Introduction: the hig dinmensional issue

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

Some possible directions:

- 

Introduction: the hig dinmensional 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 constrain 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

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

can be use to predict the outcome y given the values $x_1, x_2, \ldots, x_p$, 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

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

can be use to predict the outcome y given the values $x_1, x_2, \ldots, x_p$, 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

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

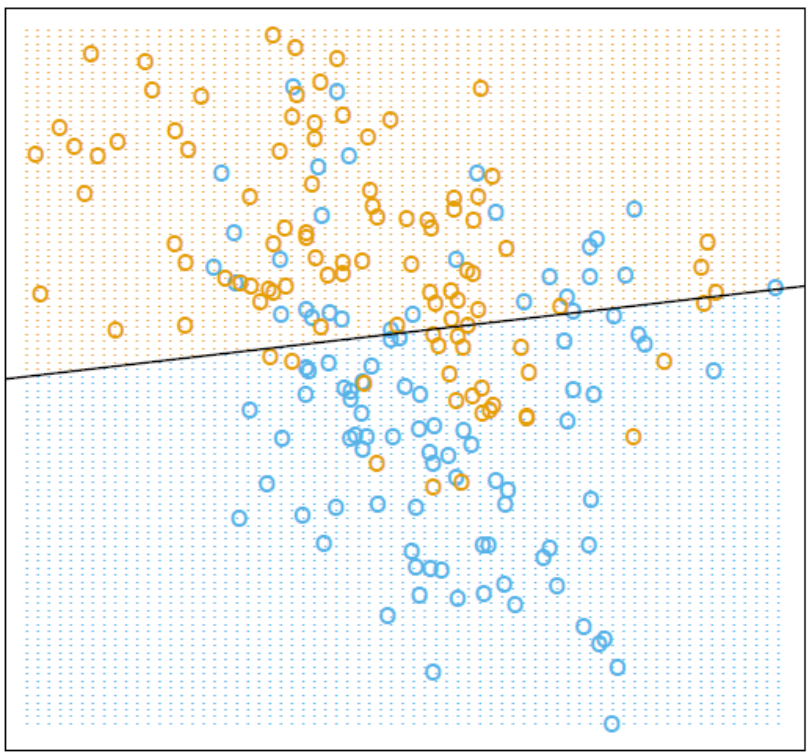
from which, when (X^TX) is non-singular, we obtain

$$\hat{\beta} = (X^TX)^{-1}X^Ty$$

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

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

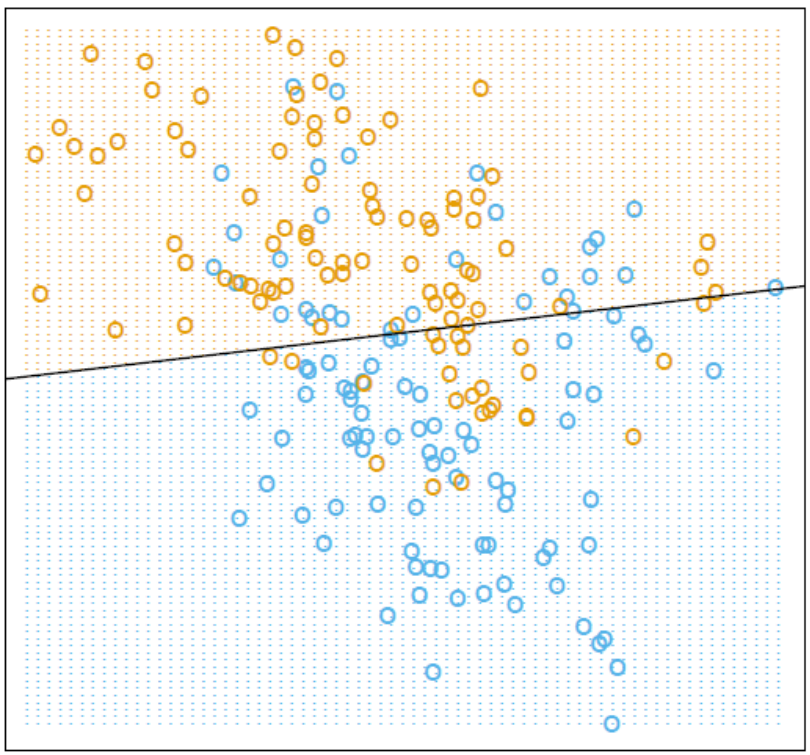
gives linear decision boundary $\{x^T \hat{\beta} = 0.5\}$

- 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

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

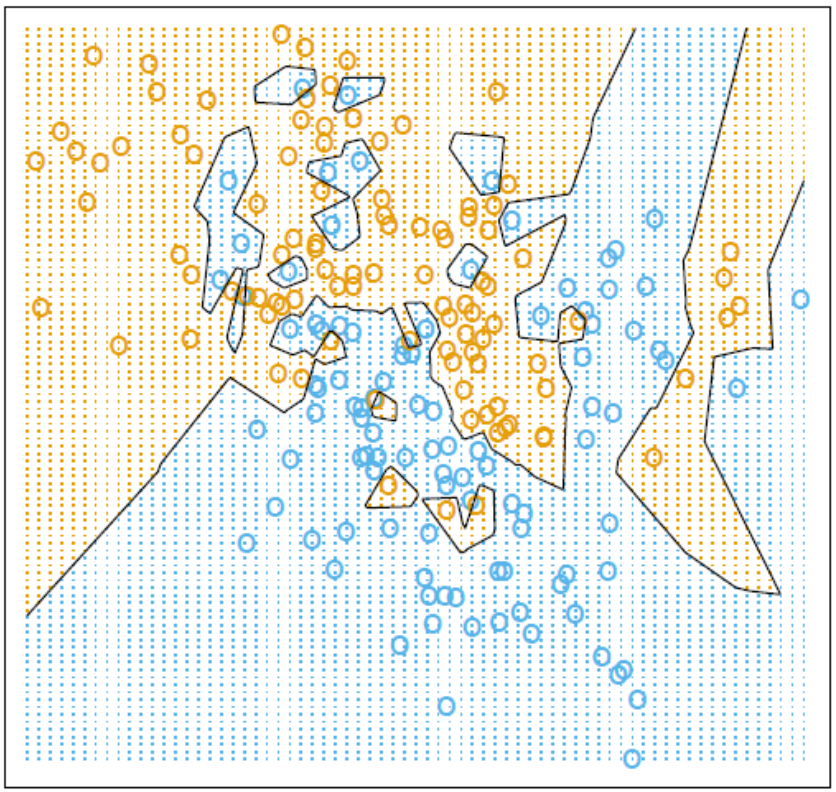
where $N_k(x)$ contains the k closest points to x.

- 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:

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



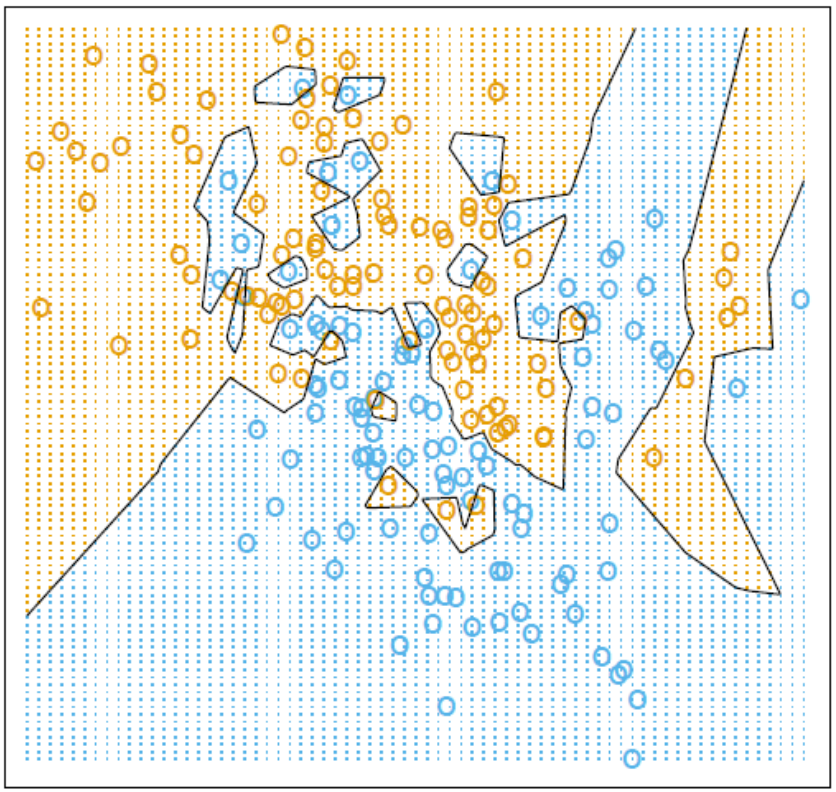
Note:

- 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:

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



Note:

- 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 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
  $f(X)$ whe
  - examp

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 $f(X)$:

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 $f(X)$:

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

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

$$\text{EPE}(f) = [redacted]$$
$$= [redacted]$$

It is th[redacted]

which [redacted]

i.e., th[redacted]

Thus, [redacted]
point $X = x$ 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:

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

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, $f(x) \approx x^T\beta$;
- $\text{argmin}_\beta E[Y - X^T\beta] \rightarrow \beta =$ [redacted]
- replacing the [redacted] leads to $\hat{\beta}$.
- Note:
  - no conditioning on [redacted]
  - we have used our k [redacted] pool over all values
  - 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

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

Linear regression:

- assumes a function linear in its arguments, $f(x) \approx x^T\beta$;
- $\text{argmin}_\beta E[Y - X^T\beta] \to \beta = E[XX^T]^{-1}E[XY]$;
- 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,

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

- there are two approximations:
  - [redacted]
  - [redacted]

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 [redacted]
- for $N \to \infty$, all [redacted]
- for $k \to \infty$, $\hat{f}(x)$ [redacted]
- under mild regularity condition on $p(X, Y)$,

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

- is this an universal solution?
  - [redacted]
  - [redacted]

Statistical decision theory: estimation of optimal f

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

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

- 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]

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

[redacted] estimates than those obtained with the conditional mean;
  - the L_1 loss function has [redacted] $\rightarrow$ 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 $\rightarrow$ numerical difficulties.

Statistical decision theory: other loss functions

What happens with a categorical outcome G?

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

Statistical decision theory: other loss functions

What happens with a categorical outcome G?

- similar concept, different loss function;
- $G \in \mathcal{G} = \{1, \ldots, K\} \rightarrow \hat{G} \in \mathcal{G} = \{1, \ldots, K\}$;
- $L(G, \hat{G}) = L_{G,\hat{G}}$ 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

$$\hat{G} = \text{ar}$$

When using the 0-1

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

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

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

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,

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

When using the 0-1 loss function

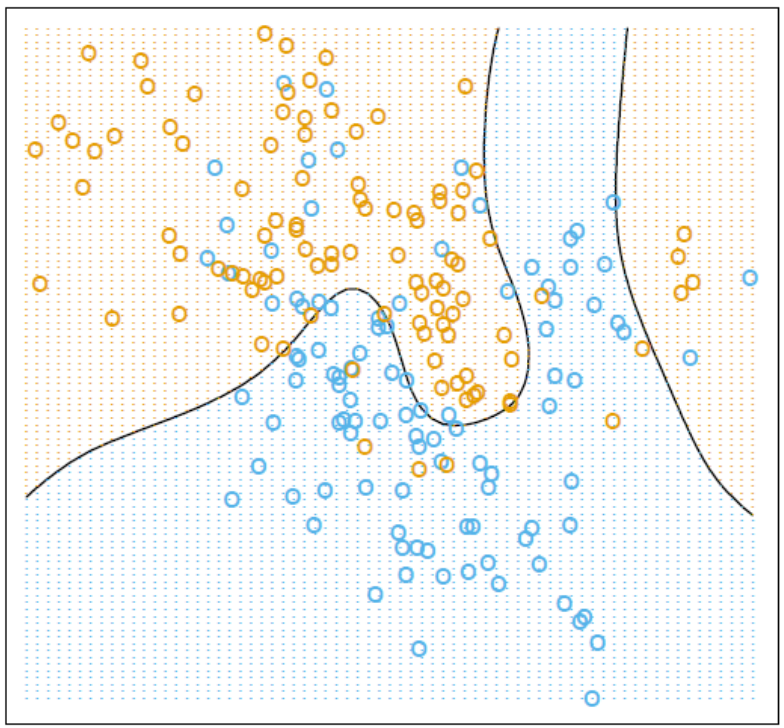
$$\begin{aligned} \hat{G} =& \text{argmin}_{g \in \mathcal{G}} \sum_{k=1}^{K} \{1 - I(G = g_k))\} \Pr(G = g_k | X = x) \\ =& \text{argmin}_{g \in \mathcal{G}} \{1 - \Pr(G = g_k | X = x)\} \\ =& \text{argmax}_{g \in \mathcal{G}} \Pr(G = g_k | X = x) \end{aligned}$$

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 [redacted]



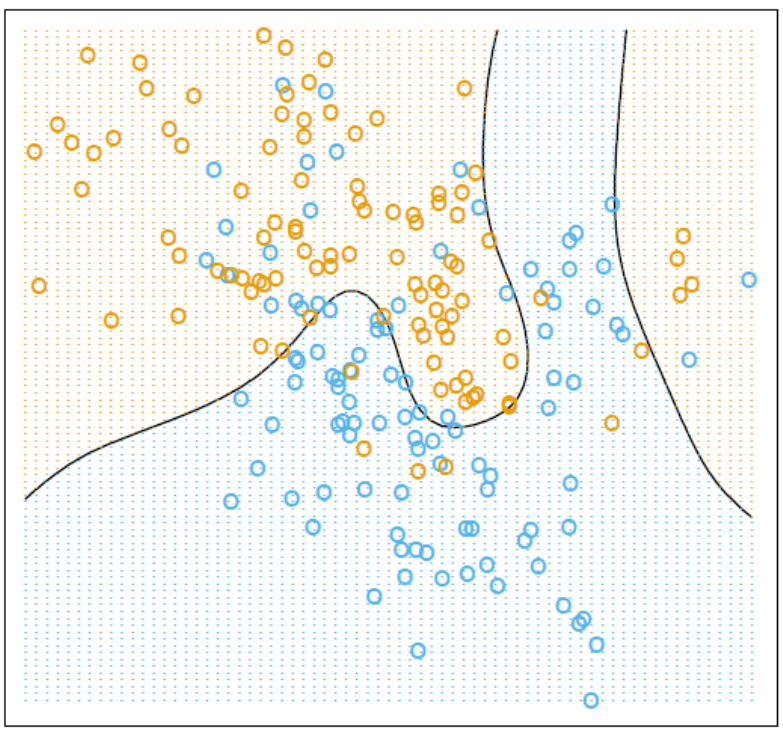
- k nearest neighbor:
  - $\hat{G}(x) =$ [redacted]
  - approxi[redacted]
- regression:
  - [redacted]
  - [redacted]

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:
  - $\hat{G}(x) =$ category with largest frequency in k 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, [redacted]
- k-nearest neighbor, less [redacted] less [redacted]

For large set of training data:

- always possible to use k nearest neighbors?
- Breaks down in [redacted] (Bellman, 1961).

Local methods in high dimensions

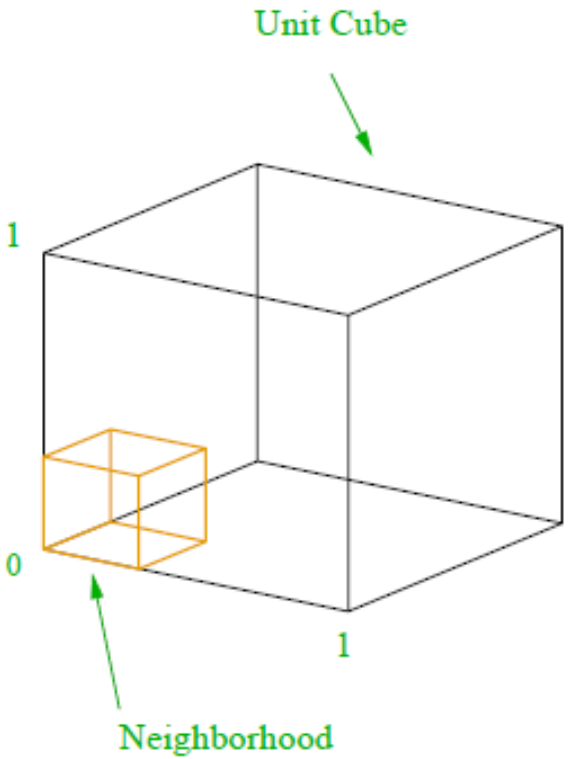
The two (extreme) methods seen so far:

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

For large set of training data:

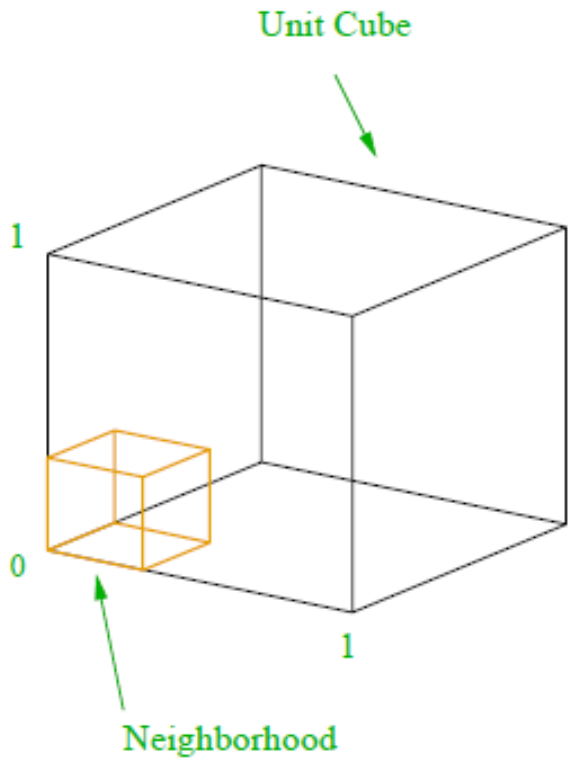
- always possible to use k nearest neighbors?
- Breaks down in high dimensions $\rightarrow$ curse of dimensionality (Bellman, 1961).

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;
- $e_p(r) =$ [redacted]

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;
- $e_p(r) = r^{1/p}$ $\left(e^p = r \Leftrightarrow e = r^{1/p}\right)$;

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

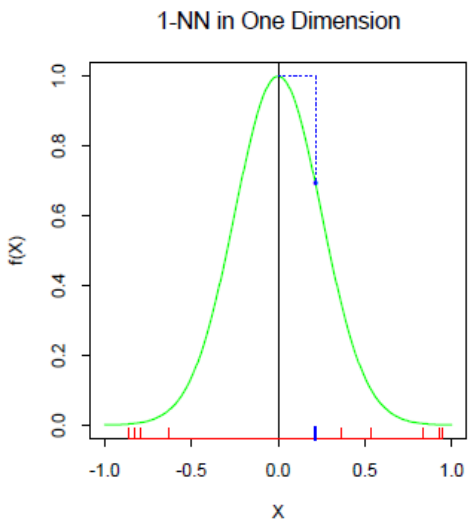
$$\text{MSE}(x_0) =$$

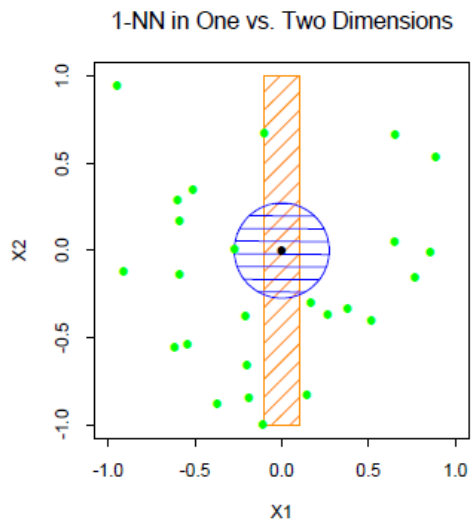
$$=$$

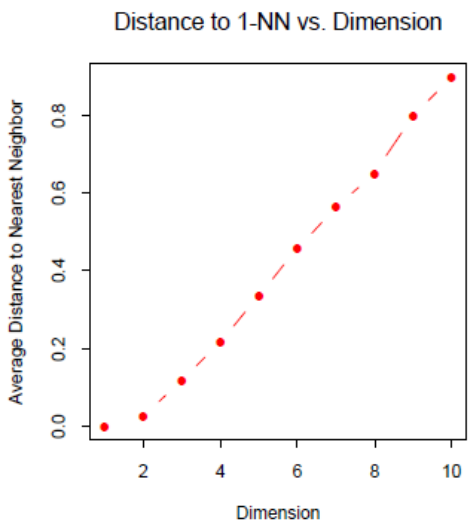
$$=$$

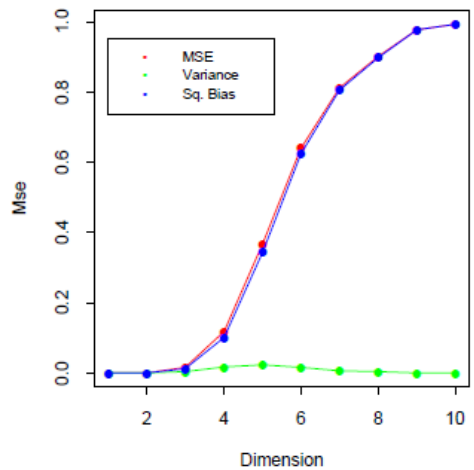
$$=$$

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









Local methods in high dimensions: curse of dimensionality

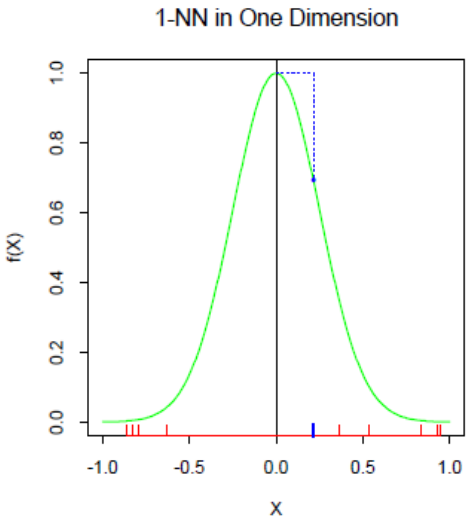
Assume $Y = f(X) = e^{-8||X||^2}$

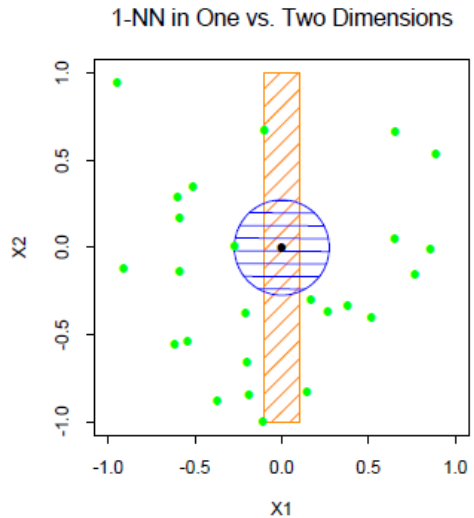
and use the 1-nearest neighbor to predict y_0 at $x_0 = 0$, i.e.

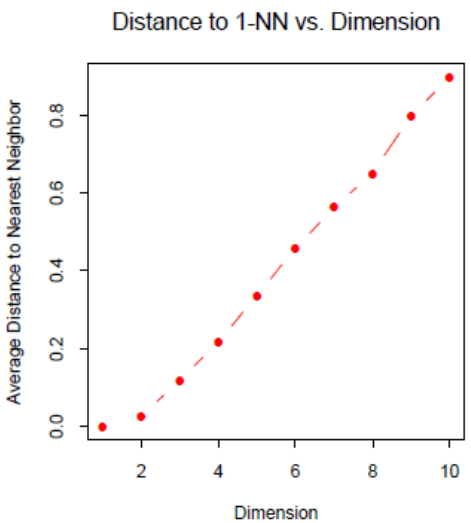
$\hat{y}_0 = y_i$ s.t. x_i nearest observed

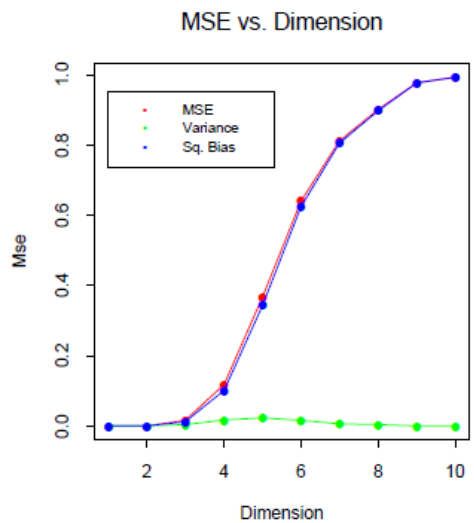
$$
\begin{aligned}
\text{MSE}(x_0) = & \\
&= E_{\mathcal{T}}[\hat{y}_0 - f(x_0)]^2 \\
&= E_{\mathcal{T}}[\hat{y}_0 - E_{\mathcal{T}}(\hat{y}_0)]^2 \\
&\quad + [E_{\mathcal{T}}(\hat{y}_0) - f(x_0)]^2 \\
&= \text{Var}(\hat{y}_0) + \text{Bias}^2(\hat{y}_0)
\end{aligned}
$$

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



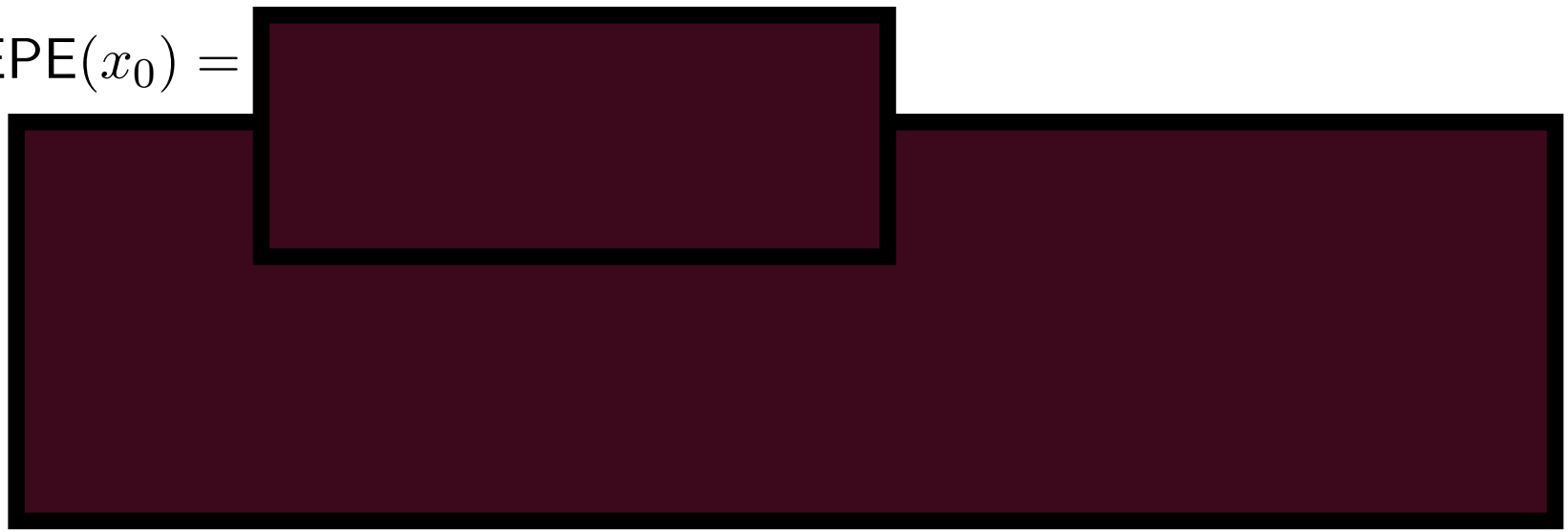






Local methods in high dimensions: EPE in the linear model

- 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 $\hat{\beta} = (X^TX)^{-1}X^Ty$

$\text{EPE}(x_0) =$ 

True and assumed linear model

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

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 $\hat{\beta} = (X^TX)^{-1}X^Ty$

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

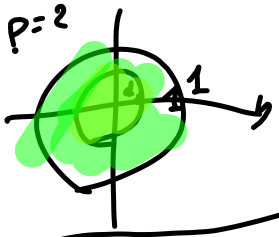
True and assumed linear model

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

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

$t_i :=$ distance between x_i and origin

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



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

$$\text{volume of a ball with radius } r = \frac{\pi^{p/2}}{(p/2)!} r^p$$

$p=2 \quad \pi r^2$

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

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

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

$$\Pr[\forall i, t_i \geq d] = 1/2$$

$$(1 - d^p)^N = 1/2$$

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

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

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

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

K-nearest neighbour

$$E[Y|X=x]$$

average → neighbour

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

Statistics vs Machine Learning

Statistical approach:

Starting from the model

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

additive model → approximation of the truth

statistics $\hat{f}(x)$ approx $f(x)$

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

BUT

we add an error term which captures:
- measurements errors;
- effects of non-measured variables;
- ...

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

most natural approach, least square

Machine learning approach

Assume $Y = f(x) + \varepsilon$

→ start from $\hat{f}(x)$, possibly simple $\hat{f}(x) = c$ (inizialization)

→ evaluate $\hat{f}(x)$ on our training set

e.g. $\sum_{i=1}^{N} (y_i - \hat{f}^{[k]}(x_i))^2$

→ modify $\hat{f}(x)$ to improve the prediction

$$\sum_{i=1}^{N} (y_i - \hat{f}^{[k+1]}(x_i))^2 < \sum_{i=1}^{N} (y_i - \hat{f}^{[k]}(x_i))^2$$

k is the step of the algorithm

→ use training set (x_i, y_i)

→ learning by examples

→ focus is on the learner, that does not aim to approximate a true $f(x)$

↓

statistical / mathematical approach

(x_i, y_i) points of a p+1 dimensional space

$$f(x): \underset{\text{in } \mathbb{R}^p}{X} \to \underset{\text{in } \mathbb{R}}{Y}$$

↳ goal: giving an approximation of $f(x)$ working at points in X given T

Parametric approach

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

$$Y = f(x; \theta) + \varepsilon \qquad f_\theta(x) = f(x; \theta)$$

e.g.:

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

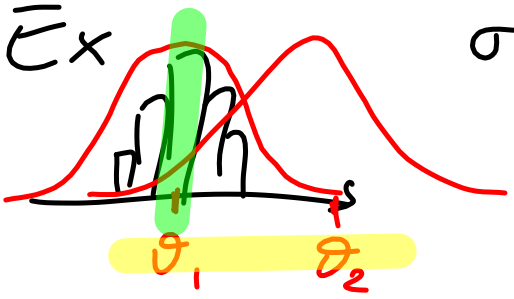
$$h_1(x) = x_1$$
$$h_2(x) = x_1^2$$

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

a function of θ

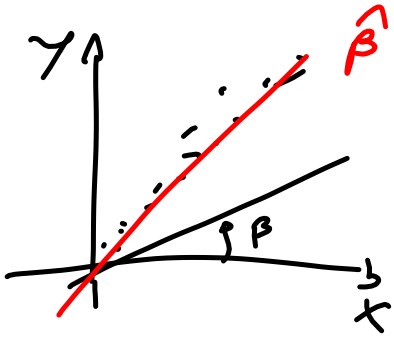
$$\hat{\theta} = \arg\min_{\theta} RSS(\theta)$$

Ex



$\sigma^2 = 1$

$\hat{\theta} = \theta_1$



simplest cases
- least squares

more complicated frameworks
- maximum likelihood estimation

Likelihood estimation

$Y_i \sim P$ $\quad p(y)$ is indexed by θ

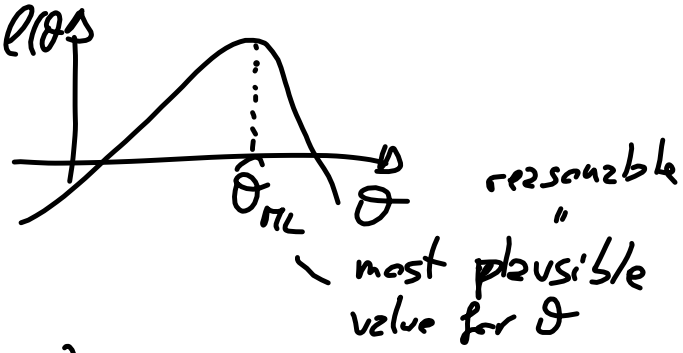
Y_i i.i.d. $\quad p(y;\theta)$

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

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

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

$$\hat{\theta}_{ML} = \arg\max_{\theta} \ell(\theta)$$



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

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

Restricted estimators

instead of estimating θ as

$$\hat{\theta} = \arg\min_{\theta} RSS(\theta)$$

we look for

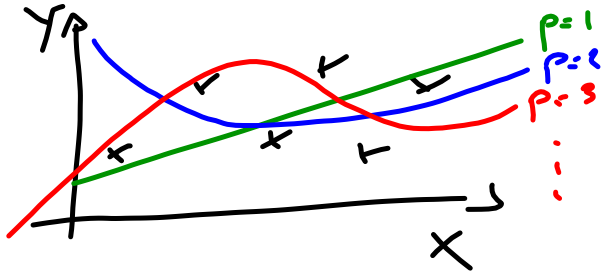
$$\hat{\theta} = \arg\min_{\theta} PRSS(\theta)$$

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

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

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

Model selection and variance - bias trade - off



$$f(x,\theta) = \sum_{j=1}^{p} \theta_j x^j$$

given enough parameters θ_j, we will always be able to find a line which passes through all points (no bias)

penalization for the number of parameters

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

Bias - variance trade off

y $f(x_0)$

$Y = f(x) + \epsilon$

$$\text{error} = \sigma^2 + \underbrace{\text{variance} + \text{bias}^2}_{MSE}$$

see figure 2.11

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

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

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

matrix form $(y - x\beta)^T(y - x\beta)$

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

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

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

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

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

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

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

H: hat matrix, projection matrix

Properties

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

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

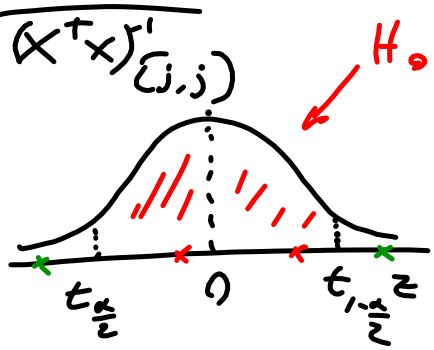
Focus on $\varepsilon \sim N(0, \sigma^2)$ ε: iid

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

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

$H_0 : \beta_j = 0 \rightarrow z_j = \dfrac{\hat{\beta}_j - 0}{sd(\hat{\beta}_j)}$

$= \dfrac{\hat{\beta}_j}{\hat{\sigma}\sqrt{(X^TX)^{-1}_{(j,j)}}}$



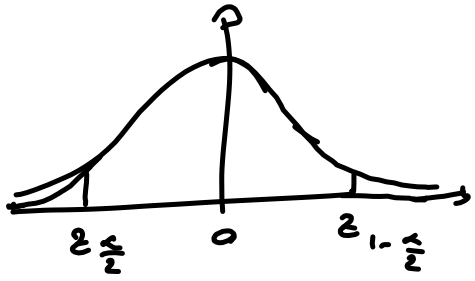
under H_0 $z_j \overset{H_0}{\sim} t_{N-p-1}$

z_{obs}

H_1 we reject H_0

when σ^2 is known

under H_0 $z_j \overset{H_0}{\sim} N(0;1)$



$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \varepsilon$

is X_1 useful to predict Y?

$\beta_j = 0 \longrightarrow$ reject H_0 Yes / do NOT reject H_0 not X_j

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

$x_i \sim h(x)$

$y_i = f(x_i) + \varepsilon_i$

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

Estimator for f linear in the y_i:

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

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

a) $\hat{f}_{LR}(x_0) = x_0^T \hat{\beta} = x_0^T (X^T X)^{-1} X^T y_i$

$\hat{\beta}$

$\Rightarrow \ell_i(x_0, \mathcal{X}) = [x_0^T (X^T X)^{-1} X^T]_{[i]}$

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

where $N_K(x_0)$ is the set of the K nearest neighbors

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

b) Decompose the conditional mean-squared error

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

stochastic | fixed

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

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

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

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

$= 0$

$$= \text{bias}^2 + \text{variance}$$

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:

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

$$\left(f(x_0) - E_{Y|X}\left[\sum_{i=1}^{n} \ell_i(x_0; X)\, y_i\right]\right)^2 + \mathrm{Var}_{Y|X}\left[\sum_{i=1}^{n} \ell_i(x_0; X)\, y_i\right]$$

$$\left(f(x_0) - \sum_{i=1}^{n} \ell_i(x_0; X)\, f(x_0)\right)^2 + \sum_{i=1}^{n} \ell_i(x_0; X)^2\, \sigma^2$$

unconditional

$$E_X\left[\left(f(x_0) - E_{Y|X}\left[\sum \ell_i(x_0; X)\, y_i\right]\right)^2\right] + E_X\left[\mathrm{Var}_{Y|X}\left[\sum_{i=1}^{n} \ell_i(x_0; X)\, y_i\right]\right]$$

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 $F \approx z^2$

$\rightarrow p_1 = p$, $p_0 = p - 1$

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

$$z = \frac{\hat{\beta}_j}{\hat{\sigma}\sqrt{[X^TX]_{(j,j)}}} \sim t_{N-p-1}$$

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

Gauss - Markov theorem

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

Best ← smallest variance
Linear ← $\hat{\vartheta} = a^T\hat{\beta}$ <-> $\vartheta = a^T\beta$
Unbiased ← $E[\hat{\theta}] = \vartheta$
Estimator

consider X fixed

$$E_{Y|X}[a^T\hat{\beta}] = E_{Y|X}[a^T(X^TX)^{-1}X^Ty]$$
$$= a^T(X^TX)^{-1}X^TX\beta$$
$$= a^T\beta$$

(where $(X^TX)^{-1}X^TX = I$)

Th: $Var(a^T\hat{\beta}) \leq Var(c^T\hat{\beta})$ → ex 3.3 (a)

$$MSE(\tilde{\vartheta}) = E[(\tilde{\theta} - \theta)^2]$$
$$= E[(\tilde{\theta} - E[\tilde{\theta}] + E[\tilde{\theta}] - \theta)^2]$$
$$= Var(\tilde{\theta}) + (E[\tilde{\theta}] - \vartheta)^2$$

smallest with LS

$= 0$

- $MSE(\hat{\theta})$ is strongly related with the prediction error

Let consider $Y_0 = f(x_0) + \varepsilon \qquad \varepsilon \sim N(0, \sigma^2)$

$\varepsilon \perp x$

$$E\left[(Y_0 - \hat{f}(x_0))^2\right] = E\left[Y_0^2 + \hat{f}(x_0)^2 - 2 Y_0 \hat{f}(x_0)\right]$$

$$= E[Y_0^2] + E\left[\hat{f}(x_0)^2\right] - 2E\left[Y_0 \hat{f}(x_0)\right]$$

$$= Var(Y_0) + \cancel{E[Y_0]^2} + Var(\hat{f}(x_0)) + E[\hat{f}(x_0)]^2 \quad (Y_0 = f(x_0))$$

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

$$= \sigma^2 + Var(\hat{f}(x_0)) + \underbrace{E[\hat{f}(x_0)]^2 - 2 f(x_0) E[\hat{f}(x_0)] + f(x_0)^2}_{(E[\hat{f}(x_0)] - Y_0)^2}$$

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

MSE

$$E[Y_0] = E[f(x_0) + \varepsilon] = E[f(x_0)] + E[\varepsilon]$$

$$= f(x_0) + 0$$

$$E[Y_0^2] = E[(f(x_0) + \varepsilon)^2]$$

$$= Var(Y_0) + E[Y_0]^2$$

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

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 $\hat{\beta}_j$:

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

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

→ the estimates may be unstable due to high variance

↓

model selection

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

→ smaller

→ larger

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

$\binom{3}{2} = \frac{3 \cdot 2}{2} = 3$

→ $\binom{8}{2} = \frac{8 \cdot 7}{2} = 28$ 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: $y = \beta_0 + \beta_2 x_2 + \beta_3 x_3 + \varepsilon$

β_3 together with β_0 and β_2

- backward elimination

p>>n case backward elimination is not possible

start : full model

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

1st step $y = \beta_0 + \beta_2 x_2 + \beta_3 x_3 + \varepsilon$

⋮

- stepwise selection
  stepback "

$RSS(\theta) + J(\theta)$
$+ \underline{\underline{\ell(p)}}$

Stagewise regression

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

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

stepwise: the estimate of β take into account all x

stagewise: introduce a new β, its estimate in only based on x_j

Ex 3.3 (a)

$\hat{\theta}_{LS} = a^T\hat{\beta} = a^T(X^TX)^{-1}X^Ty \qquad E[a^T\hat{\beta}] = a^T\beta$

$\tilde{\theta} = c^Ty$ unbiased $\qquad c^T = a^T(X^TX)^{-1}X^T + \delta^T$

$$\begin{aligned} E[\tilde{\theta}] &= E[c^Ty] \\ &= E[a^T(X^TX)^{-1}X^Ty + \delta^Ty] \\ &= a^T(X^TX)^{-1}X^TX\beta + \delta^TX\beta \\ &= a^T\beta + \delta^TX\beta \end{aligned} \quad \Rightarrow \delta^TX = 0$$

$$\begin{aligned} Var(\tilde{\theta}) &= Var(c^Ty) \\ &= c^T\sigma^2c \\ &= \sigma^2\left(a^T(X^TX)^{-1}X^T + \delta^T\right)\left(a^T(X^TX)^{-1}X^T + \delta^T\right)^T \\ &= \sigma^2\left(a^T(X^TX)^{-1}X^T + \delta^T\right)\left(X(X^TX)^{-1}a + \delta\right) \\ &= \sigma^2\Big(a^T(X^TX)^{-1}X^TX(X^TX)^{-1}a + a^T(X^TX)^{-1}X^T\delta + \\ &\quad + \delta^TX(X^TX)^{-1}a + \delta^T\delta\Big) \\ &= \sigma^2a^T(X^TX)^{-1}a + \sigma^2\delta^T\delta \end{aligned}$$

($a^T(X^TX)^{-1}X^T\delta = 0$, $\delta^TX(X^TX)^{-1}a = 0$)

$\sigma^2a^T(X^TX)^{-1}a = Var(\hat{\theta}_{LS})$ $\qquad \sigma^2\delta^T\delta \geq 0$

Variable selection

IDEA: remove irrelevant variables from the model
↑
not useful to predict/explain the response

- less variance (despite small increase of bias)
- 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 $\hat{\theta}_{IC} = \arg\min_\theta \{L(\theta) + \lambda J(\theta)\}$

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

$-2\ell(\theta)$ $\qquad$ $\sum_{j=1}^{p} \mathbb{1}[\theta_j \neq 0]$

GOAL: penalize larger models

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

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

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

n is the sample size

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

if two explenatory variables are strongly correlated → collinearity

extreme case: variables linearly dependent → super-collinearity

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

Hoerl & Kennard (1970) $X^TX \to X^TX + \lambda I_p$

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

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

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

RIDGE REGRESSION AS A SHRINKAGE METHOD

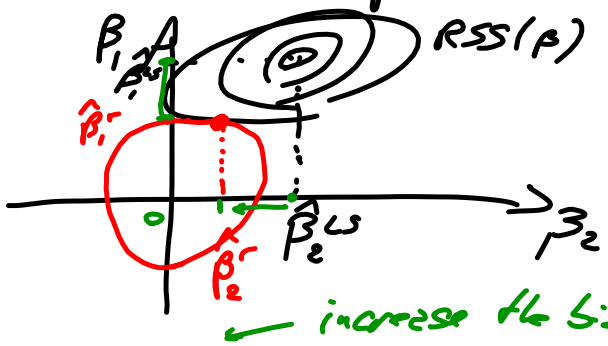
- $$\hat{\beta}^{(\lambda)}_{ridge} = \underset{\beta}{\operatorname{argmin}} \left\{ \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j\right)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \right\}$$

alternative formulation

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

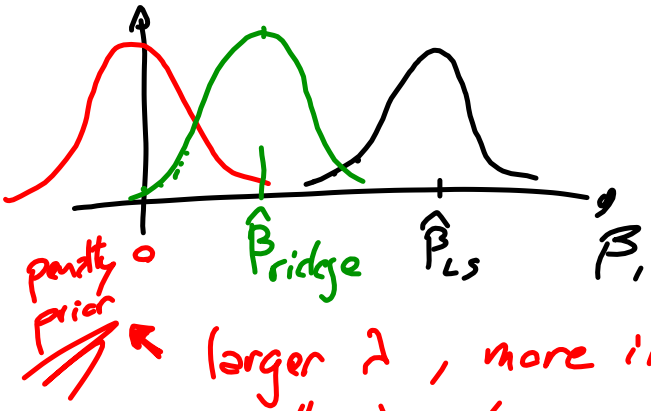
subject to $\sum_{j=1}^{p} \beta_j^2 \le t$

- one to one correspondence between λ and t



← increase the bias, reduce the variance

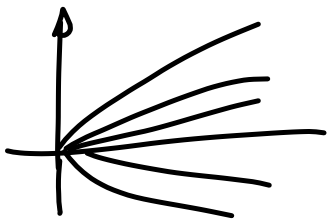
- from a Bayesian point of view, ridge estimator is the posterior mean/mode



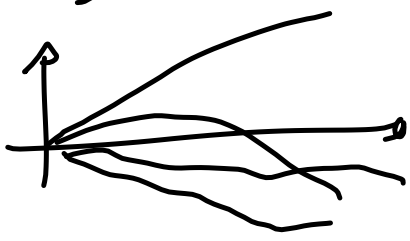
larger λ, more influence for the prior
smaller λ, less " " " "

---

if Xs are uncorrelated



if Xs are correlated



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

$$E[X_j] = 0$$
$$Var[X_j] = 1$$

Expected value of ridge estimator

$$E\left[\hat{\beta}_{ridge}^{(\lambda)}\right] = E\left[(X^TX + \lambda I_p)^{-1}X^Ty\right]$$
$$= E\left[(I_p + \lambda(X^TX)^{-1})^{-1}(X^TX)^{-1}X^Ty\right] \quad (\hat{\beta}_{LS})$$
$$= \underbrace{(I_p + \lambda(X^TX)^{-1})^{-1}}_{W_\lambda} E[\hat{\beta}_{LS}]$$
$$= W_\lambda \beta \qquad E\left[\hat{\beta}_{ridge}^{(\lambda)}\right] \neq \beta$$

$\lambda \to 0 \quad E\left[\hat{\beta}_{ridge}^{(\lambda)}\right] = \beta$

$\lambda \to \infty \quad E\left[\hat{\beta}_{ridge}^{(\lambda)}\right] = 0_{p\times 1}$ ← without intercept $\beta_0 = 0$

important $\lambda_a > \lambda_s \not\Rightarrow |\hat{\beta}_j(\lambda_a)| < |\hat{\beta}_j(\lambda_s)|$ due to correlation

Variance of the ridge estimator

$$\begin{aligned} Var\left[\hat{\beta}_{ridge}^{(\lambda)}\right] &= Var\left[W_\lambda \hat{\beta}_{LS}\right] \\ &= W_\lambda Var\left(\hat{\beta}_{LS}\right) W_\lambda^T \\ &= \sigma^2 W_\lambda (X^TX)^{-1} W_\lambda^T \end{aligned}$$

$$Var\left(\hat{\beta}_{LS}\right) - Var\left(\hat{\beta}_{ridge}^{(\lambda)}\right) = \sigma^2\left[(X^TX)^{-1} - W_\lambda (X^TX)^{-1} W_\lambda^T\right]$$

$$= \sigma^2 W_\lambda \left[(I + \lambda(X^TX)^{-1})(X^TX)^{-1}(I + \lambda(X^TX)^{-1}) - (X^TX)^{-1}\right] W_\lambda^T$$

$$= \sigma^2 W_\lambda \left[(X^TX)^{-1} + \lambda(X^TX)^{-2} + \lambda(X^TX)^{-2} + \lambda^2(X^TX)^{-3} - (X^TX)^{-1}\right] W_\lambda^T$$

$$= \sigma^2 W_\lambda \left[2\lambda(X^TX)^{-2} + \lambda^2(X^TX)^{-3}\right] W_\lambda^T > 0$$

$$\Rightarrow Var\left(\hat{\beta}_{ridge}^{(\lambda)}\right) \preccurlyeq Var\left(\hat{\beta}_{LS}\right)$$

Degrees of freedom

$\hat{Y}_{ls} = \underbrace{X(X^TX)^{-1}X^T}_{H} y \qquad df = tr(H) = p$

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

$tr(H_\lambda) = \sum_{j=1}^{p} \frac{d_j^2}{d_j^2 + \lambda} \qquad \lambda = 0 \to df = p \qquad \lambda \to \infty \to df \to 0$

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

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

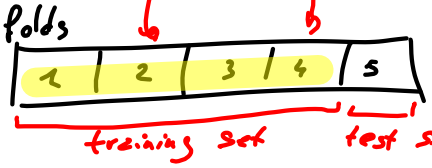
ridge : $-2\ell(\beta) + 2\sum \frac{d_j^2}{d_j^2 + \lambda}$

AIC(λ) we can find $\hat{\lambda}$ as $\hat{\lambda} = \arg\min_\lambda AIC(\lambda)$

usually λ is computed by cross-validation

sample: $x_1, x_2, \ldots, x_n$

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



$err = \sum_{i=1}^{n_5} \left(y_{i(5)} - x_{i(5)} \hat{\beta}^{(\lambda)}_{(-5)} \right)^2$

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

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

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

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

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

$\vdots$

err_5

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

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 single value of X.

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

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

$$\begin{aligned}\hat{\beta}_{LS} &= (X^TX)^{-1}X^Ty \\ &= (VDU^TUDV^T)^{-1}VDU^Ty \\ &= (VD^2V^T)^{-1}VDU^Ty \\ &= VD^{-2}V^TVDU^Ty \\ &= VD^{-2}DU^Ty\end{aligned}$$

$$\begin{aligned}\hat{y}_{LS} &= X\hat{\beta}_{LS} \\ &= UDV^TVD^{-2}DU^Ty \\ &= UD^2D^{-2}U^Ty \\ &= UU^Ty\end{aligned}$$

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

$$\begin{aligned}\hat{\beta}_{ridge}^{(\lambda)} &= (X^TX+\lambda I)^{-1}X^Ty \\ &= (VDU^TUDV^T+\lambda I_p)^{-1}VDU^Ty \\ &= (VD^2V^T+\lambda VV^T)VDU^Ty \\ &= V(D^2+\lambda I)^{-1}V^TVDU^Ty \\ &= V(D^2+\lambda I_p)^{-1}DU^Ty\end{aligned}$$

$$\begin{aligned}\hat{y}_{ridge} &= X\hat{\beta}_{ridge} \\ &= UDV^TV(D^2+\lambda I_p)^{-1}DU^Ty \\ &= UD^2(D^2+\lambda I_p)^{-1}U^Ty\end{aligned}$$

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

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

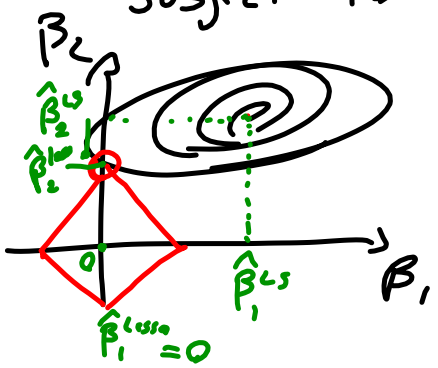
LASSO

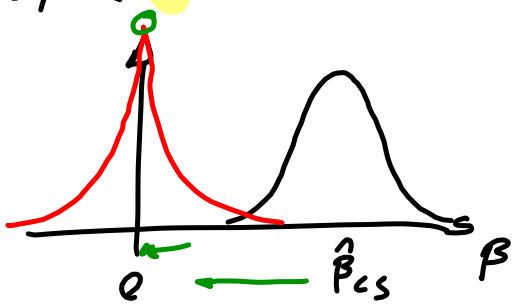
$$\hat{\beta}_{LASSO} = \underset{\beta}{\operatorname{argmin}} \left\{ \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij} \beta_j \right)^2 + \lambda \sum_{j=1}^{p} |\beta_j| \right\}$$

or, equivalently

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

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





$$\hat{\beta}_{pen} = \underset{\beta}{\operatorname{argmin}} \left\{ \sum_{i} \left(y_i - \beta_0 - \sum_{j} x_{ij} \beta_j \right)^2 + \lambda \sum_{j} |\beta|^q \right\}$$

$q = 1 \rightarrow$ LASSO

$q = 2 \rightarrow$ RIDGE REGRESSION

$1 < q < 2 \approx$ average between lasso and ridge penalties

ELASTIC NET

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

STK4030 - Statistical Learning: Advanced Regression and Classification

Riccardo De Bin

debin@math.uio.no

Procedure	Logic
BE-only	Estimate full model on $x_1, \ldots, x_k$. Repeat: while the least significant term has $P \geq \alpha_2$, remove it and re-estimate the model
BE	Estimate full model on $x_1, \ldots, x_k$. If the least significant term has $P \geq \alpha_2$, remove it and re-estimate; otherwise stop. Again: if the least significant term has $P \geq \alpha_2$, remove it and re-estimate; otherwise stop Repeat • if most significant excluded term has $P < \alpha_1$, add it and re-estimate; • if least significant included term has $P \geq \alpha_2$, remove it and re-estimate; until neither action is possible.
FS-only	Estimate null model. Repeat: while the most significant excluded term has $P < \alpha_1$, add it and re-estimate.
FS	Estimate null model. If the most significant excluded term has $P < \alpha_1$, add it and re-estimate; otherwise stop. Again: if the most significant excluded term has $P < \alpha_1$, add it and re-estimate; otherwise stop. Repeat • if least significant included term has $P \geq \alpha_2$, remove it and re-estimate; • if most significant included term has $P < \alpha_1$, add it and re-estimate; until neither action is possible.

from: Royston & Sauerbrei (2008)

Table 2.2 Myeloma study (65 patients, 48 events). Results of applying variable selection strategies.[a]

Variable	Nominal significance level, α						All-subsets
	0.05			0.157			
	Full	BE	FS	Full	BE	FS	AIC
x_1	*	✓	✓	*	✓	✓	✓
x_2			✓			✓	
x_3	*	✓		*	✓		✓
x_4		✓		*	✓		✓
x_5							
x_6		✓		*	✓		✓
x_7	*	✓		*	✓		✓
x_8				*	✓		✓
x_9							
x_{10}							
x_{11}							
x_{12}	*	✓		*	✓		✓
x_{13}	*	✓		*	✓		✓
x_{14}				*			
x_{15}							
x_{16}							

[a] BE(0.01) and FS(0.01) selected only one variable, x_1; * denotes that a variable is significant at the relevant α-level in the full model.

from: Royston & Sauerbrei (2008)

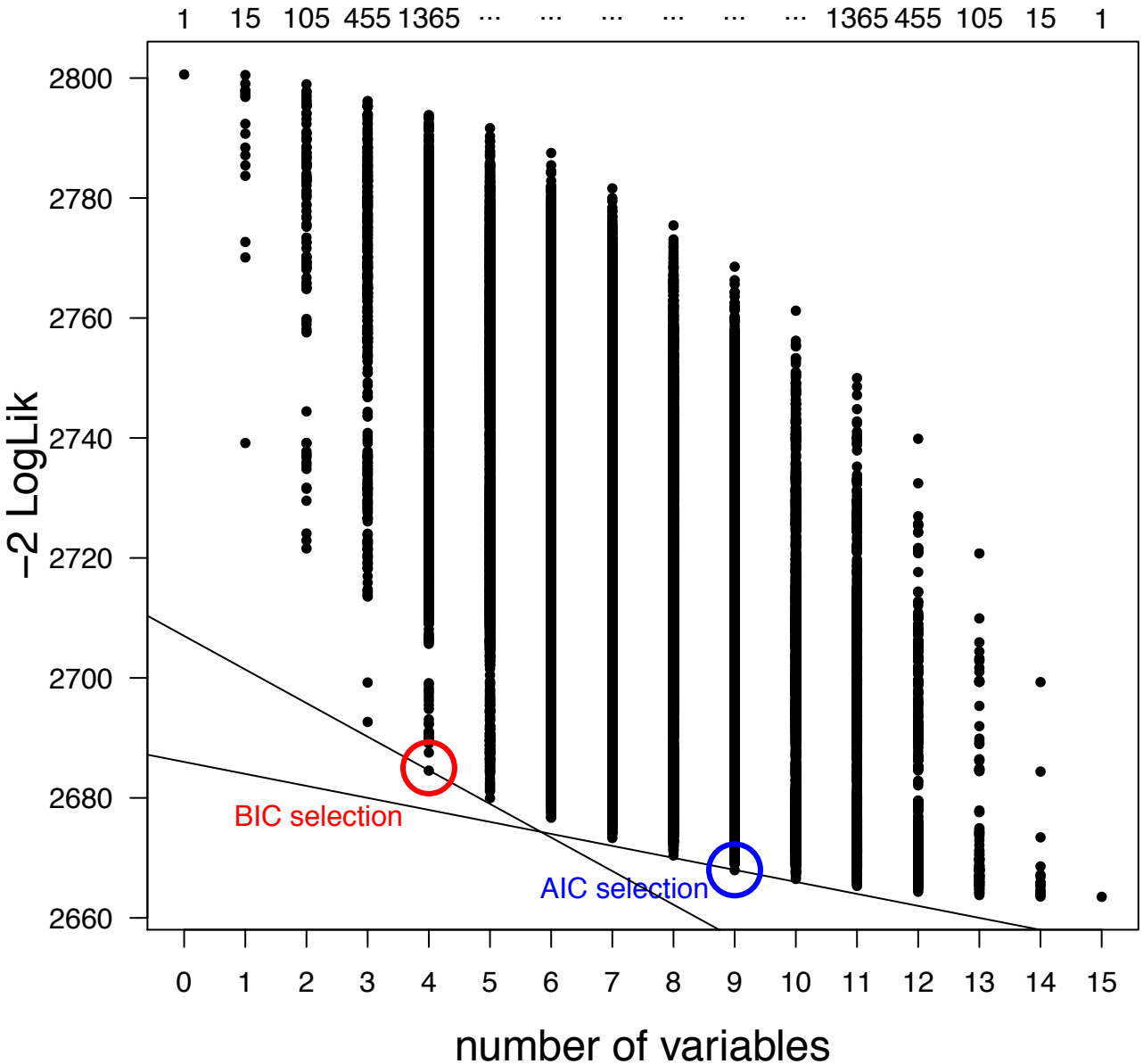

1 15 105 455 1365 ... ... 1365 455 105 15 1
2800
2780
2760
2740
2720
2700
2680
2660
−2 LogLik
BIC selection
AIC selection
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
number of variables

Table 2.3 Educational body-fat data. Full model and that selected by BE(0.05). Final three columns give details of the full model excluding x_6.

Variable	Full model			BE(0.05)			Full model excl. x_6		
	$\widehat{\beta}$	SE	$\widehat{\beta}$/SE	$\widehat{\beta}$	SE	$\widehat{\beta}$/SE	$\widehat{\beta}$	SE	$\widehat{\beta}$/SE
x_1	0.074	0.032	2.31	0.056	0.024	2.35	0.211	0.034	6.20
x_2	−0.019	0.067	−0.28				0.227	0.074	3.08
x_3	−0.249	0.191	−1.30	−0.322	0.121	−2.65	−0.915	0.212	−4.32
x_4	−0.394	0.234	−1.68				−0.378	0.278	−1.36
x_5	−0.119	0.108	−1.10				0.150	0.124	1.21
x_6	0.901	0.091	9.90	0.774	0.033	23.26	–	–	–
x_7	−0.146	0.144	−1.02				0.163	0.166	0.98
x_8	0.178	0.146	1.22				0.231	0.173	1.33
x_9	−0.041	0.245	−0.17				−0.095	0.291	−0.33
x_{10}	0.185	0.220	0.85				−0.053	0.259	−0.21
x_{11}	0.178	0.170	1.04				−0.066	0.200	−0.33
x_{12}	0.277	0.207	1.34				0.058	0.244	0.24
x_{13}	−1.830	0.529	−3.46	−1.943	0.406	−4.78	−2.692	0.620	−4.34

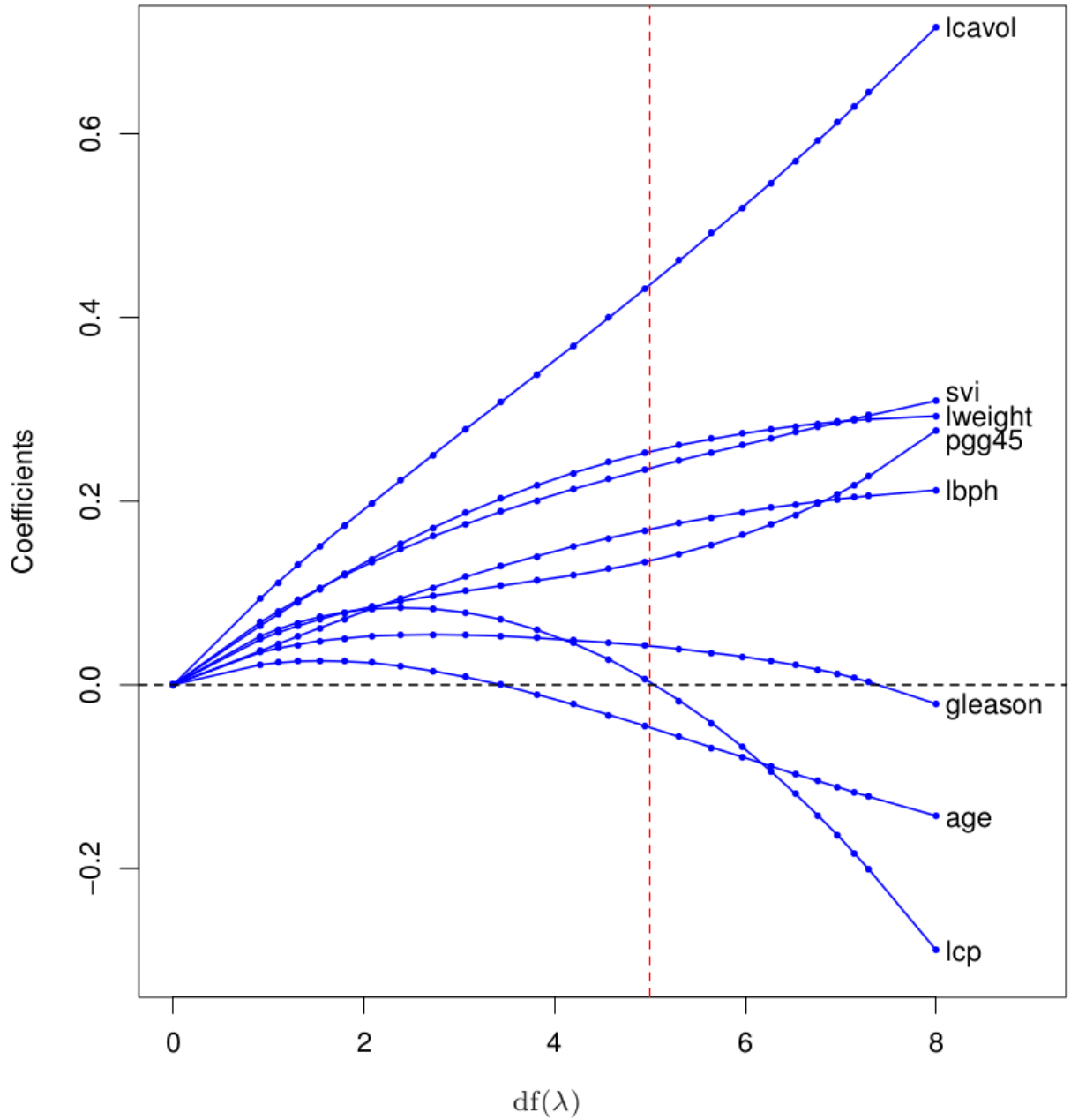
from: Royston & Sauerbrei (2008)

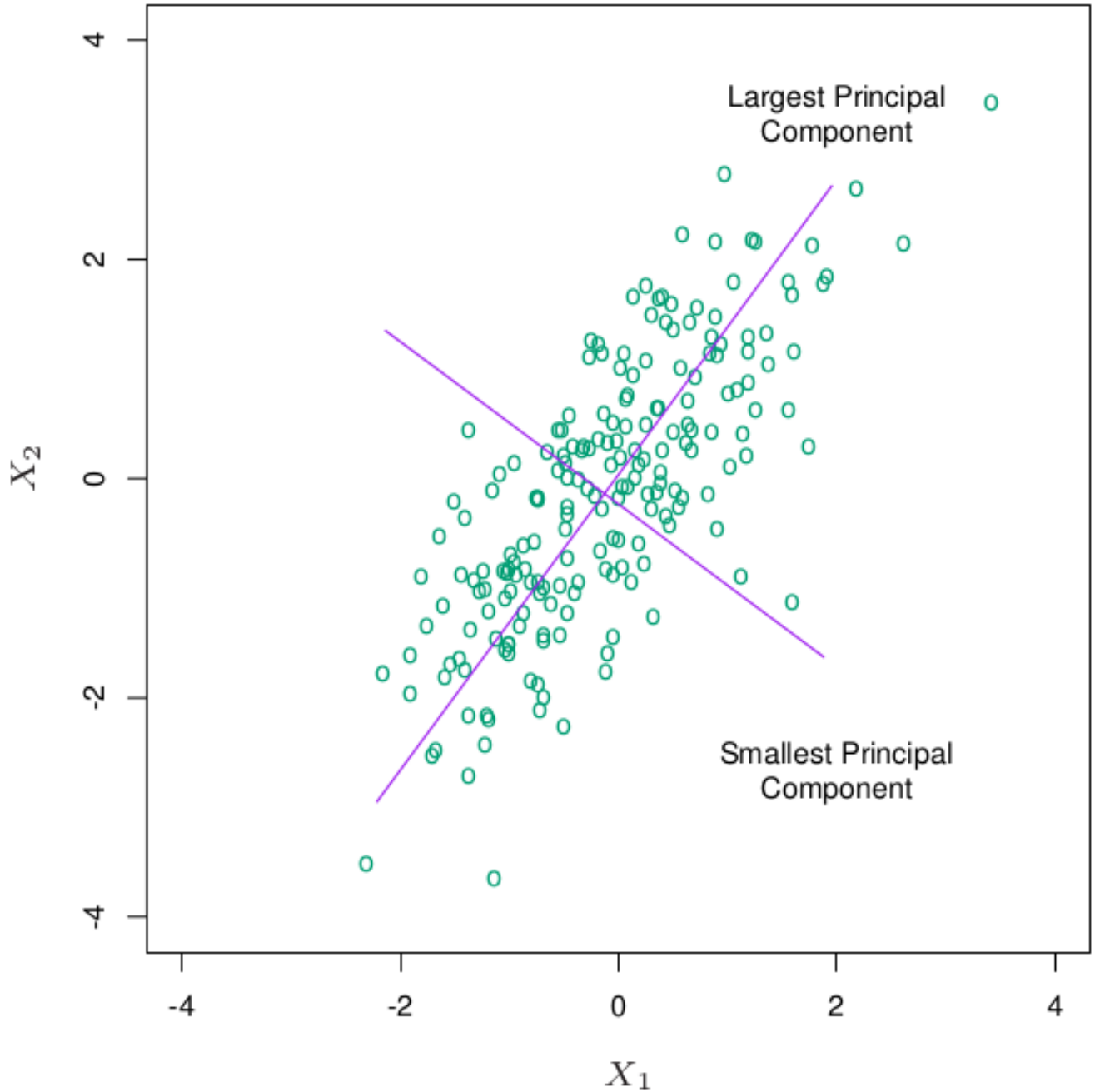
```r
library(Biobase)
library(breastCancerVDX)

# ids of genes FLOT1
idFLOT1 <- which(fData(vdx)[,5] == 10211)

# ids of ERBB2
idERBB2 <- which(fData(vdx)[,5] == 2064)

# get expression levels of probes mapping to FLOT genes
X <- t(exprs(vdx)[idFLOT1,])
X <- sweep(X, 2, colMeans(X))

# get expression levels of probes mapping to FLOT genes
Y <- t(exprs(vdx)[idERBB2,])
Y <- sweep(Y, 2, colMeans(Y))

# regression analysis
summary(lm(formula = Y[,1] ~ X[,1] + X[,2] + X[,3] + X[,4]))
```

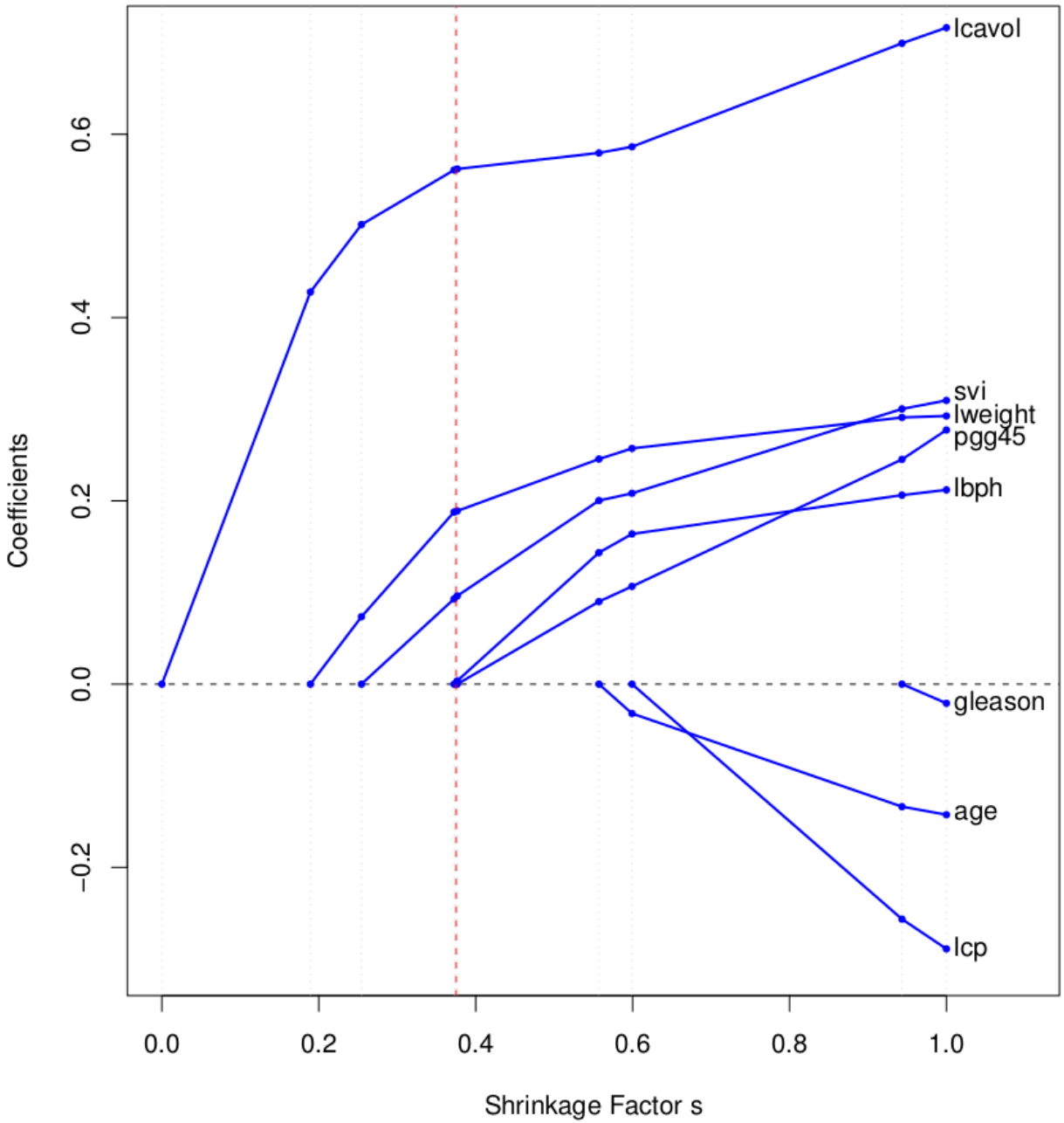
from: Van der Wieringen (2015)

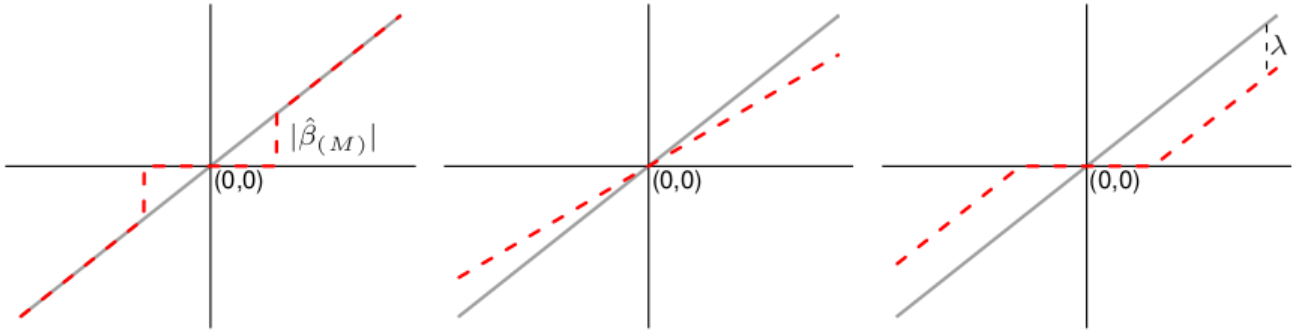
```
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   0.0000     0.0633  0.0000   1.0000
X[, 1]        0.1641     0.0616  2.6637   0.0081 **
X[, 2]        0.3203     0.3773  0.8490   0.3965
X[, 3]        0.0393     0.2974  0.1321   0.8949
X[, 4]        0.1117     0.0773  1.4444   0.1496
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1

Residual standard error: 1.175 on 339 degrees of freedom
Multiple R-squared:  0.04834,Adjusted R-squared:  0.03711
F-statistic: 4.305 on 4 and 339 DF,  p-value: 0.002072
```

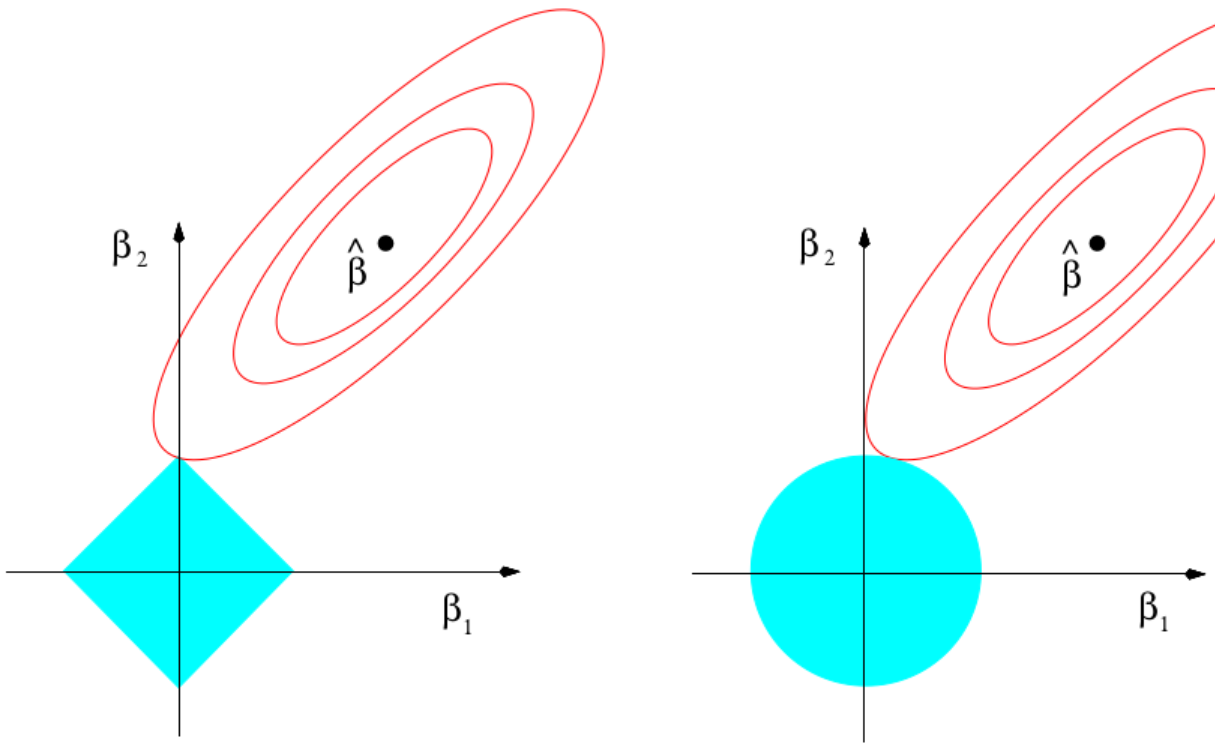
from: Van der Wieringen (2015)


lcavol
svi
lweight
pgg45
lbph
gleason
age
lcp
Coefficients
0.6
0.4
0.2
0.0
−0.2
0
2
4
6
8
df(λ)


Largest Principal
Component
Smallest Principal
Component
X_1
X_2
-4
-2
0
2
4


lcavol
svi
lweight
pgg45
lbph
gleason
age
lcp
Coefficients
0.6
0.4
0.2
0.0
−0.2
0.0
0.2
0.4
0.6
0.8
1.0
Shrinkage Factor s


$|\hat{\beta}_{(M)}|$
(0,0)
(0,0)
λ
(0,0)


β_2
$\hat{\beta}$
β_1
β_2
$\hat{\beta}$
β_1

Exercise 3.6

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

where

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

$$p(\beta) \hookrightarrow \mathcal{N}(0, \tau I)$$

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

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

Exercise 3.10

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

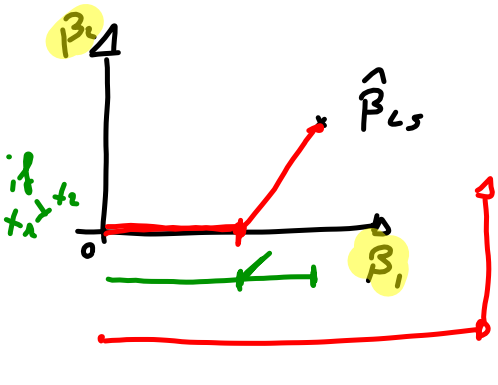
($p_1 - p_0 = 1$)

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

we know (ex 3.1), $F_{1,N-p-1} \triangleq z_j^2$

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

Least angle regression



$\hat{\beta}_{LS}^{(1)}$

LS

$\hat{\beta}_{LS}^{(2)}$

we update $\hat{\beta}_1$ until

$cor(x_1, r) < cor(x_2, r)$

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 β_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 $\hat{y} = \hat{\beta}_0 + \hat{\beta}_1^{LS} x_1$

LAR

$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1^{(2)} x_1$

- 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

$z_m = X v_m$ ← eigenvectors

z_1 = direction with the largest variability
z_2 = " " the 2nd largest variability s.t. $z_1 \perp z_2$
⋮
z_p = " " the least variability $z_p \perp z_1, \dots, z_{p-1}$

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

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

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

$$= \hat{\theta}_0 + \sum_{m=1}^{M} \hat{\theta}_m X v_m$$

$$= \hat{\theta}_0 + X \sum_{m=1}^{M} \hat{\theta}_m v_m$$

$$= \hat{\theta}_0 + X \hat{\beta}_{PCR}$$

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

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

$$\hat{\beta}_{PCR} = \sum_{j=1}^{p} v_j \mathbb{1}[j \leq M] v_j^T y = \sum_{j=1}^{M} v_j v_j^T y$$

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: $\hat{\varphi}_{1j} = \frac{\langle x_j, y \rangle}{\langle x_j, x_j \rangle}$ (with $\langle x_j, x_j \rangle = 1$) → $z_1 = \sum_{j=1}^{p} \hat{\varphi}_{1j} x_j$ — 1st pls direction

$\hat{\vartheta}_1 = \frac{\langle z_1, y \rangle}{\langle z_1, z_1 \rangle}$

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

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

$z_2 = \sum_{j=1}^{p} \hat{\varphi}_{2j} x_j^{(2)}$

$\hat{\vartheta}_2 = \frac{\langle z_2, y \rangle}{\langle z_2, z_2 \rangle}$

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

$\alpha^T S v_\ell = 0 \quad \ell = 1, \dots, m-1$

(S: sample covariance matrix)

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

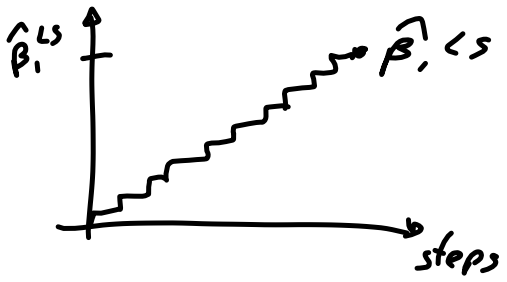
s.t. $\|\alpha\| = 1$

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

uncorrelated directions

In practice, the variance component tends to dominate the correlation one → similar results!

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 categorial variable
- working with genetic data, group genes belonging to the same pathway

$$\hat{\beta}_{gL} = \arg\min_{\beta} \left\{ \sum_{i=1}^{n} (y_i - \hat{\beta}_0 - \sum_{\ell=1}^{L} X_\ell \beta_\ell) + \lambda \sum_{\ell=1}^{L} \sqrt{p_\ell} \|\beta_\ell\|_2 \right\}$$

(p_ℓ: group size)

$\| \ \|_2$ = Euclidean distance
it is 0 $\Leftrightarrow$ all components are 0

- adaptive LASSO
- relaxed LASSO

} → 2-step procedure

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

$$\frac{dJ(\beta)}{d\beta} = \lambda \, \text{sign}(\beta) \left[\mathbb{1}[|\beta| \le \lambda] + \frac{(a\lambda - |\beta|)_+}{(a-1)\lambda} \mathbb{1}(|\beta| > \lambda) \right]$$

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

$$\hat{f}_o(x) = \hat{f}_a(x) = \hat{f}_c(x)$$

$$\hat{f}_o(x) > f_a(x) \wedge \hat{f}_c(x) > \hat{f}_a(x)$$

until now: classification based on linear regression

$$\forall k \quad \hat{f}_k(x) = \hat{\beta}_{k0} + \hat{\beta}_k^T x$$

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

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

If $\delta_k(x)$ or $Pr[G = K \mid X = x]$ are linear in x → linear decision boundaries

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

Examples

(i) $\hat{\delta}_k(x) = \hat{p}_k(x) = \hat{\beta}_{k0} + \hat{\beta}_{k1}x_1 + \hat{\beta}_{k2}x_2 + \hat{\beta}_{k3}x_1^2 + \hat{\beta}_{k4}x_2^2$

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

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

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

} logit transformation $\log \frac{p}{1-p}$

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

Linear regression of an Indicator Matrix

- codify each of the classes $1,\dots,K$ with an indicator variable

ex. $K=3 \Rightarrow$

	class 1	2	3	
	1	0	0	1st obs. is of class 1
	0	1	0	2nd and 3rd " 2
	0	1	0	
	0	0	1	4th " 5th " 3
	0	0	1	

$= Y$, $N \times K$

$\rightarrow$ linear regression: $\underset{N\times K}{\hat{Y}} = \underset{N\times p}{X}\underset{p\times p}{(X^TX)^{-1}}\underset{p\times N}{X^T}\underset{N\times K}{Y}$

new observation x_{new} $\quad \hat{y} = \hat{f}(x_{new}) = \underset{1\times p}{x_0}\underset{p\times K}{\hat{B}}$

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

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

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 $\sum_{k=1}^{K} \hat{f}_k(x) = 1$

→ $\hat{f}_k(x)$ 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 <u>masking effect</u>

- only if $K \geq 3$

As we saw if 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 $Pr(G=k \mid X=x)$

Suppose:

- $f_k(x)$ is the density of x conditional to class $G=k$
  $Pr[X=x \mid G=k]$
- $\hat{\pi}_k(x)$ is the prior probability to be in class k, $Pr[G=k]$

Then

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

$$= \frac{f_k(x)\, \hat{\pi}_k(x)}{\sum_{\ell=1}^{K} f_\ell(x)\, \hat{\pi}_\ell(x)}$$

We can choose $f_k(x)$ and $\hat{\pi}_k(x)$ as we prefer...

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

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

In particular, for LDA we suppose $\Sigma_k = \Sigma \quad \forall k$

We can then compare two classes by the log-ratio

$$\log \frac{Pr[G=k|X=x]}{Pr[G=\ell|X=x]} = \log \frac{\frac{1}{(2\pi)^{p/2}|\Sigma|^{1/2}} \exp\left\{-\frac{1}{2}(x-\mu_k)^T\Sigma^{-1}(x-\mu_k)\right\}\hat{\pi}_k / D}{\frac{1}{(2\pi)^{p/2}|\Sigma|^{1/2}} \exp\left\{-\frac{1}{2}(x-\mu_\ell)^T\Sigma^{-1}(x-\mu_\ell)\right\}\hat{\pi}_\ell / D}$$

$$= \log\frac{\hat{\pi}_k}{\hat{\pi}_\ell} - \frac{1}{2}\left(x^T\Sigma^{-1}x - 2\underbrace{x^T\Sigma^{-1}\mu_k}_{\text{scalar} = \mu_k^T\Sigma^{-1}x} + \mu_k^T\Sigma^{-1}\mu_k - x^T\Sigma^{-1}x + 2x^T\Sigma^{-1}\mu_\ell - \mu_\ell^T\Sigma^{-1}\mu_\ell\right)$$

$$= \log\frac{\hat{\pi}_k}{\hat{\pi}_\ell} - \frac{1}{2}(\mu_k+\mu_\ell)^T\Sigma^{-1}(\mu_k-\mu_\ell) + x^T\Sigma^{-1}(\mu_k-\mu_\ell)$$

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

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

$$\rightarrow \hat{G}(x) = \underset{k}{\operatorname{argmax}}\ \hat{\delta}_k(x)$$

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

- $\hat{\pi}_k = N_k/N = \dfrac{\#\text{ observations in class } k}{\text{total \# of observations}}$
- $\hat{\mu}_k = \sum_{g_i=k} x_i / N_k$
- $\hat{\Sigma} = \sum_{k=1}^{K}\sum_{g_i=k}(x_i-\mu_k)(x_i-\mu_k)^T/(N-K)$

With two classes, there is a simple correspondance 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

$x \in \mathbb{R}^p$ Y 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 2 if $\delta_2(x) > \delta_1(x)$

$$x^T\Sigma^{-1}\hat{\mu}_2 - \frac{1}{2}\hat{\mu}_2^T\Sigma^{-1}\hat{\mu}_2 + \log\frac{N_2}{N} > x^T\Sigma^{-1}\hat{\mu}_1 - \frac{1}{2}\hat{\mu}_1^T\Sigma^{-1}\hat{\mu}_1 + \log\frac{N_1}{N}$$

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

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

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

$Y = t_1U_1 + t_2U_2$ $\quad t_1 = -\frac{N}{N_1}, \; t_2 = \frac{N}{N_2}$

$1 = U_1 + U_2$

U_1	U_2	$=1$
1	0	1
0	1	1
0	1	1
1	0	1

Note that $\hat{\mu}_i = \frac{X^TU_i}{N_i} \Rightarrow X^TU_1 = N_1\hat{\mu}_1$, $X^TU_2 = N_2\hat{\mu}_2$

Therefore $X^TY = X^T(t_1U_1 + t_2U_2) = t_1N_1\hat{\mu}_1 + t_2N_2\hat{\mu}_2$

Our goal is to minimize $(Y-\beta_0 1-X\beta)^T(Y-\beta_0 1-X\beta)$

$$\begin{cases} \frac{\partial RSS}{\partial \beta_0} & -(Y-\beta_0 1-X\beta)^T 1 - 1^T(Y-\beta_0 1-X\beta) = 0 \\ \frac{\partial RSS}{\partial \beta} & -(Y-\beta_0 1-X\beta)^T X - X^T(Y-\beta_0 1-X\beta) = 0 \end{cases}$$

$$\rightarrow \begin{cases} 2\beta_0 \overbrace{1^T 1}^{N} - 2\,1^T(Y-X\beta) = 0 \\ 2X^TX\beta - 2X^TY + 2\beta_0 X^T 1 = 0 \end{cases}$$

$$\beta_0 = \frac{1}{N} 1^T (Y-X\beta)$$

$$\rightarrow X^TX\beta - X^TY + \frac{1}{N}X^T 1 1^T Y - \frac{1}{N}X^T 1 1^T X\beta = 0$$

$$\underbrace{\left(X^TX - \frac{1}{N}X^T 1 1^T X\right)\beta}_{LHS} = \underbrace{X^TY - \frac{1}{N}X^T 1 1^T Y}_{RHS}$$

$$RHS = t_1 N_1 \hat{\mu}_1 + t_2 N_2 \hat{\mu}_2 - \frac{1}{N}\left(X^T(U_1+U_2)(U_1+U_2)^T Y\right.$$

$$= t_1 N_1 \hat{\mu}_1 + t_2 N_2 \hat{\mu}_2 - \frac{1}{N}\left(X^T U_1 + X^T U_2\right)(U_1+U_2)^T(t_1 U_1 + t_2 U_2)$$

$$= t_1 N_1 \hat{\mu}_1 + t_2 N_2 \hat{\mu}_2 - \frac{1}{N}\left(N_1\hat{\mu}_1 + N_2\hat{\mu}_2\right)\left(t_1 U_1^T U_1 + \cancel{t_1 U_2^T U_1}^{=0} + t_2 \cancel{U_1^T U_2}^{=0} + t_2 U_2^T U_2\right)$$

$$= t_1 N_1 \hat{\mu}_1 + t_2 N_2 \hat{\mu}_2 - \frac{1}{N}\left(N_1\hat{\mu}_1 + N_2\hat{\mu}_2\right)\left(t_1 N_1 + t_2 N_2\right)$$

$$= \underline{t_1 N_1 \hat{\mu}_1} + t_2 N_2 \hat{\mu}_2 - \underline{\frac{N_1^2}{N} t_1 \hat{\mu}_1} - \frac{N_1 N_2}{N}\hat{\mu}_1 t_2 - \frac{N_1 N_2}{N}\hat{\mu}_2 t_1 - \frac{N_2^2}{N} t_2 \hat{\mu}_2$$

$$= t_1\hat{\mu}_1\left(N_1 - \frac{N_1^2}{N}\right) + t_2\hat{\mu}_2\left(N_2 - \frac{N_2^2}{N}\right) - \frac{N_1 N_2}{N}\hat{\mu}_1 t_2 - \frac{N_1 N_2}{N}\hat{\mu}_2 t_1$$

$$= t_1\hat{\mu}_1\left(\frac{N_1(N_1+N_2) - N_1^2}{N}\right) + t_2\hat{\mu}_2\left(\frac{N_2(N_1+N_2) - N_2^2}{N}\right) - \frac{N_1 N_2}{N}\hat{\mu}_1 t_2 - \frac{N_1 N_2}{N}\hat{\mu}_2 t_1$$

$$= \frac{N_1 N_2}{N}(t_1 - t_2)(\hat{\mu}_1 - \hat{\mu}_2)$$

$$LHS = X^TX - \frac{1}{N} X^T 1 1^T X$$

$$\Sigma = \sum_{k=1}^{K} \sum_{i:g_i=k} (x_i - \hat{\mu}_k)(x_i - \hat{\mu}_k)^T / (N-2)$$

$$(N-2)\Sigma = X^TX - N_1 \hat{\mu}_1 \hat{\mu}_1^T - N_2 \hat{\mu}_2 \hat{\mu}_2^T$$

$$\text{so } X^TX = (N-2)\Sigma + N_1 \hat{\mu}_1 \hat{\mu}_1^T + N_2 \hat{\mu}_2 \hat{\mu}_2^T$$

$$\frac{1}{N} X^T 1 1^T X = \frac{1}{N} X^T (U_1 + U_2)(U_1 + U_2)^T X$$

$$= \frac{1}{N} (N_1 \hat{\mu}_1 + N_2 \hat{\mu}_2)(N_1 \hat{\mu}_1 + N_2 \hat{\mu}_2)^T$$

$$= \frac{N_1^2 \hat{\mu}_1 \hat{\mu}_1^T}{N} + \frac{N_1 N_2}{N} \hat{\mu}_1 \hat{\mu}_2^T + \frac{N_1 N_2}{N} \hat{\mu}_2 \hat{\mu}_1^T + \frac{N_2^2}{N} \hat{\mu}_2 \hat{\mu}_2^T$$

$$= (N-2)\Sigma + N_1 \hat{\mu}_1 \hat{\mu}_1^T + N_2 \hat{\mu}_2 \hat{\mu}_2^T - \frac{N_1^2}{N} \hat{\mu}_1 \hat{\mu}_1^T - \frac{N_1 N_2}{N} \hat{\mu}_1 \hat{\mu}_2^T - \frac{N_1 N_2}{N} \hat{\mu}_2 \hat{\mu}_1^T - \frac{N_2^2}{N} \hat{\mu}_2 \hat{\mu}_2^T$$

$$= (N-2)\Sigma + \left(\frac{N_1(N_1+N_2) - N_1^2}{N}\right) \hat{\mu}_1 \hat{\mu}_1^T + \left(\frac{N_2(N_1+N_2) - N_2^2}{N}\right) \hat{\mu}_2 \hat{\mu}_2^T + \\ - \frac{N_1 N_2}{N} \hat{\mu}_1 \hat{\mu}_2^T - \frac{N_1 N_2}{N} \hat{\mu}_2 \hat{\mu}_1^T$$

$$= (N-2)\Sigma + \frac{N N_1 N_2}{N N} (\hat{\mu}_2 - \hat{\mu}_1)^T (\hat{\mu}_2 - \hat{\mu}_1)$$

$$\Sigma_B = \frac{N_1 N_2}{N^2} (\hat{\mu}_2 - \hat{\mu}_1)^T (\hat{\mu}_2 - \hat{\mu}_1)$$

$$= (N-2)\Sigma + N \Sigma_B$$

$$[(N-2)\Sigma + N\Sigma_B]\beta = \frac{N_1 N_2}{N} (t_1 - t_2)(\hat{\mu}_1 - \hat{\mu}_2)$$

$$= \frac{N_1 N_2}{N} \left(-\frac{N}{N_1} - \frac{N}{N_2}\right)(\hat{\mu}_1 - \hat{\mu}_2)$$

$$= \frac{N_1 N_2}{N} \left(-\frac{N(N_1+N_2)}{N_1 N_2}\right)(\hat{\mu}_1 - \hat{\mu}_2)$$

$$= N(\hat{\mu}_2 - \hat{\mu}_1)$$

(c) Note that $\hat{\Sigma}\hat{\beta} = (\hat{\mu}_2 - \hat{\mu}_1)\underbrace{\frac{N_1 N_2}{N^2}(\hat{\mu}_2 - \hat{\mu}_1)^T\hat{\beta}}_{\text{scalar}}$, it goes in the direction of $(\hat{\mu}_2 - \hat{\mu}_1)$

Both the LHS and the RHS of

$$[(N-2)\Sigma + N\Sigma_B]\beta = N(\hat{\mu}_2 - \hat{\mu}_1)$$

go in the direction of $(\hat{\mu}_2 - \hat{\mu}_1)$, so the solution must be proportional to $\Sigma^{-1}(\hat{\mu}_2 - \hat{\mu}_1)$

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

(d) t_1 and t_2 were chosen arbitrarily, so the result holds

(e)
$$\begin{aligned}
\hat{\beta}_0 &= \frac{1}{N}\underline{1}^T(\underline{Y} - X\hat{\beta}) \\
&= \frac{1}{N}(U_1 + U_2)^T(t_1U_1 + t_2U_2) - \frac{1}{N}(U_1 + U_2)^T X\hat{\beta} \\
&= \frac{1}{N}\left(t_1\underbrace{U_1^TU_1}_{N_1} + t_2\underbrace{U_1^TU_2}_{0} + t_1\underbrace{U_2^TU_1}_{0} + t_2\underbrace{U_2^TU_2}_{N_2}\right) - \frac{1}{N}(U_1^TX + U_2^TX)\hat{\beta} \\
&= \frac{1}{N}\left(-\frac{N}{N_1}\cdot N_1 + \frac{N}{N_2}\cdot N_2\right) - \frac{1}{N}(N_1\hat{\mu}_1 + N_2\hat{\mu}_2)\beta \\
&= -\frac{1}{N}(N_1\hat{\mu}_1 + N_2\hat{\mu}_2)\hat{\beta}
\end{aligned}$$

$$\begin{aligned}
\hat{f}(x) &= \hat{\beta}_0 + x^T\hat{\beta} \\
&= -\frac{1}{N}(N_1\hat{\mu}_1 + N_2\hat{\mu}_2)\hat{\beta} + x^T\hat{\beta} \\
&= \frac{1}{N}(Nx^T - N_1\hat{\mu}_1 - N_2\hat{\mu}_2)\hat{\beta} \\
&= \frac{1}{N}(Nx^T - N_1\hat{\mu}_1 - N_2\hat{\mu}_2)\lambda\hat{\Sigma}^{-1}(\hat{\mu}_2 - \hat{\mu}_1) \qquad \lambda \in \mathbb{R}
\end{aligned}$$

$$\hat{f}(x) > 0 \iff Nx^T\lambda\hat{\Sigma}^{-1}(\hat{\mu}_2 - \hat{\mu}_1) > (N_1\hat{\mu}_1 + N_2\hat{\mu}_2)\lambda\hat{\Sigma}^{-1}(\hat{\mu}_2 - \hat{\mu}_1)$$

$$x^T\hat{\Sigma}^{-1}(\hat{\mu}_2 - \hat{\mu}_1) > \frac{1}{N}(N_1\hat{\mu}_1 + N_2\hat{\mu}_2)\hat{\Sigma}^{-1}(\hat{\mu}_2 - \hat{\mu}_1)$$

LDA (4.11): $x^T\hat{\Sigma}^{-1}(\hat{\mu}_2 - \hat{\mu}_1) > \frac{1}{2}(\hat{\mu}_1 + \hat{\mu}_2)\hat{\Sigma}^{-1}(\hat{\mu}_2 - \hat{\mu}_1)$

The two rules are equal if $N_1 = N_2 = \frac{N}{2}$

Linear Discriminant Analysis

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

$$Pr[G = k \mid X = x] = \frac{f_k(x)\,\hat{\pi}_k}{\sum_{l=1}^{K} f_l(x)\,\hat{\pi}_l}$$

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

$\Sigma_k = \Sigma \quad \forall k \rightarrow$ LDA

When we do NOT assume, 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

$$\hat{\Sigma}_k = \sum_{j:k} (x_i - \hat{\mu}_k)(x_i - \hat{\mu}_k)^T / (N_k - 1)$$

- similar techniques, difference in # of parameters to estimate
  - for LDA, $\#\text{pars} = (K-1) \times (p-1)$
    - $(K-1)$: all differences to contrast $\delta_k - \delta_1$
    - $(p-1)$: # parameters to estimate for each difference
  - for QDA, $\#\text{pars} = (K-1) \times \left[\left[p(p+3)/2\right] + 1\right]$

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

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

where $\alpha \in [0;1]$ should be chosen (i.e., by cross-validation)
α controls the amount of shrinkage

$\alpha = 0 \longrightarrow$ LDA
$\alpha = 1 \longrightarrow$ 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 then α

- Combining

$$\hat{\Sigma}_k(\alpha,\gamma) = \alpha \hat{\Sigma}_k + (1-\alpha)\left[\gamma \Sigma + (1-\gamma)\sigma^2 I\right]$$

which

we obtain a general family for the covariate metrix, depends on α, γ

Reduced-rank LDA

Fisher: find the best combination $z = aX$ such that the between-class variance is maximized relative to the within-class variance

Total variance $T = B + W$

x_i from class k
g_i: class of x_i

$$T = \frac{1}{N}\sum_{i=1}^{N}(x_i - \bar{x})(x_i - \bar{x})^T$$

$$= \frac{1}{N}\sum_{k=1}^{K}\sum_{g_i=k}\left((x_i - \bar{x}_k) + (\bar{x}_k - \bar{x})\right)\left((x_i - \bar{x}_k) + (\bar{x}_k - \bar{x})\right)^T$$

$$= \frac{1}{N}\left[\sum_{k=1}^{K}\sum_{g_i=k}(x_i - \bar{x}_k)(x_i - \bar{x}_k) + \sum_{k=1}^{K}\left[\sum_{g_i=k}(x_i - \bar{x}_k)\right](\bar{x}_k - \bar{x})^T + \right.$$

$$\left. + \sum_{k=1}^{K}(\bar{x}_k - \bar{x})\sum_{g_i=k}(x_i - \bar{x}_k)^T + \sum_{k=1}^{K}\sum_{g_i=k}(\bar{x}_k - \bar{x})(\bar{x}_k - \bar{x})^T\right]$$

$$= \frac{1}{N}\sum_{k=1}^{K}\sum_{g_i=k}(x_i - \bar{x}_k)(x_i - \bar{x}_k)^T + \frac{1}{N}\sum_{k=1}^{K} N_k(\bar{x}_k - \bar{x})(\bar{x}_k - \bar{x})^T$$

(the cross terms vanish because $\sum_{g_i=k}(x_i - \bar{x}_k) = 0$)

- first term: within-class variance W
- second term: between-class variance B

a: $a^T B a$ is maximized
$a^T W a$ is minimized

a_1: $\arg\max_a \frac{a^T B a}{a^T W a}$ or $\max_a a^T B a$ subject to $a^T W a = 1$

→ generalized eigenvalue problem, a corresponds to the largest eigenvalue of $W^{-1}B$

a_2: $a_2 \perp a_1$, $\arg\max_a \frac{a^T B a}{a^T W a}$

a_3 ...

$a_1, a_2, \ldots$ are called:
- discriminant coordinates
- canonical variates

Motivations
- initially data reduction for visualization
- can be seen as a restricted classification rule

↑ the centroids lie in the L-dimensional subspace of $\mathbb{R}^p$

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

$$\log \frac{Pr[G=1 \mid X=x]}{Pr[G=K \mid X=x]} = \beta_{10} + \beta_1^T x$$

$$\log \frac{Pr[G=2 \mid X=x]}{Pr[G=K \mid X=x]} = \beta_{20} + \beta_2^T x$$

$$\vdots$$

$$\log \frac{Pr[G=K-1 \mid X=x]}{Pr[G=K \mid X=x]} = \beta_{K-1\,0} + \beta_{K-1}^T x$$

- specifies $K-1$ log-odd
- based on the logit transformation

$K=2$ (≥ 2)

$$\log \frac{p}{1-p} = x\beta \iff p = \frac{e^{x\beta}}{1+e^{x\beta}}$$

$$Pr[G=1 \mid X=x] = p = \frac{e^{x\beta^T}}{1+e^{x\beta^T}}$$

$$Pr[G=2 \mid X=x] = 1 - Pr[G=1 \mid X=x] = 1-p = \frac{1}{1+e^{x\beta^T}}$$

$$\frac{e^{x\beta^T}}{1+e^{x\beta^T}} + \frac{1}{1+e^{x\beta^T}} = 1$$

$\beta = \beta_0 \beta$

$X = (1, X)$

$$\hat{\beta} = \arg\max_{\beta} L(\beta)$$

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

$$= \prod_{i=1}^{N} \binom{1}{y_i} p(x_i;\beta)^{y_i} (1 - p(x_i;\beta))^{1-y_i}$$

$$\ell(\beta) = \sum_{i=1}^{N} y_i \log p(x_i;\beta) + (1-y_i) \log (1 - p(x_i;\beta))$$

$$= \sum_{i=1}^{N} \left[y_i \log \frac{e^{x\beta^T}}{1+e^{x\beta^T}} + \log\left(1 - \frac{e^{x\beta^T}}{1+e^{x\beta^T}}\right) - y \log\left(1 - \frac{e^{x\beta^T}}{1+e^{x\beta^T}}\right) \right]$$

$$= \sum_{i=1}^{N} \left[y_i x\beta^T - y_i \log(1+e^{x\beta^T}) + \log\left(\frac{1+e^{x\beta^T} - e^{x\beta^T}}{1+e^{x\beta^T}}\right) + y \log\left(\frac{1}{1+e^{x\beta^T}}\right) \right]$$

$$= \sum_{i=1}^{N} y_i x_i \beta^T - \log(1+e^{x\beta^T})$$

$$\ell_\beta(\beta) = \frac{\partial \ell(\beta)}{\partial \beta} = \sum_{i=1}^{N} \left[y_i x_i - \frac{e^{x\beta^T}}{1+e^{x\beta^T}} x_i \right] = 0$$

$$= \sum_{i=1}^{N} x_i (y_i - p(x_i;\beta)) = 0$$

system of $p+1$ equations not linear in β

→ to find the solution, we can use the Newton-Raphson algorithm

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \quad \leftarrow \ell_\beta(\beta)$$

$$\leftarrow \ell_{\beta\beta}(\beta) = \frac{\partial^2 \ell(\beta)}{\partial \beta \partial \beta^T}$$

$$\ell_{\beta\beta}(\beta) = -\sum_{i=1}^{N} \left(x_i e^{x_i\beta^T}(1+e^{x_i\beta^T}) - x_i e^{x_i\beta^T} e^{x_i\beta^T} \right) x_i^T \Big/ (1+e^{x\beta^T})^2$$

$$= -\sum_{i=1}^{N} x_i \frac{e^{x_i\beta^T}(1 + e^{x_i\beta^T} - e^{x_i\beta^T})}{(1+e^{x_i\beta^T})^2} x_i^T$$

$$= -\sum_{i=1}^{N} x_i \frac{e^{x_i\beta^T}}{(1+e^{x_i\beta^T})^2} x_i^T$$

Newton-Raphson

$$\beta_{new} = \beta_{old} - \ell_{\beta\beta}(\beta)^{-1}\ell_{\beta}(\beta)$$

$$\ell_{\beta}(\beta) = X^T(y-p) \qquad p = \frac{e^{x\beta^T}}{1+e^{x\beta^T}}$$

$$\ell_{\beta\beta}(\beta) = -X^T W X \qquad \text{where } W = \begin{bmatrix} p(1-p) & & \\ & \ddots & \\ & & p(1-p) \end{bmatrix}$$

$$\frac{e^{x\beta^T}}{1+e^{x\beta^T}} \cdot \frac{1}{1+e^{x\beta^T}} = \frac{e^{x\beta^T}}{(1+e^{x\beta^T})^2}$$

p $\quad$ $(1-p)$

β in W and p are β_{old}

$$\beta_{new} = \beta_{old} + \underbrace{(X^T W X)^{-1}}_{\ell_{\beta\beta}(\beta)} \underbrace{X^T(y-p)}_{\ell_{\beta}(\beta)}$$

$$= (X^T W X)^{-1} X^T W \underbrace{(X\beta_{old} + W^{-1}(y-p))}_{z}$$

$$= (X^T W X)^{-1} X^T W z \longrightarrow \text{weighted least square}$$

- 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

$$\hat{\beta} = \arg\min_{\beta} \left\{ -\ell(\beta) + \lambda \sum_{j=1}^{p} |\beta_j| \right\}$$

$$= \arg\max_{\beta} \left\{ \ell(\beta) - \lambda \sum_{j=1}^{p} |\beta_j| \right\}$$

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

Exercise:

- try to reproduce Table 3.3 with the data from the South Africa heart disease example

$\hat{\beta}$	LS	best(α=0.05)	LASSO	RIDGE	LDA	QDA

- read ch 4.4.5

Logistic regression vs LDA

LDA $\log \frac{Pr[G=0|X=x]}{Pr[G=1|X=x]} = \log \frac{\pi_0}{\pi_1} - \frac{1}{2}(\mu_0+\mu_1)^T\Sigma^{-1}(\mu_0-\mu_1) + x^T\Sigma^{-1}(\mu_0-\mu_1)$

$= \alpha_{01} + \alpha_1^T x$

similar to the logistic regression

$= \beta_{01} + \beta_1^T x$

$Pr[G=k, X=x] = Pr(x)\, Pr[G=k|X=x]$

both LDA and logistic regress

LDA also model this part

$Pr(x) = \sum_{k=1}^{K} \pi_k \phi(x; \mu_k, \Sigma)$

Gaussian

$\frac{e^{\beta_{0k} + \beta_k^T x}}{1 + \sum_{\ell=1}^{K} e^{\beta_{0\ell} + \beta_\ell^T x}}$

Model assessment (1) and model selection (2)

(1) evaluate the performance (in term of prediction) of a selected model

(2) select the best model (for prediction)

GOAL of a 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

$$Err_{\mathcal{T}} = E[L(Y, \hat{f}(x)) \mid \mathcal{T}]$$

remember, 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

$$Err = E[L(Y, \hat{f}(x))] = E[Err_{\mathcal{T}}]$$

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_T

↳ the training error is NOT a good estimate of Err_T

↳ $\overline{err} = \frac{1}{N}\sum_{i=1}^{N} L(y_i, \hat{f}(x_i))$

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 $\mathcal{G}$

indicator function $\mathbb{1}(x \neq x_0) \begin{cases} 1 & x \neq x_0 \\ 0 & x = x_0 \end{cases}$

typical loss functions in this case $L(G, \hat{G}(x)) = \begin{cases} \mathbb{1}(G \neq \hat{G}(x)) & \text{0-1 loss} \\ -2 \log L(\beta) & \text{deviance} \end{cases}$

$-2\log L(\beta) = \ell(\beta)$

- $-2\,\ell(\beta)$ is a general loss-function, it can be used in all 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

$$L(\beta) = \frac{1}{\sqrt{2\pi}^1} \exp\left\{-\frac{1}{2\cdot 1}\sum_{i=1}^{N}(y_i - x_i^T\beta)^2\right\}$$

$$\ell(\beta) = -\frac{1}{2}\underbrace{\sum_{i=1}^{N}(y_i - x_i^T\beta)^2}_{\text{sum of squares}}$$

$-2\,\ell(\beta)$ = squared loss

In an ideal situation
↳ a lot of data

we can randomly split the observations in three independent sets

- 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

$Y = f(x) + \varepsilon$, $E[\varepsilon] = 0$

$Var[\varepsilon] = \sigma_\varepsilon^2$

$$Err(x_0) = E\left[(Y - \hat{f}(x_0))^2 \mid X = x_0\right]$$

$$= \sigma_\varepsilon^2 + \left(E[\hat{f}(x_0)] - f(x_0)\right)^2 + E\left[\left(\hat{f}(x_0) - E[\hat{f}(x_0)]\right)^2\right]$$

$$= \text{irreducible error} + \text{bias}^2 + \text{variance}$$

where

- σ_ε^2, the variance of the target around the true mean, so we cannot do anything about that
- bias^2, 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

$$\hat{f}(x_0) = \frac{1}{K} \sum_{x_i \in N_K(x_0)} y_i$$

$$Err(x_0) = \sigma_\varepsilon^2 + \left[f(x_0) - \frac{1}{K}\sum_{\ell=1}^{K} f(x_{(\ell)})\right]^2 + \frac{\sigma_\varepsilon^2}{K}$$

$$E[\hat{f}(x)] = E\left[\frac{1}{K}\sum_{\ell=1}^{K} Y_{(\ell)}\right] = \frac{1}{K}\sum_{\ell=1}^{K} E[Y_{(\ell)}] = \frac{1}{K}\sum_{\ell=1}^{K} f(x_{(\ell)})$$

$$Var[\hat{f}(x_0)] = Var\left[\frac{1}{K}\sum_{\ell=1}^{K} Y_{(\ell)}\right] = \frac{1}{K^2}\sum_{\ell=1}^{K} Var[Y_{(\ell)}] = \frac{1}{K^2}\sum_{\ell=1}^{K} \sigma_\varepsilon^2 = \frac{K}{K^2}\sigma_\varepsilon^2 = \frac{\sigma_\varepsilon^2}{K}$$

$$\#K \propto \frac{1}{\text{complexity}}$$

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

$$\frac{1}{N}\sum_{i=1}^{N} Err(x_i) = \sigma_\varepsilon^2 + \frac{1}{N}\sum_{i=1}^{N}\left[f(x_i) - E[\hat{f}(x_i)]\right]^2 + \frac{p}{N}\sigma_\varepsilon^2$$

$p \uparrow$

called the in-sample error

p = # 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

$$E_{x_0}\left[f(x_0) - E[\hat{f}(x_0)]\right]^2 = E_{x_0}\left[f(x_0) - x^T\hat{\beta}_*\right] + E_{x_0}\left[x^T\beta - E[x_0^T\hat{\beta}_\alpha]\right]^2$$

$$\hat{\beta}_* = \arg\min_\beta E\left[f(x) - x^T\beta\right]^2$$

Average [model bias]^2 + Ave [Estimation bias]^2

difference between the true function and the best-fitting linear approximation

error between the average estimate and the best model

additional bias that we add when, e.g., add shrinkage

OLS = 0

difference between truth and best model

- we can reduce it only increasing the model space (more variables, more complex relationship - interaction -, ...)

Optimism of the Training Error

Let us start from the definitions of last time:

$$Err_{\mathcal{T}} = E_{(X_0, Y_0)}\left[L(Y_0, \hat{f}(X_0)) \mid \mathcal{T}\right]$$

Test error, where

- X_0, Y_0 are new (test) point → 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

$$Err = E_{\mathcal{T}}\left[E_{(X_0, Y_0)}\left[L(Y_0, \hat{f}(X_0)) \mid \mathcal{T}\right]\right]$$

gives the expected error

We also saw the training error

$$\overline{err} = \frac{1}{N}\sum_{i=1}^{N} L(y_i, \hat{f}(x_i))$$

is NOT a good estimate of the test error, because it underestimate it.

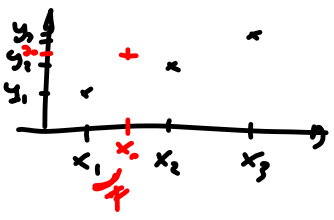
Same data used for both training $\hat{f}(x)$ and to test its performance → optimistic estimate

The test error can be thought as a <u>extra-sample</u> error (the estimation of the error is computed on new points, $x_0 \neq x_i$

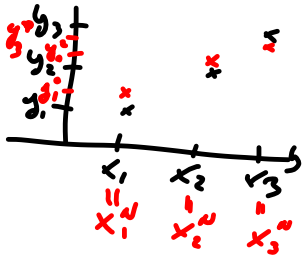
We are going to evaluate the optimism of $\overline{err}$ in the <u>in-sample</u> case, i.e. we have new observations in the same points of training set

$$Err_{in} = \frac{1}{N}\sum_{i=1}^{N} E_{Y^0}\left[L(Y_i^0, \hat{f}(x_i) \mid \mathcal{T}\right]$$

<u>extra-sample</u>



in sample



- only Y_0 is random, new y_i^0 for $x_1, \dots, x_N$

Definition: we define optimism the difference between Err_{in} and $\overline{err}$

$$op := Err_{in} - \overline{err}$$

op is generally positive, as $\overline{err}$ is computed on training data

Definition: average optimism is

$$w := E_y[op]$$

↑ 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

$$w = \frac{2}{N}\sum_{i=1}^{N} Cov(\hat{y}_i, y_i)$$

where Cov denotes the covariance → 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 $Cov(\hat{y}_i, y_i)$ → the higher the optimism

As a consequence:

$$E_y[Err_{in}] = E_y[\overline{err}] + \frac{2}{N}\sum_{i=1}^{N} Cov(\hat{y}_i, y_i)$$

When $\hat{y}_i$ is obtained by a linear fit in the inputs, $Y = f(x) + \varepsilon$

$$\sum_{i=1}^{N} Cov(\hat{y}_i, y_i) = d\,\sigma_\varepsilon^2$$ ← effective number of parameters

therefore

$$\rightarrow E_y[Err_{in}] = E_y[\overline{err}] + 2\frac{d}{N}\sigma_\varepsilon^2 \quad (*)$$

e.g. in linear regression, the number of covariates

- optimism increases with the number of inputs;
- " decreases " " sample size.

Methods we will see:
- C_p, 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 (*)

$$E_y[\overline{Err}_{in}] = E_Y[\overline{err}] + 2\frac{d}{N}\sigma_\varepsilon^2 \quad (*)$$

Write the general form of the in-sample estimate

$$\widehat{Err}_{in} = \overline{err} + \hat{\omega}$$

- when we have linearity and squared errors, from (*)

$$C_p = \overline{err} + 2\frac{d}{N}\hat{\sigma}_\varepsilon^2$$

NB: there are other versions of C_p, different versions may give different numbers, but they all lead to the same model

where:
- $\overline{err}$ is computed through the squared loss;
- d is # of parameters
- $\hat{\sigma}_\varepsilon^2$ is an estimate of the noise variance — it is computed using the full model, because it has the smallest bias

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 ($N \to \infty$)

$$-2E[\log P_\theta(Y)] \approx -\frac{2}{N}E\left[\sum_{i=1}^{N}\log P_{\hat{\theta}}(y_i)\right] + 2\frac{d}{N}$$

loglik → the maximized log-likelihood $\ell(\hat{\theta})$ is the mle maximum likelihood estimate

- $P_\theta(Y)$ is the family of densities for Y (containing the "true" density)

E.g., in the logistic regression: $AIC = -\frac{2}{N}\text{loglik} + 2\frac{d}{N}$

Gaussian regression: $AIC \propto C_p$ — 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

$$AIC(\alpha) = \overline{err}(\alpha) + 2\frac{d(\alpha)}{N}\hat{\sigma}_\varepsilon^2$$

- usually minimizing AIC is not the best solution to find the value of the tuning parameter ⟶ cross-validation is a better approach

The effective number of parameters

- generalized the concept of number of parameters, in order to extend the previous approaches to more complex situation

Suppose

$y = (y_1, \dots, y_N)$ outcome

$\hat{y} = (\hat{y}_1, \dots, \hat{y}_N)$ predictions

Linear regression

$\hat{y} = \underbrace{X (X^T X)^{-1} X^T}_{S} y$

Linear methods $\hat{y} = Sy$

Ridge regression

$\hat{y} = \underbrace{X(X^T X + \lambda I_p)^{-1} X^T}_{S} y$

where S is a $N \times N$ matrix which:

- depends on X
- independent on Y

Effective number of parameters (or effective degrees of freedom)

$df(S) = \text{trace}(S)$

We should replace trace(S) to d to obtain the correct criterion (value of the)

If $y = f(X) + \varepsilon$, $Var(\varepsilon) = \sigma_\varepsilon^2 \rightarrow \sum_{i=1}^{N} Cov(\hat{y}_i, y_i) = \text{trace}(S)\,\sigma_\varepsilon^2$

So $df(\hat{y}) = \frac{\sum_{i=1}^{N} Cov(\hat{y}_i, y_i)}{\sigma_\varepsilon^2}$ $\rightarrow$ Ex 7.5

The Bayesian approach and BIC

The BIC is an alternative criterion to AIC,
↳ Bayesian Information Criterion

$$\frac{1}{N}BIC = -\frac{2}{N}\log \text{lik} + \log N \cdot \frac{d}{N}$$

AIC = 2
BIC = $\log(n)$

Despite they are quite similar, AIC and BIC come from completely different ideas. BIC comes from the Bayesian approach to model selection

$$Pr(\mathcal{M}_m|z) \propto Pr(\mathcal{M}_m) Pr(z|\mathcal{M}_m)$$
$$\propto Pr(\mathcal{M}_m) \int Pr(z|\mathcal{M}_m, \theta_m) Pr(\theta_m|\mathcal{M}_m) d\theta_m$$

To choose between two models, we compare their posterior probabilities

$$\frac{Pr(\mathcal{M}_0|z)}{Pr(\mathcal{M}_1|z)} = \frac{Pr(\mathcal{M}_m)}{Pr(\mathcal{M}_1)} \cdot \frac{Pr(z|\mathcal{M}_m)}{Pr(z|\mathcal{M}_1)}$$

Bayes factor

in a large number of case = 1 (give the same prior probability to the two model) $\Rightarrow$ the choice between the two models is based on the Bayes factor

Approximately

$$Pr(z|\mathcal{M}_m) = \log(z|\hat{\theta}_m, \mathcal{M}_m) - \frac{d_m}{2}\log N + O(1)$$

m/e

If the loss function is $-2 \log Pr[z|\hat{\theta}, \mathcal{M}_m)$, we obtain BIC

- We may select the model with smallest BIC

  correspond to selecting the model with highest posterior probability

Note that $\frac{e^{-\frac{1}{2}BIC_m}}{\sum_{l=1}^{M} e^{-\frac{1}{2}BIC_l}}$ 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 ($N \to \infty$, 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

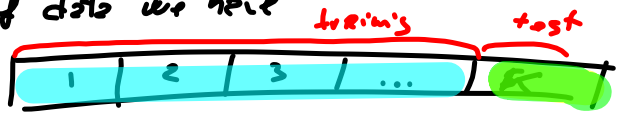
$$Err = E[L(Y, \hat{f}(x))]$$

independent test set

the average test error when $\hat{f}(x)$ is applied to a new sample

If we had enough data ⟶ training set
                       ⟶ test "

↳ in general, we have not, so we mimic this split using the limited amount of data we have



folds

derived using all data but the κ-th part

- divide the observations in K folds
- we use, in turn, K-1 folds to train the model (derive $\hat{f}^{-\kappa}(x_i)$)
- we evaluate the model in the remaining fold

$$CV(\hat{f}) = \sum_{\kappa=1}^{K} \frac{n_\kappa}{N} \sum_{i \in \kappa} L(y_i, \hat{f}^{-\kappa}(x_i))$$

- 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 ⟶ the training sets are really similar to each other)
- usual choices are $K = 5$ or $K = 10$

[Fig 7.8]

⟶ 1-Err vs N
⟶ the classifier is ok until $\approx N = 100$ (then is flat)

- if $N = 200$, $K = 5$ ⟶ training $N_t = 160$ ✓ $160 > 100$
- if $N = 50$, $K = 5$ ⟶ " $N_t = 40$ ✗ $40 \not> 100$

Notes:
- CV estimates Err and not the Err_t
- if we want to select a tuning parameter via CV

$\hat{f}^{-\kappa}(x, \alpha)$ is the model selected using α and fitted on the observations which do not belong to the κ-th fold

$$CV(\hat{f}, \alpha) = \frac{1}{N} \sum_{\kappa=1}^{K} \sum_{i=1}^{n_\kappa} L(y_i, \hat{f}(x_i, \alpha))$$

$$\hat{\alpha} = \arg\min_{\alpha} CV(\hat{f}, \alpha)$$

Generalized cross-validation
- convenient approximation to the LOOCV, for squared loss function

$$\hat{y} = Sy$$

LOOCV: $\frac{1}{N} \sum_{\kappa=1}^{K} \sum_{i=1}^{n_\kappa} \left(y_i - \hat{f}^{-i}(x_i)\right)^2 \to \frac{1}{N} \sum_{i=1}^{N} \left(y_i - \hat{f}^{-i}(x_i)\right)^2 = \frac{1}{N} \sum_{i=1}^{N} \left(\frac{y_i - \hat{f}(x_i)}{1 - S_{ii}}\right)^2$

where S_{ii} is the ith 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

$$Err_{in} = \frac{1}{N}\sum_{i=1}^{N} E_{Y^0}\left[(Y_i^0 - \hat{f}(x_i))^2\right]$$

$$\overline{err} = \frac{1}{N}\sum_{i=1}^{N} (y_i - \hat{f}(x_i))^2 \qquad y = f(x) + \varepsilon$$

$$E_Y[op] = \frac{2}{N}\sum_{i=1}^{N} Cov(\hat{y}_i, y)$$

$$E_Y[op] = E_Y\left[\frac{1}{N}\sum_{i=1}^{N}\left\{E_{Y^0}\left[(Y_i^0 - \hat{f}(x_i))^2\right] - (y_i - \hat{f}(x_i))^2\right\}\right]$$

$$= E_Y\left[\frac{1}{N}\sum_{i=1}^{N}\left\{E_{Y^0}[Y_i^{0\,2}] - 2E_{Y^0}[Y_i^0]\hat{f}(x_i) + \hat{f}(x_i)^2 + E[\hat{f}(x_i)] - E[\hat{f}(x_i)] - y_i^2 + 2E_Y[y_i\hat{f}(x_i)] - \hat{f}(x_i)^2\right\}\right]$$

$$= \frac{1}{N}\sum_{i=1}^{N} E_Y\left[E_{Y^0}[Y_i^{0\,2}]\right] - 2E_Y[Y_i^0]E_Y[\hat{f}(x_i)] - E_Y[y_i^2] + 2E_Y[y_i\hat{f}(x_i)]$$

- $E_Y\left[E_{Y^0}[Y_i^{0\,2}] - f(x)^2 + f(x)^2\right] = \sigma_\varepsilon^2 + f(x)^2$
- $E_Y\left[y^2 - f(x)^2 + f(x)^2\right] = \sigma_\varepsilon^2 + f(x)^2$
- $E_{Y^0}[Y_i^0] = f(x) = E_Y[y]$ $\qquad \hat{f}(x) = \hat{y}$

$$E_Y[op] = \frac{1}{N}\sum_{i=1}^{N}\left\{-2E_Y[y]E_Y[\hat{y}] + 2E_Y[y_i\hat{y}_i]\right\}$$

$$= \frac{2}{N}\sum_{i=1}^{N} Cov(y_i, \hat{y}_i)$$

Ex 7.5

For $\hat{y} = Sy$ show that $\sum_{i=1}^{N} Cov(y_i, \hat{y}_i) = trace(S)\sigma^2$

$$\sum_{i=1}^{N} Cov(y_i, \hat{y}_i) = trace(Cov(y, \hat{y}))$$
$$= trace(Cov(y, Sy))$$
$$= trace(S\, Cov(y, y))$$
$$= trace(S\, Var(y))$$
$$= trace(S\sigma^2)$$
$$= trace(S)\sigma^2$$

Cov matrix
$$\begin{pmatrix} Cov(y_1,y_1) & Cov(y_1,y_2) & \cdots & Cov(y_1,y_n) \\ Cov(y_2,y_1) & Cov(y_2,y_2) & & \vdots \\ \vdots & & \ddots & \\ Cov(y_n,y_1) & \cdots & & Cov(y_n,y_n) \end{pmatrix}$$
$\sum_{i=1}^{rank(M)} m_{ii} = trace\ M$

Exercise 7.7

Use the approximation: $\frac{1}{(1-x)^2} \approx 1 + 2x$

to show similarities between Cp/AIC and GCV

$$C_p = \overline{err} + 2\frac{d}{N}\hat{\sigma}_\varepsilon^2 \qquad AIC = -\frac{2}{N} loglik + 2\frac{d}{N}$$

$$GCV = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{y_i - \hat{f}(x_i)}{1 - trace(S)/N}\right)^2$$

$$C_p = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{f}(x_i))^2 + 2\frac{d}{N}\hat{\sigma}_\varepsilon^2$$

$trace(S) = \frac{\sum Cov(y_i, \hat{y}_i)}{\sigma^2}$

$\hat{\sigma}_\varepsilon^2 = \frac{1}{N-1}\sum(y_i - \hat{f}(x_i))^2$

$$GCV = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{f}(x_i))^2 \cdot \left(\frac{1}{1 - \frac{trace(S)}{N}}\right)^2$$
$$= \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{f}(x_i))^2\left(1 + 2\cdot\frac{trace(S)}{N}\right)$$
$$= \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{f}(x_i))^2 + \frac{2}{N^2}trace(S)\sum(y_i - \hat{f}(x_i))^2 = \overline{err} + \frac{2}{N}trace(S)\hat{\sigma}_\varepsilon^2$$

Gaussian regression

$$AIC \propto -\frac{2}{N}\log\left(\exp\left\{-\frac{1}{2\hat{\sigma}_\varepsilon^2}\sum_{i=1}^{N}(y_i - \hat{f}(x_i))^2\right\}\right) + \frac{2d}{N}$$
$$= +\frac{1}{N\hat{\sigma}_\varepsilon^2}\sum_{i=1}^{N}(y_i - \hat{f}(x_i))^2 + \frac{2d}{N}$$
$trace(S)$
$$\propto \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{f}(x_i))^2 + \frac{2d}{N}\sigma_\varepsilon^2$$

Bootstrap methods

→ what is bootstrap
→ how to use bootstrap for error estimation $\hat{E}[Err_\tau]$

IDEA: generate bootstrap sample from the empirical distribution computed on original sample
→ by resampling with replacement from the original sample

- suppose $\tau = \{(x_1, y_1), \dots, (x_n, y_n)\}$
- by resampling, $\tau_1^* = \{(x_1^*, y_1^*), \dots, (x_n^*, y_n^*)\}$
- repeat for B large, $\tau_1^*, \tau_2^*, \dots, \tau_B$

Based on the generated bootstrap sample (which mimic new experiments) we can estimate any aspect of the distribution of a map

Example

original sample $\tau = \{z_1, z_2, z_3, z_4\} = \{1, 3, 4, 6\}$

generate B bootstrap sample by resampling with replacement from τ

$\tau_1^* = \{z_1^*, z_2^*, z_3^*, z_4^*\}$
$= \{1, 4, 1, 6\}$

$\tau_2^* = \{4, 4, 3, 3\}$

$\vdots$

$\tau_B^* = \{1, 1, 1, 1\}$

$(X, Y) \quad Cov(X, Y)$

$\tau \ (x_1; y_1), (x_2; y_2), \dots, (x_n; y_n)$
$(4; 3), (1; 3), \dots, (4; 2)$

$\tau_1^* = \{(1; 3), (4; 2), \dots, (4; 2)\}$

Bootstrap approach for prediction error estimation

WRONG APPROACH
- estimate our $\hat{f}(x)$ from each bootstrap sample
- evaluate how well $\hat{f}^*_b(x)$ estimate y

$$\widehat{Err}_3^{\text{wrong}} = \frac{1}{B}\sum_{b=1}^{B}\left(\frac{1}{N}\sum_{i=1}^{N} L(y, \hat{f}^*_b(x_i))\right)$$

⚠ : training and test set are not independent

$E[\widehat{Err}_3^{\text{wrong}}] = 0.184$, 1NN , $Y \perp X$

- $y_i \in \tau^*_b$ → error = 0
- $y_i \notin \tau^*_b$ → error = 0.5

$= 0.5 \times Pr[y_i \notin \tau^*_b] + 0 \times Pr[y_i \in \tau^*_b]$

$Pr[y_i \notin \tau^*_b] \approx 0.368$

$Pr[\text{observation } i \notin \text{bootstrap sample } b]$

$Pr[\tau^*_{b(i)} \neq y_i] = \frac{N-1}{N}$ ← same for all positions

$\Rightarrow Pr[y_i \notin \tau^*_b] = \left(\frac{N-1}{N}\right)^N = e^{-1} \approx 0.368$

$E[\widehat{Err}_3^{\text{wrong}}]\Big|_{\substack{X \perp Y \\ 1NN}} \approx 0.5 \times e^{-1} = 0.184$

An important fact

$Pr[\text{observation } i \text{ belongs to a bootstrap sample } b] = 1 - e^{-1} \approx \boxed{0.632}$

approximation of $1 - \left(\frac{N-1}{N}\right)^N$

CORRECT APPROACH

NB.: bootstrap sample has the same size of the original sample

$\mathcal{T} = \{z_1, \dots, z_N\}$

$\mathcal{T}_b^* = \{z_1^*, \dots, z_N^*\}$ resampling with replacement

→ there are original observations which are included more than once
⇒ there are (also) original observations which are not included at all
→ these can be used as a test set as they are not used in the training process.

$$\widehat{Err}^{(1)} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{|C^{-i}|}\sum_{b\in C^{-i}} L\left(y_i, \hat{f}_b^*(x_i)\right)$$

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 $0.632N$ → 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

$$\widehat{Err}^{(.632)} = 0.368\ \overline{err} + 0.632\ \widehat{Err}^{(1)}$$

In general, it works well, but in some case it fails, like in our 1NN ×⊥Y

$\overline{err} = 0 \rightarrow \widehat{Err}^{(.632)} = 0.632\ \widehat{Err}^{(1)}$

Further improvements ".632+ estimator"

- based on the quantity γ, 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 seperately, we compute the prediction error for each combination of y_i and x_j

$$\hat{\gamma} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{N}\sum_{j=1}^{N} L\left(y_i, \hat{f}(x_j)\right)$$

- $\hat{\gamma}$ is used to compute the overfitting rate

$$\hat{R} = \frac{\widehat{Err}^{(1)} - \overline{err}}{\hat{\gamma} - \overline{err}} \qquad 0 \le \hat{R} \le 1$$

↖ no overfitting

- finally

$$\widehat{Err}^{(.632+)} = (1-\hat{w})\,\overline{err} + \hat{w}\,Err^{(1)}$$

where $w = \dfrac{0.632}{1 - 0.368\hat{R}}$

Generalized Additive Models

- extensions of the (generalized) linear model

Linear model
- powerfull tool
- can be used in several cases (regression, classification, ...)

Main limitation
- it suppose linear effects, often not true in reality
  ($\hat{\beta}$ is the increments in y when the corresponding x increase of 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

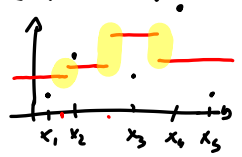
- we already saw that we can use $f_j(x_j) = x_j^2$, $f_j(x_j) = \log x_j$
- we can be more general, and use a nonparametric function (splines, kernel, ...)

splines → § 5.2

kernel → § 6.1, § 6.2

- cubic splines → bottom right of Fig 5.2
- natural splines → since the estimation outside the observations' range can be dangerous, the line is forced to be linear outside the range

Kernel method
- extension of k-nn

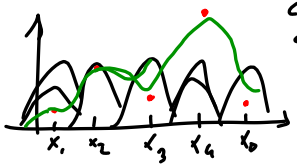


2-nn
- use the y_i of the 2 nearest neighbours
- constant
- ugly and unecessery discontinuities

kernel method
- weights through a kernel $D\left(\frac{|x-x_0|}{h_\lambda(x_0)}\right)$

different kinds of kernels

h, λ are parameters
h controls the shape λ the width of the kernel



GAM

for functional effect, we can

- $E[Y|X] = \alpha + \beta_1 f_1(x_1) + \beta_2 f_2(x_2) + \dots + \beta_p f_p(x_p)$
  $f_i(x_i) = \log(x_i)$
  → 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

↳ link function

e.g. logistic model

$g(\cdot) = \text{logit}$ $\mu(x) = P[Y=1|X=x]$

$\log\left(\frac{\mu(x)}{1-\mu(x)}\right) = \alpha + \beta_1 x_1 + \dots + \beta_p x_p$

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
- semiparametric model $g(\mu(x)) = \underbrace{X^T\beta}_{\text{parametric}} + \underbrace{f(z)}_{\text{non-parametric}}$

e.g. semiparametric model : Cox model
$\lambda(t) = \underbrace{\lambda_0(t)}_{\text{non-parametric}} \underbrace{\exp(X^T\beta)}_{\text{parametric part}}$

8.2.2 Maximum likelihood inference

Today
- Boosting (AdaBoost : the first popular boosting algorithm)
- statistical interpretation of boosting (L2 Boosting)
- likelihood-based boosting (together with model-based boosting (gradient boosting), statistical boosting)

Y_i iid with density $p(y_i;\vartheta)$

e.g. $Y_i \sim N(\mu, \sigma^2) \to p(y_i;\underbrace{\mu,\sigma^2}_{\vartheta}) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{-\frac{1}{2\sigma^2}(y_i-\mu)^2\right\}$

$L(\vartheta;y) = \prod_{i=1}^{n} p(y_i;\vartheta)$ likelihood

$\ell(\vartheta;y) = \sum_{i=1}^{n} \log p(y_i;\vartheta)$ log-likelihood

$\hat{\vartheta} = \arg\max_{\vartheta} \ell(\vartheta;y) = \arg\max_{\vartheta} L(\vartheta;y)$

$\ell_\vartheta(\vartheta;y) = \frac{\partial \ell(\vartheta;y)}{\partial \vartheta}$ score function

$\ell_{\vartheta\vartheta}(\vartheta;y) = \frac{\partial^2 \ell(\vartheta;y)}{\partial \vartheta \partial \vartheta^T}$

$\to j(\vartheta) = -\ell_{\vartheta\vartheta}(\vartheta;y)$ observed information

$\to i(\vartheta) = E_\vartheta[j(\vartheta)]$ expected information

If ϑ_0 is the true parameter

$$\hat{\vartheta} \xrightarrow{n\to\infty} N(\vartheta_0; j(\vartheta_0)^{-1})$$

Estimation

$$\hat{\vartheta} \sim N(\hat{\vartheta}; j(\hat{\vartheta})^{-1}) \quad \text{or} \quad \hat{\vartheta} \sim N(\hat{\vartheta}; i(\vartheta)^{-1})$$

confidence intervals $\hat{\vartheta} \pm z_{1-\frac{\alpha}{2}} \sqrt{j^{-1}(\hat{\vartheta})}$

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 (GAM)

Starting challenge:

"can a committee of blockheads 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 (iteratively) apply a weak estimator to modifications of the data → AdaBoost

at each iteration AdaBoost gives more weight to the misspecified observations

Consider a two class classification problem

$Y_i \in \{-1, 1\}$

X_i: the vector of inputs

AdaBoost algorithm

① inizielize, weights are $\left(\frac{1}{N}, \frac{1}{N}, \ldots, \frac{1}{N}\right) = w^{[0]}$

② for m from 1 to m_stop (M)

a) fit the weak estimator $\hat{g}(x)$ to the weighted data;

b) compute the weighted in-sample missclassification rate

$$err^{[m]} = \frac{\sum_{i=1}^{N} w_i^{[m-1]} \mathbb{1}\left[y_i \neq \hat{g}^{[m]}(x_i)\right]}{\sum_{i=1}^{N} w_i^{[m-1]}}$$

c) compute $\alpha_m = \log\left(\frac{1 - err^{[m]}}{err^{[m]}}\right)$

α_m is used to weight the contribution of the $\hat{g}^{[m]}(x)$ to the final estimate (classification)

d) update the weights

$$\tilde{w}_i = w^{[m-1]} \exp\left\{\alpha_m \mathbb{1}\left[y_i \neq \hat{g}^{[m]}(x_i)\right]\right\} \quad i = 1, \ldots, N$$

reweight only missclassified observations

ok

$$w_i^{[m]} = \frac{\tilde{w}_i}{\sum_{i=1}^{N} \tilde{w}_i}$$

③ compute final result

$$\hat{f}_{AdaBoost} = sgn\left(\sum_{m=1}^{m\_stop} \alpha^{[m]} \hat{g}^{[m]}(x)\right)$$

Example

step [0] $w = \left(\frac{1}{10}, \frac{1}{10}, \dots, \frac{1}{10}\right)$

step [1] $err = \frac{\sum_{i=1}^{N} \frac{1}{10} \mathbb{1}[y_i \neq \hat{g}_i]}{\sum_{i=1}^{N} \frac{1}{10}} = \frac{\frac{3}{10}}{\frac{10}{10}} = 0.3$

$\alpha_1 = \log \frac{1-err}{err} = \log 0.7 - \log 0.3 \simeq 0.84$

$\tilde{w} = \left(\underbrace{\exp(0.84)\frac{1}{10}}_{\simeq 0.23}, 0.23, 0.23, 0.1, \dots, 0.1\right)$

misclassified in the first iteration

$w^{[1]} \propto (0.17, 0.17, 0.17, 0.07, \dots, 0.07)$

Statistical view of Boosting

- functional gradient descendent algorithm
- forward stagewise (additive) modelling

→ see AdaBoost as 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 $\hat{f}^{[m-1]} = \sum_{k=1}^{m-1} \alpha_k \hat{g}^{[k]}(x)$ ($1 \dots m-1$ given in boosting)
- the goal is to find $(\alpha_m, g_m) = \arg\min_{\alpha, g} \sum_{i=1}^{N} \exp\{-y \underbrace{\sum_{k=1}^{m} \alpha_k g^{[k]}}_{f}\}$

$$\alpha_m, g_m = \arg\min_{\alpha, g} \sum_{i=1}^{N} \exp\{-y (\underbrace{\sum_{k=1}^{m-1} \alpha_k \hat{g}^{[k]}}_{\hat{f}^{[m-1]}} + \alpha g)\}$$

$$= \arg\min_{\alpha, g} \sum_{i=1}^{N} w_i^{[m-1]} \exp\{-y \alpha g\} \quad (*)$$

where $w_i = \exp\{-y \hat{f}^{[m-1]}\}$, which do not depend neither on α nor on g (given from the previous iteration)

$y \in \{-1, 1\}$

$g \in \{-1, 1\}$

- two step procedure: first we minimize with respect to g

$$\arg\min_{g} \sum_{i=1}^{N} w_i^{[m-1]} \exp(-y \alpha g)$$

if $y = g \rightarrow yg = 1$

if $y \neq g \rightarrow yg = -1$

$$= \arg\min_{g} \left\{\sum_{g=y} w_i^{[m-1]} e^{-\alpha} + \sum_{g \neq y} w_i^{[m-1]} e^{\alpha}\right\}$$

$\pm e^{-\alpha} \sum_{g \neq y} w_i$

$$= \arg\min_{g} \left\{e^{-\alpha} \sum_{i=1}^{N} w_i^{[m-1]} + \left(e^{\alpha} - e^{-\alpha}\right) \sum_{g \neq y} w_i\right\}$$

$$\arg\min \left\{\sum_{g=y} w_i^{[m-1]} e^{-\alpha} + \sum_{g \neq y} w_i^{[m-1]} e^{-\alpha} = \sum_{g \neq y} w_i^{[m-1]} e^{-\alpha} + \sum_{g \neq y} w_i^{[m-1]} e^{\alpha}\right\}$$

$$= \arg\min_{g} \left\{e^{-\alpha} \sum_{i=1}^{N} w_i^{[m-1]} + \left(e^{\alpha} - e^{-\alpha}\right) \sum_{i=1}^{N} w_i^{[m-1]} \mathbb{1}[g_i \neq y_i]\right\}$$

$$g = \arg\min_{g} \left\{\sum_{i=1}^{N} w_i^{[m-1]} \mathbb{1}[g_i \neq y_i]\right\}$$

$$= \arg\min_{\alpha, g} \sum_{i=1}^{N} w_i^{[m-1]} \exp\{-y\alpha g\}$$

$$\arg\min_{\alpha} \sum w_i^{[m-1]} \exp\{-y\alpha g\} \qquad \frac{\partial L}{\partial \alpha} = \Sigma$$

$$y = g \longrightarrow \sum_{g=y} w_i^{[m-1]} \exp\{-\alpha\}$$

$$y \neq g \longrightarrow \sum_{g \neq y} w_i^{[m-1]} \exp\{\alpha\}$$

$$\frac{\partial L}{\partial \alpha} = -\sum_{g=y} w_i^{[m-1]} \exp\{-\alpha\} + \sum_{g \neq y} w_i^{[m-1]} \exp\{\alpha\} = 0$$

multiply both terms for e^{α}

$$= -\sum_{g=y} w_i^{[m-1]} + \sum_{g \neq y} w_i^{[m-1]} \exp\{2\alpha\} = 0$$

$$e^{2\alpha} \sum_{g \neq y} w_i^{[m-1]} = \sum_{g=y} w_i^{[m-1]}$$

$$e^{2\alpha} = \frac{\sum_{i=1}^{N} w_i^{[m-1]} - \sum_{g \neq y} w_i^{[m-1]}}{\sum_{g \neq y} w_i^{[m-1]}}$$

$$\alpha = \frac{1}{2} \log\left(\frac{1 - err}{err}\right)$$

where $err = \dfrac{\sum_{i=1}^{N} w_i^{[m-1]} \mathbb{1}(y \neq g)}{\sum_{i=1}^{N} w_i^{[m-1]}}$

- $\hat{g}^{(m)}$ minimizer of the weighted misclassification
- $\alpha = \frac{1}{2} \log\left(\frac{1 - err}{err}\right)$

Our general classifier is updated as

$$\hat{f}^{[m]} = \hat{f}^{[m-1]} + \alpha_m \hat{g}^{[m]}$$

which causes the weights of the next iteration to be

$$w_i^{[m]} = w_i^{[m-1]} \cdot \exp\{-\alpha y_i \hat{g}_i^{[m]}\}$$

Since $-y_i \hat{g}_i^{[m]} = -\sum_{g_i = y_i} 1 + \sum_{g_i \neq y_i} (+1) = -\sum_{g_i \neq y_i} 1 + \sum_{g_i \neq y_i} 1$

$$= 2 \sum_{g_i \neq y_i} 1 - 1$$

$$w_i^{[m]} = w_i^{[m-1]} \cdot \exp\{+\alpha \left(2 \sum_{g_i \neq y_i} 1 - 1\right)\}$$

$$= w_i^{[m-1]} \exp\{2\alpha \mathbb{1}(y_i \neq g_i) - \alpha\}$$

$$= w_i^{[m-1]} e^{-\alpha} e^{2\alpha \mathbb{1}(y_i \neq g_i)}$$

$e^{-\alpha}$: constant for each observation → can be ignored

$2\beta = 2\alpha$ 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

$$f^*(x) = \arg\min_{f(x)} E_{Y|x}[e^{-Y f(x)}] = \frac{1}{2} \log \frac{Pr[Y=1|x]}{Pr[Y=-1|x]}$$

alternatively: $Pr[Y=1|x] = \frac{1}{1+e^{-2f^*(x)}}$

$\frac{1}{2}$ 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

$$\pi = Pr[Y=1|x] = \frac{e^{f(x)}}{e^{f(x)} + e^{-f(x)}} = \frac{1}{1+e^{-2f(x)}}$$

$y' = \frac{y+1}{2} \in \{0;1\}$

then

$$\ell(\pi) = y' \log \pi + (1-y') \log(1-\pi)$$

$$\Rightarrow -\ell(\pi) = \log(1+e^{-2yf(x)})$$

$\log_2$

Exercise 10.2

$$f^*(x) = \arg\min_{f(x)} E_{Y|x}\left[e^{-Yf(x)}\right]$$

$$\frac{\partial}{\partial f(x)} E_{Y|x}\left[e^{-Yf(x)}\right] = E_{Y|x}\left[-Ye^{-Yf(x)}\right]$$

$$E_{Y|x}\left[-Ye^{-Yf(x)}\right] = 0 \qquad Y = \begin{cases} -1 & \Pr[Y=-1|x] \\ 1 & \Pr[Y=1|x] \end{cases}$$

$$+(+1)e^{+(+1)f(x)}\Pr[Y=-1|x] - (1)e^{-1\cdot f(x)}\Pr[Y=1|x] = 0$$

$$e^{f(x)}e^{f(x)}\Pr[Y=-1|x] = e^{-f(x)}\Pr[Y=1|x]$$

multiply both sides for $e^{f(x)}$

$$e^{2f(x)}\Pr[Y=-1|x] = \Pr[Y=1|x]$$

$$e^{2f(x)} = \frac{\Pr[Y=1|x]}{\Pr[Y=-1|x]}$$

$$f(x) = \frac{1}{2}\log\left(\frac{\Pr[Y=1|x]}{\Pr[Y=-1|x]}\right)$$

Exercise 10.4

$X = (x_1, \ldots, x_{10})$ $\quad x_j \sim N(0;1)$ $\quad x_j$ iid $\quad N = 2000$

$$Y_i = \begin{cases} 1 & \text{if } \sum_{j=1}^{10} x_j^2 > \chi^2_{10}(0.5) \\ -1 & \text{if } \sum_{j=1}^{10} x_j^2 \leq \chi^2_{10}(0.5) \end{cases}$$

	x_1	x_2	...	x_{10}
1	x_{11}	x_{12}	...	$x_{1,10}$
2				
⋮				
N				

apply (x^2, 1, sum)

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 : - stump - tree - ...

↳ AdaBoost : weight more missclassified observations

for classification → AdaBoost $e^{-Yf(x)}$

from classification to regression
- loss function : RSS
- weak estimator : $\nu(X^TX)^{-1}X^Ty$ (LS estimator; weak estimator)
  - penalty parameter $0 < \nu < 1$ → makes our LSE "weak" default = 0.1
- modification of the data : $y \to u$ residuals
  - focusing on the not explain part of the variation
  - outcome

$\nu \to 0$
u

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

$\hat{b}^{[m]} = \nu (X^T X)^{-1} X^T v$ → $\nu = 0.1$

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

$\hat{\beta}_{L_2Boost} = \sum_{m=1}^{m\_stop} \hat{b}^{[m]} = \hat{\beta}^{[m\_stop]}$

Note:

$m \to \infty$, $\hat{\beta}_{L_2Boost} \longrightarrow \hat{\beta}_{OLS}$

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 => more steps (iterations)
larger values => less steps

X must be centred $E[X_j] = 0$
(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

↝ negative gradient : $-\frac{\partial L(y, f(x))}{\partial f(x)}$

e.g. $\frac{\partial \sum \frac{1}{2}(y - X^T\beta)^2}{\partial \underbrace{X^T\beta}_{f(x)}} = \frac{2}{2}\underbrace{\sum (y - X^T\beta)}_{\text{residuals}}$

$\frac{1}{K} \exp\{-\frac{1}{2}(y - x^T\beta)\}$

In general:

1. Initialization : $\hat{f}(x) = 0$ or $\hat{f}(x) = \bar{y}$
2. for m from 1 to m_stop
   (a) derive $U = -\frac{\partial L(Y, f(x))}{\partial f(x)}$ ← RSS, lin. reg.
   (b) fit our weak estimator : $\hat{b} = g(U, X, U)$ ← lin. reg. residuals; $(X^TX)^{-1}X^Ty$; lin reg
   (c) update the estimate $\hat{f}^{[m]} = \hat{f}^{[m-1]} + \hat{b}$
3. finalization $\hat{f}_{boost} = \hat{f}^{[m\_stop]}$

GAM : $y = \sum_{j=1}^{p} f_j(x_j)$

L_2 Boost for High Dimensional Data

- one of the advantages of boosting is that we can handle HDD
  → component wise version of boosting

Componentwise boosting

- Linear (Gaussian) regression model

① Initialization $\hat{\beta}_j^{[0]} \equiv 0 \quad j = 1, \dots, p$

$(u = y - X\hat{\beta}_j^{[0]} = y)$

② for m from 1 to mstop

a) compute possible updates for each dimension of the regression coefficient vector seperately (fit a weak estimator on each dimension of x)

$$b_j = \frac{\sum_{i=1}^{n} x_i^{(j)} u_i}{\sum_{i=1}^{n} x_i^{(j)2}} \qquad j = 1, \dots, p$$

b) select the best update among the p possibilities

$$j^* : \arg\min_j \sum_{i=1}^{n} (u_i - x^{(j)} b_j)^2$$

c) update the j^*-th regression coefficient

$$\hat{\beta}^{[m]} = \hat{\beta}^{[m-1]} + (0, \dots, 0, b_{j^*}, 0, \dots, 0)$$

d) modify the data $u = y - X\hat{\beta}^{[m]}$

---

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

$\mu(y) = X^T\beta$ → glmboost

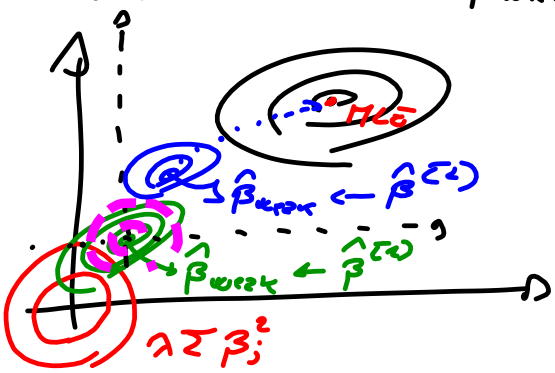
Gaussian regression, logistic regression, ⋮

Likelihood-based boosting

- fully statistical approach $\rightsquigarrow$ 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

$$\hat{\beta}_{weak} = (X^TX + \lambda I)^{-1}X^Ty$$

same meaning of the parameter

Boosting Ridge (Gaussian regression)

① initialize $\hat{\beta}^{[0]} = (0, \dots 0)$ $\rightarrow$ $U = y - X\hat{\beta}^{[0]}$

② fit the weak estimator $\hat{b} = (X^TX + I\lambda)X^Ty$

$$\frac{\partial \ell_{pen}(\beta)}{\partial \beta} = 0 \qquad \ell_{pen} \propto \frac{1}{2}(y - X\beta)(y - X\beta)^T + \frac{\lambda}{2}\beta\beta^T \qquad \lambda \rightarrow \frac{\lambda}{2}$$

$$\frac{\partial \ell_{pen}}{\partial \beta} = -X^T(y - X\beta) + \lambda\beta = 0$$

$$-X^Ty + X^TX\beta + \lambda\beta = 0$$

$$\beta(X^TX + \lambda I) = X^Ty$$

$$\hat{b} = \hat{\beta} = (X^TX + \lambda I)^{-1}X^Ty$$

kernel of a Gaussian log-likelihood

$\sigma^2 = 1$ $\exp\left\{-\frac{1}{2}(y - X\beta)(y - X\beta)^T\right\}$

(b) $\hat{\beta}^{[m]} = \hat{\beta}^{[m-1]} + \hat{b}$

(c) modification of the data (add an offset in the log-likelihood)

$$\frac{1}{2}\left(y - X\hat{\beta}^{[m]} - X\beta\right)\left(y - X\hat{\beta}^{[m]} - X\beta\right)^T$$

③ $\hat{\beta}_{cBoost} = \sum_{m=1}^{m.stop} \hat{b}^{[m]}$

weak estimator is the ridge estimator

Gaussian regression → likelihood-based boosting
gradient boosting → same model when we choose ν and λ in the right way

Exercise

$$\hat{y} = X\hat{\beta}_{boost} = \sum_{m=1}^{m.stop} S(I-S)^m y = I - (I-S)^{m.stop+1} y$$

where $S = Xb$

using $b = (X^TX + \lambda I)^{-1}X^T y$ Boosting with ridge estimator

Show through SVD that Boosting ridge and ridge regression provide different shrinkage effect

ridge regression : $\frac{d_j^2}{d_j^2 + \gamma}$

Boosting Ridge : $\left(1 - \left(1 - \frac{d_j^2}{d_j^2 + \lambda}\right)^{m.stop+1}\right)$

Likelihood-based boosting

- Loss function
- weak estimator
- modification of the data

L_2 Boost

$u_i := \frac{\partial L(y, f(x))}{\partial f(x)}$

penalized version of OLS

$(X^TX)^{-1}X^Tu$

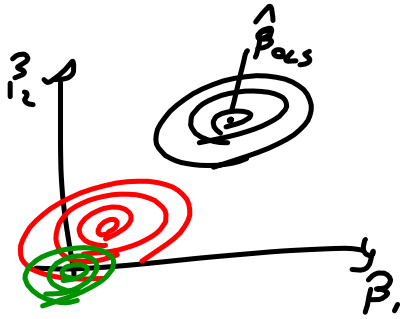
u X L_2 OLS minimizer of L_2 loss

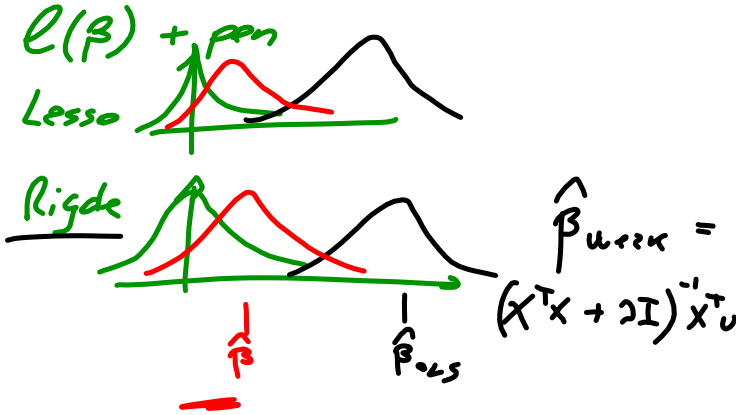
L_2 is negative log-likelihood

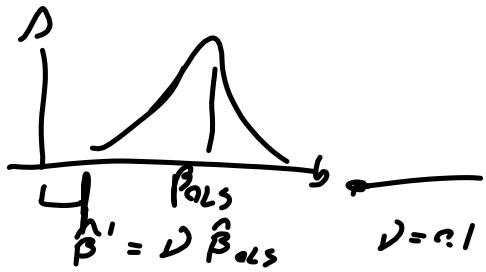
parametric version

$L(y, f(x, \beta)) :=$ negative log-likelihood

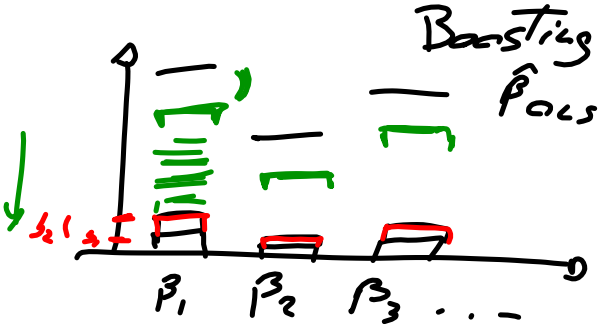
weak estimator → instead of minimizing the negative log-likelihood we minimize a penalized version of it







L_2 Boost



$\hat{\beta}^{[1]}_{boost}$

$\hat{\beta}^{[2]}_{boost} = \hat{b}^{[1]} + \hat{b}^{[2]}$

Algorithm

1) Initialization $\hat{\beta} = 0_n$ $u = y$

2) for m from 1 to m_steps

$\ell_{pen}(0 + \beta; \lambda)$

$\frac{\partial \ell_{pen}(0+\beta;\lambda)}{\partial \beta} = 0$

Ridge penalty

$(X^T X + I\lambda)^{-1} X^T u$

3) $\hat{b}^{[m]} = (X^T X + \lambda I)^{-1} X^T u$

$\ell_{pen}(\Sigma \hat{b}^{[m]} + \beta; \lambda)$

Gaussian case $\ell_{pen}(\Sigma b^{[m]} + \beta; 1) =$

$(y - \Sigma b^{[m]} X - X\beta)^T (y - \Sigma b^{[m]} X - X\beta) + \lambda \beta\beta^T$

u u

Exercise

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

$x\hat{b}^{[0]}$ $\hat{f}^{[0]}_{boost}(x) = X(X^TX + \lambda I)^{-1}X^T y = Sy$ ✓

$\hat{f}^{[1]}_{boost}(x) = X(X^TX + \lambda I)^{-1}X^T u$ $\quad u = y - x\hat{\beta}^{[m-1]} = y - X(X^TX + \lambda I)^{-1}X^T y$

$x\hat{b}^{[1]}$

$= X(X^TX + \lambda I)^{-1}X^T (y - X(X^TX + \lambda I)^{-1}X^T y)$

$= S(I - S)y$

$\hat{f}^{[2]}_{boost}(x) = X(X^TX + \lambda I)^{-1}X^T u$

$= X(X^TX + \lambda I)^{-1}X^T (y - x\hat{b}^{[1]} - x\hat{b}^{[0]})$

$= X(X^TX + \lambda I)^{-1}X^T (y - X(X^TX + \lambda I)^{-1}(y - X(X^TX + \lambda I)^{-1}X^T y) - X(X^TX + \lambda I)^{-1}X^T y)$

$= S(y - S(I - S)y - Sy)$

$= S(I - S + S^2 - S)y = S(I - S)^2 y$

$\vdots$

$x\hat{b}^{[m]} = S(I - S)^m y$

m-th improvement

$\hat{\beta}^{[m\_step]}_{boost} = \sum_{j=0}^{m\_step} \hat{b}^{[j]}$

$\hat{y} = x\hat{\beta}^{[m\_step]} = \sum_{j=0}^{m\_step} S(I - S)^j y = \underbrace{\left(I - (I - S)^{m\_step+1}\right)}_{H_m} y$

$\hat{y} = H_m y$ where $H = I - (I - S)^{m\_stop+1}$

$= I - (I - X(X^TX + \lambda I)^{-1}X^T)^{m\_stop+1}$

use Singular Value Decomposition

$X = UDV^T$ where U spans the column space of X, $U^TU = I$

V " " row " " , $V^TV = VV^T = I$

D diagonal matrix with singular values

$d_1 \geq d_2 \geq \dots \geq d_p$

$$H_m = I - \left(I - UDV^T\left(VDU^TUDV^T + \lambda I\right)^{-1}VDU^T\right)^{m\_stop+1}$$

$$= I - (I - UDV^TV(D^2 + \lambda I)^{-1}V^TVDU^T)^{m\_stop+1}$$

$$= I - (I - UD^2(D^2+\lambda I)^{-1}U^T)^{m\_stop}$$

$$= U(I - (I - \tilde{D})^{m\_stop+1})U^T \qquad \tilde{D} = (D^2 + \lambda I)^{-1}D^2$$

$$\tilde{d}_j^2 = \frac{d_j^2}{d_j^2 + \lambda I}$$

Ridge estimator (from lecture 4, notes page 8)

ridge $X\hat{\beta}_{ridge} = UD^2(D^2 + \gamma I)^{-1}U^T$

ridge-boosting $X\hat{\beta}_{boost} = U(I - (I - D^2(D^2+\lambda I)^{-1})^{m\_stop+1})U^T$

$D^2(D^2 + \gamma I)^{-1} = I - (I - D^2(D^2+\lambda I)^{-1})^{m\_stop}$

no γ, λ, m_stop s.t. $\neq 0$

$m\_stop = 0$.

apply only the first time the weak estimator (ridge)

$D^2(D^2 + \gamma I)^{-1} = I - (I - D^2(D^2+\lambda I)^{-1})^0$

$= I - I + D^2(D^2+\lambda I)^{-1}$

only if $m\_stop = 0$

$\gamma \neq \lambda$

L_2 Boost is an algorithm that goes under the umbrella of gradient boosting + likelihood-based boosting } → 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

CAM with implicit variable selection by likelihood-based boosting

• general on statistical boosting: Mayr et al. (2014)
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)

loss function negative (binomial) log-likelihood

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 → have 0 bias, minimize the variance
(Gauss-Markov theorem : OLS is BLUE)
best estimator

We saw approaches which accept an increase in bias to have a (larger) decrease in variance

$$E\left[(Y_0 - \hat{f}(x_0))^2\right] = E\left[Y_0^2 - 2Y_0\hat{f}(x_0) + \hat{f}(x_0)^2\right]$$

$$= E[Y_0^2] - f(x_0) + f(x_0) + E\left[\hat{f}(x_0)^2\right] - E\left[\hat{f}(x_0)\right]^2 + E\left[\hat{f}(x_0)\right]^2$$

$$- 2\left(E[Y_0\hat{f}(x_0)] - E[Y_0]E[\hat{f}(x_0)] + E[Y_0]E[\hat{f}(x_0)]\right)$$

$\text{Cov}(Y_0, \hat{f}(x_0)) = 0$

$$= \sigma^2 + \text{Var}\left(\hat{f}(x_0)\right) + \left(E[\hat{f}(x_0)] - \underbrace{f(x_0)}_{\hat{R}(x_0)}\right)^2$$

$=$ irreducible error + variance + bias^2

$Y = f(x_0) + \varepsilon \qquad f(x_0) \perp \varepsilon$

- prediction error
  - training error (underestimating the prediction error)
  - test error (computed on independent data)

- comparing methods based on prediction error
  - training error + estimate optimism 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
  - forward selection
  - best subset selection
  - stepping criteria (AIC, BIC, significance level (t-test, F-test))

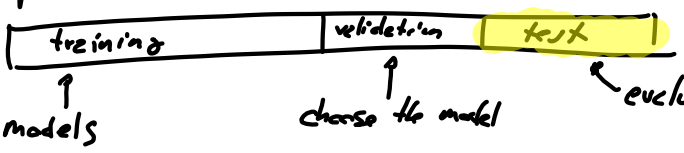
... and combinations (stepwise selection, stepback selection)

X splines

- problems with high-dimensional data
  - classical approaches (OLS) do not work

either
- add constrains (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 act as a training set) without involving the observation 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

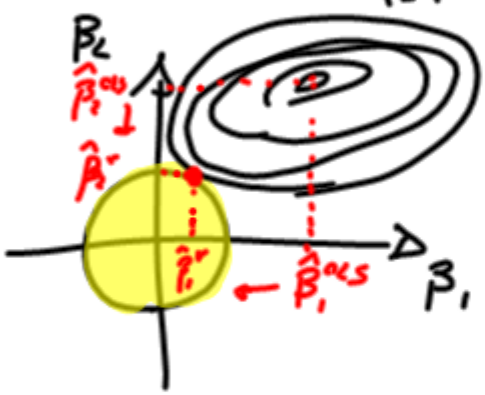
a) 1: lasso

2: ridge regression

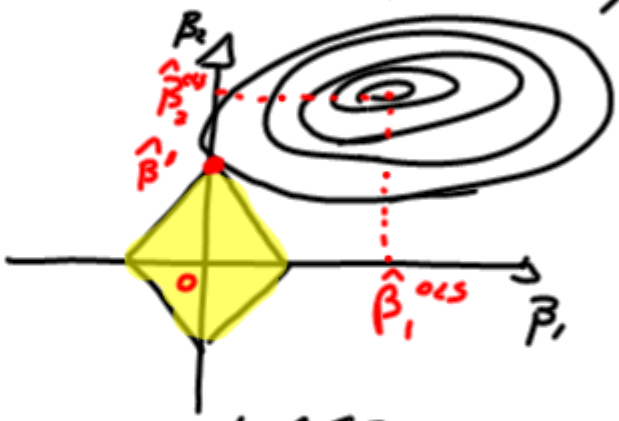
$$PRSS_\lambda^{lasso} = \sum_{i=1}^{N} (y_i - \beta_1 x_1 - \beta_2 x_2)^2 + \lambda(|\beta_1| + |\beta_2|)$$

$$PRSS_\lambda^{ridge} = \sum_{i=1}^{N} (y_i - \beta_1 x_1 - \beta_2 x_2)^2 + \lambda(\beta_1^2 + \beta_2^2)$$

b)



RIDGE



LASSO

Exercise 2

a) $\hat{\beta}_{ridge} = (X^TX + \lambda I)^{-1} X^T y$

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

$$\hat{\beta}_{ridge} = \underbrace{(X^TX + \lambda I)^{-1}(X^TX)}_{A} \underbrace{(X^TX)^{-1} X^T y}_{\hat{\beta}_{ols}}$$

$$= A\hat{\beta}_{ols}$$

b) $N \to \infty$, $X^T X \simeq N\Sigma$

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}^{-1} = \frac{1}{ad-bc}\begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$$

$$\hat{\beta}_{ridge} = A\hat{\beta}_{ols} = (X^TX + \lambda I)^{-1} X^T X \hat{\beta}_{ols}$$

$$\overset{N \text{ large}}{\approx} (N\Sigma + \lambda I)^{-1} N\Sigma \hat{\beta}_{ols}$$

$$= (\Sigma + \frac{\lambda I}{N})^{-1}\Sigma\hat{\beta}_{ols}$$

$$= \begin{pmatrix} 1+\frac{\lambda}{N} & \rho \\ \rho & 1+\frac{\lambda}{N} \end{pmatrix}^{-1}\begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix}\begin{pmatrix} \hat{\beta}_{ols}^{[1]} \\ \hat{\beta}_{ols}^{[2]} \end{pmatrix}$$

$$= \frac{1}{(1+\frac{\lambda}{N})^2-\rho^2}\begin{pmatrix} 1+\frac{\lambda}{N} & -\rho \\ -\rho & 1+\frac{\lambda}{N} \end{pmatrix}\begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix}\begin{pmatrix} \hat{\beta}_{ols}^{[1]} \\ \hat{\beta}_{ols}^{[2]} \end{pmatrix}$$

$$= \frac{1}{(1+\frac{\lambda}{N})^2-\rho^2}\begin{pmatrix} 1+\frac{\lambda}{N}-\rho^2 & \frac{\lambda}{N}\rho \\ \frac{\lambda}{N}\rho & 1+\frac{\lambda}{N}-\rho^2 \end{pmatrix}\begin{pmatrix} \hat{\beta}_{ols}^{[1]} \\ \hat{\beta}_{ols}^{[2]} \end{pmatrix}$$

$$\Sigma = \begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix} \qquad \frac{\lambda I}{N} = \frac{\lambda}{N}\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} + \begin{pmatrix} \frac{\lambda}{N} & 0 \\ 0 & \frac{\lambda}{N} \end{pmatrix}$$

$$\hat{\beta}_{ridge}^{[1]} = \frac{1}{(1+\frac{\lambda}{N})^2-\rho^2}\left[(1+\frac{\lambda}{N}-\rho^2)\hat{\beta}_{ols}^{[1]} + \frac{\lambda}{N}\rho\,\hat{\beta}_{ols}^{[2]}\right]$$

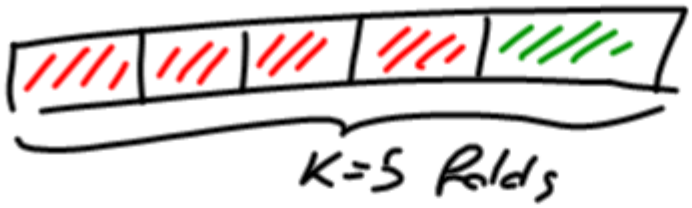
$$\hat{\beta}_{ridge}^{[2]} = \frac{1}{(1+\frac{\lambda}{N})^2-\rho^2}\left[\frac{\lambda}{N}\rho\,\hat{\beta}_{ols}^{[1]} + (1+\frac{\lambda}{N}-\rho^2)\hat{\beta}_{ols}^{[2]}\right]$$

c) Lasso part : exploit variable selection property of LASSO (i.e., remove irrelevant predictors)

Ridge part : better handle of correlation structures in X

Exercise 3

a) K-fold cross-validation



training set
validation set

1) for each value of λ, $\hat{\beta}^{-K}_{lasso}(\lambda)$ training

$k=1$

$$PECV_k(\lambda) = L\left(y^{[k]}, X^{[k]}\hat{\beta}^{-k}_{lasso}(\lambda)\right)$$ validation

2) $k=2$

⋮

final) $$PECV_{tot}(\lambda) = \sum_{k=1}^{K} PECV_k(\lambda)$$

$$\lambda^* = \arg\min_{\lambda}\{PECV_{tot}(\lambda)\}$$

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 estimator to modifications of the data to minimize a loss function"
- iterative procedure

Bagging: bootstrap samples $x_i^* = (x_{i1}^*, x_{i2}^*, \ldots, x_{iN}^*)$

$\hat{\beta}_b$ on x_i^*

$$\hat{\beta}_{BAGGING} = \frac{1}{B}\sum_{b=1}^{B} \hat{\beta}_b$$

b) not in the program this year

c) point a) apply the weak estimator on the reweighted observations

point d) $\alpha_m \, \mathbb{1}[y_i \neq G_m(x_i)]$

point e) $\sum_{m=1}^{m\text{-step}} \alpha_m G_m(x)$