear regression $f(x) = \beta_0 + \beta_1 x_1 + \dots + \beta_p x_p$

$$\hat{\beta} = \underset{\beta}{\operatorname{argmin}} RSS(\beta)$$

$$\sum_{i=1}^n |y_i - \beta_0 - \sum_{j=1}^p x_{ij} \beta_j|^2$$

$$\text{matrix form } (y - X\beta)^T (y - X\beta)$$

$$\frac{\partial RSS(\beta)}{\partial \beta} =$$



$$\frac{\partial^2 RSS(\beta)}{\partial \beta^2} =$$

Linear methods for regression

$E[Y|X]$ is linear in inputs

$$E[Y|X] = X^T \beta$$

→ simple

→ often adequate

→ easy to interpret

→ often outperforms fancier methods

↓
simplicity!

→ sample size is small

→ sparse data

→ low signal-to-noise ratio

linear methods

regression
(ch 3)

classification
(ch 4)

Regression

- consider continuous Y

- linear regression $f(x) = \beta_0 + \beta_1 x_1 + \dots + \beta_p x_p$

$$\hat{\beta} = \underset{\beta}{\operatorname{argmin}} RSS(\beta)$$

$$\sum_{i=1}^n |y_i - \beta_0 - \sum_{j=1}^p x_{ij} \beta_j|^2$$

$$\text{matrix form } (y - X\beta)^T (y - X\beta)$$

$$\frac{\partial RSS(\beta)}{\partial \beta} = -2X^T(y - X\beta)$$

$$\frac{\partial^2 RSS(\beta)}{\partial \beta^2} = 2X^T X$$

$$\frac{\partial RSS(\beta)}{\partial \beta} = 0$$

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

$\beta = 0$

$$\hat{\beta} = (x^T x)^{-1} x^T y$$

$$\frac{\partial^2 RSS(\beta)}{\partial \beta^2} = \begin{bmatrix} & \\ & \\ & \end{bmatrix} > 0 \Rightarrow \hat{\beta} \text{ minimum}$$

$$\hat{y} = x \hat{\beta} = \underbrace{x (x^T x)^{-1} x^T}_{H} y$$

hat matrix
projection matrix

Properties

$$V_{cr}(\hat{\beta}) =$$

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

$$\hat{\sigma}^2 =$$

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

Focus on $\varepsilon \sim \mathcal{N}(0, \sigma^2)$ ε : iid

$$\hat{\beta} \sim \mathcal{N}(\beta, (x^T x)^{-1} \sigma^2)$$

$$(N-p-1)\hat{\sigma}^2 \sim \sigma^2 \chi_{N-p-1}^2$$

$$\frac{\partial RSS(\beta)}{\partial \beta} = 0 \quad X^T(y - X\beta) = 0$$

$$X^T y - X^T X \beta = 0$$

$$\hat{\beta} = (X^T X)^{-1} X^T y$$

$$\frac{\partial^2 RSS(\beta)}{\partial \beta^2} = 2X^T X > 0 \Rightarrow \hat{\beta} \text{ minimum}$$

$$\hat{y} = X \hat{\beta} = \underbrace{X(X^T X)^{-1} X^T}_H y$$

hat matrix
projection matrix

Properties

$$V_{cr}(\hat{\beta}) = (X^T X)^{-1} \sigma^2$$

$$\hat{\sigma}^2 = \frac{1}{N-p-1} \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

Focus on $\varepsilon \sim N(0, \sigma^2)$ ε : iid

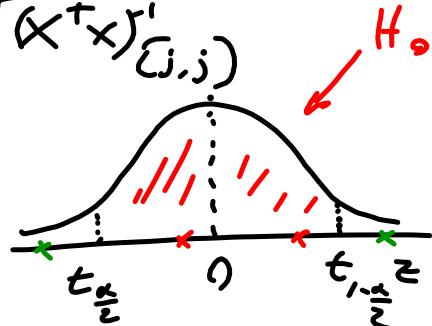
$$\hat{\beta} \sim N(\beta, (X^T X)^{-1} \sigma^2)$$

$$(N-p-1)\hat{\sigma}^2 \sim \sigma^2 \chi_{N-p-1}^2$$

$$H_0: \beta_j = 0 \rightarrow z_j = \frac{\hat{\beta}_j - 0}{sd(\hat{\beta}_j)} = \frac{\hat{\beta}_j}{\hat{\sigma} \sqrt{(\mathbf{x}^T \mathbf{x})_{jj}}}$$

Under H_0 , $z_j \sim t_{N-p-1}$

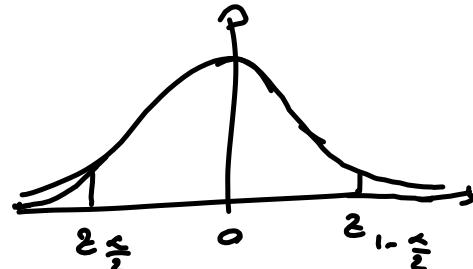
z_{obs}



H_1 \Rightarrow reject H_0

when σ^2 is known

under H_0 , $z_j \stackrel{H_0}{\sim} \mathcal{N}(0; 1)$



$$Y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \epsilon$$

is x_1 useful to predict y ?

$\beta_1 = 0 \rightarrow \epsilon \begin{cases} \text{reject } H_0 & \text{Yes} \\ \text{do NOT reject } H_0 & \text{not Yes} \end{cases}$

Are (x_2, x_3) useful to predict y ?

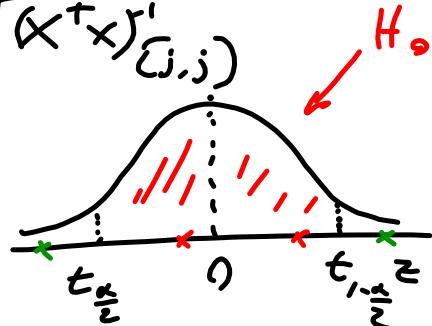
$H_0: \beta_2, \beta_3 = 0$

$F =$

$$H_0: \beta_j = 0 \rightarrow z_j = \frac{\hat{\beta}_j - 0}{sd(\hat{\beta}_j)} = \frac{\hat{\beta}_j}{\hat{\sigma} \sqrt{(X^T X)^{-1}_{(j,j)}}}$$

Under H_0 , $z_j \sim t_{n-p-1}$

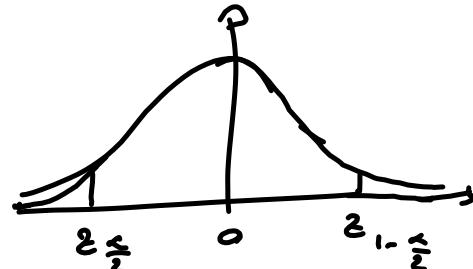
z_{obs}



H_1 \Rightarrow reject H_0

when σ^2 is known

under H_0 , $z_j \stackrel{H_0}{\sim} \mathcal{N}(0; 1)$



$$Y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \epsilon$$

is x_2 useful to predict y ?

$\beta_2 = 0 \rightarrow \epsilon \begin{cases} \text{reject } H_0 & \text{Yes} \\ \text{do NOT reject } H_0 & \text{not Yes} \end{cases}$

Are (x_2, x_3) useful to predict y ?

$H_0: \beta_2, \beta_3 = 0$

$$F = \frac{(RSS_0 - RSS_1) / (p_1 - p_0)}{RSS_1 / (n - p_1 - 1)}$$

Today

- 1) 3 exercises
 - 2) Gauss-Markov theorem
orthogonalization
 - 3) Model selection
 - 4) Shrinkage methods (ch 3.4)
-

Ex 2.7

N pairs (x_i, y_i) drawn iid from

$$x_i \sim h(x)$$

$$y_i = f(x_i) + \varepsilon_i$$

$$\varepsilon_i \sim N(0, \sigma^2)$$

Estimator for f linear in the y_i :

$$\hat{f}(x_0) = \sum_{i=1}^N \ell_i(x_0; X) y_i$$

where weights $\ell_i(x_0; X) \perp\!\!\!\perp y_i$, but depend on X , the entire training sequence of x_i .

a) $\hat{f}_{LR}(x_0) = X_0^T \hat{\beta} =$

$$\Rightarrow \ell_i(x_0, X) =$$

$$\hat{f}_{KNN}(x_0) = \text{Ave} \{ y_i \mid x_i \in N_k(x_0) \}$$

where $N_k(x_0)$ is the set of the k nearest neighbors

$$\Rightarrow \ell_i(x_0, X) = \begin{cases} 1/k & \text{if } x_i \in N_k(x_0) \\ 0 & \text{otherwise} \end{cases}$$

Today

- 1) 3 exercises
 - 2) Gauss-Markov theorem
orthogonalization
 - 3) Model selection
 - 4) Shrinkage methods (ch 3.4)
-

Ex 2.7

\checkmark drawn
N pairs (x_i, y_i) iid from

$$x_i \sim h(x)$$

$$y_i = f(x_i) + \varepsilon_i$$

$$\varepsilon_i \sim N(0, \sigma^2)$$

Estimator for f linear in the y_i :

$$\hat{f}(x_0) = \sum_{i=1}^N \ell_i(x_0; X) y_i$$

where weights $\ell_i(x_0; X) \perp\!\!\!\perp y_i$, but depend on X , the entire training sequence of x_i .

$$\begin{aligned} a) \hat{f}_{LR}(x_0) &= X_0^T \hat{\beta} = \underbrace{x_0^T (X^T X)^{-1} X^T}_{\hat{\beta}} y_i \\ &\Rightarrow \ell_i(x_0, X) = [x_0^T (X^T X)^{-1} X^T]_{[1]} \end{aligned}$$

$$\hat{f}_{KNN}(x_0) = \text{Ave} \{ y_i \mid x_i \in N_k(x_0) \}$$

where $N_k(x_0)$ is the set of the k nearest neighbors

$$\Rightarrow \ell_i(x_0, X) = \begin{cases} 1/k & \text{if } x_i \in N_k(x_0) \\ 0 & \text{otherwise} \end{cases}$$

b) Decompose the conditional mean-squared error

=
= A
=

c) unconditional

$$E_x \left[(\hat{f}(x_0) - f(x_0))^2 \right] = E_x \left[\text{Var}_{Y|X}(\hat{f}(x_0)) + \text{Var}_x(f(x_0)) \right]$$

d) conditional:

$$(f(x_0) - E_{Y|X}[\hat{f}(x_0)])^2 + \text{Var}_{Y|X}(\hat{f}(x_0))$$

$$(f(x_0) - E_{Y|X}[\sum_i \ell_i(x_0; \chi) g_i])^2 + \text{Var}_{Y|X}[\sum_i \ell_i(x_0; \chi) g_i]$$

$$(f(x_0) - \sum_i \ell_i(x_0; \chi) f(x_0))^2 + \sum_i \ell_i(x_0; \chi)^2 \sigma^2$$

unconditional

$$E_x \left[(f(x_0) - E_{Y|X}[\sum_i \ell_i(x_0; \chi) g_i])^2 \right] + E_x \left[\text{Var}_{Y|X}[\sum_i \ell_i(x_0; \chi) g_i] \right]$$

b) Decompose the conditional mean-squared error

$$E_{Y|X} \left[(f(x_0) - \hat{f}(x_0))^2 \right]$$

student fixed

$$E_{Y|X} \left[(f(x_0) - E_{Y|X}[\hat{f}(x_0)])^2 + E_{Y|X}[\hat{f}(x_0)] - \hat{f}(x_0))^2 \right]$$

$$= E_{Y|X} \left[(f(x_0) - E_{Y|X}[\hat{f}(x_0)])^2 + (E_{Y|X}[\hat{f}(x_0)] - \hat{f}(x_0))^2 \right.$$

$$\left. + 2 (f(x_0) - E_{Y|X}[\hat{f}(x_0)]) (E_{Y|X}[\hat{f}(x_0)] - \hat{f}(x_0)) \right]$$

$$= \underbrace{(f(x_0) - E_{Y|X}[\hat{f}(x_0)])^2}_{\text{bias}^2} + \underbrace{\text{Var}_{Y|X}(\hat{f}(x_0))}_{\text{variance}} +$$

$$+ 2 (f(x_0) - E_{Y|X}[\hat{f}(x_0)]) E_{Y|X} \left[\underbrace{E_{Y|X}[\hat{f}(x_0)] - \hat{f}(x_0)}_{\text{error}} \right]$$

= bias² + variance

c) unconditional

$$E_{Y|X} \left[(f(x_0) - \hat{f}(x_0))^2 \right]$$

$$E_X \left[E_{Y|X} \left[(f(x_0) - \hat{f}(x_0))^2 \right] \right]$$

d) conditional:

$$(f(x_0) - E_{Y|X}[\hat{f}(x_0)])^2 + \text{Var}_{Y|X}(\hat{f}(x_0))$$

$$(f(x_0) - E_{Y|X} \left[\sum_i \ell_i(x_0; X) g_i \right])^2 + \text{Var}_{Y|X} \left[\sum_i \ell_i(x_0; X) g_i \right]$$

$$(f(x_0) - \sum_i \ell_i(x_0; X) f(x_0))^2 + \sum_i \ell_i(x_0; X)^2 \sigma^2$$

unconditional

$$E_X \left[(f(x_0) - E_{Y|X} \left[\sum_i \ell_i(x_0; X) g_i \right])^2 \right] = E_X \left[\text{Var}_{Y|X} \left[\sum_i \ell_i(x_0; X) g_i \right] \right]$$

Ex 3.1

$$F = \frac{(RSS_0 - RSS_1) / (P_1 - P_0)}{RSS_1 / (N - P_1 - 1)}$$

when we are testing only one parameter $F \approx z^2$

$$\rightarrow P_1 = P, P_0 = P - 1$$

$$F = \frac{(RSS_0 - RSS_1)}{RSS_1 / (N - P - 1)} \sim F_{1, N-P-1}$$

$$z = \frac{\hat{\beta}_j}{\hat{\sigma} [x'x]_{jj}} \sim t_{N-P-1}$$

$$\Rightarrow z^2 \approx F$$

Gauss-Markov theorem

$\hat{\beta}_{LS}$ is BLUE

B est ←
L inear ←
U nbiased ←
E stimator

consider X

$$\begin{aligned} E[\hat{\beta}] &= \\ &= \\ &= \\ &= q^T \beta \quad \perp \end{aligned}$$

$$\text{Th: } \text{Var}(\hat{\beta}) \leq \text{Var}(\hat{\beta}) \rightarrow \text{ex 3.3(a)}$$

$$\text{MSE}(\hat{\beta}) =$$

$$=$$

$$=$$

$$=$$

smallest with LS

||
0

Ex 3.1

$$F = \frac{(RSS_0 - RSS_1) / (P_0 - P_1)}{RSS_1 / (N - P_1 - 1)}$$

when we are testing only one parameter $F \approx z^2$

$$\rightarrow P_1 = P, P_0 = P - 1$$

$$F = \frac{(RSS_0 - RSS_1)}{RSS_1 / (N - P - 1)} \sim F_{1, N-P-1}$$

$$z = \frac{\hat{\beta}_j}{\hat{\sigma}[x'x]_{jj}} \sim t_{N-P-1}$$

$$(t_{N-P-1})^2 \stackrel{d}{=} F_{1, N-P-1} \Rightarrow z^2 \approx F$$

Gauss-Markov theorem

$\hat{\beta}_{LS}$ is BLUE

B est \leftarrow smallest variance

L inear $\leftarrow \hat{\theta} = \alpha^\top \hat{\beta} \leftrightarrow \theta = \alpha^\top \beta$

U nbiased $\leftarrow E[\hat{\theta}] = \theta$

E stimator

consider X fixed

$$\begin{aligned} E[\hat{\theta}] &= E[\alpha^\top (X^\top X)^{-1} X^\top y] \\ &= \alpha^\top (X^\top X)^{-1} X^\top X \beta \\ &= \alpha^\top \beta \end{aligned}$$

$$\boxed{\text{Th: } \text{Var}(\hat{\theta}) \leq \text{Var}(\tilde{\theta})} \quad \rightarrow \text{ex 3.3(a)}$$

$$\text{MSE}(\hat{\theta}) = E[(\hat{\theta} - \theta)^2]$$

$$= E[(\hat{\theta} - E[\hat{\theta}] + E[\hat{\theta}] - \theta)^2]$$

$$= \text{Var}(\hat{\theta}) + (E[\hat{\theta}] - \theta)^2$$

smallest with LS

||
0

Ex 3.3 (a)

$$\hat{\theta}_{LS} =$$

$$\tilde{\theta} = c$$

$$E[\tilde{\theta}] =$$

$$\begin{aligned} &= - \\ &= - \\ &= - \\ &= - \\ &= \end{aligned}$$

$$Var(\tilde{\theta})$$

$$\begin{aligned} &= \cdot \\ &= - \\ &= - \\ &= - \\ &= a \\ &= \end{aligned}$$

Ex 3.3 (a)

$$\hat{\theta}_{LS} = \alpha^T \hat{\beta} = \alpha^T (X^T X)^{-1} X^T y \quad E[\alpha^T \hat{\beta}] = \alpha^T \beta$$

$$\tilde{\theta} = c^T y \quad \text{unbiased} \quad c^T = \alpha^T (X^T X)^{-1} X^T + \delta^T$$

$$E[\tilde{\theta}] = E[c^T y]$$

$$\begin{aligned} &= E[\alpha^T (X^T X)^{-1} X^T y + \delta^T y] \\ &= \alpha^T (X^T X)^{-1} X^T X \beta + \delta^T X \beta \\ &= \underline{\alpha^T \beta} + \underline{\delta^T X \beta} \Rightarrow \underline{\delta^T X} = 0 \end{aligned}$$

$$\text{Var}(\tilde{\theta}) = \text{Var}(c^T y)$$

$$\begin{aligned} &= c^T \sigma^2 c \\ &= \sigma^2 (\alpha^T (X^T X)^{-1} X^T + \delta^T) (\alpha^T (X^T X)^{-1} X^T + \delta^T)^T \\ &= \sigma^2 (\alpha^T (X^T X)^{-1} X^T + \delta^T) (X (X^T X)^{-1} \alpha + \delta) \\ &= \sigma^2 \left(\alpha^T (X^T X)^{-1} X^T X (X^T X)^{-1} \alpha + \alpha^T (X^T X)^{-1} X^T \delta + \right. \\ &\quad \left. + \delta^T X (X^T X)^{-1} \alpha + \delta^T \delta \right) \\ &= \sigma^2 \alpha^T (X^T X)^{-1} \alpha + \sigma^2 \delta^T \delta \\ &\quad \parallel \quad \text{VI} \\ &= \text{Var}(\hat{\theta}_{LS}) \quad 0 \end{aligned}$$

• $MSE(\hat{f})$ is strongly related with the prediction error

Let consider $y_0 = f(x_0) + \varepsilon$ $\varepsilon \sim N(0, \sigma^2)$
 $\varepsilon \perp x$

$$E[(y_0 - \hat{f}(x_0))^2] = E[y_0^2 + \hat{f}(x_0)^2 - 2y_0\hat{f}(x_0)]$$

=
|
= 1
=

Jons variant

1.5c

$$\begin{aligned} E[y_0] &= E[f(x_0) + \varepsilon] = E[f(x_0)] + E[\varepsilon] \\ &\stackrel{!}{=} f(x_0) + 0 \end{aligned}$$

$$\begin{aligned} E[y_0^2] &= E[(f(x_0) + \varepsilon)^2] \\ &\stackrel{!}{=} \text{Var}[y_0] + E[y_0]^2 \\ &\stackrel{!}{=} \sigma^2 + \underline{\underline{f(x_0)^2}} \end{aligned}$$

• $MSE(\hat{f})$ is strongly related with the prediction error

Let consider $y_0 = f(x_0) + \varepsilon$ $\varepsilon \sim N(0, \sigma^2)$
 $\varepsilon \perp x$

$$E[(y_0 - \hat{f}(x_0))^2] = E[y_0^2 + \hat{f}(x_0)^2 - 2y_0\hat{f}(x_0)]$$

$$= E[y_0^2] + E[\hat{f}(x_0)^2] - 2E[y_0\hat{f}(x_0)]$$

$$= Var(y_0) + E[y_0^2] + Var(\hat{f}(x_0)) + E[\hat{f}(x_0)]^2$$

$$- 2f(x_0)E[\hat{f}(x_0)]$$

$$= \sigma^2 + Var(\hat{f}(x_0)) + E[\hat{f}(x_0)]^2 - 2f(x_0)E[\hat{f}(x_0)] + f(x_0)^2$$

$$= \sigma^2 + Var(\hat{f}(x_0)) + \text{bias}^2$$

$(E[\hat{f}(x_0)] - y_0)^2$

MSE

$$E[y_0] = E[f(x_0) + \varepsilon] = E[f(x_0)] + E[\varepsilon]$$

$$= f(x_0) + 0$$

$$E[y_0^2] = E[(f(x_0) + \varepsilon)^2]$$

$$= Var(y_0) + E[y_0]^2$$

$$= \sigma^2 + f(x_0)^2$$

$\underline{\underline{}}$

Suppose $Y = X\beta + \varepsilon$

Univariable
if $Y = X_1\beta_1 + \varepsilon$

$$\hat{\beta} = \frac{\sum x_i y_i}{\sum x_i^2} = \frac{\langle x, y \rangle}{\langle x, x \rangle}$$

Multivariable
if $Y = X_1\beta_1 + X_2\beta_2 + \varepsilon$

only if X is orthogonal , $\hat{\beta}_j = \frac{\langle x_j, y \rangle}{\langle x_j, x_j \rangle}$

alternative formula for variance of $\hat{\beta}_j$:

$$\text{Var}(\hat{\beta}_j) = \frac{\sigma^2}{\langle e_p, e_p \rangle} = \frac{\sigma^2}{\|e_p\|^2}$$

→ variables may share the same information useful for explaining/predicting y

→ the estimates may be unstable due to high variance

↓
model selection

$\Sigma_{\text{group}} =$



→ smaller

→ larger

Suppose $Y = X\beta + \varepsilon$

Univariable
if $Y = X_1\beta_1 + \varepsilon$

$$\hat{\beta} = \frac{\sum x_i y_i}{\sum x_i^2} = \frac{\langle x, y \rangle}{\langle x, x \rangle}$$

Multivariable
if $Y = X_1\beta_1 + X_2\beta_2 + \varepsilon$

only if X is orthogonal , $\hat{\beta}_j = \frac{\langle x_j, y \rangle}{\langle x_j, x_j \rangle}$

alternative formula for variance of $\hat{\beta}_j$:

$$\text{Var}(\hat{\beta}_j) = \frac{\sigma^2}{\langle e_p, e_p \rangle} = \frac{\sigma^2}{\|e_p\|^2}$$

→ variables may share the same information useful for explaining/predicting y

→ the estimates may be unstable due to high variance

↓
model selection

$$z_{\text{score}} = \frac{\hat{\beta}}{\text{sd}(\hat{\beta})} \rightarrow \text{smaller}$$

\rightarrow larger

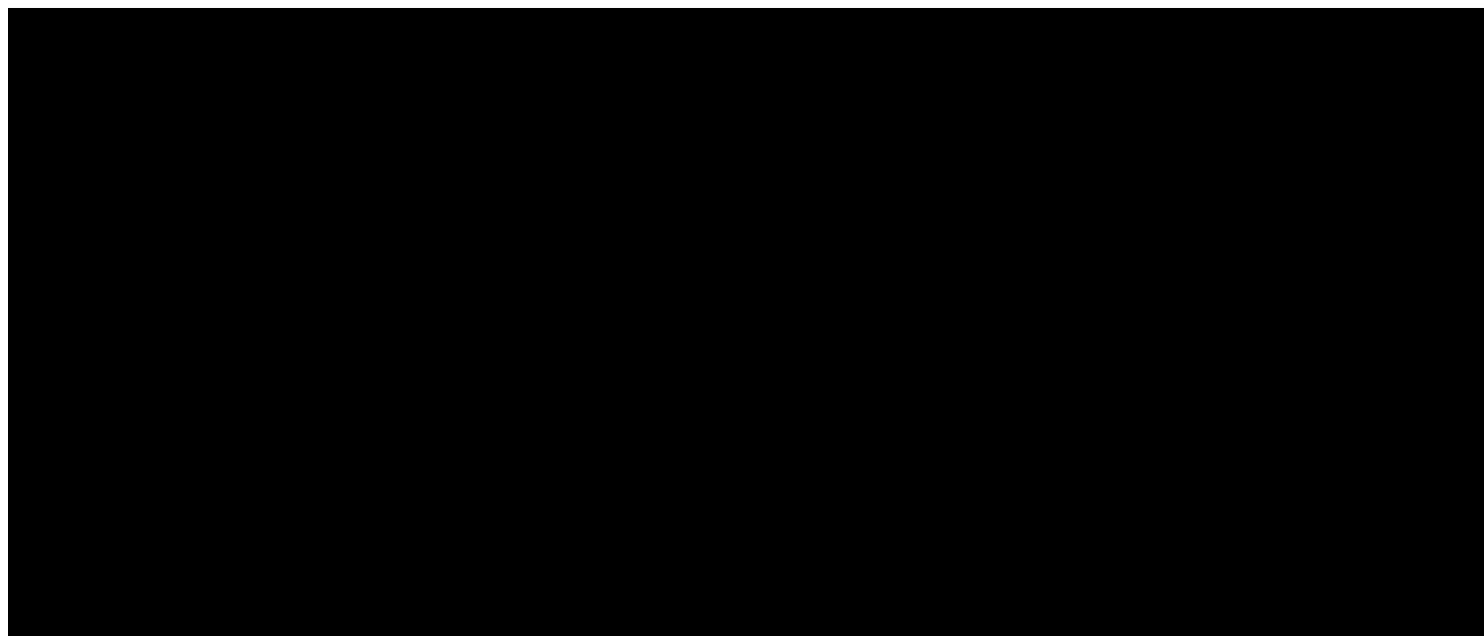
Model Selection

- [REDACTED]
- [REDACTED]

→ portability

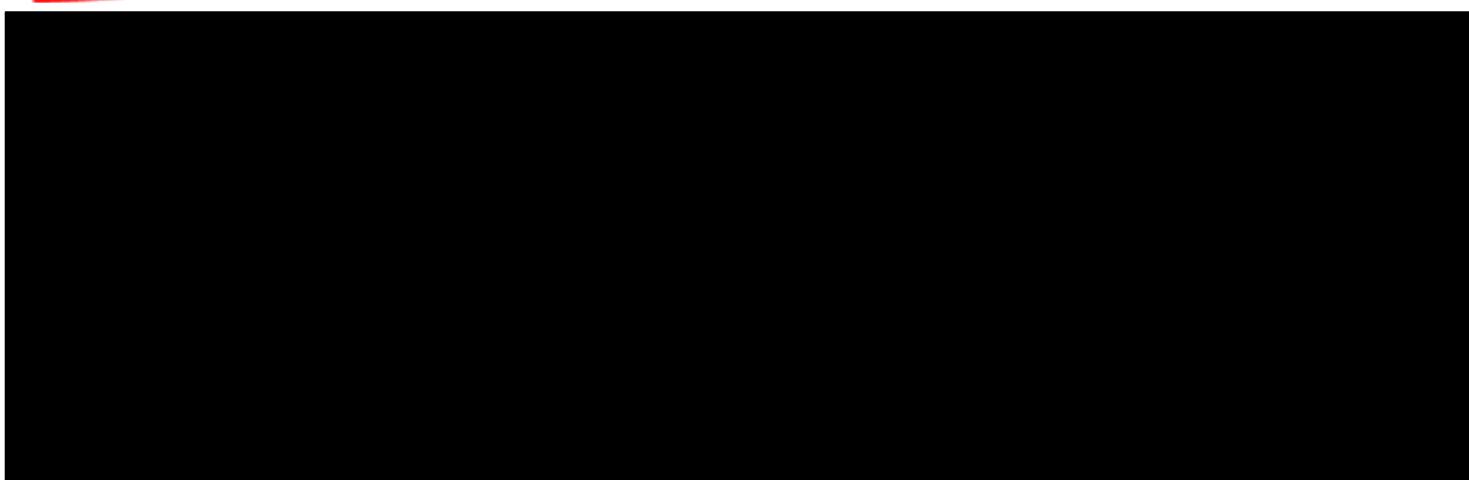
-
- best subset technique

$$x = (x_1, x_2, \dots)$$



-
- Stepwise techniques

- forward selection



Model Selection

- prediction accuracy
- interpretability
- portability

- best subset technique

$x = (x_1, x_2, x_3)$

models : 0 variables $y = \beta_0 + \varepsilon$

1 variable $y = \beta_0 + \beta_1 x_1 + \varepsilon$

$$y = \beta_0 + \beta_2 x_2 + \varepsilon$$

$$y = \beta_0 + \beta_3 x_3 + \varepsilon$$

2 variables $y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \varepsilon$

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \varepsilon$$

$\binom{3}{2} = \frac{3 \cdot 2}{2} = 3 \rightarrow y = \beta_0 + \beta_2 x_2 + \beta_3 x_3 + \varepsilon$

$\rightarrow \binom{8}{2} = \frac{8 \cdot 7}{2} = 28$ picture

- Stepwise techniques

- forward selection

start : $y = \beta_0 + \varepsilon$ x_1, x_2, x_3

1st step : $y = \beta_0 + \beta_2 x_2 + \varepsilon$ x_1, x_3

2nd step : $y = \beta_0 + \beta_2 x_2 + \beta_3 x_3 + \varepsilon$

$\hat{\beta}_3$ together with $\hat{\beta}_0$ and $\hat{\beta}_2$

- backward elimination

penalized

- stepwise selection
step back "

$$\text{RSS}(\boldsymbol{\beta}) + J(\boldsymbol{\beta}) \\ + \underline{\ell(p)}$$

staged regression

stepwise: the estimate

- backward elimination

start : full model

Prin case
backward
elimination is

$$y = \beta_0 + \underline{\beta_1 x_1} + \underline{\beta_2 x_2} + \underline{\beta_3 x_3} + \varepsilon \quad \text{not possible}$$

1st step

$$y = \beta_0 + \beta_2 x_2 + \beta_3 x_3 + \varepsilon$$

:

:

- stepwise selection
step back " "

$$\text{RSS}(\theta) + J(\theta) \\ + \underline{\ell |P|}$$

stagewise regression

$$y = \beta_0 + \beta_1 x_2 + \varepsilon$$

$$y = \beta_0 + \beta_1 x_1 + \underline{\beta_2 x_2} + \varepsilon$$

stepwise: the estimate
of β take into account
all x

stagewise: introduce
2 new β , its estimate
is only based on x_j

b[1]=mean(y)

b[2..n]=0

r=(y-X*b)

index, maxCorr = max(transpose(r)*X)

while(abs(maxCorr) > someThreshold)

b[index]=b[index]+regress(r,X[1..n][index])

r=(y-X*b)

index, maxCorr = max(transpose(r)*X)

Variable Selection

IDEA: remove irrelevant variables from the model

↑
not useful to predict/explain the response

- less variance (despite small increase of \$S_{res}\$)
- better interpretability
- better portability

When a variable is considered irrelevant

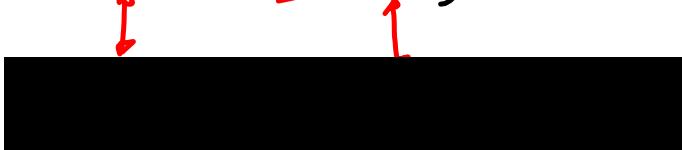
- [REDACTED]
- [REDACTED]

INFORMATION CRITERIA

IDEA: instead of $\hat{\theta} = \arg \min_{\theta} L(\theta)$,

we find $\hat{\theta}_{IC} = \arg \min_{\theta} \{L(\theta) + \lambda J(\theta)\}$

$$\hat{\theta}_{IC} = \underline{X} \underline{\beta}$$



GOAL: penalize larger models

$$L(\theta) =$$

$$J(\theta) =$$

$$\lambda = \left\{ \begin{array}{l} \text{Akaike information criterion} \\ \text{Bayesian} \end{array} \right.$$

variables in the model

Akaike information criterion

Bayesian " "

n is the sample size

NB: using AIC for model selection is like using $\alpha=0.157$

Variable Selection

IDEA: remove irrelevant variables from the model

↑
not useful to predict/explain the response

- less variance (despite small increase of \$S_{res}\$)
- better interpretability
- better portability

When a variable is considered irrelevant

- p-value of a test $> \alpha$, usually $\alpha=0.05$
- its inclusion increases a' information criterion

INFORMATION CRITERIA

IDEA: instead of $\hat{\theta} = \arg \min_{\theta} L(\theta)$,

We find $\hat{\theta}_{IC} = \arg \min_{\theta} \{L(\theta) + \lambda J(\theta)\}$

$$\hat{\theta}_{IC} = \underline{X} \underline{\beta}$$

$$-2 \ell(\theta)$$

$$\sum_{j=1}^p \mathbf{1}[\theta_j \neq 0]$$

GOAL: penalize larger models

$$L(\theta) = -2 \ell(\theta)$$

$$J(\theta) = \sum_{j=1}^p \mathbf{1}[\theta_j \neq 0] \quad \# \text{variables in the model}$$

$$\lambda = \begin{cases} 2 & AIC \quad \text{Akaike information criterion} \\ \log(n) & BIC \quad \text{Bayesian} \end{cases}$$

n is the sample size

NB: using AIC for model selection is like using $\alpha=0.157$

if two explanatory variables are strongly correlated \rightarrow collinearity

extreme case: variables linearly dependent
 \rightarrow super-collinearity

in the case of super-collinearity $(X^T X)^{-1}$ is not invertible
 (not full rank)

Häderl & Kennard (1970) $X^T X \rightarrow X^T X + \lambda I_p$

$$I_p = \begin{pmatrix} 1 & & & \\ & \ddots & & \\ & & 1 & \\ & 0 & \cdots & 1 \end{pmatrix}$$

$$\boxed{\hat{\beta}_{\text{ridge}}^{(\lambda)} = (X^T X + \lambda I_p)^{-1} X^T y} \quad \lambda \in [0; \infty)$$

when $\lambda \in (0; \infty)$ $(X^T X + \lambda I_p)^{-1}$ exists

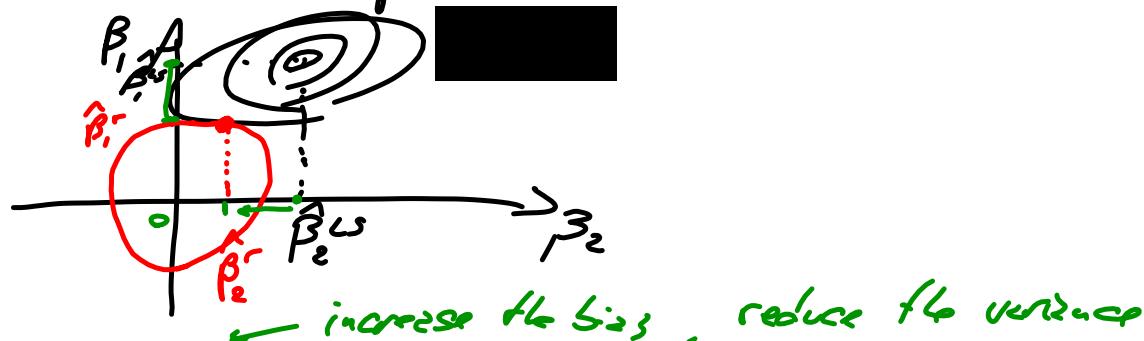
RIDGE REGRESSION AS A SHRINKAGE METHOD

- $\hat{\beta}_{\text{ridge}}^{(\lambda)} = \boxed{\dots}$

alternative formulation

- $\hat{\beta}_{\text{ridge}}^{(\lambda)} = \boxed{\dots}$

- one-to-one correspondence between λ and t



if two explanatory variables are strongly correlated \rightarrow collinearity

extreme case: variables linearly dependent
 \rightarrow super-collinearity

in the case of super-collinearity $(X^T X)^{-1}$ is not invertible
 (not full rank)

Häerl & Kennard (1970) $X^T X \rightarrow X^T X + \lambda I_p$

$$I_p = \begin{pmatrix} 1 & & & \\ & \ddots & & \\ & & 1 & \\ & 0 & \cdots & 1 \end{pmatrix}$$

$$\hat{\beta}_{\text{ridge}}^{(\lambda)} = (X^T X + \lambda I_p)^{-1} X^T y \quad \lambda \in [0; \infty)$$

when $\lambda \in (0; \infty)$ $(X^T X + \lambda I_p)^{-1}$ exists

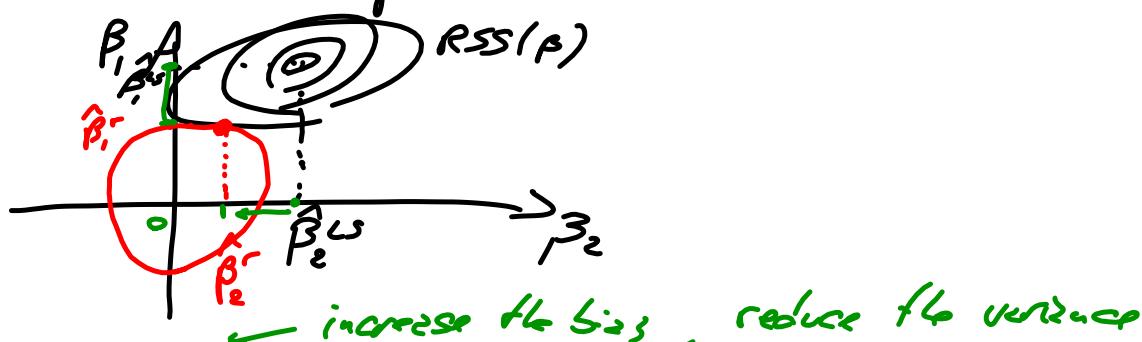
RIDGE REGRESSION AS A SHRINKAGE METHOD

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

alternative formulation

- $\hat{\beta}_{\text{ridge}}^{(\lambda)} = \underset{\beta}{\operatorname{argmin}} \left\{ \sum_{i=1}^n (y_i - \beta_0 - \sum_{j=1}^p x_{ij} \beta_j)^2 \right\}$
 subject to $\sum_{j=1}^p \beta_j^2 \leq t$

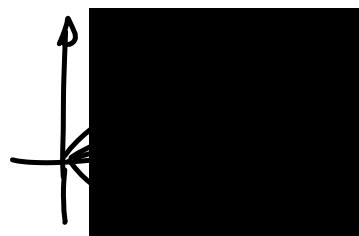
- one-to-one correspondence between λ and t



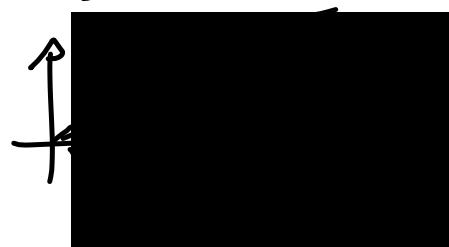
- from a Bayesian point of view, ridge estimator is the [redacted]



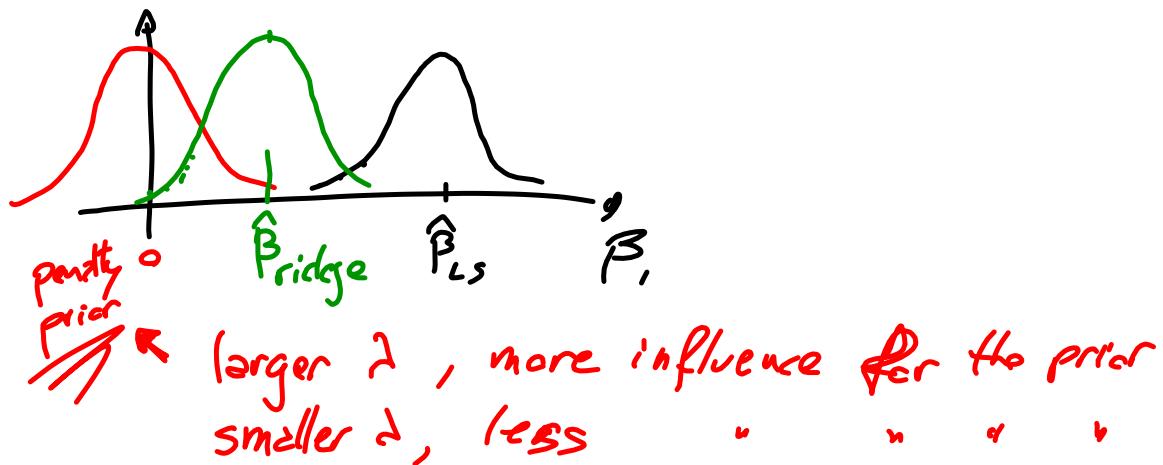
if X_s are uncorrelated



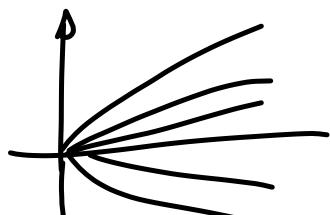
if X_s are correlated



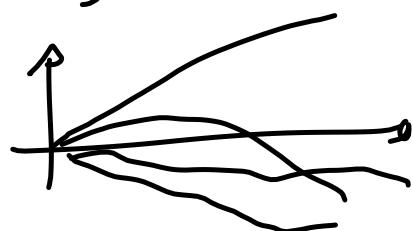
- from a Bayesian point of view, ridge estimator is the posterior mean/mode



if X_s are uncorrelated



if X_s are correlated



IMPORTANT! We need to [REDACTED]
our explanatory variables before applying
the ridge regression

$$\begin{aligned} \text{[REDACTED]} &= 0 \\ \text{[REDACTED]} &= 1 \end{aligned}$$

Expected value of ridge estimator

$$E[\hat{\beta}_{\text{ridge}}^{(j)}] = \frac{1}{1 + \lambda_j} \beta_j$$

$$\lambda \rightarrow 0$$

$$E[\hat{\beta}_{\text{ridge}}^{(j)}] = \frac{1}{1 + 0} \beta_j = \beta_j$$

$$\lambda \rightarrow \infty$$

$$E[\hat{\beta}_{\text{ridge}}^{(j)}] = \frac{1}{1 + \infty} \beta_j = 0$$

without intercept
 $\beta_0 = 0$

important

$$\lambda_a > \lambda_s \neq$$

$$\frac{1}{1 + \lambda_a} \beta_a \neq \frac{1}{1 + \lambda_s} \beta_s$$

IMPORTANT! We need to standardize our explanatory variables before applying the ridge regression

$$E[X_j] = 0$$

$$\text{Var}[X_j] = 1$$

Expected value of ridge estimator

$$\begin{aligned} E[\hat{\beta}_{\text{ridge}}^{(\lambda)}] &= E[(X^T X + \lambda I_p)^{-1} X^T y] \quad \hat{\beta}_{LS} \\ &= E[(I_p + \lambda (X^T X)^{-1})^{-1} (X^T X)^{-1} X^T y] \\ &= (I_p + \lambda (X^T X)^{-1})^{-1} E[\hat{\beta}_{LS}] \\ &= W_{\lambda} \beta \quad E[\hat{\beta}_{\text{ridge}}^{(\lambda)}] \neq \beta \end{aligned}$$

$$\lambda \rightarrow 0 \quad E[\hat{\beta}_{\text{ridge}}^{(\lambda)}] = \beta$$

$$\lambda \rightarrow \infty \quad E[\hat{\beta}_{\text{ridge}}^{(\lambda)}] = \sigma_{p+1} \leftarrow \begin{array}{l} \text{without intercept} \\ \beta_0 = 0 \end{array}$$

$$\text{important} \quad \lambda_a > \lambda_s \quad \Rightarrow |\hat{\beta}_j(\lambda_a)| < |\hat{\beta}_j(\lambda_s)|$$

due to correlation

Variance of the ridge estimator

$$\text{Var}[\hat{\beta}_{\text{ridge}}^{(1)}] = \frac{1}{1 - \frac{1}{1 + \frac{1}{n}}}$$

$$\text{Var}(\hat{\beta}_{LS}) - \text{Var}(\hat{\beta}_{\text{ridge}}^{(1)}) =$$

$$= \frac{1}{1 - \frac{1}{1 + \frac{1}{n}}}$$

Variance of the ridge estimator

$$\begin{aligned}\text{Var}(\hat{\beta}_{\text{ridge}}^{(s)}) &= \text{Var}(W_s \hat{\beta}_{LS}) \\ &= W_s \text{Var}(\hat{\beta}_{LS}) W_s^T \\ &= \sigma^2 W_s (X^T X)^{-1} W_s^T\end{aligned}$$

$$\begin{aligned}\text{Var}(\hat{\beta}_{LS}) - \text{Var}(\hat{\beta}_{\text{ridge}}^{(s)}) &= \sigma^2 \left[(X^T X)^{-1} - W_s (X^T X)^{-1} W_s^T \right] \\ &= \sigma^2 W_s \left[(\underline{I} + \lambda (X^T X)^{-1}) \underline{(X^T X)^{-1} (\underline{I} + \lambda (X^T X)^{-1})} - \underline{(X^T X)^{-1}} \right] W_s^T \\ &= \sigma^2 W_s \left[\cancel{(X^T X)^{-1}} + \lambda (X^T X)^{-2} + \lambda (X^T X)^{-2} + \lambda^2 (X^T X)^{-3} - \cancel{(X^T X)^{-1}} \right] W_s^T \\ &\stackrel{!}{=} \sigma^2 W_s \left[2\lambda (X^T X)^{-2} + \lambda^2 (X^T X)^{-3} \right] W_s^T > 0 \\ \Rightarrow \text{Var}(\hat{\beta}_{\text{ridge}}^{(s)}) &\leq \text{Var}(\hat{\beta}_{LS})\end{aligned}$$

Degrees of freedom

$$\hat{Y}_{LS} = \underbrace{\mathbf{X}(X^T X)^{-1} X^T g}_H \quad df = \text{tr}(H) = p$$

$$\hat{Y}_{RIDGE} = \underbrace{\mathbf{X}(X^T X + \lambda I)^{-1} X^T g}_{H_\lambda} \quad \text{effective } df = \text{tr}(H_\lambda)$$

$$\text{tr}(H_\lambda) = \sum_{j=1}^p \frac{d_j^{-2}}{d_j^{-2} + \lambda}$$

$$\lambda = 0 \rightarrow$$

$$\lambda \rightarrow \infty \rightarrow$$

$$AIC : L(\beta) + 2J(\theta)$$

$$\text{least square} : -2\ell(\beta) +$$

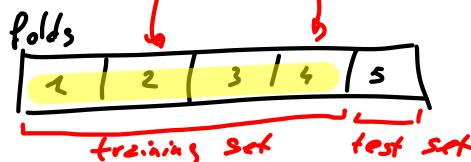
$$\text{ridge} : -2\ell(\beta) +$$

AIC(λ) we can find $\hat{\lambda}$ es es

usually λ is computed by

sample: x_1, x_2, \dots, x_n

split in K folds ($K=5$ or $K=10$)



$$\text{err} = \sum_{i=1}^n (y_{(5)} - x_{(5)} \hat{\beta}_{(5)})^2$$

train the model on $K-1$ folds $\rightarrow \hat{\beta}_{(5)}$
evaluate its performance on the remaining fold err_5

repeat the procedure for K times, always using a different fold as test set

$\hat{\beta}$ computed by using all observations not belonging to fold 5

err_1 : train set $\{2, 3, 4, 5\}$
test set $\{1\}$

err_2 : train set $\{1, 3, 4, 5\}$
test set $\{2\}$

:

err_5

$$\sum_{K=1}^5 \text{err}_K(\lambda)$$

λ that minimizes

Degrees of freedom

$$\hat{Y}_{LS} = \underbrace{\mathbf{X}(X^T X)^{-1} X^T g}_H \quad df = \text{tr}(H) = p$$

$$\hat{Y}_{RIDGE} = \underbrace{\mathbf{X}(X^T X + \lambda I)^{-1} X^T g}_{H_\lambda} \quad \begin{matrix} \text{effective} \\ df = \text{tr}(H_\lambda) \end{matrix}$$

$$\text{tr}(H_\lambda) = \sum_{j=1}^p \frac{d_j^{-2}}{d_j^{-2} + \lambda} \quad \begin{matrix} \lambda = 0 \rightarrow df = p \\ \lambda \rightarrow \infty \rightarrow df \rightarrow 0 \end{matrix}$$

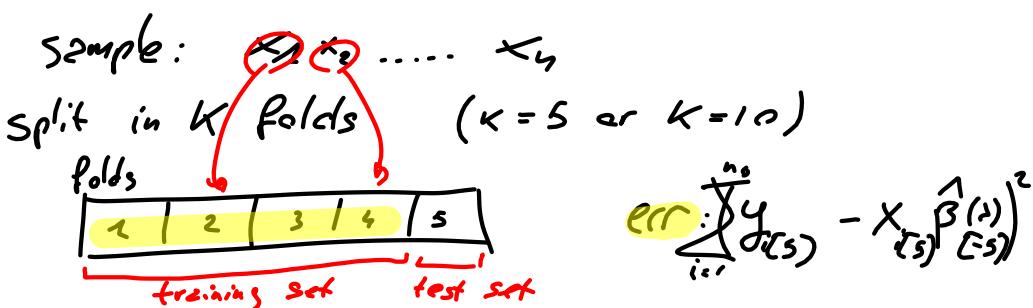
$$AIC : L(\beta) + 2J(\theta)$$

$$\text{least square} : -2\ell(\beta) + 2p$$

$$\text{ridge} : -2\ell(\beta) + 2 \sum \frac{d_j^{-2}}{d_j^{-2} + \lambda}$$

$AIC(\lambda)$ we can find $\hat{\lambda}$ es $\arg \min_{\lambda} AIC(\lambda)$

usually λ is computed by cross-validation



train the model on $K-1$ folds $\rightarrow \hat{\beta}_{(5)}$
evaluate its performance on the remaining fold err_5

- repeat the procedure for K times, always using a different fold as test set

$\hat{\beta}$ computed by using all observation not belonging to fold 5

err_1 : train set $\{2, 3, 4, 5\}$
test set $\{1\}$

err_2 : train set $\{1, 3, 4, 5\}$
test set $\{2\}$

:

err_5

$$\sum_{K=1}^5 err_K(\lambda) \quad \lambda \text{ that minimizes}$$

More about shrinkage

$$X = UDV^T$$

$U = n \times n$ orthogonal matrix, whose columns span the column space of X

$V = p \times n$ orthogonal matrix, whose columns span the row space
of X

$D = n \times n$ diagonal matrix, d_{ii} are single value of X_i .

$$U^T U = I_n = V^T V$$

LS estimator $\hat{\beta}_{LS}$

$\hat{\beta}_{\zeta s}$

the first time in the history of the world, the people of the United States have been called upon to determine whether they will submit to the law of force, or the law of the Constitution. We consider the question to be, whether the Southern Slaveholding States have a right to secede from the Federal Union; and, if so, whether the Federal Government has a right to suppress them by force. The former question is the more important, because it is the only one that can be decided by the people themselves. The latter question is of less importance, because it can only be decided by the Federal Government, and the people have no voice in it. The former question is also the more difficult, because it requires a knowledge of the Constitution, and the latter question is easier, because it only requires a knowledge of the law of force.

$$\hat{y}_{1,5} = - - - =$$

$$\frac{d_1}{d_2}$$

$\hat{\beta}_{\text{ridge}}^{(1)}$

卷之三

六

$$\sum_{j=1}^p \frac{d_j}{d_j^2 + \epsilon}$$

See page 66

More about shrinkage

$$\underset{n \times p}{X} = UDV^T$$

$U = n \times n$ orthogonal matrix, whose columns span the column space of X

$V = p \times n$ orthogonal matrix, whose columns span the row space of X

$D = n \times n$ diagonal matrix, d_{ii} are singular values of X :

$$U^T U = I_n = VV^T$$

LS estimator $\hat{\beta}_{LS}$

$$\begin{aligned}\hat{\beta}_{LS} &= (X^T X)^{-1} X^T y \\ &= (V D U^T U D V^T)^{-1} V D U^T y \\ &= (V D^2 V^T)^{-1} V D U^T y \\ &= V D^2 V^T V D U^T y \\ &= V D^2 \underset{\lambda}{\underbrace{D^T D}} U^T y\end{aligned}$$

$$\begin{aligned}\hat{y}_{LS} &= X \hat{\beta}_{LS} \\ &= U D V^T V \underset{\lambda}{\underbrace{D^{-2} D^T D}} U^T y \\ &= U D^2 D^{-2} U^T y \\ &= U U^T y\end{aligned}$$

$$\sum_{j=1}^p \frac{d_j}{d_j^2}$$

$$\begin{aligned}\hat{\beta}_{Ridge}^{(\lambda)} &= (X^T X + \lambda I_p)^{-1} X^T y \\ &= (V D U^T U D V^T + \lambda I_p)^{-1} V D U^T y \\ &= (V D^2 V^T + \lambda V V^T) V D U^T y \\ &= V (\lambda I_p + \underset{\lambda}{\underbrace{D^T D}}) V^T V D U^T y \\ &= V (\lambda I_p + \underset{\lambda}{\underbrace{D^T D}})^{-1} D U^T y\end{aligned}$$

$$\begin{aligned}\hat{y}_{Ridge} &= X \hat{\beta}_{Ridge} \\ &= U D V^T V (\lambda I_p + \underset{\lambda}{\underbrace{D^T D}})^{-1} D U^T y \\ &= U D^2 (\lambda I_p + \underset{\lambda}{\underbrace{D^T D}})^{-1} U^T y\end{aligned}$$

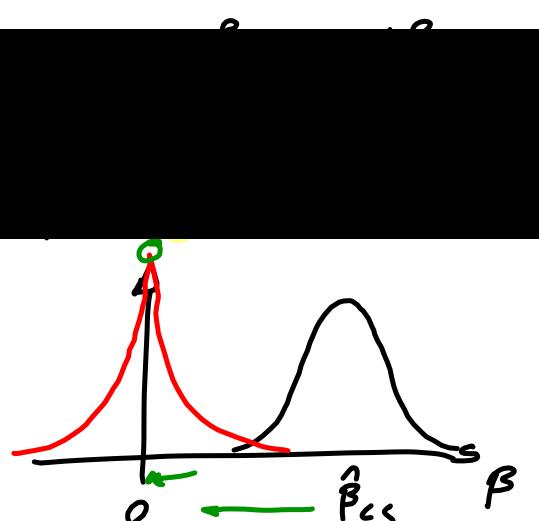
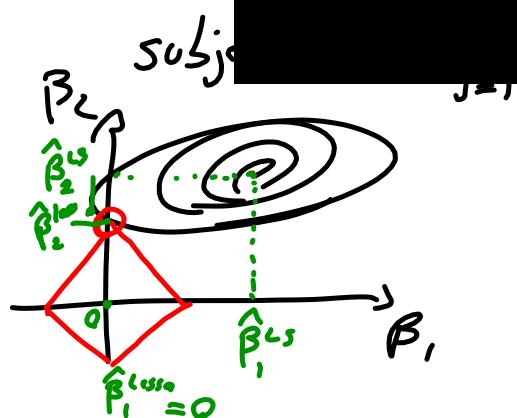
$$\sum_{j=1}^p \frac{d_j}{d_j^2 + \lambda}$$

LASSO

$$\hat{\beta}_{\text{LASSO}} = \arg \min_{\beta} \|y - X\beta\|_2^2 + \lambda \|\beta\|_1$$

or, equivalently

$$\hat{\beta}_{\text{LASSO}} = \arg \min_{\beta} \|y - X\beta\|_2^2 + \lambda \sum_j |\beta_j|$$



$$\hat{\beta}_{\text{pen}} = \arg \min_{\beta} \left\{ \sum_i (y_i - \beta_0 - \sum_j x_{ij} \beta_j)^2 + \lambda \sum_j |\beta_j|^q \right\}$$

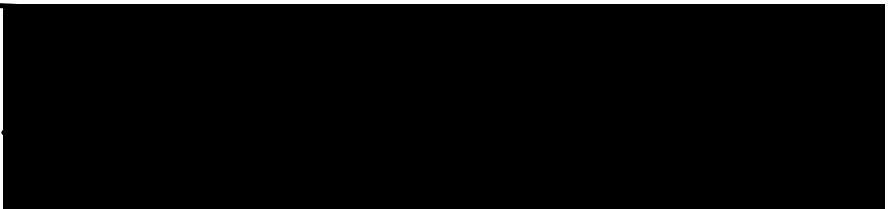
$q=1 \rightarrow \text{LASSO}$

$q=2 \rightarrow \text{RIDGE REGRESSION}$

$1 < q < 2 \approx \text{average between Lasso and ridge penalties}$

ELASTIC NET

penalty :



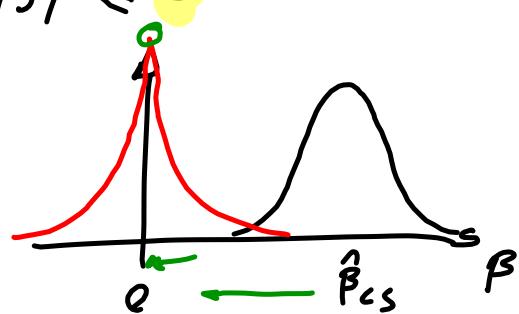
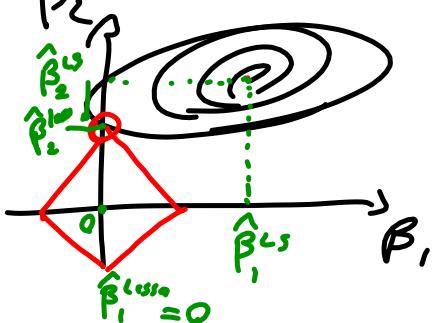
LASSO

$$\hat{\beta}_{\text{LASSO}} = \arg \min_{\beta} \left\{ \sum_{i=1}^n (y_i - \beta_0 - \sum_{j=1}^p x_{ij} \beta_j)^2 + \lambda \sum_{j=1}^p |\beta_j| \right\}$$

or, equivalently

$$\hat{\beta}_{\text{LASSO}} = \arg \min_{\beta} \left\{ \sum_{i=1}^n (y_i - \beta_0 - \sum_{j=1}^p x_{ij} \beta_j)^2 \right\}$$

subject to $\sum_{j=1}^p |\beta_j| \leq t$



$$\hat{\beta}_{\text{pen}} = \arg \min_{\beta} \left\{ \sum_i (y_i - \beta_0 - \sum_j x_{ij} \beta_j)^2 + \lambda \sum_j |\beta_j|^q \right\}$$

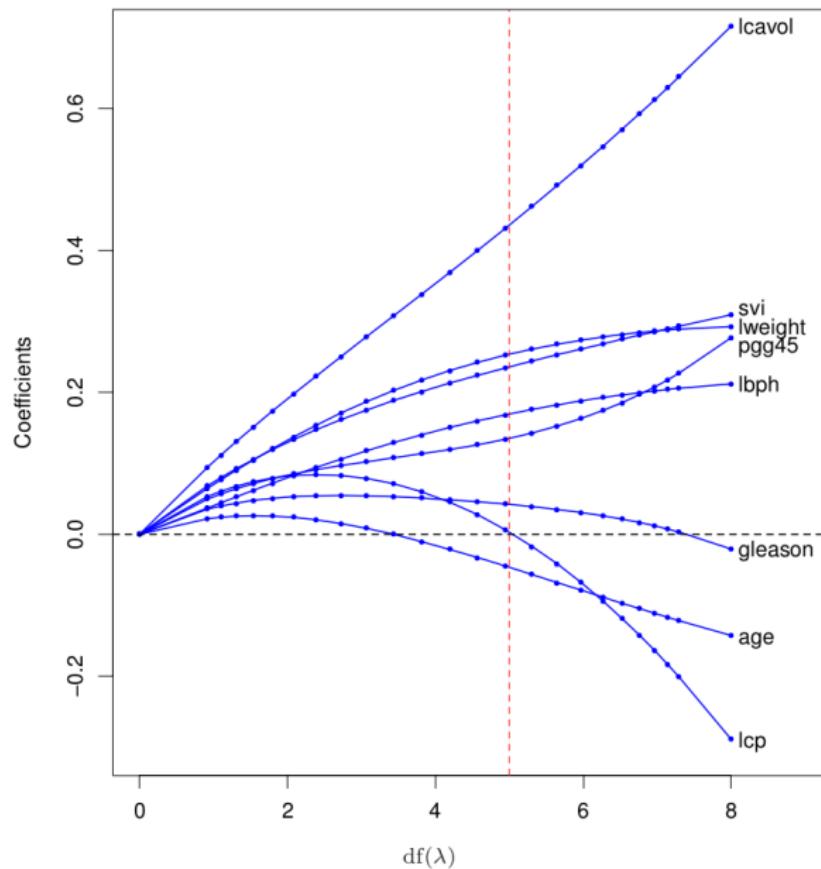
$q=1 \rightarrow \text{LASSO}$

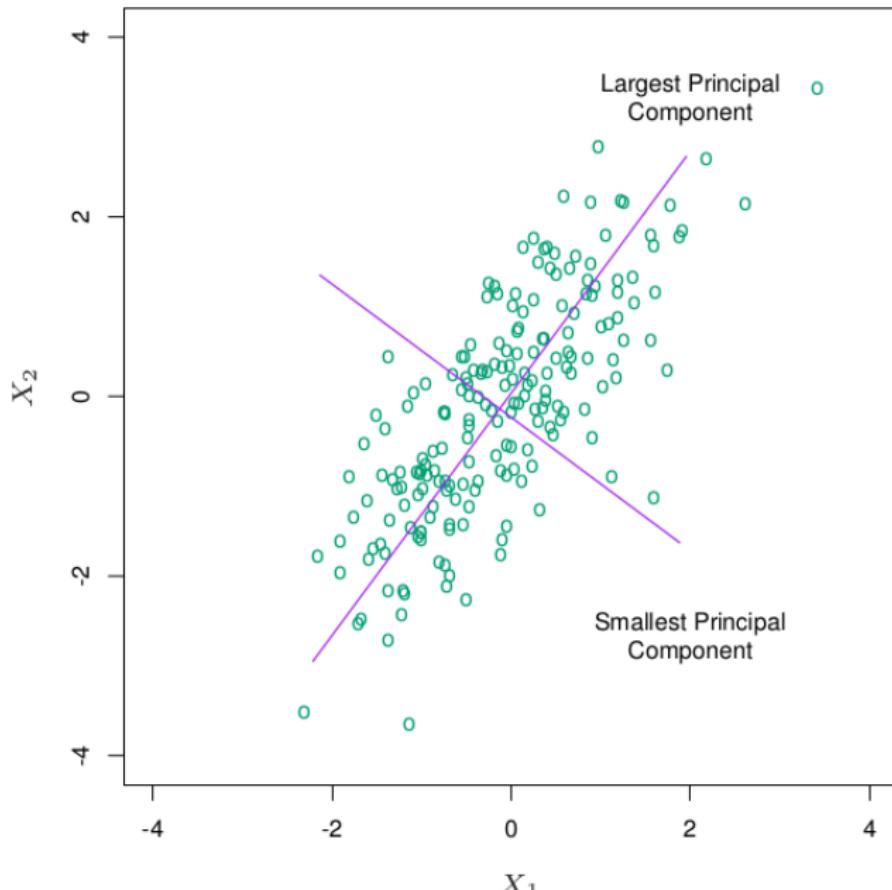
$q=2 \rightarrow \text{RIDGE REGRESSION}$

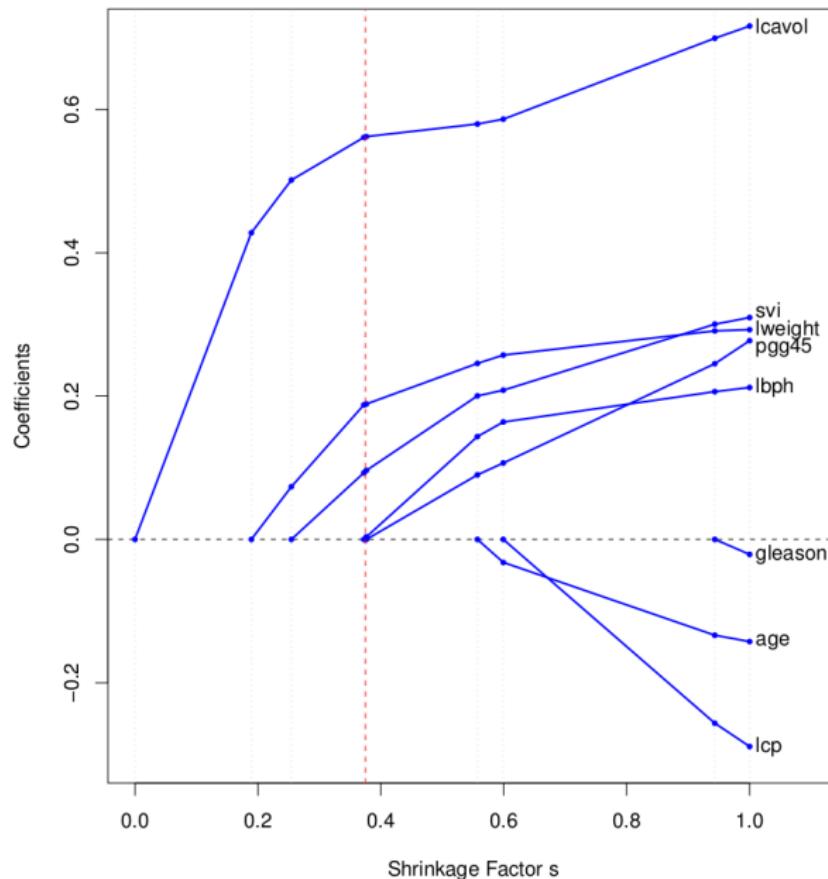
$1 < q < 2 \approx \text{average between Lasso and ridge penalties}$

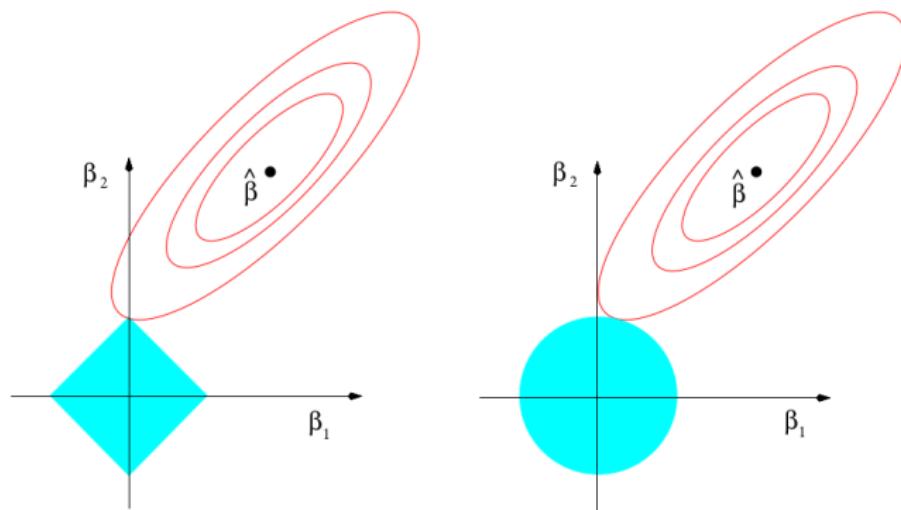
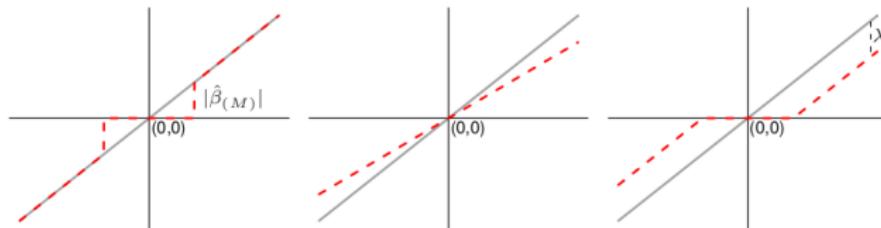
ELASTIC NET

$$\text{penalty : } \lambda \left(\alpha \sum_{j=1}^p \beta_j^2 + (1-\alpha) \sum_j |\beta_j| \right)$$









Exercise 3.6

$$P(\beta | y) \propto$$

where

$$p(y|\beta) \rightarrow N(X\beta, \sigma^2 I)$$

$$p(\beta) \rightarrow N(0, \tau^2 I)$$

$$\frac{1}{\sqrt{p} \cdot \sigma} \cdot \text{const}$$

$$P(\beta | y) \propto$$

$\frac{1}{\sqrt{\sigma^2 + \tau^2 \beta^T \beta}}$
d.e.x
L

Exercise 3.10

$$F = \frac{(RSS_0 - RSS_1) / (\underbrace{p_1 - p_0}_{=1})}{RSS_1 / (N - p_1 - 1)}$$

→ find the β_j s.t. $\beta_j = 0$ lead to the smallest $RSS_0 - RSS_1$

we know (ex 3.1), $F_{1, N-p_1-1} \stackrel{d}{=} z_j^2$

⇒ the β_j which, when set equal to 0, increases the least the RSS is that with

Exercise 3.6

$$P(\beta | y) \propto p(y | \beta) p(\beta)$$

where

$$p(y | \beta) \hookrightarrow N(X\beta, \sigma^2 I)$$

$$p(\beta) \hookrightarrow N(0, \tau^2 I)$$

$$\frac{1}{\sqrt{\beta^\top \beta + \sigma^2 n}} \text{ const}$$

$$P(\beta | y) \propto \text{const.} \exp \left\{ -\frac{1}{2\sigma^2} (y - X\beta)^\top (y - X\beta) \right\} \exp \left\{ -\frac{1}{2\tau^2} \beta^\top \beta \right\}$$

$$\propto \exp \left\{ (y - X\beta)^\top (y - X\beta) + \frac{\sigma^2}{\tau^2} \beta^\top \beta \right\}$$

OBS: Feil i siste linje.

Mode = Max posterior = Max av det i klammen = Ridge-estimatet!

Exercise 3.10

$$F = \frac{(RSS_0 - RSS_1) / (\underbrace{p_1 - p_0}_{=1})}{RSS_1 / (n - p_1 - 1)}$$

\rightarrow find the β_j s.t. $\beta_j = 0$ lead to the smallest $RSS_0 - RSS_1$

we know (ex 3.1), $F_{1, n-p_1-1} \stackrel{d}{=} z_j^2$

\Rightarrow the β_j which, when set equal to 0, increases the least the RSS is that with the smallest z_j^2 .

until all the variables are in the model, and ends at the full least-squares fit. Algorithm 3.2 provides the details. The termination condition in step 5 requires some explanation. If $p > N - 1$, the LAR algorithm reaches a zero residual solution after $N - 1$ steps (the -1 is because we have centered the data).

Algorithm 3.2 Least Angle Regression.

- 1.
- 2.
- 3.
- 4.
- 5.

Suppose \mathcal{A}_k is the active set of variables at the beginning of the k th step, and let $\beta_{\mathcal{A}_k}$ be the coefficient vector for these variables at this step; there will be $k - 1$ nonzero values, and the one just entered will be zero. If $\mathbf{r}_k = \mathbf{y} - \mathbf{X}_{\mathcal{A}_k} \beta_{\mathcal{A}_k}$ is the current residual, then the direction for this step is

$$\delta_k = (\mathbf{X}_{\mathcal{A}_k}^T \mathbf{X}_{\mathcal{A}_k})^{-1} \mathbf{X}_{\mathcal{A}_k}^T \mathbf{r}_k. \quad (3.55)$$

The coefficient profile then evolves as $\beta_{\mathcal{A}_k}(\alpha) = \beta_{\mathcal{A}_k} + \alpha \cdot \delta_k$. Exercise 3.23 verifies that the directions chosen in this fashion do what is claimed: keep the correlations tied and decreasing. If the fit vector at the beginning of this step is $\hat{\mathbf{f}}_k$, then it evolves as $\hat{\mathbf{f}}_k(\alpha) = \hat{\mathbf{f}}_k + \alpha \cdot \mathbf{u}_k$, where $\mathbf{u}_k = \mathbf{X}_{\mathcal{A}_k} \delta_k$ is the new fit direction. The name “least angle” arises from a geometrical interpretation of this process; \mathbf{u}_k makes the smallest (and equal) angle with each of the predictors in \mathcal{A}_k (Exercise 3.24). Figure 3.14 shows the absolute correlations decreasing and joining ranks with each step of the LAR algorithm, using simulated data.

By construction the coefficients in LAR change in a piecewise linear fashion. Figure 3.15 [left panel] shows the LAR coefficient profile evolving as a function of their L_1 arc length ². Note that we do not need to take small

²The L_1 arc-length of a differentiable curve $\beta(s)$ for $s \in [0, S]$ is given by $\text{TV}(\beta, S) = \int_0^S \|\dot{\beta}(s)\|_1 ds$, where $\dot{\beta}(s) = \partial \beta(s) / \partial s$. For the piecewise-linear LAR coefficient profile, this amounts to summing the L_1 norms of the changes in coefficients from step to step.

until all the variables are in the model, and ends at the full least-squares fit. Algorithm 3.2 provides the details. The termination condition in step 5 requires some explanation. If $p > N - 1$, the LAR algorithm reaches a zero residual solution after $N - 1$ steps (the -1 is because we have centered the data).

Algorithm 3.2 Least Angle Regression.

1. Standardize the predictors to have mean zero and unit norm. Start with the residual $\mathbf{r} = \mathbf{y} - \bar{\mathbf{y}}$, $\beta_1, \beta_2, \dots, \beta_p = 0$.
 2. Find the predictor \mathbf{x}_j most correlated with \mathbf{r} .
 3. Move β_j from 0 towards its least-squares coefficient $\langle \mathbf{x}_j, \mathbf{r} \rangle$, until some other competitor \mathbf{x}_k has as much correlation with the current residual as does \mathbf{x}_j .
 4. Move β_j and β_k in the direction defined by their joint least squares coefficient of the current residual on $(\mathbf{x}_j, \mathbf{x}_k)$, until some other competitor \mathbf{x}_l has as much correlation with the current residual.
 5. Continue in this way until all p predictors have been entered. After $\min(N - 1, p)$ steps, we arrive at the full least-squares solution.
-

Suppose \mathcal{A}_k is the active set of variables at the beginning of the k th step, and let $\beta_{\mathcal{A}_k}$ be the coefficient vector for these variables at this step; there will be $k - 1$ nonzero values, and the one just entered will be zero. If $\mathbf{r}_k = \mathbf{y} - \mathbf{X}_{\mathcal{A}_k} \beta_{\mathcal{A}_k}$ is the current residual, then the direction for this step is

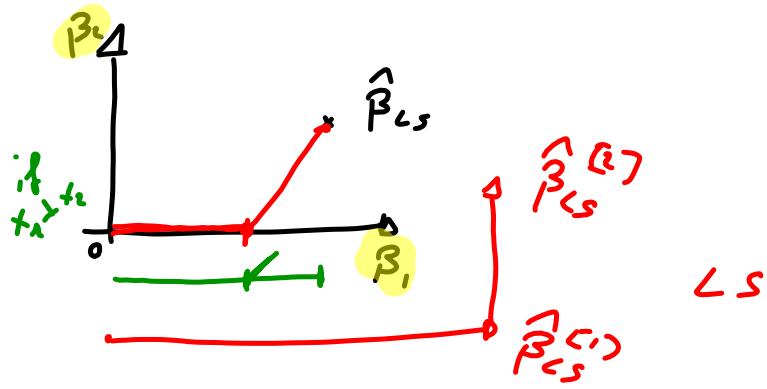
$$\delta_k = (\mathbf{X}_{\mathcal{A}_k}^T \mathbf{X}_{\mathcal{A}_k})^{-1} \mathbf{X}_{\mathcal{A}_k}^T \mathbf{r}_k. \quad (3.55)$$

The coefficient profile then evolves as $\beta_{\mathcal{A}_k}(\alpha) = \beta_{\mathcal{A}_k} + \alpha \cdot \delta_k$. Exercise 3.23 verifies that the directions chosen in this fashion do what is claimed: keep the correlations tied and decreasing. If the fit vector at the beginning of this step is $\hat{\mathbf{f}}_k$, then it evolves as $\hat{\mathbf{f}}_k(\alpha) = \hat{\mathbf{f}}_k + \alpha \cdot \mathbf{u}_k$, where $\mathbf{u}_k = \mathbf{X}_{\mathcal{A}_k} \delta_k$ is the new fit direction. The name “least angle” arises from a geometrical interpretation of this process; \mathbf{u}_k makes the smallest (and equal) angle with each of the predictors in \mathcal{A}_k (Exercise 3.24). Figure 3.14 shows the absolute correlations decreasing and joining ranks with each step of the LAR algorithm, using simulated data.

By construction the coefficients in LAR change in a piecewise linear fashion. Figure 3.15 [left panel] shows the LAR coefficient profile evolving as a function of their L_1 arc length ². Note that we do not need to take small

²The L_1 arc-length of a differentiable curve $\beta(s)$ for $s \in [0, S]$ is given by $\text{TV}(\beta, S) = \int_0^S \|\dot{\beta}(s)\|_1 ds$, where $\dot{\beta}(s) = \partial \beta(s) / \partial s$. For the piecewise-linear LAR coefficient profile, this amounts to summing the L_1 norms of the changes in coefficients from step to step.

Least angle regression



we update $\hat{\beta}_1$ until
 $\text{cor}(x_1, r) < \text{cor}(x_2, r)$
 then
 we update both $(\hat{\beta}_1, \hat{\beta}_2)$

$$r = y - \bar{y} - \sum_{j=1}^p x_j \hat{\beta}_j$$

- start with $r = y - \bar{y}$
- check the largest correlation between x_j and r
- update the regression coefficient of x_j ($\hat{\beta}_j$) until $\langle x_j, r \rangle$

- first we update $\hat{\beta}_1$

Suppose that $\hat{\beta}_j$ are ordered by importance, i.e., effect of x_i is larger than effect of x_j , $i < j$

- we reach a point where we add also $\hat{\beta}_2$ to our solution
- we are updating $\{\hat{\beta}_1, \hat{\beta}_2\}$

- we reach a point where the update is related to $\{\hat{\beta}_1, \hat{\beta}_2, \hat{\beta}_3\}$

:

once a coefficient enters in the set of active regression coefficients, it stays

FORWARD REGRESSION

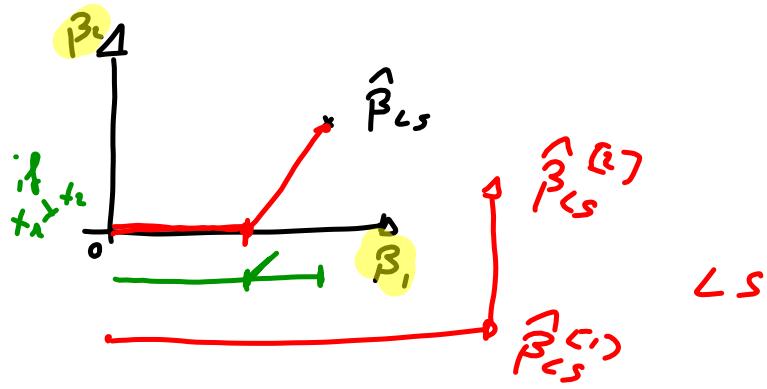
1st step $\hat{y} = \hat{\beta}_0 + \hat{\beta}_{LS}^{(1)} x_1$

LAR

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1^{(1)} x_1$$

- Lasso can be seen as a special case of a modification

Least angle regression



we update $\hat{\beta}_1$ until
 $\text{cor}(x_1, r) < \text{cor}(x_2, r)$
 then
 we update both $(\hat{\beta}_1, \hat{\beta}_2)$

$$r = y - \bar{y} - \sum_{j=1}^p x_j \hat{\beta}_j$$

- start with $r = y - \bar{y}$
- check the largest correlation between x_j and r
- update the regression coefficient of x_j ($\hat{\beta}_j$) until $\langle x_j, r \rangle$

- first we update $\hat{\beta}_1$

Suppose that $\hat{\beta}_j$ are ordered by importance, i.e., effect of x_i is larger than effect of x_j , $i < j$

- we reach a point where we add also $\hat{\beta}_2$ to our solution
- we are updating $\{\hat{\beta}_1, \hat{\beta}_2\}$

- we reach a point where the update is related to $\{\hat{\beta}_1, \hat{\beta}_2, \hat{\beta}_3\}$

:

once a coefficient enters in the set of active regression coefficients, it stays

FORWARD REGRESSION

1st step $\hat{y} = \hat{\beta}_0 + \hat{\beta}_{LS}^{(1)} x_1$

LAR

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1^{(1)} x_1$$

- Lasso can be seen as a special case of a modification of LAR, for which once the estimate of a regression coeff. reach 0, it is excluded by the set of active regression coefficient

METHODS USING DERIVED INPUT DIRECTIONS

- Principal component regression
- Partial least squares

Principal component regression

IDEA: inputs have different variabilities in different directions
 → directions with largest variability provide more information

↓ principal components

linear combinations of X based on directions of largest variability

$$z_m = X v_m$$

z_1 = direction with the largest variability

z_2 = " " the 2nd largest variability s.t. $e_1 \perp e_2$

:

z_p = " " the least variability $e_p \perp z_1, \dots, z_{p-1}$

Model

$$y = \theta_0$$

$$\hat{\theta}_0 =$$

$$\hat{\theta}_m =$$

NOTE

principal component analysis is scale dependent

↓
 IMPORTANT to first standardize X

for each x_j

$$x_{ij}^{st} = \frac{(x_{ij} - \bar{x}_j)}{sd(x_j)}$$

METHODS USING DERIVED INPUT DIRECTIONS

- Principal component regression
- Partial least squares

Principal component regression

IDEA: inputs have different variabilities in different directions
 → directions with largest variability provide more information

↓ principal components

linear combinations of X based on directions of largest variability

$$z_m = X v_m$$

v_m = eigenvectors
 z_1 = direction with the largest variability

z_2 = " " the 2nd largest variability s.t. $e_1 \perp e_2$

:

z_p = " " the least variability $e_p \perp z_1, \dots, z_{p-1}$

Model

$$y = \theta_0 + \sum_{m=1}^M \theta_m z_m + \varepsilon$$

$$\hat{\theta}_0 = \bar{y}$$

$$\hat{\theta}_m = \frac{\langle z_m, y \rangle}{\langle z_m, z_m \rangle}$$

NOTE

principal component analysis is scale dependent



IMPORTANT to first standardize X

for each x_j

$$x_{ij}^{st} = \frac{(x_{ij} - \bar{x}_j)}{sd(x_j)}$$

$$\hat{y} = \hat{\theta}_0 + \sum_{m=1}^M \hat{\theta}_m z_m$$

$$z_m = X v_m$$

$$\begin{aligned} &= \hat{\theta}_0 + \boxed{\quad} \\ &= \hat{\theta}_0 + \boxed{\quad} \\ &= \hat{\theta}_0 + \boxed{\quad} \\ &= \hat{\theta}_0 + X \hat{\beta}_{PCR} \end{aligned}$$

if $M=p$, principal component regression provides the least square estimates, because all p principal component span the space of X

- idea: use $M < p$ principal components, to exclude directions with less information

we obtain results similar to ridge regression

$$\hat{\beta}_{\text{ridge}} = \sum_{j=1}^p$$

$$\hat{\beta}_{PCR} = \sum_{j=1}^M$$

$$\hat{\beta}_{LS} = \sum_{j=1}^p u_j u_j^\top y$$

$$y = \sum_{j=1}^M u_j u_j^\top y$$

$$\begin{aligned}
 \hat{y} &= \hat{\theta}_0 + \sum_{m=1}^M \hat{\theta}_m z_m & z_m &= X v_m \\
 &= \hat{\theta}_0 + \sum_{m=1}^M \hat{\theta}_m X v_m \\
 &= \hat{\theta}_0 + X \sum_{m=1}^M \hat{\theta}_m v_m \\
 &= \hat{\theta}_0 + X \hat{\beta}_{PCR}
 \end{aligned}$$

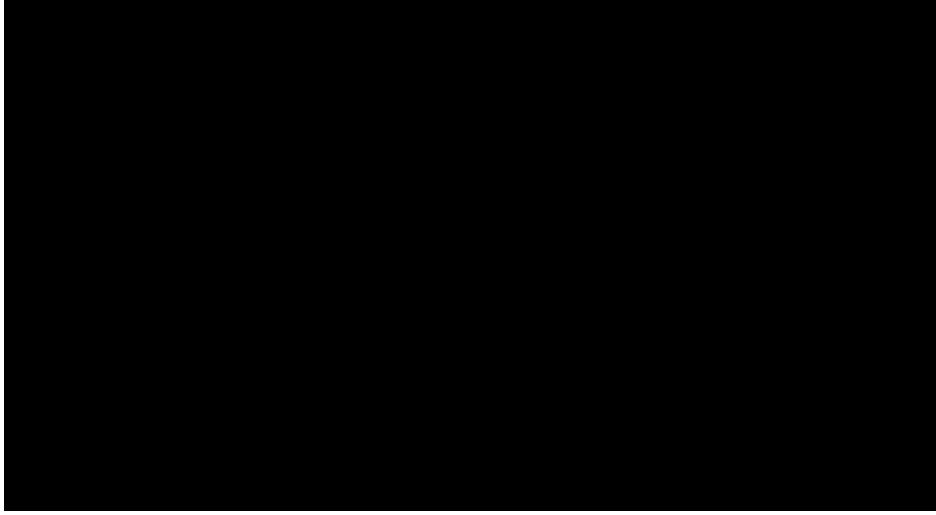
- idea: use $M < p$ principal components, to exclude directions with less information

if $M = p$, principal component regression provides the least square estimates, because all p principal component span the space of X

we obtain results similar to ridge regression

$$\hat{\beta}_{\text{ridge}} = \sum_{j=1}^p u_j \frac{d_j^2}{d_j^2 + \lambda} u_j^T y \quad \hat{\beta}_{LS} = \sum_{j=1}^p u_j u_j^T y$$

$$\hat{\beta}_{PCR} = \sum_{j=1}^M u_j \underbrace{\mathbb{1}_{[j \leq M]}}_{\text{red box}} u_j^T y = \sum_{j=1}^M u_j u_j^T y$$

Algorithm 3.3 Partial Least Squares.

In the prostate cancer example, cross-validation chose $M = 2$ PLS directions in Figure 3.7. This produced the model given in the rightmost column of Table 3.3.

What optimization problem is partial least squares solving? Since it uses the response \mathbf{y} to construct its directions, its solution path is a nonlinear function of \mathbf{y} . It can be shown (Exercise 3.15) that partial least squares seeks directions that have high variance *and* have high correlation with the response, in contrast to principal components regression which keys only on high variance (Stone and Brooks, 1990; Frank and Friedman, 1993). In particular, the m th principal component direction v_m solves:

$$\begin{aligned} & \max_{\alpha} \text{Var}(\mathbf{X}\alpha) \\ & \text{subject to } \|\alpha\| = 1, \alpha^T \mathbf{S}v_\ell = 0, \ell = 1, \dots, m-1, \end{aligned} \tag{3.63}$$

where \mathbf{S} is the sample covariance matrix of the \mathbf{x}_j . The conditions $\alpha^T \mathbf{S}v_\ell = 0$ ensures that $\mathbf{z}_m = \mathbf{X}\alpha$ is uncorrelated with all the previous linear combinations $\mathbf{z}_\ell = \mathbf{X}v_\ell$. The m th PLS direction $\hat{\varphi}_m$ solves:

$$\begin{aligned} & \max_{\alpha} \text{Corr}^2(\mathbf{y}, \mathbf{X}\alpha) \text{Var}(\mathbf{X}\alpha) \\ & \text{subject to } \|\alpha\| = 1, \alpha^T \mathbf{S}\hat{\varphi}_\ell = 0, \ell = 1, \dots, m-1. \end{aligned} \tag{3.64}$$

Further analysis reveals that the variance aspect tends to dominate, and so partial least squares behaves much like ridge regression and principal components regression. We discuss this further in the next section.

If the input matrix \mathbf{X} is orthogonal, then partial least squares finds the least squares estimates after $m = 1$ steps. Subsequent steps have no effect

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 $\mathbf{x}_j^{(0)} = \mathbf{x}_j$, $j = 1, \dots, p$.
2. For $m = 1, 2, \dots, p$
 - (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 : $\mathbf{x}_j^{(m)} = \mathbf{x}_j^{(m-1)} - [\langle \mathbf{z}_m, \mathbf{x}_j^{(m-1)} \rangle / \langle \mathbf{z}_m, \mathbf{z}_m \rangle] \mathbf{z}_m$, $j = 1, 2, \dots, p$.
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 $\hat{\mathbf{y}}^{(m)} = \mathbf{X}\hat{\beta}^{pls}(m)$. These linear coefficients can be recovered from the sequence of PLS transformations.

In the prostate cancer example, cross-validation chose $M = 2$ PLS directions in Figure 3.7. This produced the model given in the rightmost column of Table 3.3.

What optimization problem is partial least squares solving? Since it uses the response \mathbf{y} to construct its directions, its solution path is a nonlinear function of \mathbf{y} . It can be shown (Exercise 3.15) that partial least squares seeks directions that have high variance *and* have high correlation with the response, in contrast to principal components regression which keys only on high variance (Stone and Brooks, 1990; Frank and Friedman, 1993). In particular, the m th principal component direction v_m solves:

$$\begin{aligned} & \max_{\alpha} \text{Var}(\mathbf{X}\alpha) \\ & \text{subject to } \|\alpha\| = 1, \alpha^T \mathbf{S}v_\ell = 0, \ell = 1, \dots, m-1, \end{aligned} \tag{3.63}$$

where \mathbf{S} is the sample covariance matrix of the \mathbf{x}_j . The conditions $\alpha^T \mathbf{S}v_\ell = 0$ ensures that $\mathbf{z}_m = \mathbf{X}\alpha$ is uncorrelated with all the previous linear combinations $\mathbf{z}_\ell = \mathbf{X}v_\ell$. The m th PLS direction $\hat{\varphi}_m$ solves:

$$\begin{aligned} & \max_{\alpha} \text{Corr}^2(\mathbf{y}, \mathbf{X}\alpha) \text{Var}(\mathbf{X}\alpha) \\ & \text{subject to } \|\alpha\| = 1, \alpha^T \mathbf{S}\hat{\varphi}_\ell = 0, \ell = 1, \dots, m-1. \end{aligned} \tag{3.64}$$

Further analysis reveals that the variance aspect tends to dominate, and so partial least squares behaves much like ridge regression and principal components regression. We discuss this further in the next section.

If the input matrix \mathbf{X} is orthogonal, then partial least squares finds the least squares estimates after $m = 1$ steps. Subsequent steps have no effect

Partial least squares

- in the construction of principal components, we do not take into account y

- ↓
- in the construction of derived input directions, we also consider y

as for principal component regression it is important to first standardize X

$$1^{\text{st}} \text{ step: } \hat{\varphi}_{1j} = \frac{\langle x_j, y \rangle}{\langle x_j, x_j \rangle} \quad \rightarrow \underline{z_1} = \sum_{j=1}^p \hat{\varphi}_{1j} x_j$$

$$\hat{\theta}_1 = \frac{\langle e_1, y \rangle}{\langle e_1, e_1 \rangle}$$

$$\underline{x}_j^{(2)} = x_j - \frac{\langle z_1, x_j \rangle}{\langle z_1, z_1 \rangle} \quad j = 1, \dots, p$$

$$\hat{\varphi}_{2j} = \frac{\langle x_j^{(2)}, y \rangle}{\langle x_j^{(2)}, x_j^{(2)} \rangle}$$

$$\underline{z}_2 = \sum_{j=1}^p \hat{\varphi}_{2j} \underline{x}_j^{(2)}$$

$$\hat{\theta}_2 = \frac{\langle e_2, y \rangle}{\langle e_2, e_2 \rangle}$$

Differences:

PCR :

PLS :

Mathematically

PCR

$$\max_{\alpha} \text{Var}(X\alpha) \quad \text{s.t. } \|\alpha\| = 1$$

$$\alpha^T S V e = 0 \quad l = 1, \dots, m$$

sample covariance matrix

PLS

$$\max_{\alpha} \text{Cor}^2(y, X\alpha) \text{Var}(X\alpha)$$

uncorrelated directions

In practice, the s.t. $\|\alpha\| = 1$

variance component tends to dominate the correlation one

$$\alpha^T S \hat{\varphi}_l = 0 \quad l = 1, \dots, m-1$$

→ similar results!

Partial least squares

- in the construction of principal components, we do not take into account y

↓
- in the construction of derived input directions, we also consider y

as for principal component regression it is important to first standardize X

1st step: $\hat{\varphi}_{1j} = \frac{\langle x_j, y \rangle}{\langle x_j, x_j \rangle}$ → $\underline{z}_1 = \sum_{j=1}^p \hat{\varphi}_{1j} x_j$
 $\hat{\theta}_1 = \frac{\langle \underline{z}_1, y \rangle}{\langle \underline{z}_1, \underline{z}_1 \rangle}$

$$\underline{x}_j^{(2)} = x_j - \frac{\langle \underline{z}_1, x_j \rangle}{\langle \underline{z}_1, \underline{z}_1 \rangle} \quad j = 1, \dots, p$$

$$\hat{\varphi}_{2j} = \frac{\langle \underline{x}_j^{(2)}, y \rangle}{\langle \underline{x}_j^{(2)}, \underline{x}_j^{(2)} \rangle}$$

$$\underline{z}_2 = \sum_{j=1}^p \hat{\varphi}_{2j} \underline{x}_j^{(2)}$$

$$\hat{\theta}_2 = \frac{\langle \underline{z}_2, y \rangle}{\langle \underline{z}_2, \underline{z}_2 \rangle}$$

Differences:

PCR: derived input directions are ^{the} principal components of X , constructed by looking at the variability within X

PLS: directions take into consideration both the variability and the correlation with y

Mathematically

PCR $\max_{\alpha} \text{Var}(X\alpha) \quad \text{s.t. } \|\alpha\| = 1$

$$\alpha^T S V e = 0 \quad l = 1, \dots, m$$

sample covariance matrix

PLS $\max_{\alpha} \text{Cor}^2(y, X\alpha) \text{Var}(X\alpha)$

uncorrelated directions

In practice, the s.t. $\|\alpha\| = 1$

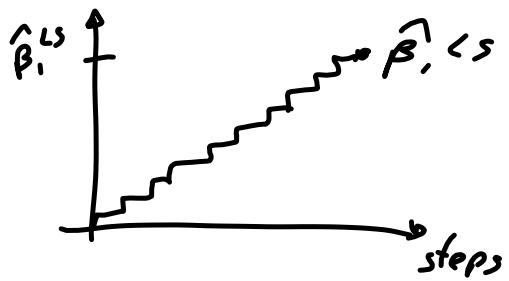
variance component

tends to dominate the

correlation one → similar results!

$$\alpha^T S \hat{\varphi}_l = 0 \quad l = 1, \dots, m-1$$

Incremental forward stagewise regression



similar to LAR, but each update involves only one parameter each time

$$\hat{\beta}_j^{(m)} = \hat{\beta}_j^{(m-1)} + \underbrace{\delta}_{\text{step}}$$

→ we will go back when we will talk about boosting

→ grouped LASSO

different construction of penalty to take into account structures in the data

- dummy variables related to the same categorical variable
- working with genetic data, group genes belonging to the same pathway

$$\hat{\beta}_{gL} = \arg \min_{\beta} \left\{ \sum_{i=1}^n (y_i - \hat{\beta}_0 - \sum_{e=1}^E X_e \beta_e) + \right.$$

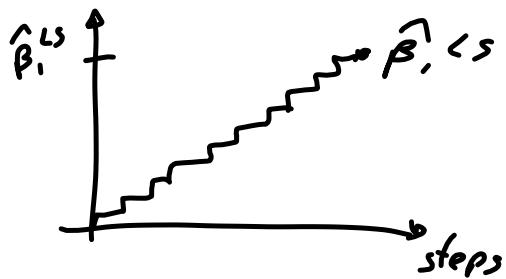
$\| \beta \|_2$ = Euclidean distance
if $\beta = 0 \Leftrightarrow$ all components are 0

- adaptive LASSO
 - relaxed LASSO
- } → 2-step procedure

- SCAD $L(\beta) + J(\beta)$

$$\frac{d J(\beta)}{d \beta} =$$

Incremental forward stagewise regression



similar to LAR, but each update involves only one parameter each time

$$\hat{\beta}_j^{(m)} = \hat{\beta}_j^{(m-1)} + \underbrace{\delta}_{\text{step}}$$

→ we will go back when we will talk about boosting

→ grouped LASSO

different construction of penalty to take into account structures in the data

- dummy variables related to the same categorical variable
- working with genetic data, group genes belonging to the same pathway

$$\hat{\beta}_{gL} = \arg \min_{\beta} \left\{ \sum_{i=1}^n (y_i - \hat{\beta}_0 - \sum_{e=1}^E X_e \beta_e) + \lambda \sum_{e=1}^E \sqrt{p_e} \|\beta_e\|_2 \right\}$$

group size

$\|\cdot\|_2$ = Euclidean distance

if $\|\cdot\|_2 = 0 \Leftrightarrow$ all components are 0

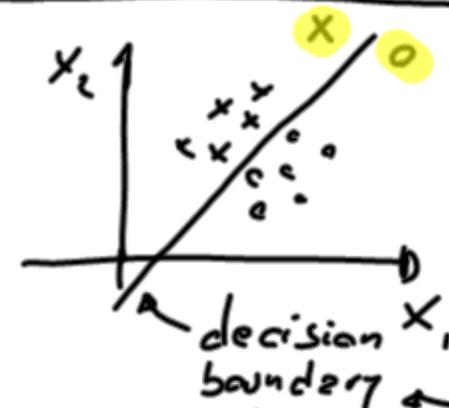
- adaptive LASSO
 - relaxed LASSO
- } → 2-step procedure

- SCAD $L(\beta) + J(\beta)$

$$\frac{d J(\beta)}{d \beta} = \lambda \operatorname{sign}(\beta) \left[\mathbb{1}(|\beta| \leq \lambda) + \frac{(\alpha^2 - |\beta|)}{(\alpha - 1)\lambda} \mathbb{1}(|\beta| > \lambda) \right]$$

Larger coefficients are shrunken less severely!

Linear models for classification



- we saw that we can divide the input space in regions, and assign a label to each of them

when linear

linear methods
for classification

$$\hat{f}_0(x) = \hat{f}_0(x) = \hat{f}_0(x)$$

$$\hat{f}_0(x) > \hat{f}_0(x) \wedge \hat{f}_0(x) > \hat{f}_0(x)$$

until now: classification based on linear regression

$$\forall k \quad \hat{f}_k(x) = \boxed{\text{ }}$$

decision boundary between two classes k and ℓ

$$x: \hat{f}_k(x) = \hat{f}_\ell(x)$$

discriminant functions $\delta_k(x)$

it is
 a member of
 a class of
 methods, namely
 methods based on
 discriminant functions

$$\left. \begin{array}{l} f_A(x) = \Pr[G=A \mid X=x] \\ f_B(x) = \Pr[G=B \mid X=x] \\ f_C(x) = \Pr[G=C \mid X=x] \end{array} \right\} \text{posterior probabilities}$$

If $\delta_k(x)$ or $\Pr[G=k \mid X=x]$ are $\boxed{\text{ }}$ \rightarrow linear decision boundaries

Linear models for classification



- we saw that we can divide the input space in regions, and assign a label to each of them



when linear { linear methods
for classification

$$\hat{f}_o(x) = \hat{f}_q(x) = \hat{f}_o(c)$$

$$\hat{f}_0(x) > f_0(x) \quad \wedge \quad \hat{f}_0(x) > \hat{f}_0(x)$$

until now: classification based on linear regression

$$\forall k \quad \hat{f}_k(x) = \hat{\beta}_{\text{int}} + \hat{\beta}_k^T x$$

decision boundary between two classes \hat{t} and \hat{c}

$$X: \hat{f}_k(x) = \hat{f}_\ell(x)$$

discriminant functions $\delta_k(x)$

\hat{t} and \hat{c} if it is
a member of
a class of
methods, namely,
methods based on
discriminant functions

$$f_A(x) = \Pr[G = A | X = x] \quad \left\{ \begin{array}{l} \text{posterior} \\ \text{probabilities} \end{array} \right.$$

$$f_B(x) = \Pr[G = B \mid x = x]$$

$$f_c(x) = \Pr[G=c \mid X=x]$$

If $\delta_k(x)$ or $\Pr[G=k | X=x]$ are linear in $x \rightarrow$ linear decision boundaries

Actually, we only need monotone transformation of $\delta_k(x)$ or $\Pr[G=k | X=x]$ to be linear

Example

$$(i) \hat{\delta}_k(x) - \hat{f}_k(x) = \hat{\beta}_{k0} + \hat{\beta}_{k1}x_1 + \hat{\beta}_{k2}x_2 + \hat{\beta}_{k3}\underline{x_1} + \hat{\beta}_{k4}\underline{x_2}$$

the relationship is linear in the augmented input space, but the decision boundaries are quadratic in the original space

(ii) when there are two classes

$$\Pr[G=1 | X=x] = \frac{1}{1 + e^{-\hat{Y}}} \quad \left. \begin{array}{l} \text{logit} \\ \text{transformation} \end{array} \right\}$$

$$\Pr[G=2 | X=x] = \frac{e^{-\hat{Y}}}{1 + e^{-\hat{Y}}}$$

$$= \beta_0 + \beta^T x$$

Linear regression of an Indicator Matrix

- codify each of the classes 1, ..., K with an indicator variable

$$\text{ex. } \underset{N \times 3}{X} \Rightarrow \underset{N \times K}{\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ \vdots & \vdots & \vdots \\ 0 & 0 & 1 \end{pmatrix}} = Y \quad \begin{array}{ll} 1^{\text{st}} \text{ obs. is of class 1} \\ 2^{\text{nd}} \text{ and } 3^{\text{rd}} \text{ .. 2} \\ 4^{\text{th}} \text{ .. 5}^{\text{th}} \text{ .. 3} \end{array}$$

$$\rightarrow \text{linear regression: } \hat{Y} = \underset{N \times K}{\underbrace{\begin{pmatrix} Np & p_{11} & Np & p_{12} & Np & \dots & Np & p_{1K} \end{pmatrix}}_{N \times p}}$$

$$\text{new observation } x_{\text{new}} \quad \hat{y} = \hat{f}(x_{\text{new}}) = \begin{pmatrix} \hat{f}_1(x_{\text{new}}) & \hat{f}_2(x_{\text{new}}) & \hat{f}_3(x_{\text{new}}) \end{pmatrix}^T$$

$$\hat{G}(x_{\text{new}}) = \begin{pmatrix} \hat{\delta}_1(x_{\text{new}}) & \hat{\delta}_2(x_{\text{new}}) & \hat{\delta}_3(x_{\text{new}}) \end{pmatrix}^T$$

Actually, we only need monotone transformation of $\delta_k(x)$ or $\Pr[G=k | X=x]$ to be linear

Example

$$(i) \hat{\delta}_k(x) = \hat{\beta}_{k0} + \hat{\beta}_{k1}x_1 + \hat{\beta}_{k2}x_2 + \hat{\beta}_{k3}\underline{x_1} + \hat{\beta}_{k4}\underline{x_2}$$

the relationship is linear in the augmented space, but the decision boundaries are quadratic in the original space

(ii) when there are two classes

$$\Pr[G=1 | X=x] = \frac{\exp(\beta_0 + \beta^T x)}{1 + \exp(\beta_0 + \beta^T x)}$$

logit transformation

$$\Pr[G=2 | X=x] = \frac{1}{1 + \exp(\beta_0 + \beta^T x)}$$

$$\log \frac{P}{1-P}$$

$$\log \frac{\Pr[G=1 | X=x]}{\Pr[G=2 | X=x]} = \beta_0 + \beta^T x$$

Linear regression of an Indicator Matrix

- codify each of the classes 1, ..., K with an indicator variable

$$\text{ex. } x^{(3)} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix} = Y \quad \begin{array}{ll} 1^{\text{st}} \text{ obs. is of class 1} \\ 2^{\text{nd}} \text{ and } 3^{\text{rd}} \text{ .. 2} \\ 4^{\text{th}} \text{ .. 5}^{\text{th}} \text{ .. 3} \end{array}$$

$$\rightarrow \text{linear regression: } \hat{Y} = X (X^T X)^{-1} X^T Y$$

$$\text{new observation } x_{\text{new}} \quad \hat{y} = \hat{f}(x_{\text{new}}) = \underbrace{x_{\text{new}} \hat{\beta}}_{1 \times p \text{ and } p \times 1} \quad \hat{\beta} \in \mathbb{R}^{K \times 1}$$

$$\left(\hat{f}_1(x_{\text{new}}) \quad \hat{f}_2(x_{\text{new}}) \quad \hat{f}_3(x_{\text{new}}) \right)$$

$$\hat{G}(x_{\text{new}}) = \operatorname{argmax}_k \hat{f}_k(x)$$

Why does it work

$$E[Y_k | X=x] = \Pr[G=k | X=x]$$

Are $\hat{f}_k(x)$ reasonable estimates of $\Pr[G=k | X=x]$?

Yes and no

→ if intercept is included $\sum_{k=1}^K$ [redacted]

→ $\hat{f}_k(x)$ can be <0 or >1 → problems happen when

many times it works ←
despite this issue

A bigger problem is the so called [redacted]

- only if $K \geq 3$

As we saw in Fig 4.3, when $K=3$, it is sufficient to have a [redacted]

More generally, if we have K classes, we need a curve of [redacted]

Why does it work

$$E[Y_k | X=x] = \Pr[G=k | X=x]$$

Are $\hat{f}_k(x)$ reasonable estimates of $\Pr[G=k | X=x]$?

Yes and no

- if intercept is included $\sum_{k=1}^K \hat{f}_k(x) = 1$
- $\hat{f}_k(x)$ can be <0 or >1 → problems happen when the new observation is outside the training hull, due to the rigidity of linear regression
- many times it works despite this issue

A bigger problem is the so called masking effect

- only if $K \geq 3$

As we see in Fig 4.3, when $K=3$, it is sufficient to have a quadratic rule.

More generally, if we have K classes, we need a curve of degree $K-1$

LINEAR DISCRIMINANT ANALYSIS

- From the decision theory (Section 2.4) we know that for optimal classification we need to know the class posterior $\Pr(G=k | X=x)$

Suppose:

- $f_k(x)$ is the density of x conditional to class $G=k$
 $\Pr[X=x | G=k]$
- $\pi_k(x)$ is the prior probability to be in class k , $\Pr[G=k]$

Then

$$\Pr[G=k | X=x] = \frac{\Pr[X=x | G=k] \Pr[G=k]}{\Pr[X=x]} = \frac{\Pr[X=x | G=k] \Pr[G=k]}{\sum \Pr[X=x | G=k] \Pr[G=k]}$$

We can choose $f_k(x)$ and $\pi_k(x)$ as we prefer...

- when $f_k(x)$ is from a [] distribution, then Linear Discriminant Analysis (LDA)
Quadratic Discriminant Analysis (QDA)

$$f_k(x) =$$

In particular, for LDA we suppose

LINEAR DISCRIMINANT ANALYSIS

- From the decision theory (Section 2.4) we know that for optimal classification we need to know the class posterior $\Pr(G=k | X=x)$

Suppose:

- $f_k(x)$ is the density of x conditional to class $G=k$
 $\Pr[X=x | G=k]$
- $\pi_k(x)$ is the prior probability to be in class k , $\Pr[G=k]$

Then

$$\begin{aligned} \Pr[G=k | X=x] &= \frac{\Pr[X=x | G=k] \Pr[G=k]}{\Pr[X=x]} = \sum \Pr[X=x | G=k] \Pr[G=k] \\ &= \frac{f_k(x) \pi_k(x)}{\sum_{e=1}^E f_e(x) \pi_e(x)} \end{aligned}$$

We can choose $f_k(x)$ and $\pi_k(x)$ as we prefer...

- when $f_k(x)$ is from a multivariate Gaussian distribution, then Linear Discriminant Analysis (LDA)
 Quadratic Discriminant Analysis (QDA)

$$f_k(x) = \frac{1}{(2\pi)^{D/2} |\Sigma_k|^{1/2}} \exp \left\{ -\frac{1}{2} (x - \mu_k)^T \Sigma_k^{-1} (x - \mu_k) \right\}$$

In particular, for LDA we suppose $\Sigma_k = \sum f_k$

We can then compare two classes by the log-ratio

$$\log \frac{\Pr[G=k | X=x]}{\Pr[G=0 | X=x]} =$$

$$= \log \frac{\hat{\pi}_k}{\hat{\pi}_0} - \frac{1}{2} (\underline{x}^T \Sigma)$$

$$= \log \frac{\hat{\pi}_k}{\hat{\pi}_0} - \frac{1}{2} (\mu_k -$$

Note

- the decision boundaries are NOT the perpendicular bisectors of the segments joining the centroids (happens only if $\Sigma = \sigma^2 I$)
- the linear discriminant function $\delta_k(x)$ is

$$\delta_k(x) = \underline{x}^T \Sigma$$

$$\rightarrow \hat{G}(x) =$$

Since we do not know the values of the parameters $\hat{\pi}_k, \mu_k, \Sigma$, we need to estimate them

$$\cdot \hat{\pi}_k = \frac{\text{number of } k \text{ samples}}{\text{total number of samples}}$$

$$\cdot \hat{\mu}_k = \frac{\sum \text{samples of class } k}{\text{number of samples}}$$

$$\cdot \hat{\Sigma} = \frac{\sum (x - \hat{\mu}_k)(x - \hat{\mu}_k)^T}{\text{number of samples}}$$

With two classes, there is a simple correspondence between LDA and

With $I \geq 2$, there are substantial differences

LDA does NOT suffer from the masking effect.

We can then compare two classes by the log-ratio

$$\log \frac{\Pr[G=k | X=x]}{\Pr[G=0 | X=x]} = \log \frac{\frac{1}{(2\pi)^{n/2} |\Sigma|^{1/2}} \exp \left\{ -\frac{1}{2} (x - \mu_k)^\top \Sigma^{-1} (x - \mu_k) \right\} \hat{h}_k / D}{\frac{1}{(2\pi)^{n/2} |\Sigma|^{1/2}} \exp \left\{ -\frac{1}{2} (x - \mu_0)^\top \Sigma^{-1} (x - \mu_0) \right\} \hat{h}_0 / D}$$

$$= \log \frac{\hat{h}_k}{\hat{h}_0} - \frac{1}{2} \left(x^\top \Sigma^{-1} x - 2x^\top \Sigma^{-1} \mu_k + \mu_k^\top \Sigma^{-1} \mu_k - x^\top \Sigma^{-1} x + 2x^\top \Sigma^{-1} \mu_0 - \mu_0^\top \Sigma^{-1} \mu_0 \right)$$

$$= \log \frac{\hat{h}_k}{\hat{h}_0} - \frac{1}{2} (\mu_k + \mu_0)^\top \Sigma^{-1} (\mu_k - \mu_0) + x^\top \Sigma^{-1} (\mu_k - \mu_0)$$

Note

- the decision boundaries are NOT the perpendicular bisectors of the segments joining the centroids (happens only if $\Sigma = \sigma^2 I$)
- the linear discriminant function $\delta_k(x)$ is

$$\delta_k(x) = x^\top \Sigma^{-1} \mu_k - \frac{1}{2} \mu_k^\top \Sigma^{-1} \mu_k + \log \hat{h}_k$$

$$\rightarrow \hat{G}(x) = \arg \max_k \delta_k(x)$$

Since we do not know the values of the parameters \hat{h}_k, μ_k, Σ , we need to estimate them

$$\cdot \hat{h}_k = \frac{N_k}{N} = \frac{\# \text{ observations in class } k}{\text{total } \# \text{ of observations}}$$

$$\cdot \hat{\mu}_k = \sum_{g_i=k} x_i / N_k$$

$$\cdot \hat{\Sigma} = \sum_{k=1}^K \sum_{g_i=k} (x_i - \hat{\mu}_k) (x_i - \hat{\mu}_k)^\top / (N - K)$$

With two classes, there is a simple correspondence between LDA and classification by linear regression (\rightarrow Ex. 4.2)

With $K > 2$, there are substantial differences

LDA does NOT suffer from the masking effect.

Exercise 4.2

$x \in \mathbb{R}^p$ γ has 2 classes

size N_1 size N_2

target code: $\frac{N}{N_1}$ target code: $\frac{N}{N_2}$

(a) LDA classifies to class ℓ if $\delta_\ell(x) > \delta_i(x)$

$$x^\top \underline{\Sigma^{-1} \hat{\mu}_2} - \frac{1}{2} \underline{\hat{\mu}_2^\top \Sigma^{-1} \hat{\mu}_2} + \underline{\log \frac{N_2}{N}} >$$

$$x^\top \underline{\Sigma^{-1} \hat{\mu}_1} - \frac{1}{2} \underline{\hat{\mu}_1^\top \Sigma^{-1} \hat{\mu}_1} + \underline{\log \frac{N_1}{N}}$$

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

$$(b) \min \sum_{i=1}^n (y_i - \beta_0 - x_i^\top \beta)^2$$

γ = target vector, target code $\frac{N}{N_1}, \frac{N}{N_2}$

$$\underline{\gamma = t_1 U_1 + t_2 U_2} \quad t_1 = -\frac{N}{N_1}, t_2 = \frac{N}{N_2}$$

$$\underline{1 = U_1 + U_2}$$

$$\begin{matrix} U_1 & U_2 \\ \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{pmatrix} \end{matrix}$$

Note that $\hat{\mu}_i = \frac{x^\top U_i}{N_i}$ $\Rightarrow \underline{x^\top U_1 = N_1 \hat{\mu}_1}$
 $\underline{x^\top U_2 = N_2 \hat{\mu}_2}$

Therefore $\underline{x^\top \gamma} = x^\top (t_1 U_1 + t_2 U_2) = \underline{t_1 N_1 \hat{\mu}_1 + t_2 N_2 \hat{\mu}_2}$

Linear Discriminant Analysis

$$\delta_K(x) = x^T \Sigma^{-1} \mu_K - \frac{1}{2} \mu_K^T \Sigma^{-1} \mu_K + \log \pi_K$$

$$\Pr[G=K | X=x] = \frac{f_K(x) \pi_K}{\sum_{k=1}^K f_k(x) \pi_k}$$

$f_k(x)$ Gaussian \rightarrow LDA, QDA

$$\Sigma_K = \sum \pi_k \rightarrow \text{LDA}$$

When we do NOT assume $\Sigma_K = \sum \pi_k$, then

$$\delta_K(x) = -\frac{1}{2} \log |\Sigma_K| - \frac{1}{2} (x - \mu_K)^T \Sigma_K^{-1} (x - \mu_K) + \log \pi_K$$

QDA the quadratic term does not simplify

- estimates for QDA are the same as those for LDA, but

$$\hat{\Sigma}_K = \sum \pi_k (x - \mu_k)(x - \mu_k)^T$$

- similar techniques, difference in # of parameters to estimate

- for LDA, $\# \text{params} = K(K-1)/2$ \rightarrow # parameters to estimate all differences to contrast \quad for each difference $S_K - \underline{S_1}$

- for QDA, $\# \text{params} = (K-1) \times \left[\frac{p(p+3)}{2} + 1 \right]$

Note: both methods perform quite well in a large number of situations

- data support only linear (or quadratic) decision boundaries
- Gaussian models are stable

Linear Discriminant Analysis

$$\delta_K(x) = x^T \mu_K - \frac{1}{2} \mu_K^T \Sigma_K^{-1} \mu_K + \log \pi_K$$

$$\Pr[G=K | X=x] = \frac{f_K(x) \pi_K}{\sum_{k=1}^K f_k(x) \pi_k}$$

$f_k(x)$ Gaussian \rightarrow LDA, QDA

$$\Sigma_K = \sum \pi_k \rightarrow \text{LDA}$$

When we do NOT assume $\Sigma_K = \sum \pi_k$, then QDA

$$\delta_K(x) = -\frac{1}{2} \log |\Sigma_K| - \frac{1}{2} (x - \mu_K)^T \Sigma_K^{-1} (x - \mu_K) + \log \pi_K$$

QDA the quadratic term does not simplify

- estimates for QDA are the same as those for LDA, but
 $\hat{\Sigma}_K = \sum_{j=1}^K (x_j - \hat{\mu}_K)(x_j - \hat{\mu}_K)^T / (N_K - 1)$
- similar techniques, difference in # of parameters to estimate
 - for LDA, # pers = $(K-1) \times (P-1)$ \rightarrow # parameters to estimate all differences to contrast for each difference $S_K - \delta_1$
 - for QDA, # pers = $(K-1) \times \left[\frac{P(P+3)}{2} + 1 \right]$

Note: both methods perform quite well in a large number of situations

- data support only linear (or quadratic) decision boundaries
- Gaussian models are stable

Regularized Discriminant Analysis

- idea: create a sort of compromise between LDA and QDA
- we allow differences among the covariance matrices, but we shrink them toward Σ & similar to ridge

$$\hat{\Sigma}_K(\alpha) = \boxed{\text{expression}}$$

where $\alpha \in [0; 1]$ should be chosen (i.e., by cross-validation)
 α controls the amount of shrinkage

$$\alpha = 0 \rightarrow \text{LDA}$$

$$\alpha = 1 \rightarrow \text{QDA}$$

- Further possibility is to shrink Σ toward $\sigma^2 I$

$$\hat{\Sigma} = \gamma \hat{\Sigma} + (1-\gamma) \sigma^2 I$$

where $\gamma \in [0; 1]$ has a similar meaning than α

- Combining

$$\hat{\Sigma}_K(\alpha, \gamma) = \alpha \hat{\Sigma}_K + (1-\alpha) [\gamma \hat{\Sigma} + (1-\gamma) \sigma^2 I]$$

we obtain a general family for the covariance matrix, depends which on α, γ

Regularized Discriminant Analysis

- idea: create a sort of compromise between LDA and QDA
- we allow differences among the covariance matrices, but we shrink them toward $\bar{\Sigma}$ & similar to ridge

$$\hat{\Sigma}_K(\alpha) = \underline{\alpha} \hat{\Sigma}_K + (\underline{1}-\alpha) \hat{\Sigma}$$

where $\alpha \in [0; 1]$ should be chosen (i.e., by cross-validation)
 α controls the amount of shrinkage

$$\alpha = 0 \rightarrow \text{LDA}$$

$$\alpha = 1 \rightarrow \text{QDA}$$

- Further possibility is to shrink $\hat{\Sigma}$ toward $\sigma^2 I$

$$\hat{\Sigma} = \gamma \hat{\Sigma} + (1-\gamma) \sigma^2 I$$

where $\gamma \in [0; 1]$ has a similar meaning than α

- Combining

$$\hat{\Sigma}_K(\alpha, \gamma) = \alpha \hat{\Sigma}_K + (1-\alpha) [\gamma \hat{\Sigma} + (1-\gamma) \sigma^2 I]$$

we obtain a general family for the covariate matrix, depends which on α, γ

Logistic regression

- model the posterior probabilities of the K classes, s.t.

$$\begin{aligned} \bullet & \Pr[G=1 | X=x] + \Pr[G=2 | X=x] + \dots + \Pr[G=K | X=x] = 1 \\ \bullet & \Pr[G=1 | X=x] \geq 0 \\ \bullet & \Pr[G=1 | X=x] \leq \Pr[G=2 | X=x] \leq \dots \leq \Pr[G=K | X=x] \end{aligned}$$

Logistic regression models

$$\log \frac{\Pr[G=1 | X=x]}{\Pr[G=K | X=x]} = \beta_{10} + \beta_1^T x$$

$$\log \frac{\Pr[G=2 | X=x]}{\Pr[G=K | X=x]} = \beta_{20} + \beta_2^T x$$

:

$$\log \frac{\Pr[G=K-1 | X=x]}{\Pr[G=K | X=x]} = \beta_{K-10} + \beta_{K-1}^T x$$

- specifies $K-1$ log-odd
- based on the logit transformation

$$\begin{array}{c} K=2 \\ \geq 2 \end{array} \quad \boxed{\beta} = X\beta \Leftrightarrow P = \boxed{e^{X\beta^T}} / (1 + e^{X\beta^T})$$

$$\Pr[G=1 | X=x] = P = \frac{e^{X\beta^T}}{1 + e^{X\beta^T}}$$

$$\Pr[G=2 | X=x] = 1 - \Pr[G=1 | X=x] = 1 - P = \frac{1}{1 + e^{X\beta^T}}$$

$$\frac{e^{X\beta^T}}{1 + e^{X\beta^T}} + \frac{1}{1 + e^{X\beta^T}} = 1$$

$$\begin{aligned} \beta &= \beta_0 \ \beta \\ x &= (1, x) \end{aligned}$$

$$\hat{\beta} = \boxed{\beta}$$

Logistic regression

- model the posterior probabilities of the K classes, s.t.
 - linear functions in x
 - sum of them = 1
 - they $\in [0; 1]$

Logistic regression models

$$\log \frac{\Pr[G=1 | X=x]}{\Pr[G=K | X=x]} = \beta_{10} + \beta_1^T x$$

$$\log \frac{\Pr[G=2 | X=x]}{\Pr[G=K | X=x]} = \beta_{20} + \beta_2^T x$$

:

$$\log \frac{\Pr[G=K-1 | X=x]}{\Pr[G=K | X=x]} = \beta_{K-10} + \beta_{K-1}^T x$$

- specifies $K-1$ log-odd
- based on the logit transformation

$$\begin{array}{l} K=2 \\ \geq 2 \end{array} \quad \log \frac{P}{1-P} = x\beta \Leftrightarrow P = \frac{e^{x\beta}}{1+e^{x\beta}}$$

$$\Pr[G=1 | X=x] = P = \frac{e^{x\beta^T}}{1+e^{x\beta^T}}$$

$$\Pr[G=2 | X=x] = 1 - \Pr[G=1 | X=x] = 1 - P = \frac{1}{1+e^{x\beta^T}}$$

$$\frac{e^{x\beta^T}}{1+e^{x\beta^T}} + \frac{1}{1+e^{x\beta^T}} = 1$$

$$\begin{aligned} \beta &= \beta_0 \ \beta \\ x &= (1, x) \end{aligned}$$

$$\hat{\beta} = \arg \max_{\beta} L(\beta)$$

$$L(\beta) =$$

$$=$$

$$=$$

$$\ell(\beta) =$$

$$=$$

$$=$$

$$=$$

$$=$$

$$=$$

$$\ell_{\beta}(\beta) = \frac{\partial \ell(\beta)}{\partial \beta} = \sum_{i=1}^n \left[y_i x_i - \frac{e^{x_i \beta^T}}{1 + e^{x_i \beta^T}} x_i \right] = 0$$

$$=$$

system of $p+1$
equations not
linear in β

→ to find the solution, we can use the

$$x_{n+1} =$$

$$\ell_{\beta\beta}(\beta) = - \sum_{i=1}^n \left(x_i \frac{e^{x_i \beta^T} (1 + e^{x_i \beta^T}) - x_i e^{x_i \beta^T} e^{x_i \beta^T}}{(1 + e^{x_i \beta^T})^2} x_i^T \right) / (1 + e^{x_i \beta^T})^2$$

$$= - \sum_{i=1}^n x_i \frac{e^{x_i \beta^T} (1 + e^{x_i \beta^T} - e^{x_i \beta^T})}{(1 + e^{x_i \beta^T})^2} x_i^T$$

$$= - \sum_{i=1}^n x_i \frac{e^{x_i \beta^T}}{(1 + e^{x_i \beta^T})^2} x_i^T$$

$$\begin{aligned} L(\beta) &= \prod_{i=1}^n f(y_i; P(x_i; \beta)) \\ &= \prod_{i=1}^n \cancel{\left(\frac{1}{\rho(x_i; \beta)} \right)^{y_i} \left(1 - \rho(x_i; \beta) \right)^{1-y_i}} \end{aligned}$$

$$\begin{aligned} l(\beta) &= \sum_{i=1}^n y_i \log \rho(x_i; \beta) + (1-y_i) \log (1-\rho(x_i; \beta)) \\ &= \sum_{i=1}^n \left[y_i \log \frac{e^{x_i \beta^\top}}{1+e^{x_i \beta^\top}} + \log \left(1 - \frac{e^{x_i \beta^\top}}{1+e^{x_i \beta^\top}} \right) - y_i \log \left(1 - \frac{e^{x_i \beta^\top}}{1+e^{x_i \beta^\top}} \right) \right] \\ &= \sum_{i=1}^n \left[y_i x_i \beta^\top - y_i \log (1+e^{x_i \beta^\top}) + \log \left(\frac{1+e^{x_i \beta^\top} - e^{x_i \beta^\top}}{1+e^{x_i \beta^\top}} \right) + y_i \log \left(\frac{1}{1+e^{x_i \beta^\top}} \right) \right] \\ &= \sum_{i=1}^n y_i x_i \beta^\top - \log (1+e^{x_i \beta^\top}) \end{aligned}$$

$$\begin{aligned} \ell_\beta(\beta) &= \frac{\partial \ell(\beta)}{\partial \beta} = \sum_{i=1}^n \left[y_i x_i - \frac{e^{x_i \beta^\top}}{1+e^{x_i \beta^\top}} x_i \right] = 0 \\ &= \sum_{i=1}^n x_i (y_i - \rho(x_i; \beta)) = 0 \quad \text{system of } p+1 \\ &\quad \text{equations not linear in } \beta \end{aligned}$$

→ to find the solution, we can use the Newton-Raphson algorithm

$$x_{n+1} = x_n - \frac{\ell(x_n)}{\ell'(x_n)} \leftarrow \ell_\beta(\beta)$$

$$\ell_{\beta\beta}(\beta) = \frac{\partial^2 \ell(\beta)}{\partial \beta \partial \beta^\top}$$

$$\begin{aligned} \ell_{\beta\beta}(\beta) &= - \sum_{i=1}^n \left(x_i \frac{e^{x_i \beta^\top} (1+e^{x_i \beta^\top}) - x_i e^{x_i \beta^\top} e^{x_i \beta^\top}}{(1+e^{x_i \beta^\top})^2} \right) x_i^\top / (1+e^{x_i \beta^\top})^2 \\ &= - \sum_{i=1}^n x_i \frac{e^{x_i \beta^\top} (1+e^{x_i \beta^\top} - e^{x_i \beta^\top})}{(1+e^{x_i \beta^\top})^2} x_i^\top \\ &= - \sum_{i=1}^n x_i \frac{e^{x_i \beta^\top}}{(1+e^{x_i \beta^\top})^2} x_i^\top \end{aligned}$$

Newton-Raphson

$$\beta_{\text{new}} = \beta_{\text{old}} - \ell_{\beta\beta}(\beta)^{-1} \ell_\beta(\beta)$$

$$\ell_\beta(\beta) = x^\top (y - p)$$

$$\ell_{\beta\beta}(\beta) = -x^\top W x$$

$$\frac{e^{x^\top \beta^*}}{1+e^{x^\top \beta^*}} \cdot \frac{1}{1+e^{x^\top \beta^*}} = \frac{e^{x^\top \beta^*}}{(1+e^{x^\top \beta^*})^2}$$

$$p = \frac{e^{x^\top \beta^*}}{1+e^{x^\top \beta^*}}$$

where $W = \begin{bmatrix} & \\ & \end{bmatrix}$

$$p(1-p)$$

$$\beta_{\text{new}} = \beta_{\text{old}} + \frac{\ell_{\beta\beta}(\beta)}{\ell_\beta(\beta)}$$

β in W and p are β_{old}

$$= (x^\top W x)^{-1} x^\top W (x \beta_{\text{old}} + W^{-1}(y - p))$$

$$\approx (x^\top W x)^{-1} x^\top W \rightarrow \text{weighted least square}$$

- repeat the steps of the Newton-Raphson algorithm until it converges to $\hat{\beta}$

L_1 regularized logistic regression

The L_1 penalty (LASSO) can be applied to the logistic regression as well

$$\hat{\beta} = \arg \min_{\beta}$$

$$= \arg \max_{\beta}$$

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

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

Exercise:

- try to reproduce Table 3.3 with the data from the South Africa heart disease example

| $\hat{\beta}$ | LS | LOGIT($\alpha=0.05$) | LASSO | RIDGE | LDA | QDA |
|---------------|----|------------------------|-------|-------|-----|-----|
| | | | | | | |

- read ch 4.4.5

Newton-Raphson

$$\beta_{\text{new}} = \beta_{\text{old}} - \ell_{\beta\beta}(\beta)^{-1} \ell_\beta(\beta)$$

$$\ell_\beta(\beta) = x^\top(y - p)$$

$$\ell_{\beta\beta}(\beta) = -x^\top W x$$

$$\frac{e^{x^\top \beta^*}}{1+e^{x^\top \beta^*}} \cdot \frac{1}{1+e^{x^\top \beta^*}} = \frac{e^{x^\top \beta^*}}{(1+e^{x^\top \beta^*})^2}$$

$$p = \frac{e^{x^\top \beta^*}}{1+e^{x^\top \beta^*}}$$

where $W = \begin{bmatrix} & \\ & \end{bmatrix}$

$$p(1-p)$$

$$\beta_{\text{new}} = \beta_{\text{old}} + \frac{\ell_{\beta\beta}(\beta)}{\ell_\beta(\beta)}$$

β in W and p are β_{old}

$$= (x^\top W x)^{-1} x^\top W (x \beta_{\text{old}} + W^{-1}(y - p))$$

$$\approx (x^\top W x)^{-1} x^\top W \rightarrow \text{weighted least square}$$

- repeat the steps of the Newton-Raphson algorithm until it converges to $\hat{\beta}$

L_1 regularized logistic regression

The L_1 penalty (LASSO) can be applied to the logistic regression as well

$$\hat{\beta} = \arg_{\beta} \min \left\{ -\ell(\beta) + \lambda \sum_{j=1}^p |\beta_j| \right\}$$

$$= \arg_{\beta} \max \left\{ \ell(\beta) - \lambda \sum_{j=1}^p |\beta_j| \right\}$$

$$= \arg_{\beta} \max \left\{ \sum_{i=1}^n \left[y_i (\beta_0 + \beta^\top x) - \log(1 + e^{\beta_0 + \beta^\top x}) - \lambda \sum_{j=1}^p |\beta_j| \right] \right\}$$

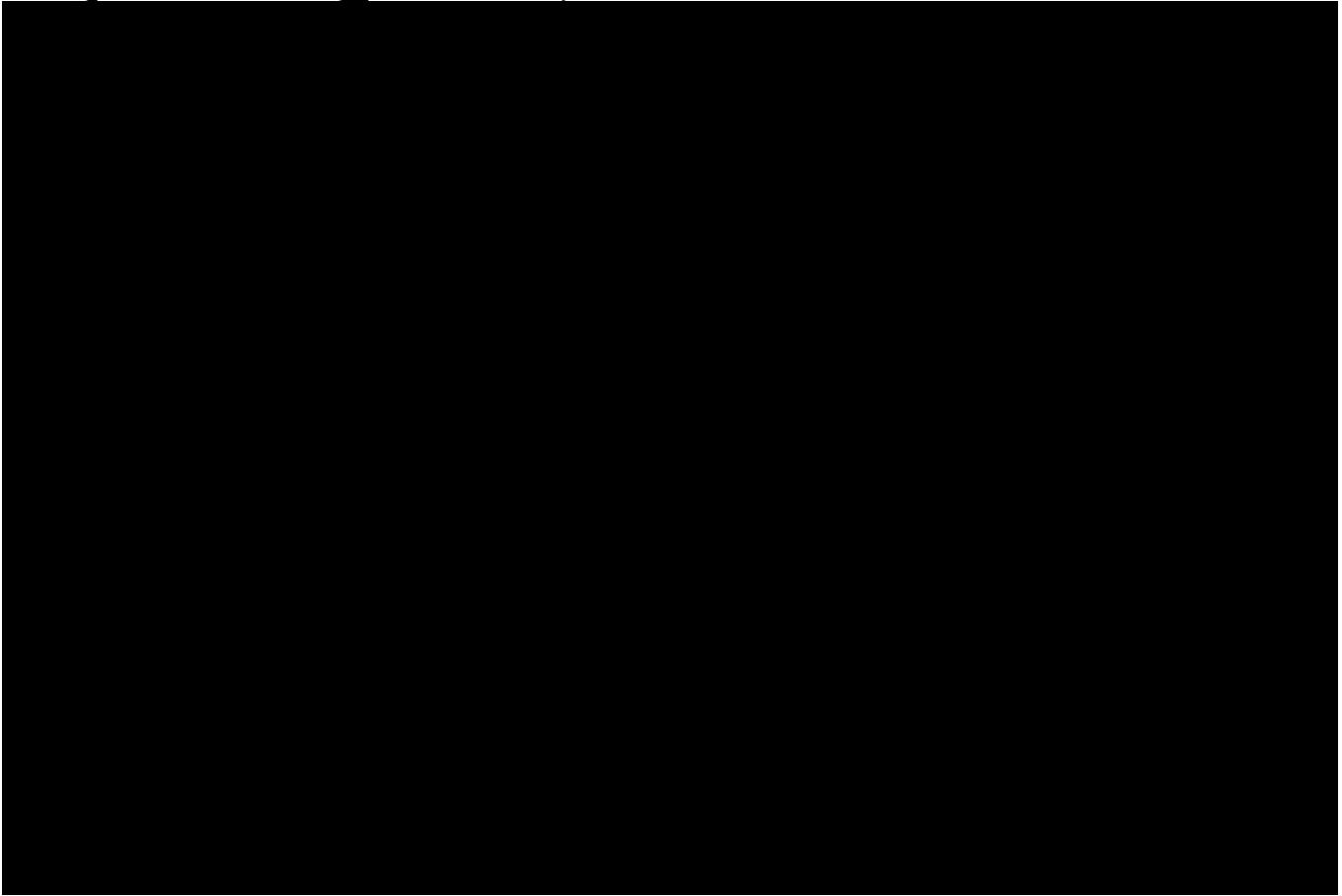
Exercise:

- try to reproduce Table 3.3 with the data from the South Africa heart disease example

| $\hat{\beta}$ | LS | BEA($\alpha=0.05$) | LASSO | RIDGE | LDA | QDA |
|---------------|----|----------------------|-------|-------|-----|-----|
| | | | | | | |

- read ch 4.4.5

Logistic regression vs LDA



Logistic regression vs LDA

$$\text{LDA} \quad \log \frac{\Pr[G=0 | X=x]}{\Pr[G=1 | X=x]} = \underbrace{\log \frac{\hat{\pi}_0}{\hat{\pi}_1} - \frac{1}{2}(\mu_0 + \mu_1)^T \Sigma^{-1} (\mu_0 - \mu_1)}_{+ x^T \Sigma^{-1} (\mu_0 - \mu_1)} \\ = \underbrace{\alpha_0}_{\alpha_0} + \underbrace{\alpha_1^T x}_{\alpha_1^T x} \\ = \underbrace{\beta_0}_{\beta_0} + \underbrace{\beta_1^T x}_{\beta_1^T x}$$

similar to the logistic regression

$$\Pr[G=k, X=x] = \Pr(x) \Pr[G=k | X=x]$$

both LDA and logistic regress.

$$\Pr(x) = \sum_{k=1}^K \hat{\pi}_k \phi(x; \mu_k, \Sigma)$$

Gaussian

$$\frac{e^{\beta_0 x + \beta_1^T x}}{1 + \sum_{k=1}^K e^{\beta_0 x + \beta_1^T x}}$$

Model assessment and model selection

- ① evaluate the performance (in term of prediction) of a selected model
- ② select the best model's (for prediction)

GOAL of \Rightarrow prediction model

generalization: a prediction model must be valid in broad generality, and not valid for a specific dataset

Bias, variance, model complexity

Define:

Y = target variable

X = input matrix

$\hat{f}(x)$ prediction rule, that is trained on a training set \mathcal{T}

Error is measured through a loss function

$$L(Y, \hat{f}(x))$$

that should penalize differences between Y and $\hat{f}(x)$

typical choice
for continuous
outcomes

$$L(Y, \hat{f}(x)) = \begin{cases} (Y - \hat{f}(x))^2 & \text{quadratic loss} \\ |Y - \hat{f}(x)| & \text{absolute loss} \end{cases}$$

Test error (generalization error) is a prediction error computed on an independent sample

$$\text{Err}_{\mathcal{T}} = \boxed{\quad}$$

relabel, X, Y

\mathcal{T} is fixed, is the specific training sets on which we derive our prediction rule

In general, we would like to minimize the expected prediction error

$$\text{Err} = \boxed{\quad}$$

We do not want a model with the smallest prediction error for the specific training set, but for the general case

Model assessment and model selection

- ① evaluate the performance (in term of prediction) of a selected model
- ② select the best model's (for prediction)

GOAL of \Rightarrow prediction model

generalization: a prediction model must be valid in broad generality, and not valid for a specific dataset

Bias, variance, model complexity

Define:

Y = target variable

X = input matrix

$\hat{f}(x)$ prediction rule, that is trained on a training set \mathcal{T}

Error is measured through a loss function

$$L(Y, \hat{f}(x))$$

that should penalize differences between Y and $\hat{f}(x)$

typical choice
for continuous
outcomes

$$L(Y, \hat{f}(x)) = \begin{cases} (Y - \hat{f}(x))^2 & \text{quadratic loss} \\ |Y - \hat{f}(x)| & \text{absolute loss} \end{cases}$$

Test error (generalization error) is a prediction error computed on an independent sample

$$\text{Err}_{\mathcal{T}} = E[L(Y, \hat{f}(x)) \mid \mathcal{T}]$$

relabel, X, Y

\mathcal{T} is fixed, is the specific training sets on which we derive our prediction rule

In general, we would like to minimize the expected prediction error

$$\text{Err} = E[L(Y, \hat{f}(x))] = E[\text{Err}_{\mathcal{T}}]$$

We do not want a model with the smallest prediction error for the specific training set, but for the general case

Since we have our training set, we are going to estimate Err_τ

↳ the training error is NOT a good estimate of Err_τ

$$\text{↳ } \bar{\text{err}} = \frac{1}{N} \sum_{i=1}^N L(y_i; f(x_i))$$

We do NOT want to minimize the training error.

We saw that by increasing the model complexity, we can always decrease the training error

OVERFITTING ↳ our model is specific for the training set
 ↳ generalizes poorly

Similar story for categorical outcome

G : target variable → takes K values in G

typical loss functions
in this case

$$L(G, \hat{G}(x)) =$$

indicator function
 $I(g \neq x_0) \begin{cases} 1 & g \neq x_0 \\ 0 & g = x_0 \end{cases}$

0-1 loss
deviance

- $-2 \ell(\beta)$ is a general loss function, it can be used in cases (Binomial, Gamma, Poisson, log-normal, ...)

- the factor -2 is added to make the loss function be equal to the squared loss in the Gaussian case

$$L(\beta) =$$

$$\ell(\beta) =$$

Since we have our training set, we are going to estimate Err_τ

↳ the training error is NOT a good estimate of Err_τ

$$\text{↳ } \bar{\text{err}} = \frac{1}{N} \sum_{i=1}^N L(y_i; f(x_i))$$

We do NOT want to minimize the training error.

We saw that by increasing the model complexity, we can always decrease the training error

OVERFITTING ↳ our model is specific for the training set
 ↳ generalizes poorly

Similar story for categorical outcome

G : target variable → takes K values in \mathcal{G}

typical loss functions
in this case

$$L(G, \hat{G}(x)) = \begin{cases} 1 & (G \neq \hat{G}(x)) \\ -2 \log L(\beta) & \end{cases}$$

indicator function
 $1_{\{G \neq \hat{G}(x)\}}$ { 1 if $G \neq \hat{G}(x)$
0 if $G = \hat{G}(x)$

0-1 loss

deviance

- 2 $L(\beta)$ is a general loss function, it can be used in cases (Binomial, Gamma, Poisson, log-normal, ...)

- the factor -2 is added to make the loss function be equal to the squared loss in the Gaussian case

$$L(\beta) = \frac{1}{2n} \exp \left\{ -\frac{1}{2} \sum_{i=1}^n (y_i - x_i^\top \beta)^2 \right\}$$

$$L(\beta) = -\frac{1}{2} \sum_{i=1}^n (y_i - x_i^\top \beta)^2$$

Sum of squares

-2 $L(\beta)$ = Squared loss

In an ideal situation
 ↳ a lot of data

we can randomly split the observations in three
independent sets



- training set :

- validation set :

- test set : data

↳ this set must be considered only at the end of the analysis,
 i.e. only when we have already chosen the best model
 Must be ignored for model selection
 - avoid

How to split the data in the three sets :

- no general rule

- back suggestion is : 50% training set
 25% validation set
 25% test set

- it depends:

- + sample size
- + on the signal-to-noise ratio
- + complexity of the model we are considering

In an ideal situation
 ↳ a lot of data

we can randomly split the observations in three independent sets

| training | validation | test |
|-----------------|------------|------------------|
| model selection | | model assessment |

- training set : contains the data on which we fit our models
- validation set : data we use to identify the best model
- test set : data to assess the performance of the selected model

↳ this set must be considered only at the end of the analysis,
 i.e. only when we have already chosen the best model
 Must be ignored for model selection
 - avoid overoptimism

How to split the data in the three sets :

- no general rule
- back suggestion is : 50% training set
 25% validation set
 25% test set

- it depends:
 - + sample size
 - + on the signal-to-noise ratio
 - + complexity of the model we are considering

The Bias-Variance Decomposition

$$Y = f(x) + \varepsilon \quad , \quad E[\varepsilon] = 0$$

$$Var[\varepsilon] = \sigma_\varepsilon^2$$

$$Err(x_0) = \sigma_\varepsilon^2 + \text{irreducible error}$$

F 1 2 1 1 2

where

- σ_ε^2 , the variance of the target around the true mean, so we cannot do anything about that
- bias², the squared difference between the average of our estimates and the true mean
- variance, the expected squared difference between $\hat{f}(x)$ and its mean

kNN

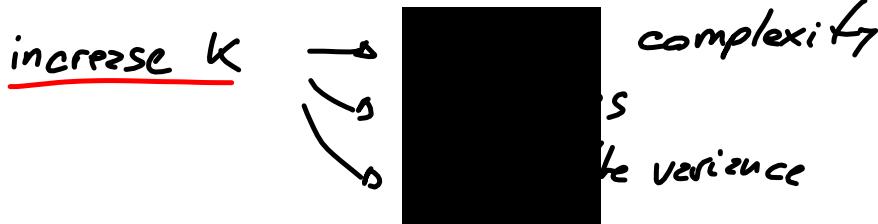
$$\hat{f}(x_0) = \frac{1}{K} \sum_{x_i \in N_K(x_0)} y_i$$

$$Err(x_0) = \sigma_\varepsilon^2 + \left[f(x_0) - \frac{1}{K} \sum_{k=1}^K f(x_k) \right]^2 + \frac{\sigma_\varepsilon^2}{K}$$

$$E[\hat{f}(x)] = \frac{1}{K} \sum_{k=1}^K f(x_k)$$

$$Var[\hat{f}(x)] = \frac{\sigma_\varepsilon^2}{K}$$

$$\#K \propto \frac{1}{\text{complexity}}$$



The Bias-Variance Decomposition

$$Y = f(x) + \varepsilon \quad , \quad E[\varepsilon] = 0$$

$\text{Var}[\varepsilon] = \sigma_\varepsilon^2$

$$\begin{aligned} \text{Err}(x_o) &= E[(Y - \hat{f}(x_o))^2 | X = x_o] \\ &= \sigma_\varepsilon^2 + (E[\hat{f}(x)] - f(x_o))^2 + E[(\hat{f}(x_o) - E[\hat{f}(x_o)])^2] \\ &= \underset{\substack{\text{irreducible} \\ \text{error}}}{\sigma_\varepsilon^2} + \text{bias}^2 + \text{variance} \end{aligned}$$

where

- σ_ε^2 , the variance of the target around the true mean, so we cannot do anything about that
- bias², the squared difference between the average of our estimates and the true mean
- variance, the expected squared difference between $\hat{f}(x)$ and its mean

kNN

$$\hat{f}(x_o) = \frac{1}{K} \sum_{x_i \in N_K(x_o)} y_i$$

$$\text{Err}(x_o) = \sigma_\varepsilon^2 + \left[f(x_o) - \frac{1}{K} \sum_{k=1}^K f(x_{e_k}) \right]^2 + \frac{\sigma_\varepsilon^2}{K}$$

$$E[\hat{f}(x)] = E\left[\frac{1}{K} \sum_{k=1}^K Y_{e_k}\right] = \frac{1}{K} \sum_{k=1}^K E[Y_{e_k}] = \frac{1}{K} \sum_{k=1}^K f(x_{e_k})$$

$$\text{Var}[\hat{f}(x)] = \text{Var}\left[\frac{1}{K} \sum_{k=1}^K Y_{e_k}\right] = \frac{1}{K^2} \sum_{k=1}^K \text{Var}[Y_{e_k}] = \frac{1}{K} \sum_{k=1}^K \sigma_\varepsilon^2 = \frac{K}{K} \sigma_\varepsilon^2 = \frac{\sigma_\varepsilon^2}{K}$$

$$\#K \propto \frac{1}{\text{complexity}}$$

increase K → decrease complexity
 ↓ more bias
 ↓ reduce the variance

Similar for linear model $\hat{f}(x; \beta) = x^T \beta$

- slightly more difficult to derive, it turns out to be

$$\frac{1}{N} \sum_{i=1}^n \text{Err}(x_i) = \sigma_\epsilon^2 + \frac{1}{N} \sum_{i=1}^n [\hat{f}(x_i) - E[\hat{f}(x_i)]]^2 + \frac{P}{N} \sigma_\epsilon^2$$

called the in-sample error

$P = \# \text{ of variables}$

increasing the model complexity,
we increase the variance
component of the error

For regularized regression (e.g., lasso, ridge, ...) the form is the same,
but there is an additional dependence on the tuning (complexity) parameter α

- We can go more into details on the bias component

$$E_{x_*} [\hat{f}(x_*) - E[\hat{f}(x_*)]]^2 = E_{x_*} [\hat{f}(x_*) - x_*^T \hat{\beta}_*]^2 + E_{x_*} [x_*^T \beta - E[x_*^T \beta]]^2$$

$$\hat{\beta}_* = \arg \min_{\beta} E[\hat{f}(x) - x^T \beta]^2$$

$$\text{Average [model bias]}^2 + \text{Avg [Estimation bias]}^2$$

difference between the
true function and the best fitting
linear approximation

error between the average
estimate and the best model

difference between truth and best model

additional bias that we add
when, e.g., add shrinkage

- we can reduce it only increasing the model
space (more variables, more complex relationship - interaction -,...)

OLS = 0

Optimism of the Training Error

Let us start from the definitions of test time:

$$\bar{E}_{\text{err}} = \boxed{\dots}$$

Test error, where

- X_o, Y_o are new (test) point \rightarrow random

- T is fixed $T = \{(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)\}$

Averaging over all training sets

$$\bar{E}_{\text{rr}} = \boxed{\dots}$$

gives the expected error

We also saw the training error

$$\bar{e}_{\text{rr}} = \boxed{\dots}$$

is NOT a good estimate of the test error, because it underestimates it.

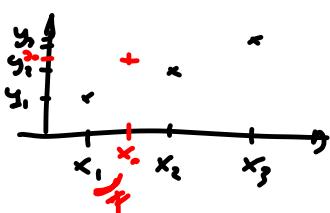
Same data used for both training $\hat{f}(x)$ and to test its performance \rightarrow optimistic estimate

The test error can be thought as extra-sample error (the estimation of the error is computed on new points $x_o \neq x_i$)

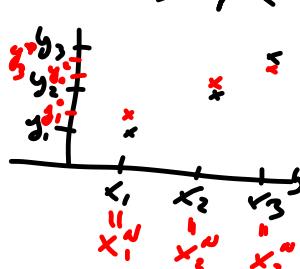
We are going to evaluate the optimism of \bar{e}_{rr} in the in-sample case, i.e. we have new observations in the same points of training set

$$\bar{E}_{\text{rr,in}} = \boxed{\dots}$$

extra-sample



in sample



-only Y_o is random, new y_i for x_1, \dots, x_N

Optimism of the Training Error

Let us start from the definitions of test time:

$$\text{Err}_{\mathcal{T}} = E_{(x_0, y_0)} [L(Y_0, \hat{f}(x_0)) | \mathcal{T}]$$

Test error, where

- x_0, Y_0 are new (test) point \rightarrow random

- \mathcal{T} is fixed $\mathcal{T} = \{(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)\}$

Averaging over all training sets

$$\text{Err} = E_{\mathcal{T}} [E_{(x_0, y_0)} [L(Y_0, \hat{f}(x_0)) | \mathcal{T}]]$$

gives the expected error

We also saw the training error

$$\bar{\text{err}} = \frac{1}{N} \sum_{i=1}^N L(y_i, \hat{f}(x_i))$$

is NOT a good estimate of the test error, because it underestimates it.

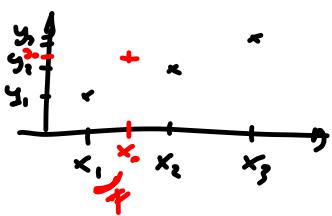
Same data used for both training $\hat{f}(x)$ and to test its performance \rightarrow optimistic estimate

The test error can be thought as extra-sample error (the estimation of the error is computed on new points $x_0 \neq x_i$)

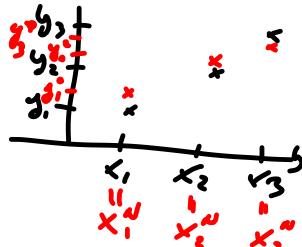
We are going to evaluate the optimism of $\bar{\text{err}}$ in the in-sample case, i.e. we have new observations in the same points of training set

$$\text{Err}_{in} = \frac{1}{N} \sum_{i=1}^N E_{Y_0} [L(Y_i, \hat{f}(x_i)) | \mathcal{T}]$$

extra-sample



in sample



-only Y_0 is random, new y_i for x_1, \dots, x_N

Definition: we define optimism the difference between Err_{in} and $\overline{\text{err}}$

$$\text{op} := \boxed{\dots}$$

op is generally positive, as $\overline{\text{err}}$ is computed on training data

Definition: average optimism is

$$w := E_y[\text{op}]$$

we are computing the expected value over the training set outcome

For a reasonable number of loss functions, including 0/1 loss and squared error, it can be shown that

$$w = \boxed{\dots}$$

where Cov denotes the covariance $\rightarrow \text{Ex 7.4}$

Note:

- optimism depends on how much
- the harder we fit the data, the
 \rightarrow the higher the optimism

$$\boxed{\dots}$$

As a consequence:

$$E_y[\underline{\text{Err}_{\text{in}}}] = E_y[\overline{\text{err}}] + \frac{2}{N} \sum_{i=1}^N \text{Cov}(\hat{y}_i, y_i)$$

When \hat{y}_i is obtained by a linear fit in the inputs, $Y = f(x) \cdot \varepsilon$

$$\sum_{i=1}^N \text{Cov}(\hat{y}_i, y_i) = \boxed{\dots} \quad \leftarrow \text{effective number of parameters}$$

therefore

$$\rightarrow E_y[\underline{\text{Err}_{\text{in}}}] = \boxed{\dots} \quad (\star)$$

- optimism increases with the
- " decreases "

e.g. in linear regression, the number of covariates

Methods we will see:

- Cp, AIC and BIC estimate the prediction error by estimating

$$\boxed{\dots}$$

- cross-validation and bootstrap-based procedures try to estimate $\boxed{\dots}$

Note:

- in-sample error is generally NOT of interest (we are mainly interested in new data, including new points in X)
- to select the best model / tuning the complexity parameter, we are more interested in the relative difference in errors rather than the absolute one.

Definition: we define optimism the difference between Err_{in} and $\overline{\text{err}}$

$$\text{op} := \text{Err}_{\text{in}} - \overline{\text{err}}$$

op is generally positive, as $\overline{\text{err}}$ is computed on training data

Definition: average optimism is

$$\omega := E_y[\text{op}]$$

we are computing the expected value over the training set outcome

For a reasonable number of loss functions, including 0-1 loss and squared error, it can be shown that

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

where Cov denotes the covariance $\rightarrow \text{Ex 7.4}$

Note:

- optimism depends on how much y_i affects its own prediction
- the harder we fit the data, the larger the value of $\text{Cov}(\hat{y}_i, y_i)$
 \rightarrow the higher the optimism

As a consequence:

$$E_y[\underline{\text{Err}_{\text{in}}}] = E_y[\overline{\text{err}}] + \frac{2}{N} \sum_{i=1}^N \text{Cov}(\hat{y}_i, y_i)$$

When \hat{y}_i is obtained by a linear fit in the inputs, $Y = f(x) \cdot \varepsilon$

$$\sum_{i=1}^N \text{Cov}(\hat{y}_i, y_i) = d \sigma_\varepsilon^2 \quad \leftarrow \begin{matrix} \text{effective number} \\ \text{of parameters} \end{matrix}$$

therefore

$$\rightarrow E_y[\underline{\text{Err}_{\text{in}}}] = E_y[\overline{\text{err}}] + 2 \frac{d}{N} \sigma_\varepsilon^2 \quad (*)$$

- optimism increases with the number of inputs;
- decreases " " sample size.

e.g. in linear regression, the number of covariates

Methods we will see:

- Cp, AIC and BIC estimate the prediction error by estimating the training error and the optimism (work when estimates are linear in their parameters)
- cross-validation and bootstrap-based procedure try to estimate directly the expected error

Note:

- in-sample error is generally NOT of interest (we are mainly interested in new data, including new points in X)
- to select the best model / tuning the complexity parameter, we are more interested in the relative difference in errors rather than the absolute one.

Estimates of Err_{in}

Start from (*)

$$E_g[\bar{\text{err}}_{in}] = E_Y[\bar{\text{err}}] + 2 \frac{d}{N} \hat{\sigma}_\epsilon^2 \quad (*)$$

Write the general form of the in-sample estimate

$$\hat{\bar{\text{err}}}_{in} = \bar{\text{err}} + \hat{w}$$

- when we have linearity and squared errors, from (*)

$$C_P = \bar{\text{err}} + 2 \frac{d}{N} \hat{\sigma}_\epsilon^2$$

where:

- $\bar{\text{err}}$ is computed through the squared loss;

- d is # of parameters

- $\hat{\sigma}_\epsilon^2$ is an estimate of the noise variance

it is computed using the full model, because it has the smallest bias

NB: There are other versions of C_P , different versions may give different numbers, but they all lead to the same model

Similar idea for AIC (Akaike Information Criterion)

- we start again from (*), but we want to be more general
(the error is computed through a likelihood approach)

We are using the asymptotic result ($N \rightarrow \infty$)

$$-2E[\log P_\theta(Y)] \approx -\frac{2}{N} E\left[\sum_{i=1}^N \log P_\theta(y_i)\right] + 2 \frac{d}{N}$$

$P_\theta(Y)$ is the family of densities for Y (containing the "true" density)

loglik ↴ the maximized log-likelihood
 $\ell(\hat{\theta})$
is the mle maximum likelihood estimate

E.g., in the logistic regression: $AIC = -\frac{2}{N} \log \text{lik} + 2 \frac{d}{N}$

Gaussian regression: $AIC \propto \text{G}$

AIC ↴ C_P is a special case of AIC

To find the best model, we choose that with the smallest AIC

- straightforward in the simplest cases, we need more attention in more complex situations. In particular, we need to find a reasonable measure for the model complexity

Note: for regularized/penalized regression

$$AIC(\alpha) = \bar{\text{err}}(\alpha) + 2 \times \frac{d(\alpha)}{N} \hat{\sigma}_\epsilon^2$$

• usually minimizing AIC is not the best solution to find the value of the tuning parameter α → cross-validation is better approach

The effective number of parameters

- generalized the concept of number of parameters, in order to extend the previous approaches to more complex situations

Suppose

$$\mathbf{y} = (y_1, \dots, y_N) \text{ outcome}$$

$$\hat{\mathbf{y}} = (\hat{y}_1, \dots, \hat{y}_N) \text{ predictions}$$

Linear regression

$$\hat{\mathbf{y}} = \underbrace{\mathbf{x}(\mathbf{x}^T \mathbf{x})^{-1} \mathbf{x}^T}_{S} \mathbf{y}$$

Linear methods $\hat{\mathbf{y}} = S\mathbf{y}$ Ridge regression

where S is a $N \times N$ matrix which:

- depends on \mathbf{x}
- independent on \mathbf{y}

$$\hat{\mathbf{y}} = \underbrace{\mathbf{x}(\mathbf{x}^T \mathbf{x} + \lambda \mathbf{I}_p)^{-1} \mathbf{x}^T}_{S} \mathbf{y}$$

Effective number of parameters (or effective degrees of freedom)

$$df(S) = \text{trace}(S)$$

value of λ

We should replace $\text{trace}(S)$ to d to obtain the correct criterion

If $\mathbf{y} = f(\mathbf{x}) + \varepsilon$, $\text{Var}(\varepsilon) = \sigma_\varepsilon^2 \rightarrow \sum_{i=1}^n \text{Cov}(\hat{y}_i, y_i) = \text{trace}(S) \sigma_\varepsilon^2$

So $df(\hat{\mathbf{y}}) = \frac{\sum_{i=1}^n \text{Cov}(\hat{y}_i, y_i)}{\sigma_\varepsilon^2} \rightarrow \text{Ex 7.5}$

The Bayesian approach and BIC

The BIC is an alternative criterion to AIC,

↳ Bayesian Information Criterion

$$\frac{1}{N} \text{BIC} = -\frac{2}{N} \log \text{lik} + \log N \cdot \frac{d}{N}$$

$$\frac{\text{AIC}}{\text{BIC}} = e^{\frac{2}{N}}$$

Despite they are quite similar, AIC and BIC come from completely different ideas. BIC comes from the Bayesian approach to model selection

$$\Pr(M_m | z) \propto \Pr(M_m) \Pr(z | M_m)$$

$$\propto \Pr(M_m) \int \Pr(z | M_m, \theta_m) \Pr(\theta_m | M_m) d\theta_m$$

To choose between two models, we compare their posterior probabilities

$$\frac{\Pr(M_0 | z)}{\Pr(M_e | z)} = \frac{\Pr(M_m)}{\Pr(M_e)} \cdot \frac{\Pr(z | M_m)}{\Pr(z | M_e)}$$

↑
in a large number
of case = 1
(give the same prior probability to the two model)

Bayes factor
the choice between the two models is based on the Bayes factor

Approximately

$$\Pr(z | M_m) = \log(z | \hat{\theta}_m, M_m) - \frac{d_m}{2} \log N + O(1)$$

If the loss function is $-2 \log \Pr(z | \hat{\theta}, M_m)$, we obtain BIC

- We ^{may} select the model with smallest BIC

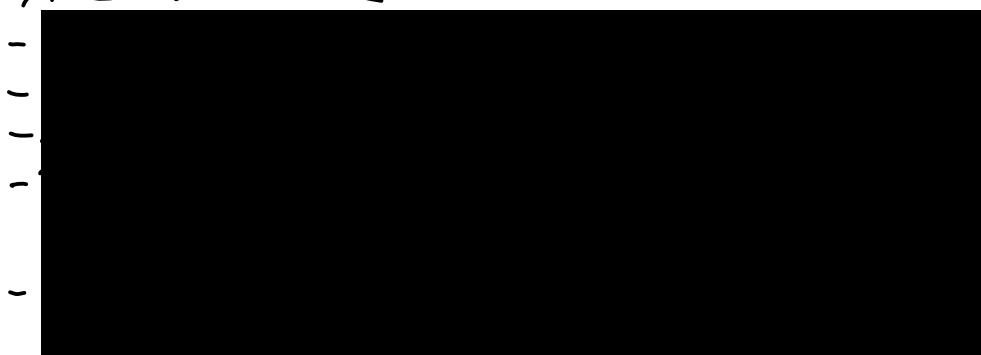
correspond to selecting the model with highest

Note that

$$\frac{e^{-\frac{1}{2} \text{BIC}_m}}{\sum_{e=1}^{m-1} e^{-\frac{1}{2} \text{BIC}_e}}$$

is the posterior probability of selecting the model m

AIC vs BIC



The Bayesian approach and BIC

The BIC is an alternative criterion to AIC,

↳ Bayesian Information Criterion

$$\frac{1}{N} \text{BIC} = -\frac{2}{N} \log \text{lik} + \log N \cdot \frac{c}{N}$$

$$\frac{\text{AIC}}{\text{BIC}} = e^{\frac{2}{N}}$$

Despite they are quite similar, AIC and BIC come from completely different ideas. BIC comes from the Bayesian approach to model selection

$$\Pr(M_m | z) \propto \Pr(M_m) \Pr(z | M_m)$$

$$\propto \Pr(M_m) \int \Pr(z | M_m, \theta_m) \Pr(\theta_m | M_m) d\theta_m$$

To choose between two models, we compare their posterior probabilities

$$\frac{\Pr(M_0 | z)}{\Pr(M_e | z)} = \frac{\Pr(M_m)}{\Pr(M_e)} \cdot \frac{\Pr(z | M_m)}{\Pr(z | M_e)}$$

↑
in a large number
of case = 1
(give the same prior probability to the two model)

Bayes factor
the choice between the two models is based on the Bayes factor

Approximately

$$\Pr(z | M_m) = \log(z | \hat{\theta}_m, M_m) - \frac{c_m}{2} \log N + O(1)$$

If the loss function is $-2 \log \Pr(z | \hat{\theta}, M_m)$, we obtain BIC

- We ^{may} select the model with smallest BIC

correspond to selecting the model with highest posterior probability

Note that

$$\frac{e^{-\frac{1}{2} \text{BIC}_m}}{\sum_{e=1}^E e^{-\frac{1}{2} \text{BIC}_e}}$$

is the posterior probability of selecting the model m

AIC vs BIC

- no clear preference
- BIC leads to a sparser Model
- AIC leads to a model with more predictors
- BIC is consistent ($N \rightarrow \infty$, the probability of selecting the true model goes to 1)
- For finite sample size, BIC tends to select a model which is too sparse

Cross-validation

The cross-validation aims to estimate the ~~extra-sample~~ error

$$Err = E[L(Y, \hat{f}(x))]$$

the average test error when $\hat{f}(x)$ is applied to a new sample

If we had enough data \hookrightarrow training set
to test "

\hookrightarrow in general, we have not, so we mimic this split using the limited amount of data we have



- divide the observations in K folds
- we use, in turn, $K-1$ folds to train the model (derive $\hat{f}^{(k)}$)
- we evaluate the model in the remaining fold

$$CV(\hat{f}) = \frac{1}{K} \sum_{k=1}^K L(Y_k, \hat{f}^{(k)}(X_k))$$

- if $K=2$, two-fold cross-validation
- if $K=N$, leave-one-out cross-validation (LOOCV)
in this case, each observation is a fold

How do we choose K

- bias-variance trade-off
- smaller the K , $\boxed{\text{large bias}}$ variance
- larger the K , $\boxed{\text{small bias}}$ variance

(the extreme case is LOOCV, where we use $N-1$ observations for training the model \rightarrow the training sets are really similar to each other)

- usual choices are $K=5$ or $K=10$

[Fig 7.8]

$\rightarrow 1\text{-}Err \text{ vs } N$

\rightarrow the classifier is OK until $\approx \underline{N=100}$ (then is flat)

• if $N=200$, $K=5 \rightarrow$ training $N_t = 160$ \checkmark $160 > \underline{100}$

• if $N=50$, $K=5 \rightarrow \dots N_t = 40 \times 40 \cancel{X} 100$

Cross-validation

The cross-validation aims to estimate the ~~extra-sample~~ error

$$Err = E[L(Y, \hat{f}(x))]$$

the average test error when $\hat{f}(x)$ is applied to a new sample

If we had enough data \hookrightarrow training set
 \hookrightarrow test "

\hookrightarrow in general, we have not, so we mimic this split using the limited amount of data we have



- divide the observations in K folds
- we use, in turn, $K-1$ folds to train the model (derive $\hat{f}_{-k}(x_k)$)
- we evaluate the model in the remaining fold

$$CV(\hat{f}) = \sum_{k=1}^{K-1} \underbrace{\sum_{i=1}^n L(y_i, \hat{f}_{-k}(x_i))}_{\text{cv}}$$

- if $K=2$, two-fold cross-validation
- if $K=N$, leave-one-out cross-validation (LOOCV)
 in this case, each observation is a fold

How do we choose K

- bias-variance trade-off
- smaller the K , larger bias, smaller variance
- larger the K , smaller bias, larger variance
 (the extreme case is LOOCV, where we use $N-1$ observations for training the model \rightarrow the training sets are really similar to each other)
- usual choices are $K=5$ or $K=10$

[Fig 7.8]

\rightarrow 1-Err vs N

\rightarrow the classifier is OK until $\approx \underline{N=100}$ (then is flat)

• if $N=200$, $K=5 \rightarrow$ training $N_t = 160$ \checkmark $160 > \underline{100}$

• if $N=50$, $K=5 \rightarrow$.. $N_t = 40$ \times $40 \cancel{>} 100$

Notes:

- CV estimates \hat{f} and not the f
- if we want to select a tuning parameter viz CV

$\hat{f}^{(k)}(x, \alpha)$ is the model selected using α and k folds on the observations which do not belong to the k -th fold

$$CV(\hat{f}, \alpha) = \frac{1}{n} \sum_{k=1}^K \sum_{i \in I_k} L(y_i, \hat{f}(x_i, \alpha))$$

$$\hat{\alpha} = \arg \min_{\alpha} CV(\hat{f}, \alpha)$$

Generalized cross-validation

- convenient approximation to the LOOCV, for squared loss function

$$\text{Loocv } \hat{f} = Sy$$

$$\frac{1}{N} \sum_{i=1}^N \left(y_i - \hat{f}^{(i)}(x_i) \right)^2 = \frac{1}{N} \sum_{i=1}^N \left(\frac{y_i - \hat{f}(x_i)}{1 - S_{ii}} \right)^2$$

where S_{ii} is the i th term on the diagonal of S

The generalized cross-validation (GCV)

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

- computational advantages;
- similarities between AIC and GCV → Ex 7.7

Notes:

- CV estimates Err and not the Err_2
- if we want to select a tuning parameter viz CV

$\hat{f}^{-k}(x, \alpha)$ is the model selected using α and k folds on the observations which do not belong to the k -th fold

$$CV(\hat{f}, \alpha) = \frac{1}{n} \sum_{k=1}^{K_n} \sum_{i=1}^{n_k} L(y_i, \hat{f}(x_i, \alpha))$$

$$\hat{\alpha} = \arg \min_{\alpha} CV(\hat{f}, \alpha)$$

Generalized cross-validation

- convenient approximation to the LOOCV, for squared loss function

$$\text{Loocv} \quad \hat{f} = Sy$$

$$N \sum_{k=1}^{n_k} \sum_{i=1}^{n_k} \rightarrow \frac{1}{N} \sum_{i=1}^N (y_i - \hat{f}_{-i}(x_i))^2 = \frac{1}{N} \sum_{i=1}^N \left(\frac{y_i - \hat{f}(x_i)}{1 - S_{ii}} \right)^2$$

where S_{ii} is the i th term on the diagonal of S

The generalized cross-validation (GCV)

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

- computational advantages;
- similarities between AIC and GCV → Ex 7.7

Exercise 7.4

$$E_{\text{err}_{in}} = \frac{1}{N} \sum_{i=1}^N E_{Y_i} \left[(Y_i^o - \hat{f}(x_i))^2 \right]$$

$$\bar{\text{err}} = \frac{1}{N} \sum_{i=1}^N (y_i - \hat{f}(x_i))^2 \quad y = f(x) + \varepsilon.$$

$$E_Y[\sigma_p] = \frac{2}{N} \sum_{i=1}^N \text{Cov}(\hat{g}_i, y)$$

$$\begin{aligned} E_Y[\sigma_p] &= E_Y \left[\frac{1}{N} \sum_{i=1}^N \left\{ E_{Y_i} \left[(Y_i^o - \hat{f}(x_i))^2 \right] - (y_i - \hat{f}(x_i))^2 \right\} \right] \\ &= E_Y \left[\frac{1}{N} \sum_{i=1}^N \left\{ E_{Y_i} [Y_i^{o2}] - 2E_{Y_i} [Y_i^o] \hat{f}(x_i) + \hat{f}(x_i)^2 - y_i^2 + 2E_Y[y_i \hat{f}(x_i)] - \hat{f}(x_i)^2 \right\} \right] \\ &= \frac{1}{N} \sum_{i=1}^N E_Y \left[E_{Y_i} [Y_i^{o2}] \right] - 2E_Y[Y_i^o] E_Y[\hat{f}(x_i)] - E_Y[y_i^2] + 2E_Y[y_i \hat{f}(x_i)] \end{aligned}$$

$$\bullet E_Y \left[E_{Y_i} [Y_i^{o2}] - \hat{f}(x_i)^2 + \hat{f}(x_i)^2 \right] = \sigma_\varepsilon^2 + f(x)^2$$

$$\bullet E_Y \left[y^2 - \hat{f}(x)^2 + \hat{f}(x)^2 \right] = \sigma_\varepsilon^2 + f(x)^2$$

$$\bullet E_{Y_i} [Y_i^o] = f(x) = E_Y[y] \quad \hat{f}(x) = \hat{y}$$

$$\begin{aligned} E_Y[\sigma_p] &= \frac{1}{N} \sum_{i=1}^N \left\{ -2E_Y[y] E_Y[\hat{y}] + 2E_Y[y_i \hat{y}_i] \right\} \\ &= \frac{2}{N} \sum_{i=1}^N \text{Cov}(y_i, \hat{y}_i) \end{aligned}$$

Bootstrap methods

- what is bootstrap
- how to use bootstrap for error estimation

$$\widehat{E[Err_T]}$$

IDEA:



- Suppose $\mathcal{T} = \{(x_1, y_1), \dots, (x_n, y_n)\}$
- by resampling, $\mathcal{T}_1^* = \{(x_1^{*1}, y_1^{*1}), \dots, (x_n^{*1}, y_n^{*1})\}$
- repeat for B large, $\mathcal{T}_1^*, \mathcal{T}_2^*, \dots, \mathcal{T}_B^*$

Based on the generated bootstrap sample (which mimic new experiments) we can estimate any aspect of the distribution of a map

Example

original sample $\mathcal{T} = \{z_1, z_2, z_3, z_4\} = \{1, 3, 4, 6\}$

generate B bootstrap sample
by resampling with replacement
from \mathcal{T}

$$\mathcal{T}_1^* = \{z_1^*, z_2^*, z_3^*, z_4^*\}$$

⋮

$$\mathcal{T}_3^* = \{1, 1, 1, 1\}$$

$$(x, y) \quad \text{Cov}(x, y)$$

$$\begin{aligned} \mathcal{T} & (x_1, y_1), (x_2, y_2), \dots, (x_n, y_n) \\ & (4, 3), (1, 3), \dots, (4, 2) \end{aligned}$$

$$\mathcal{T}_1^* = \{(1, 3), (4, 2), \dots, (4, 2)\}$$

Bootstrap methods

- what is bootstrap
- how to use bootstrap for error estimation $\widehat{E[Err_T]}$

IDEA: generate bootstrap sample from the empirical distribution computed on original sample
 → by resampling with replacement from the original sample

- suppose $\mathcal{T} = \{(x_1, y_1), \dots, (x_n, y_n)\}$
- by resampling, $\mathcal{T}_1^* = \{(x_1^{*1}, y_1^{*1}), \dots, (x_n^{*n}, y_n^{*n})\}$
- repeat for B large, $\mathcal{T}_1^*, \mathcal{T}_2^*, \dots, \mathcal{T}_B^*$

Based on the generated bootstrap sample (which mimic new experiments) we can estimate any aspect of the distribution of a map

Example

original sample $\mathcal{T} = \{z_1, z_2, z_3, z_4\} = \{1, 3, 4, 6\}$

generate B bootstrap sample
 by resampling with replacement from \mathcal{T}

$$\mathcal{T}_1^* = \{z_1^*, z_2^*, z_3^*, z_4^*\}$$

⋮

$$\mathcal{T}_3^* = \{1, 1, 1, 1\}$$

$$(x, y) \quad \text{Cov}(x, y)$$

$$\begin{aligned} \mathcal{T} & (x_1, y_1), (x_2, y_2), \dots, (x_n, y_n) \\ & (4, 3), (1, 3), \dots, (4, 2) \end{aligned}$$

$$\mathcal{T}_1^* = \{(1, 3), (4, 2), \dots, (4, 2)\}$$

Bootstrap approach for prediction error estimation

WRONG APPROACH

- estimate our $\hat{f}(x)$ from each bootstrap sample
- evaluate how well $\hat{f}_s^*(x)$ estimate y

$$\hat{E}_{\text{err}}^{\text{wrong}} = \frac{1}{B} \sum_{s=1}^B \left(\frac{1}{N} \sum_{i=1}^N L(y_i, \hat{f}_s^*(x_i)) \right)$$

⚠: training and test set are not independent

$$\begin{aligned} E[\hat{E}_{\text{err}}^{\text{wrong}}] &= 0.184, \quad 1\text{NN}, \quad Y \perp X \\ &\left(\begin{array}{l} \bullet y_i \in \mathcal{T}_s^* \rightarrow \text{error} = 0 \\ \bullet y_i \notin \mathcal{T}_s^* \rightarrow \text{error} = 0.5 \end{array} \right) \\ &= 0.5 \times \underbrace{\Pr[y_i \notin \mathcal{T}_s^*]}_{0.368} + 0 \times \underbrace{\Pr[y_i \in \mathcal{T}_s^*]}_{0} \end{aligned}$$

$\Pr[\text{observation } i \notin \text{bootstrap sample } s]$

$$\Pr[\mathcal{T}_{s[i]}^* \neq y_i] = \frac{N-1}{N} \Rightarrow \Pr[y_i \notin \mathcal{T}_s^*] = \boxed{!}$$

same for all positions

$$E[\hat{E}_{\text{err}}^{\text{wrong}}] \Big|_{\substack{X \perp Y \\ 1\text{NN}}} \approx \boxed{}$$

An important fact

$$\Pr[\text{observation } i \text{ belongs to a bootstrap sample } s] = \boxed{\frac{1}{B}}$$

approximation
of

Bootstrap approach for prediction error estimation

WRONG APPROACH

- estimate our $\hat{f}(x)$ from each bootstrap sample
- evaluate how well $\hat{f}_s^*(x)$ estimate y

$$\hat{E}_{\text{err}}^{\text{wrong}} = \frac{1}{B} \sum_{s=1}^B \left(\frac{1}{N} \sum_{i=1}^N L(y_i, \hat{f}_s^*(x_i)) \right)$$

⚠: training and test set are not independent

$$\begin{aligned} E[\hat{E}_{\text{err}}^{\text{wrong}}] &= 0.184, \quad 1\text{NN}, \quad Y \perp X \\ &\left(\begin{array}{l} \bullet y_i \in \mathcal{T}_s^* \rightarrow \text{error} = 0 \\ \bullet y_i \notin \mathcal{T}_s^* \rightarrow \text{error} = 0.5 \end{array} \right) \\ &= 0.5 \times \underbrace{\Pr[y_i \notin \mathcal{T}_s^*]}_{0.368} + 0 \times \underbrace{\Pr[y_i \in \mathcal{T}_s^*]}_{0} \end{aligned}$$

$\Pr[\text{observation } i \notin \text{bootstrap sample } s]$

$$\Pr[\mathcal{T}_{s[i]}^* \neq y_i] = \frac{N-1}{N} \Rightarrow \Pr[y_i \notin \mathcal{T}_s^*] = \left(\frac{N-1}{N}\right)^N$$

same for all positions

$$= e^{-1} \approx 0.368$$

$$E[\hat{E}_{\text{err}}^{\text{wrong}}] \Big|_{\substack{X \perp Y \\ 1\text{NN}}} \approx 0.5 \times e^{-1} = 0.184$$

An important fact

$$\Pr[\text{observation } i \text{ belongs to a bootstrap sample } s] = 1 - e^{-1}$$

≈ 0.632

approximation of $1 - \left(\frac{N-1}{N}\right)^N$

CORRECT APPROACH

$$\mathcal{T} = \{z_1, \dots, z_n\}$$

$$\mathcal{T}_s^* = \{z_1^*, \dots, z_{n_s}^*\} \text{ resampling with replacement}$$

→ there are original observations which are included more than once
 ⇒ there are original observations which are not included at all

\uparrow These can be used as a test set as they are not used in the training process.

$$\widehat{\text{Err}}^{(1)} =$$

where $|c_i|$ is the number of bootstrap samples that do not contain i

Issues

→ the average number of unique observations in the training set is → not so far from 0.5N, that is the value related to 2-fold CV

→ similar issues of training-set-size bias than 2-fold CV
 → result in a \uparrow of the error

To solve the issue, the .632 estimator has been developed

$$\widehat{\text{Err}}^{(.632)} =$$

In general, it works well, but in some case it fails, like in our $X \perp Y$

$$\overline{\text{err}} = 0 \rightarrow \widehat{\text{Err}}^{(.632)} = 0.632 \widehat{\text{Err}}^{(1)}$$

Further improvements ".632+ estimator"

- based on the quantity $\hat{\delta}$, the no-information-error rate
 error that we obtain when inputs and class label are independent
 $\hat{\delta}$ is computed by permuting x and y separately, we compute the prediction error for each combination of y_i and x_j

$$\hat{\delta} =$$

- $\hat{\delta}$ is used to compute the overfitting rate

$$\hat{R} =$$

$$0 \leq R \leq 1$$

R no overfitting

- finally

$$\widehat{\text{Err}}^{(.632+)} =$$

where $w =$

N.B.: bootstrap sample has the same size of the original sample

CORRECT APPROACH

$$\mathcal{T} = \{z_1, \dots, z_N\}$$

$$\mathcal{T}_s^* = \{z_1^*, \dots, z_{N_s^*}^*\} \text{ resampling with replacement}$$

→ there are original observations which are included more than once
 ⇒ there are original observations which are not included at all

$\hat{\epsilon}_{\text{test}}$ These can be used as a test set as they are not used in the training process.

$$\hat{\text{Err}}^{(1)} = \frac{1}{N} \sum_{i=1}^N \frac{1}{|\mathcal{C}^{(i)}|} \sum_{j \in \mathcal{C}^{(i)}} L(y_j, \hat{f}_s(x_i))$$

where $|\mathcal{C}^{(i)}|$ is the number of bootstrap samples that do not contain i

ISSUES

→ the average number of unique observations in the training set is $0.632N$ → not so far from $0.5N$, that is the value related to 2-fold CV

→ similar issues of training-set-size bias than 2-fold CV
 → result in a small overestimation of the error

To solve the issue, the .632 estimator has been developed

$$\hat{\text{Err}}^{(.632)} = 0.368 \bar{\text{err}} + 0.632 \hat{\text{Err}}^{(1)}$$

In general, it works well, but in some case it fails, like in our $X \perp Y$

$$\bar{\text{err}} = 0 \rightarrow \hat{\text{Err}}^{(.632)} = 0.632 \hat{\text{Err}}^{(1)}$$

Further improvements ".632+ estimator"

- based on the quantity $\hat{\gamma}$, the no-information-error rate
 error that we obtain when inputs and class label are independent
 $\hat{\gamma}$ is computed by permuting x and y separately, we compute the prediction error for each combination of y_i and x_j

$$\hat{\gamma} = \frac{1}{N} \sum_{i=1}^N \frac{1}{N} \sum_{j=1}^N L(y_i, \hat{f}(x_j))$$

- $\hat{\gamma}$ is used to compute the overfitting rate

$$\hat{R} = \frac{\hat{\text{Err}}^{(1)} - \bar{\text{err}}}{\hat{\gamma} - \bar{\text{err}}} \quad 0 \leq R \leq 1$$

R no overfitting

- finally

$$\hat{\text{Err}}^{(.632+)} = (1 - \hat{R}) \bar{\text{err}} + \hat{R} \hat{\text{Err}}^{(1)}$$

$$\text{where } \hat{R} = \frac{0.632}{1 - 0.368 \hat{R}}$$

N.B.: bootstrap sample has the same size of the original sample

Generalized Additive Models

- extensions of the (generalized) linear model

Linear model

- powerful tool

- can be used in several cases (regression, classification, ...)

Main limitations

- it suppose linear effects, often not true in reality

($\hat{\beta}$ is the increments in y when the corresponding x increases by 1)

In the context of regression, the (generalized) additive model has the form

$$E[Y|X_1, \dots, X_p] =$$

where

Y is the outcome

X_j are the predictors

f_j is a function which describe the effect of X_j

- we already saw that we can use $f_j(x_j) = x_j^2$, $f_j(x_j) = \log x_j$

- we can be more general, and use a nonparametric function (splines, kernel, ...)

Splines \rightarrow § 5.2

kernel \rightarrow § 6.1, § 6.2

- cubic splines \rightarrow bottom right of fig 5.2

- natural splines \rightarrow since the estimation outside the observations' range can be dangerous, the line is forced to be linear outside the range

Generalized Additive Models

- extensions of the (generalized) linear model

Linear model

- powerful tool

- can be used in several cases (regression, classification, ...)

Main limitations

- it suppose linear effects, often not true in reality

($\hat{\beta}$ is the increments in y when the corresponding x increases by 1)

In the context of regression, the (generalized) additive model has the form

$$E[Y|X_1, \dots, X_p] = \alpha + f_1(x_1) + \dots + f_p(x_p)$$

where

Y is the outcome

X_j are the predictors

f_j is a function which describe the effect of X_j

- we already saw that we can use $f_j(x_j) = x_j^2$, $f_j(x_j) = \log x_j$

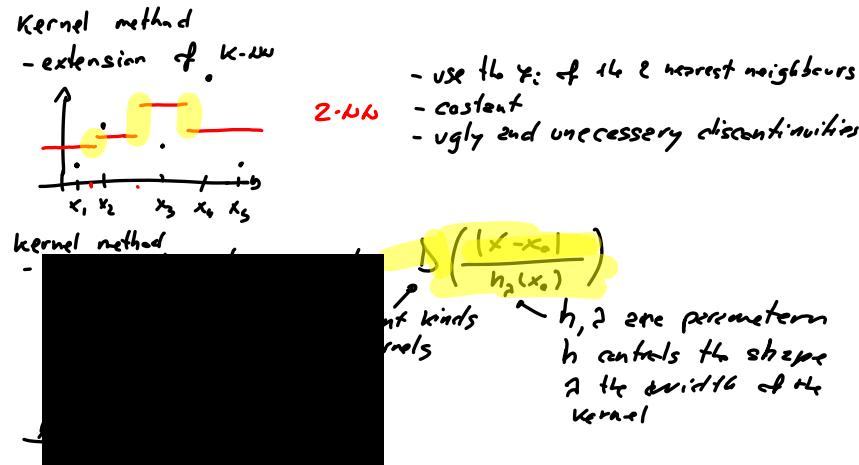
- we can be more general, and use a nonparametric function (splines, kernel, ...)

Splines \rightarrow § 5.2

kernel \rightarrow § 6.1, § 6.2

- cubic splines \rightarrow bottom right of fig 5.2

- natural splines \rightarrow since the estimation outside the observations' range can be dangerous, the line is forced to be linear outside the range

**GAM**

for functional effect, we can

$$E[Y|X] = \alpha + \beta_1 f_1(x_1) + \beta_2 f_2(x_2) + \dots + \beta_p f_p(x_p)$$

→ the least square estimator approach is usable

$$\widehat{E}[Y|X] = \alpha + f_1(x_1) + \dots + f_p(x_p)$$

→ backfitting algorithm

Generalized Additive Model

↳ extending the GLM (STK 3100)

GLM $\boxed{\text{function}} = \alpha + \beta^T X$ extending the linear model to all exponential families sampling models

e.g. logistic model

$$g(\cdot) = \text{logit} \quad \mu(x) = P[Y=1|x=x]$$

$$\log\left(\frac{\mu(x)}{1-\mu(x)}\right) = \alpha + \beta_1 x_1 + \dots + \beta_p x_p$$

GAM $\boxed{\text{function}} = \boxed{\text{function}}$

additive logistic regression : $\log\left(\frac{\mu(x)}{1-\mu(x)}\right) = \alpha + \sum_{j=1}^p f_j(x_j)$

Advantages of GAM:

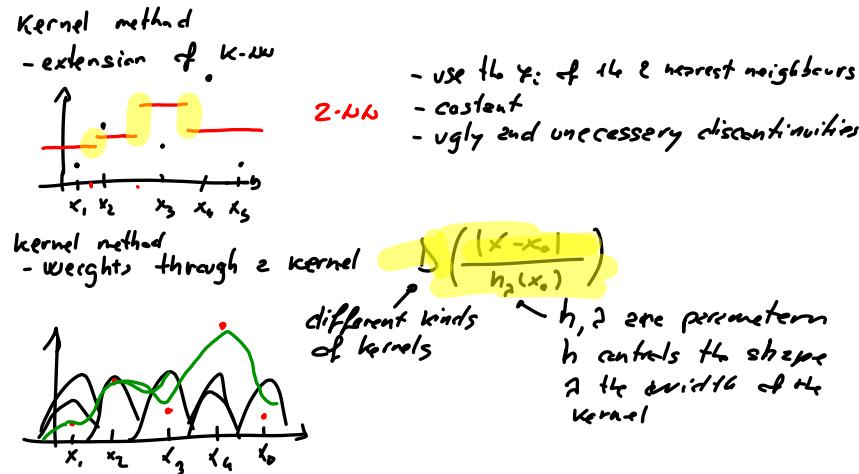
- flexibility, due to f (we can capture non-linear effects)
- interpretability, due to the additivity (not so different from the usual interpretation of GLM)

Note: not all effect need to be non-linear / linear

- semi parametric model $g(\mu(x)) = \underbrace{\beta^T x}_{\text{parametric}} + \underbrace{f(z)}_{\text{non-parametric}}$

e.g. semi parametric model: Cox model

$$\lambda(t) = \underbrace{\lambda_0(t)}_{\text{non-parametric}} \underbrace{\exp(x^T \beta)}_{\text{parametric part}}$$

**GAM**

for functional effect, we can

$$E[Y|X] = \alpha + \beta_1 f_1(x_1) + \beta_2 f_2(x_2) + \dots + \beta_p f_p(x_p)$$

$\downarrow \text{reg}(x_i)$

→ the least square estimator approach is usable

$$E[Y|X] = \alpha + f_1(x_1) + \dots + f_p(x_p)$$

→ backfitting algorithm

Generalized Additive Model

↳ extending the GLM (STK 3100)

GLM $g(\mu(x)) = \alpha + \beta^T x$ extending the linear model to all exponential families sampling models

$\downarrow \text{link function}$

e.g. logistic model

$$g(\cdot) = \text{logit} \quad \mu(x) = P[Y=1|x=x]$$

$$\log\left(\frac{\mu(x)}{1-\mu(x)}\right) = \alpha + \beta_1 x_1 + \dots + \beta_p x_p$$

GAM $g(\mu(x)) = \alpha + \sum_{j=1}^p f_j(x_j)$

additive logistic regression : $\log\left(\frac{\mu(x)}{1-\mu(x)}\right) = \alpha + \sum_{j=1}^p f_j(x_j)$

Advantages of GAM:

- flexibility, due to f (we can capture non-linear effects)
- interpretability, due to the additivity (not so different from the usual interpretation of GLM)

Note: not all effect need to be non-linear / linear

- semi parametric model $g(\mu(x)) = \underbrace{x^T \beta}_{\text{parametric}} + \underbrace{f(z)}_{\text{non-parametric}}$

e.g. semi parametric model: Cox model

$$\lambda(t) = \underbrace{\lambda_0(t)}_{\text{non-parametric}} \underbrace{\exp(x^T \beta)}_{\text{parametric part}}$$

10 Boosting

Leo Breiman: "[Boosting is] the best off-the-shelf classifier in the world"

- originally developed for classification;
- translated into the statistical world and use for all purposes (regression, ...)
- extended in its use;
- interpretable (Gini)

Starting challenge:

"Can a committee of blackheads somehow arrive at a highly reasoned decision, despite the weak judgement of the individual members?"

goal: obtain a good classifier

- apply "weak estimators", in the context of classification classifiers which lead to a solution only slightly better than a random choice

idea:

[redacted]

at each iteration AdaBoost weight \downarrow to misclassified observations

gives more

AdaBoost

10 Boosting

Leo Breiman: "[Boosting is] the best off-the-shelf classifier in the world"

- originally developed for classification;
- translated into the statistical world and use for all purposes (regression, ...)
- extended in its use;
- interpretable (Gini)

Starting challenge:

"Can a committee of blackheads somehow arrive at a highly reasoned decision, despite the weak judgement of the individual members?"

goal: obtain a good classifier

- apply "weak estimators", in the context of classification classifiers which lead to a solution only slightly better than a random choice

idea: repeatedly apply a weak estimator to modifications of the data

gives more weight to the missclassified observations

AdeBoost

Consider a two class classification problem

$$Y_i \in \{-1, 1\}$$

X_i : the vector of inputs

Adaboost algorithm

① initialize, weights are

② for m from

a)

b)

c)

d)

OK

③ co

Consider a two class classification problem

$$Y_i \in \{-1, 1\}$$

X_i : the vector of inputs

AdaBoost algorithm

① initialize, weights are $(\frac{1}{N}, \frac{1}{N}, \dots, \frac{1}{N}) = w^{[0]}$

② for m from 1 to m -step (M)

a) fit the weak estimator $\hat{g}(x)$ to the weighted data;

b) compute the weighted in-sample missclassification rate

$$\text{err}^{[m]} = \frac{\sum_{i=1}^N w_i^{[m-1]} \mathbb{1}[y_i \neq \hat{g}^{[m]}(x_i)]}{\sum_{i=1}^N w_i^{[m-1]}}$$

c) compute $\alpha_m = \log \left(\frac{1 - \text{err}^{[m]}}{\text{err}^{[m]}} \right)$

α_m is used to weight the contribution of the $\hat{g}^{[m]}(x)$ to the final estimate (classification)

d) update the weights

$$\tilde{w}_i = w^{[m-1]} \exp \left\{ \alpha_m \underbrace{\mathbb{1}[y_i \neq \hat{g}^{[m]}(x_i)]}_{\text{reweight only missclassified observations}} \right\} \quad i = 1, \dots, N$$

OK

$$w_i^{[m]} = \frac{\tilde{w}_i}{\sum_{i=1}^N \tilde{w}_i}$$

③ compute final result

$$\hat{f}_{\text{AdaBoost}} = \text{sgn} \left(\sum_{m=1}^{m-\text{stop}} \alpha^{[m]} \hat{g}^{[m]}(x) \right)$$

Example

$$\text{step } \{0\} \quad w = \left(\frac{1}{10}, \frac{1}{10}, \dots, \frac{1}{10} \right)$$

$$\text{step (i)} \quad \text{err} = \frac{\sum_{i=1}^n \frac{1}{10} \mathbf{1}[y_i \neq \hat{g}_i]}{\sum_{i=1}^n \frac{1}{10}} = \frac{\frac{3}{10}}{\frac{10}{10}} = 0.3$$

$$\alpha_1 = \log \frac{1 - err}{err} = \log 0.7 + \log 0.3 \approx 0.86$$

$$\tilde{w} = \left(\underbrace{\exp(0.84) \frac{1}{10}, 0.23, 0.23, 0.1, \dots, 0.1}_{\text{misclassified in the first iteration}} \right)$$

$$W^{(1)} \approx (0.17, 0.17, 0.17, 0.07, \dots, 0.07)$$

Statistical view of Boosting

- functional gradient descent/d algorithm
forward stagewise (additive) modelling

→ Sep Adaboost is an iterative procedure to minimize a loss-function, in particular an exponential loss-function

$$L(y, f(x)) = \exp\{-y f(x)\}$$

Consider a generic step m

- the current classifier is $\hat{f}^{(m-1)} = \sum_{k=1}^{m-1} \alpha_k \hat{g}^{(k)}(x)$ 1...m-1 given in learning
(if the algorithm had stopped at the m-1 iteration)
 - the goal is to find $(\alpha_m, g_m) = \arg \min_{\alpha_m g_m} \sum_{i=1}^n \exp \left\{ -y_i \sum_{k=1}^m \alpha_k \hat{g}^{(k)}(x_i) \right\}$

$$\alpha_m, g_m = \arg \min_{\alpha, g} \sum_{i=1}^n \exp \left\{ -y_i \left(\underbrace{\sum_{k=1}^{m-1} \alpha_k g_k}_{\hat{f}^{(m-1)}} + \alpha g \right) \right\}$$

where $w_i = \exp\{-g f^{(m-1)}\}$, which do not depend neither on m nor on g (given from the previous iteration) $g \in \{-1, 1\}$

- two step procedure: first we minimize with respect to g $\in \{1, 1\}$

$$\begin{aligned}
 & \text{argmin}_g \sum_{i=1}^n w_i^{(i-1)} \exp(-y_i g) \\
 & = \text{argmin}_g \left\{ \underbrace{\sum_{g=y}^{(i-1)} w_i e^{-\alpha}}_{\frac{e^{-\alpha} \sum w_i}{g-y}} + \underbrace{\sum_{g \neq y}^{(i-1)} w_i e^{\alpha}}_{\frac{(e^\alpha - e^{-\alpha}) \sum w_i}{g-y}} \right\} \\
 & = \text{argmin}_g \left\{ e^{-\alpha} \sum_{i=1}^{(i-1)} w_i + (e^\alpha - e^{-\alpha}) \sum_{g \neq y} w_i \right\} \\
 & \text{argmin} \left\{ \underbrace{\sum_{g=y}^{(i-1)} w_i e^{-\alpha}}_{\frac{\sum w_i e^{-\alpha}}{g-y}} + \underbrace{\sum_{g \neq y}^{(i-1)} w_i e^{-\alpha}}_{\frac{\sum w_i e^{-\alpha}}{g-y}} - \underbrace{\sum_{g \neq y}^{(i-1)} w_i e^{-\alpha}}_{\frac{\sum w_i e^{-\alpha}}{g-y}} + \underbrace{\sum_{g \neq y}^{(i-1)} w_i e^{-\alpha}}_{\frac{\sum w_i e^{-\alpha}}{g-y}} \right\} \\
 & = \text{argmin}_g \left\{ e^{-\alpha} \sum_{i=1}^{(i-1)} w_i + (e^\alpha - e^{-\alpha}) \sum_{i=1}^{(i-1)} w_i \right\} \mathbf{1}_{[g \neq y_i]} \\
 & g = \text{argmin}_g \left\{ \sum_{i=1}^n w_i \mathbf{1}_{[g_i \neq y_i]} \right\}
 \end{aligned}$$

$$\begin{aligned}
 &= \arg \min_{\alpha, g} \sum_{i=1}^n w_i^{(m-1)} \exp \{-g \alpha g\} \\
 &\quad \arg \min_{\alpha} \sum_{g=y}^{m-1} w_i^{(m-1)} \exp \{-g \alpha g\} \quad \frac{\partial L}{\partial \alpha} = \sum \\
 &\quad y=g \rightarrow \sum_{g=y}^{m-1} w_i^{(m-1)} \exp \{-\alpha\} \\
 &\quad y \neq g \rightarrow \sum_{g \neq y}^{m-1} w_i^{(m-1)} \exp \{\alpha\} \\
 &\frac{\partial L}{\partial \alpha} = - \sum_{g=y}^{m-1} w_i^{(m-1)} \exp \{-\alpha\} + \sum_{g \neq y}^{m-1} w_i^{(m-1)} \exp \{\alpha\} = 0 \\
 &\text{Multiply both terms for } e^\alpha \\
 &e^{2\alpha} \sum_{g \neq y}^{m-1} w_i^{(m-1)} = \sum_{g=y}^{m-1} w_i^{(m-1)} \\
 &e^{2\alpha} = \frac{\sum_{i=1}^n w_i^{(m-1)} - \sum_{g \neq y}^{m-1} w_i^{(m-1)}}{\sum_{g \neq y}^{m-1} w_i^{(m-1)} + \sum_{g \neq y}^{m-1} w_i^{(m-1)}} \\
 &\alpha = \frac{1}{2} \log \left(\frac{1 - \text{err}}{\text{err}} \right) \\
 \text{where } \text{err} &= \frac{\sum_{g \neq y}^{m-1} w_i^{(m-1)}}{\sum_{i=1}^n w_i^{(m-1)}}
 \end{aligned}$$

- $\hat{g}^{(m)}$ minimizer of the weighted missclassification
- $\alpha = \frac{1}{2} \log \left(\frac{1 - \text{err}}{\text{err}} \right)$

Our general classifier is updated as

$$\hat{f}^{(t+1)} = \hat{f}^{(t)} + \alpha_m \hat{g}^{(m)}$$

which causes the weights of the next iteration to be

$$w_i^{(t+1)} = w_i^{(t)} \cdot \exp \left\{ -\alpha y_i \hat{g}_i^{(m)} \right\}$$

$$\begin{aligned} \text{Since } -y_i \hat{g}_i^{(m)} &= -\sum_{g_j \neq y_i} 1 + \sum_{g_j \neq y_i} (+1) + \sum_{g_j = y_i} 1 \\ &= 2 \sum_{g_j \neq y_i} 1 - 1 \end{aligned}$$

$$w_i^{(t+1)} = w_i^{(t)} \cdot \exp \left\{ \alpha \left(2 \sum_{g_j \neq y_i} 1 - 1 \right) \right\}$$

$$= w_i^{(t)} \exp \left\{ 2\alpha \sum_{g_j \neq y_i} (y_i \neq g_j) - \alpha \right\}$$

$$= w_i^{(t)} e^{-\alpha} e^{2\alpha \sum_{g_j \neq y_i} (y_i \neq g_j)}$$

constant for each observation
→ can be ignored

2α is α in the algorithm

→ AdaBoost minimizes the exponential loss criterion by a forward stagewise procedure

Note:

- this statistical view allows us to interpret the results of the procedure.

In particular, it can be shown that the minimizer of the exponential loss

$$f^*(x) = \arg \min_{f(x)} E_{Y|x} [e^{-y f(x)}] = \frac{1}{2} \log \frac{\Pr[Y=1|x]}{\Pr[Y=-1|x]}$$

$$\text{alternatively: } \Pr[Y=1|x] = \frac{1}{1+e^{-f^*(x)}}$$

$\frac{1}{2}$ the log-odds for $\Pr[Y=1|x]$ → Ex 10.2

- other loss-functions lead to the same minimizer, for example the negative log-likelihood

$$\hat{\pi} = \Pr[Y=1|x] = \frac{e^{f(x)}}{e^{f(x)} + e^{-f(x)}} = \frac{1}{1+e^{-2f(x)}}$$

$$\cdot y' = \frac{y+1}{2} \in \{0;1\}$$

then

$$l(\hat{\pi}) = y' \log \hat{\pi} + (1-y') \log (1-\hat{\pi})$$

$$\Rightarrow -l(\hat{\pi}) = \log \left(1 + e^{-2y' f(x)} \right)$$

Exercise 10.2

$$f^*(x) = \arg \min_{f(x)} E_{Y|x} [e^{-Y f(x)}]$$

$$\frac{\partial}{\partial f(x)} E_{Y|x} [e^{-Y f(x)}] = E_{Y|x} [-Y e^{-Y f(x)}]$$

$$E_{Y|x} [-Y e^{-Y f(x)}] = 0 \quad Y = \begin{cases} -1 & \Pr[Y = -1|x] \\ 1 & \Pr[Y = 1|x] \end{cases}$$

$$+ (1) e^{-(+1) f(x)} \Pr[Y = -1|x] - (1) e^{-(-1) f(x)} \Pr[Y = 1|x] = 0$$

$$e^{f(x)} e^{f(x)} \Pr[Y = -1|x] = e^{-f(x)} \Pr[Y = 1|x] \quad \text{multiply both sides by } e^{f(x)}$$

$$e^{2f(x)} \Pr[Y = -1|x] = \Pr[Y = 1|x]$$

$$e^{2f(x)} = \frac{\Pr[Y = 1|x]}{\Pr[Y = -1|x]}$$

$$f(x) = \frac{1}{2} \log \left(\frac{\Pr[Y = 1|x]}{\Pr[Y = -1|x]} \right)$$

Exercise 10.4

$$X = (x_1, \dots, x_{10}) \quad x_j \sim N(0; 1) \quad x_j \text{ iid} \quad N = 2000$$

$$Y_i = \begin{cases} 1 & \text{if } \sum_{j=1}^{10} x_{ij}^2 > \chi_{10}^2(0.5) \\ -1 & \text{if } \sum_{j=1}^{10} x_{ij}^2 \leq \chi_{10}^2(0.5) \end{cases}$$

\rightarrow obs $\left\{ \begin{array}{c} 1 \\ 2 \\ \vdots \\ N \end{array} \right\} \quad \begin{array}{cccccc} \bar{x}_1 & \bar{x}_2 & \dots & \bar{x}_{10} & \text{apply } (\underline{x^2}, \underline{1}, \underline{\text{sum}}) \\ \underbrace{\bar{x}_{11} & \bar{x}_{12} & \dots & \bar{x}_{10} } & & & & & \end{array}$

Statistical boosting { gradient boosting
likelihood-based boosting

From the previous lecture:

AdaBoost : classifier

- weak estimator

- loss function

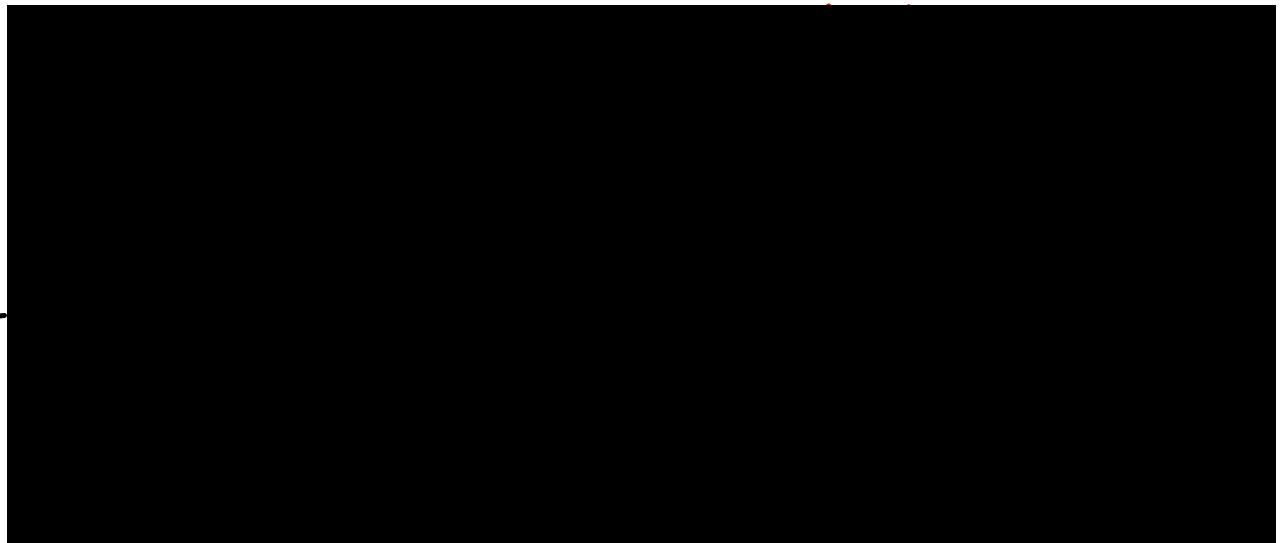
- iteratively apply a weak estimator to modifications of the data in order to minimize a loss function

↳ AdaBoost : weight more missclassified observations

AdaBoost : - stump
- tree
- ...

for classification \rightarrow AdaBoost
 $e^{-Y_t w_t}$

from classification to regression



Statistical boosting { gradient boosting
likelihood-based boosting

From the previous lecture:

AdeBoost : classifier

- weak estimator

- loss function

- iteratively apply a weak estimator to modifications of the data in order to minimize a loss function

↳ AdeBoost : weight more missclassified observations

AdeBoost : - stump
- tree
- ...

for classification \rightarrow AdeBoost^t
 $e^{-Y_t w_t}$

from classification to regression

- loss function : RSS LS estimator

- weak estimator : $\hat{y} = \frac{\nu}{\nu + 1} X^T X^{-1} X^T y$ $\nu \rightarrow 0$

penalty parameter $0 < \nu < 1$ weak estimator

↳ makes our LS "weak" default = 0.1

- modification of the data : $y \rightarrow u$ residuals

focusing on the non explain part of the variation outcome

L_2 Boost algorithm for linear regression

① Initialization:

(first modification)

② for m from 1 to m_stop

a)

b)

c)

↑

③ final estimate

Note:

$m \rightarrow \infty$,

need of an early stop (find the "right" m_stop) in order to not overfit (to find the best balance between bias and variance for the prediction error)

- m_stop is the crucial tuning parameter

if it is too small: too much bias (our model does not explain the outcome variation)

if it is too big : too much variance (we overfit the data)

complexity parameter

boosting has a second tuning parameter, λ (is not so important, because smaller values \Rightarrow more steps (iterations))
larger values \Rightarrow less steps

X must be centred $E[X_j] = 0$
(it is ok to standardize)

L_2 Boost algorithm for linear regression

① Initialization: initialize the regression coefficient estimate $\hat{\beta}^{[0]} = (0, \dots, 0)$

(first modification of the data: $v = y - X\beta = y - 0 = y$)

② for m from 1 to m_stop

a) fit the weak estimator to the modification of the data

$$\hat{b}^{[m]} = \nu (X^T X)^{-1} X^T v$$

b) update the estimate $\hat{\beta}^{[m]} = \hat{\beta}^{[m-1]} + \hat{b}^{[m]}$

c) modify the data: $v = y - X^T \hat{\beta}^{[m]}$

③ final estimate

$$\hat{\beta}_{\text{L2Boost}} = \sum_{m=1}^{m_{\text{stop}}} \hat{b}^{[m]} = \hat{\beta}^{[m_{\text{stop}}]}$$

Note:

$$m \rightarrow \infty, \hat{\beta}_{\text{L2Boost}} \rightarrow \hat{\beta}_{\text{OLS}}$$

need of an early stop (find the "right" m_{stop}) in order to not overfit (to find the best balance between bias and variance for the prediction error)

- m_{stop} is the crucial tuning parameter

if it is too small: too much bias (our model does not explain the outcome variation)

if it is too big: too much variance (we overfit the data)

complexity parameter

boosting has a second tuning parameter, ν (is not so important, because smaller values \Rightarrow more steps (iterations))
larger values \rightarrow less steps

X must be centered $E[X_j] = 0$
(it is ok to standardize)

L_2 Boost algorithm in general

- the goal is to minimize the loss function. At each step we want to identify the direction of the greatest decrease of the loss function

and negative gradient:

$$-\frac{\partial L(y, f(x))}{\partial f(x)}$$

$$\text{e.g. } \frac{\partial \sum_{i=1}^n (y_i - \hat{f}(x_i))^2}{\partial \hat{f}(x)} = \frac{2}{n} \sum_{i=1}^n (y_i - \hat{f}(x_i)) \quad \text{residuals}$$

$$\frac{1}{\kappa} \exp\left\{-\frac{1}{2} (y - \hat{f}(x))\right\}$$

In general:

$$\textcircled{1} \text{ Initialization: } \hat{f}(x) = 0 \quad \text{or} \quad \hat{f}(x) = \bar{y}$$

$$\textcircled{2} \text{ for } m \text{ from 1 to } m_{\text{stop}} \quad \text{lin. reg.}$$

(a) derive $b = \frac{\text{RSS}}{\text{lin. reg. residuals}}$

(b) fit our weak estimator:

$$\hat{f}^{[m]}(x) = \hat{f}^{[m-1]}(x) + b \cdot \text{lin. reg. residuals}$$

$$(c) \text{ update the estimate } \hat{f}^{[m]} = \hat{f}^{[m-1]} + \hat{\beta}$$

$$\textcircled{3} \text{ Finalization: } \hat{f}_{\text{boost}} = \hat{f}^{[m_{\text{stop}}]}$$

$$\text{GAM: } g = \sum_{j=1}^P f_j(x_j)$$

L_2 Boost for High Dimensional Data

- one of the advantages of boosting is that we can handle HAD
→ component wise version of boosting

L_2 Boost algorithm in general

- the goal is to minimize the loss function. At each step we want to identify the direction of the greatest decrease of the loss function

and negative gradient :

$$-\frac{\partial L(y, f(x))}{\partial f(x)}$$

$$\text{e.g. } \frac{\partial \sum_{i=1}^n (y_i - x_i^\top \beta)^2}{\partial x^\top \beta} = \cancel{\frac{2}{n}} \underbrace{\sum_{i=1}^n (y_i - x_i^\top \hat{\beta})}_{\text{residuals}}$$

$$\frac{1}{n} \exp\left\{-\frac{1}{2}(y - x^\top \beta)\right\}$$

In general:

① Initialization : $\hat{f}(x) = 0$ or $\hat{f}(x) = \bar{y}$

② for m from 1 to m stop

(a) derive $b = -\frac{\partial L(y, f(x))}{\partial f(x)}$

(b) fit our weak estimator : $\hat{g} = g(v, x, 0)$

(c) update the estimate $\hat{f}^{[m]} = \hat{f}^{[m-1]} + \hat{g}$

③ Finalization $\hat{f}_{\text{boost}} = \hat{f}^{[\text{last}]} \quad [$

GAM : $g = \sum_{j=1}^P f_j(x_j)$

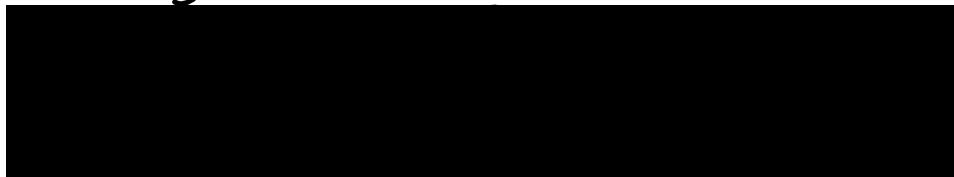
L_2 Boost for High Dimensional Data

- one of the advantages of boosting is that we can handle HAD
- component wise version of boosting

Componentwise Boosting

- Linear regression model Gaussian

①



- ② For m from 1 to m-stop

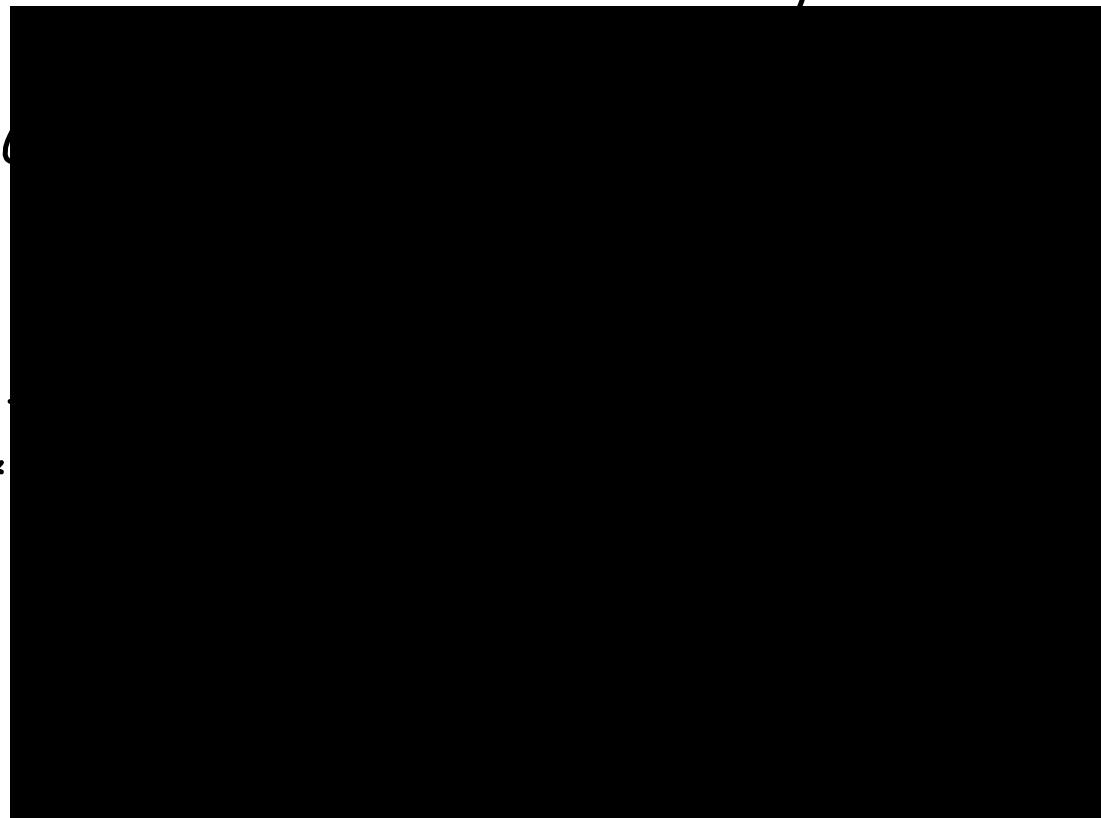
a)

b)

j*

c)

d)



$\text{fit} \in \text{GAM}$ in R with Boosting

$$y = f_1(x_1) + f_2(x_2) + \dots + f_p(x_p)$$

splines tree → function $m\text{boost}$ of the package $m\text{boost}$

$$\mu(y) = X^T \beta \rightarrow \text{glmboost}$$

R Gaussian regression,
logistic regression

:

Componentwise Boosting

- Linear regression model

① Initialization $\hat{\beta}_j^{(0)} = 0 \quad j = 1, \dots, p$
 $(v = y - X\hat{\beta}^{(0)}) = y$

② For m from 1 to m -step

- compute possible updates for each dimension of the regression coefficient vector separately
 (fit a **weak** estimator on each dimension of X)

$$\hat{b}_j = \frac{\sum_{i=1}^n x_i^{(j)} v_i}{\sum_{i=1}^n x_i^{(j)2}} \quad j = 1, \dots, p$$

- select the best update among the p possibilities

$$j^* : \arg \min_j \sum_{i=1}^n (v_i - x_i^{(j)} b_j)^2$$

- update the j^* -th regression coefficient

$$\hat{\beta}^{(m)} = \hat{\beta}^{(m-1)} + (0, \dots, 0, \hat{b}_{j^*}, 0, \dots, 0)$$

- modify the data $v = y - X\hat{\beta}^{(m)}$

Fit a GAM in R with Boosting

$$y = f_1(x_1) + f_2(x_2) + \dots + f_p(x_p)$$

splines tree → function `mboost` of the package `mboost`

$$\mu(y) = X^\top \beta \rightarrow \text{glmboost}$$

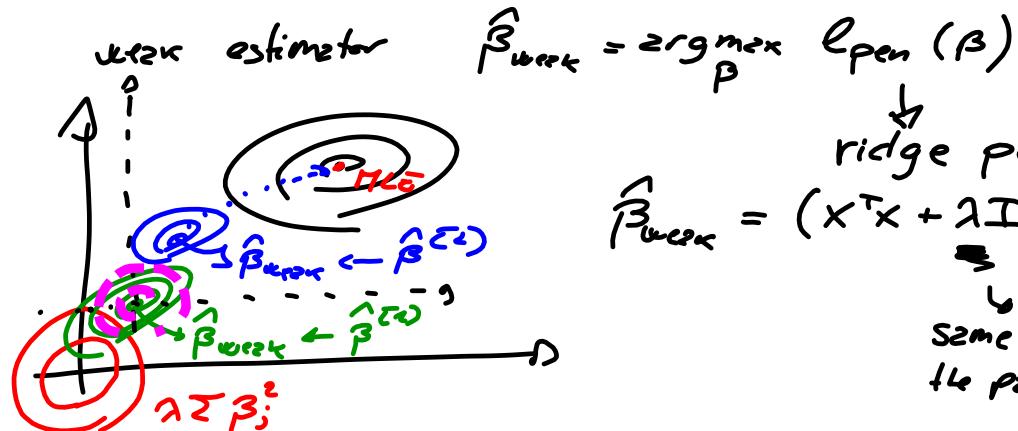
R Gaussian regression,
logistic regression

:

Likelihood-based boosting

- fully statistical approach vs likelihood-based
- weak estimator:

$$\text{standard estimator } \hat{\beta}_{MLE} = \arg \max_{\beta} \ell(\beta)$$



$$\hat{\beta}_{weak} = (X^T X + \lambda I)^{-1} X^T y$$

ridge penalty
same meaning of the parameter λ

Boosting Ridge (Gaussian regression)

- initialize $\hat{\beta}^{[0]} = (0, \dots, 0) \rightarrow v = y - X \hat{\beta}^{[0]}$
- fit the weak estimator $\hat{b} = (X^T X + \lambda I)^{-1} X^T y$

$$\frac{\partial \ell_{pen}(\beta)}{\partial \beta} = 0$$

$\ell_{pen} \downarrow$

$\frac{\partial \ell_{pen}}{\partial \beta}$

Kernel of ℓ
Gaussian log-likelihood

$$\hat{\beta}^{[1]} \exp \left\{ -\frac{1}{2} (y - X\beta)(y - X\beta)^T \right\}$$

$\hat{b} =$ [REDACTED]

$$(b) \hat{\beta}^{[n]} = \hat{\beta}^{[n-1]} + \hat{b}$$

(c)

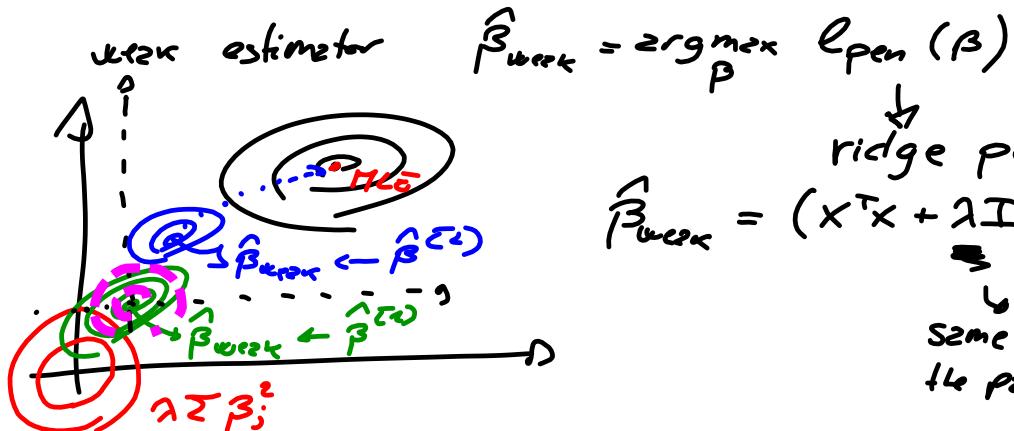
[REDACTED]

$$3 \hat{\beta}_{\text{Boost}} = \sum_{n=1}^{m, \text{step}} \hat{b}^{[n]}$$

Likelihood-based boosting

- fully statistical approach vs likelihood-based
- weak estimator:

standard estimator $\hat{\beta}_{MLE} = \arg \max_{\beta} \ell(\beta)$



weak estimator $\hat{\beta}_{weak} = \arg \max_{\beta} \ell_{pen}(\beta)$

ridge penalty

$$\hat{\beta}_{Ridge} = (X^T X + \lambda I)^{-1} X^T y$$

same meaning of the parameter ↗

Boosting Ridge (Gaussian regression)

- initialize $\hat{\beta}^{[1]} = (0, \dots, 0) \rightarrow v = y - X\hat{\beta}^{[1]}$
- fit the weak estimator $\hat{b} = (X^T X + \lambda I)^{-1} X^T y$

$$\frac{\partial \ell_{pen}(\beta)}{\partial \beta} = 0 \quad \ell_{pen}(y - X\beta)(y - X\beta)^T + \lambda \beta \beta^T \quad \checkmark$$

$$\frac{\partial \ell_{pen}}{\partial \beta} : -X^T(y - X\beta) + \lambda \beta = 0$$

Kernel of 2 Gaussian log-likelihood

$$\exp \left\{ -\frac{1}{2} (y - X\beta)(y - X\beta)^T \right\} \quad -X^T y + X^T X \beta + \lambda \beta = 0$$

$$\beta (X^T X + \lambda I) = X^T y$$

$$\hat{\beta} = \hat{\beta}^{[1]} = (X^T X + \lambda I)^{-1} X^T y$$

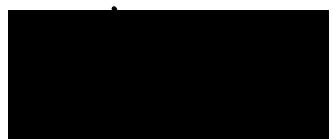
(b) $\hat{\beta}^{[n]} = \hat{\beta}^{[n-1]} + \hat{b}$

- (c) modification of the data
(add an offset in the log-likelihood)

$$\frac{1}{2} (y - X\hat{\beta}^{[n]})^T - X\beta (y - X\hat{\beta}^{[n]})^T - X\beta$$

3 $\hat{\beta}_{\text{Boost}} = \sum_{n=1}^{m, \text{step}} \hat{b}^{[n]}$

weak estimator is the ridge estimator
 Gaussian regression \rightarrow likelihood-based boosting \rightarrow gradient boosting \rightarrow some model when we



Exercise

$$\hat{y} = X \hat{\beta}_{\text{boost}} = \sum_{m=1}^{m_{\text{step}}} S(I - S)^m y = I - (I - S)^{m_{\text{step}}+1} y$$

where $S = X b$

using $b = (X^T X + \lambda I)^{-1} X^T y$ Boosting with ridge estimator

Show through SVD that Boosting ridge and ridge regression provide different shrinkage effect

ridge regression : $\frac{d_j^2}{d_j^2 + \lambda}$

Boosting Ridge : $(1 - (1 - \frac{d_j^2}{d_j^2 + \lambda}))^{m_{\text{step}}+1})$

weak estimator is the ridge estimator
 Gaussian regression \rightarrow likelihood-based boosting \rightarrow some model when we choose ψ and λ in the right way
 gradient boosting

Exercise

$$\hat{y} = X \hat{\beta}_{\text{boost}} = \sum_{m=1}^{m_{\text{step}}} S(I - S)^m y = I - (I - S)^{m_{\text{step}}+1} y$$

where $S = X b$

using $b = (X^T X + \lambda I)^{-1} X^T y$ Boosting with ridge estimator

Show through SVD that Boosting ridge and ridge regression provide different shrinkage effect

ridge regression : $\frac{d_j^2}{d_j^2 + \lambda}$

Boosting Ridge : $(1 - (1 - \frac{d_j^2}{d_j^2 + \lambda}))^{m_{\text{step}}+1})$

Likelihood-based boosting

- Loss function
- weak estimator
- modification of the data

L_2 Boost

$$\hat{y}_j = \frac{\partial L(y, f(x))}{\partial f(x)}$$

penalized version of OLS

$$\hat{y}(X^T X)^{-1} X^T u$$

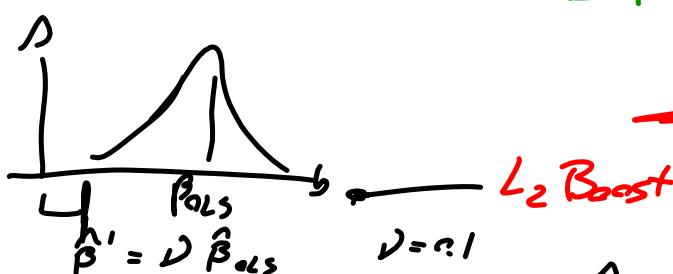
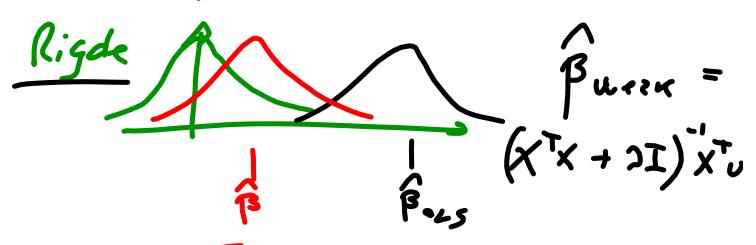
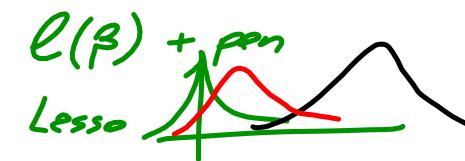
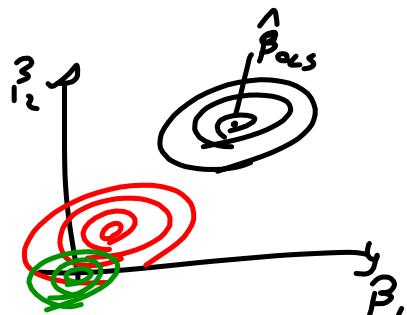
$u \propto \hat{L}_2$ OLS minimizer
 \hat{L}_2 loss

\hat{L}_2 is negative log-likelihood

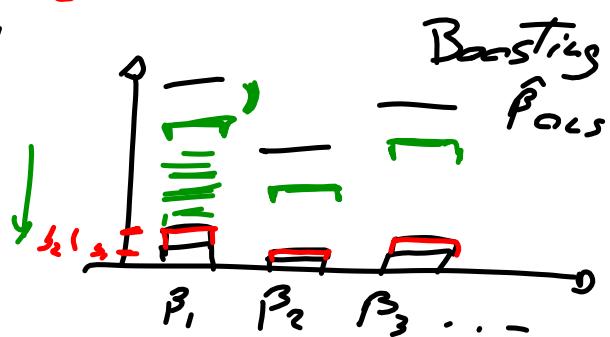
parametric version

$$L(y, f(x, \beta)) := \text{negative log-likelihood}$$

weak estimator \rightarrow instead of minimizing the negative log-likelihood, we minimize = penalized version of it



$$\hat{\beta}_{\text{boost}}^{(1)} = \hat{\beta}^{(0)} + \hat{\beta}^{(1)}$$



Algorithm

1) Initialization $\hat{\beta} = \underline{0}$ $v = y$

2) for m from 1 to m_{steps}
 $\ell_{\text{pen}}(0 + \beta; 1)$

$$\underbrace{\frac{\partial \ell_{\text{pen}}(0 + \beta; 1)}{\partial \beta} = 0}_{\text{Ridge penalty}} \quad (X^T X + I \lambda)^{-1} X^T v$$

3) $\hat{b}^{(m)} = (X^T X + \lambda I)^{-1} X^T v$

$$\ell_{\text{pen}}(\sum \hat{b}^{(m)} + \beta; 1)$$

Gaussian case $\ell_{\text{pen}}(\sum b^{(m)} + \beta; 1) =$

$$(y - \sum \hat{b}^{(m)} X - X\beta)^T (y - \sum \hat{b}^{(m)} X - X\beta) + \lambda \beta \beta^T$$

Exercise

$$\hat{f}(x) = \underline{\underline{\hat{\beta}}}$$

$$\hat{f}_b^{(0)} = X(X^T X + \lambda I)^{-1} X^T y - S y$$

$$\hat{f}_{\text{boost}}^{(1)} = X(X^T X + \lambda I)^{-1} X^T u$$

$$\hat{f}_{\text{boost}}^{(1)} = y - X \hat{\beta}^{(0)}$$

$$y - X(X^T X + \lambda I)^{-1} X^T y$$

$$= X(X^T X + \lambda I)^{-1} X^T (y - X(X^T X + \lambda I)^{-1} X^T y)$$

$$= S(I - S)y$$

$$\hat{f}_{\text{boost}}^{(2)} = X(X^T X + \lambda I)^{-1} X^T u$$

$$= X(X^T X + \lambda I)^{-1} X^T (y - \underline{X \hat{b}^{(0)}} - \underline{X \hat{b}^{(1)}})$$

$$= X(X^T X + \lambda I)^{-1} X^T (y - \underline{X(X^T X + \lambda I)^{-1} (y - X(X^T X + \lambda I)^{-1} X^T y)})$$

$$= X(X^T X + \lambda I)^{-1} X^T y$$

$$= S(y - S(I - S)y - S y)$$

$$= S(I - S + S^2 - S) y = \underline{S(I - S)^2} y$$

⋮

$$\hat{f}_b^{(m)} = S(I - S)^m y$$

m-th improvement

$$\hat{\beta}_{\text{boost}}^{(m-\text{step})} = \sum_{j=0}^{m-\text{step}} \hat{b}^{(j)}$$

$$\hat{y} = X \hat{\beta}^{(m-\text{step})} = \sum_{j=0}^{m-\text{step}} S(I - S)^j y = \underbrace{(I - (I - S)^{m-\text{step}+1})}_{H_m} y$$

$$\hat{y} = H_m y \quad \text{where } H = I - (I - S)^{m_stop+1}$$

$$= \underline{I} - \underline{(I - X(X^T X + \lambda I)^{-1} X^+)^{m_stop+1}}$$

use Singular Value Decomposition

$$X = UDV^T \quad \text{where } U \text{ spans the column space of } X, U^T U = I$$

V has n rows, $V^T V = VV^T = I$

D diagonal matrix with singular values
 $d_1 \geq d_2 \geq \dots \geq d_p$

$$\begin{aligned} H_m &= I - (I - UDV^T (UDU^T UDV^T + \lambda I)^{-1} VDU^T)^{m_stop+1} \\ &= I - (I - UDV^T (D^2 + \lambda I)^{-1} VDU^T)^{m_stop+1} \\ &= I - (I - U D^2 (D^2 + \lambda I)^{-1} U^T)^{m_stop} \\ &\vdots \\ &= U (I - (I - \tilde{D})^{m_stop+1}) U^T \quad \tilde{D} = (D^2 + \lambda I)^{-1} D^2 \\ &\quad \tilde{d}_j^2 = \frac{d_j^2}{d_j^2 + \lambda I} \end{aligned}$$

Ridge estimator (from lecture 4, notes page 8)

$$\text{ridge } X\hat{\beta}_{\text{ridge}} = U D^2 (D^2 + \lambda I)^{-1} U^T$$

$$\text{ridge-boosting } X\hat{\beta}_{\text{boost}} = U (I - (I - D^2 (D^2 + \lambda I)^{-1})^{m_stop+1}) U^T$$

no γ, λ, m_stop s.t. $D^2 (D^2 + \lambda I)^{-1} = I - (I - D^2 (D^2 + \lambda I)^{-1})^{m_stop+1}$

$$m_stop = 0.$$

apply only to first fine
the weak estimator (ridge)

$$\begin{aligned} D^2 (D^2 + \lambda I)^{-1} &= I - (I - D^2 (D^2 + \lambda I)^{-1})^{m_stop+1} \\ &= I - I + D^2 (D^2 + \lambda I)^{-1} \\ &= \lambda \end{aligned}$$

L_2 Boost is an algorithm that goes under the umbrella of gradient boosting
 + likelihood-based boosting } \rightarrow statistical boosting

L_2 Boost : • Bühlmann & Yu (2003)

Boosting with the L_2 loss: regression and classification

• Bühlmann & Hothorn (2007)

Boosting algorithms: regularization, prediction and model fitting

likelihood-based boosting : • Tutz & Binder (2006, 2007)

Boosting ridge regression

CAT with implicit variable selection
 by likelihood-based boosting

• general on statistical boosting: Mller et al. (2004)

The evolution of boosting algorithms

Advance regression and classification

Classification :

- KNN
- linear regression for classification
- LDA, QDA
- logistic regression
- AdaBoost
- L_2 Boost for classification (gradient boosting)

KNN
logistic
regression

Regression :

- linear regression and the OLS estimator
- penalized regression methods

- LASSO (L_1 penalty: $\lambda \sum_{j=1}^p |\beta_j|$)
- RIDGE (L_2 penalty: $\lambda \sum_{j=1}^p \beta_j^2$)
- Elastic-net (combination between L_1 and L_2 penalties)
- Boosting { L_2 Boost, likelihood-based boosting}
- LAR
- methods which use derived inputs
 - PCR
 - PLS

Central concept : bias-variance trade-off

OLS \rightarrow have 0 bias, minimize the variance

(Gauss-Markov theorem: OLS is BLUE)

e_i
s_i
t_i
n_i
g_i
+

We saw approaches which accept an increase in bias to have a (larger) decrease in variance

$$E[(Y_o - \hat{f}(x_o))^2] = E[Y_o^2 - 2Y_o \hat{f}(x_o) + \hat{f}(x_o)^2]$$

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

σ^2 = irreducible error
+ variance + bias²

$$Y = f(x_o) + \varepsilon \quad f(x_o) \perp \varepsilon$$

Advance regression and classification

Classification :

- KNN
- linear regression for classification
- LDA, QDA
- logistic regression
- AdaBoost
- L_2 Boost for classification (gradient boosting)

KNN
linear regression
logistic-regression

Regression :

- linear regression and the OLS estimator
- penalized regression methods
 - LASSO (L_1 penalty: $\lambda \sum_{j=1}^p |\beta_j|$)
 - RIDGE (L_2 penalty: $\lambda \sum_{j=1}^p \beta_j^2$)
 - Elastic-net (combination between L_1 and L_2 penalties)
 - Boosting { L_2 Boost, likelihood-based boosting}
 - LAR
- methods which use derived inputs
 - PCR
 - PLS

Central concept : bias-variance trade-off

OLS \rightarrow have 0 bias, minimize the variance

(Gauss-Markov theorem: OLS is BLUE)

e i g n t r n s g +

We saw approaches which accept an increase in bias to have a (larger) decrease in variance

$$E[(Y_o - \hat{f}(x_o))^2] = E[Y_o^2 - 2Y_o \hat{f}(x_o) + \hat{f}(x_o)^2]$$

$$\begin{aligned} &= E[Y_o^2] - E[\hat{f}(x_o)]^2 + E[\hat{f}(x_o)^2] - E[\hat{f}(x_o)]^2 + E[\hat{f}(x_o)]^2 \\ &\quad - 2(E[Y_o \hat{f}(x_o)]) - E[Y_o]E[\hat{f}(x_o)] + E[Y_o]E[\hat{f}(x_o)] \end{aligned}$$

$\text{cov}(Y_o, \hat{f}(x_o)) = 0$

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

$\stackrel{\text{= irreducible error}}{=} \text{variance} + \text{bias}^2$

$$Y = f(x_o) + \varepsilon \quad \hat{f}(x_o) \perp \varepsilon$$

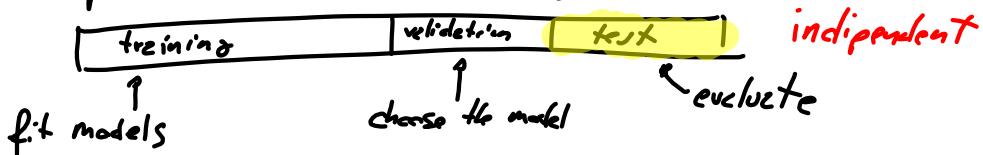
- prediction error
 - training error (underestimating the prediction error)
 - test error (computed on independent data)

- comparing methods based on prediction error
 - training error + estimate optimiser of the training error
 - AIC
 - BIC
 - try to estimate the test error
 - Cross-validation
 - Bootstrap approaches (0.632 and 0.632+ bootstrap)
- model selection
 - backward elimination ... 2nd combinations ↗ stepwise selection
 - forward selection ↗ stepback selection
 - best subset Selection
 - stopping criteria (AIC, BIC, significance level ↗ F-test)

~~splines~~

- problems with high-dimensional data
 - classical approaches (OLS) do not work
 - add constraints (LASSO - RIDGE - ...)
 - stagewise approaches (boosting)
 - || update only one dimension per time

- independence between training and test set



- CV: $k-1$ fold and the remaining fold are totally independent
training/test split

- when we apply a method which require standardization, we first standardize the training set (or the set which temporarily acts as a training set) without involving the observations in the test set. Then the observations on the test set will be "standardized" using means and standard deviations computed on the training set.

Exam 2015

Exercise 1:

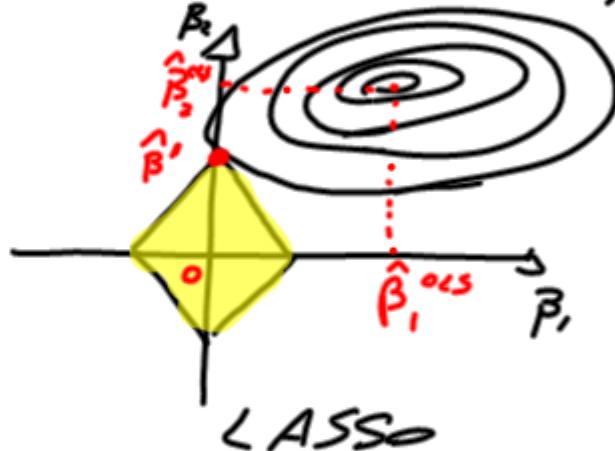
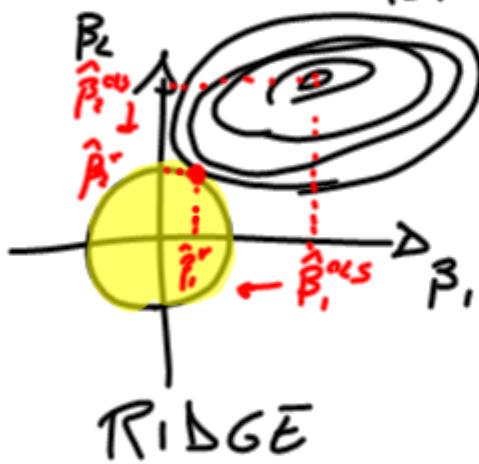
1: Lasso

2: ridge regression

$$\text{PRSS}_{\lambda}^{\text{Lasso}} = \sum_{i=1}^N (y_i - \beta_1 x_1 - \beta_2 x_2)^2 + \lambda(|\beta_1| + |\beta_2|)$$

$$\text{PRSS}_{\lambda}^{\text{ridge}} = \sum_{i=1}^N (y_i - \beta_1 x_1 - \beta_2 x_2)^2 + \lambda(\beta_1^2 + \beta_2^2)$$

b)



RIDGE

LASSO

Exercise 2

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

$$\hat{\beta}_{\text{OLS}} = (X^T X)^{-1} X^T y$$

$$\begin{aligned}\hat{\beta}_{\text{ridge}} &= \underbrace{(X^T X + \lambda I)^{-1} (X^T X)}_A \underbrace{(X^T X)^{-1} X^T y}_{\hat{\beta}_{\text{OLS}}} \\ &= A \hat{\beta}_{\text{OLS}}\end{aligned}$$

$$b) N \rightarrow \infty, X^T X \approx N \Sigma$$

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}^{-1} = \frac{1}{ad - bc} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$$

$$\hat{\beta}_{\text{ridge}} = A \hat{\beta}_{\text{OLS}} = (X^T X + \lambda I)^{-1} X^T \hat{\beta}_{\text{OLS}}$$

$$\underset{N \text{ large}}{\approx} (N \Sigma + \lambda I)^{-1} N \Sigma \hat{\beta}_{\text{OLS}} \quad \Sigma = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$$

$$= (\Sigma + \frac{\lambda I}{N})^{-1} \Sigma \hat{\beta}_{\text{OLS}}$$

$$= \begin{pmatrix} 1 + \frac{\lambda}{N} & 1 \\ 1 & 1 + \frac{\lambda}{N} \end{pmatrix}^{-1} \begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix} \begin{pmatrix} \hat{\beta}_{\text{OLS}}^{(1)} \\ \hat{\beta}_{\text{OLS}}^{(2)} \end{pmatrix}$$

$$= \frac{1}{(1 + \frac{\lambda}{N})^2 - \rho^2} \begin{pmatrix} 1 + \frac{\lambda}{N} & -\rho \\ -\rho & 1 + \frac{\lambda}{N} \end{pmatrix} \begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix} \begin{pmatrix} \hat{\beta}_{\text{OLS}}^{(1)} \\ \hat{\beta}_{\text{OLS}}^{(2)} \end{pmatrix}$$

$$= \frac{1}{(1 + \frac{\lambda}{N})^2 - \rho^2} \begin{pmatrix} 1 + \frac{\lambda}{N} - \rho^2 & \lambda + \frac{\lambda}{N} \rho - \lambda \\ \frac{\lambda}{N} \rho & 1 + \frac{\lambda}{N} - \rho^2 \end{pmatrix} \begin{pmatrix} \hat{\beta}_{\text{OLS}}^{(1)} \\ \hat{\beta}_{\text{OLS}}^{(2)} \end{pmatrix}$$

$$\hat{\beta}_{\text{ridge}}^{(1)} = \frac{1}{(1 + \frac{\lambda}{N})^2 - \rho^2} \left[\left(1 + \frac{\lambda}{N} - \rho^2 \right) \hat{\beta}_{\text{OLS}}^{(1)} + \frac{\lambda}{N} \rho \hat{\beta}_{\text{OLS}}^{(2)} \right]$$

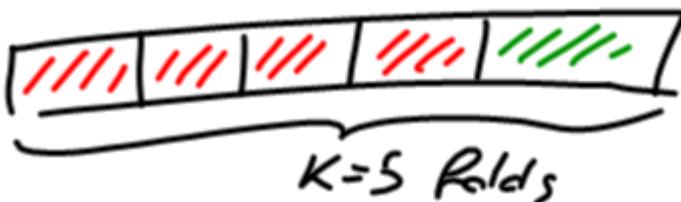
$$\hat{\beta}_{\text{ridge}}^{(2)} = \frac{1}{(1 + \frac{\lambda}{N})^2 - \rho^2} \left[\frac{\lambda}{N} \hat{\beta}_{\text{OLS}}^{(1)} + \left(1 + \frac{\lambda}{N} - \rho^2 \right) \hat{\beta}_{\text{OLS}}^{(2)} \right]$$

c) Lasso part : exploit variable selection property of LASSO (i.e., remove irrelevant predictors)

Ridge part : better handle of correlation structures in X

Exercice 3

a) K-fold cross-validation



training set
validation set

1) for each value of λ , $\kappa = 1$

$$\hat{\beta}_{\text{less}, \kappa}(\lambda) \quad \begin{array}{c} \lambda - K \\ \hline \end{array} \quad \begin{array}{c} \text{training} \\ \text{validation} \end{array}$$

$$\underline{\text{PECV}_n(\lambda)} = L\left(y^{(1)}, \hat{x}^{\lambda - K} \underline{\hat{\beta}_{\text{less}, \kappa}(\lambda)}\right)$$

2) $\kappa = 2$

:

$$\text{final } \text{PECV}_{\text{tot}}(\lambda) = \sum_{\kappa=1}^K \text{PECV}_{\kappa}(\lambda)$$

$$\lambda^* = \underset{\lambda}{\operatorname{arg\,min}} \{ \text{PECV}_{\text{tot}}(\lambda) \}$$

b) 2-fold CV : more bias, less variance

Loocv : less bias, more variance

2 additional aspects : Loocv computationally more intense
Loocv deterministic

Exercise 4

a) Boosting :
 "repeatedly apply a weak estimation to
 modifications of the data to minimize
 a loss function"
-iterative procedure

Bagging : bootstrap samples $x_i^* = (x_{i1}^*, x_{i2}^*, \dots, x_{in}^*)$
 $\hat{\beta}_b$ on x_i^*
 $\hat{\beta}_{\text{BAGGING}} = \frac{1}{B} \sum_{b=1}^B \hat{\beta}_b$

- b) not in the program this year
- c) point a) apply the weak estimator on the reweighted observations
- point d) $\sum_{m=1}^{n_{\text{step}}} \alpha_m \mathbf{1} [y_i \neq G_m(x_i)]$
- point e) $\sum_{m=1}^{n_{\text{step}}} \alpha_m G_m(x)$