STK-IN4300
Statistical Learning Methods in Data Science

Riccardo De Bin

debin@math.uio.no

Outline of the lecture

- Model Assessment and Selection
  - Cross-Validation
  - Bootstrap Methods
- Methods using Derived Input Directions
  - Principal Component Regression
  - Partial Least Squares
- Shrinkage Methods
  - Ridge Regression

Cross-Validation: k-fold cross-validation

The cross-validation aims at estimating the expected test error,

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

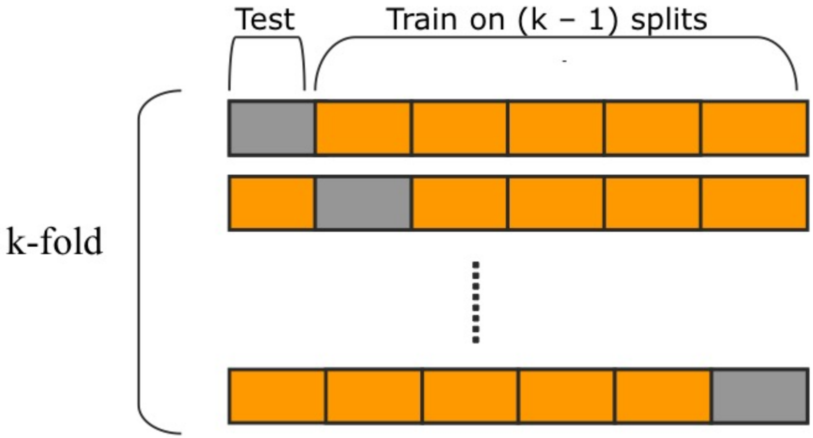
- with enough data, we can split them in a training and test set;
- since usually it is not the case, we mimic this split by using the limited amount of data we have,
  - split data in K folds $\mathcal{F}_1, \dots, \mathcal{F}_K$, approximatively same size;
  - use, in turn, $K-1$ folds to train the model (derive $\hat{f}^{-k}(X)$);
  - evaluate the model in the remaining fold,

$$CV(\hat{f}^{-k}) = \frac{1}{|\mathcal{F}_k|} \sum_{i \in \mathcal{F}_k} L(y_i, \hat{f}^{-k}(x_i))$$

  - estimate the expected test error as an average,

$$CV(\hat{f}) = \frac{1}{K} \sum_{k=1}^{K} \frac{1}{|\mathcal{F}_k|} \sum_{i \in \mathcal{F}_k} L(y_i, \hat{f}^{-k}(x_i)) \overset{|\mathcal{F}_k| = \frac{N}{K}}{=} \frac{1}{N} \sum_{i=1}^{N} L(y_i, \hat{f}^{-k}(x_i)).$$

Cross-Validation: k-fold cross-validation



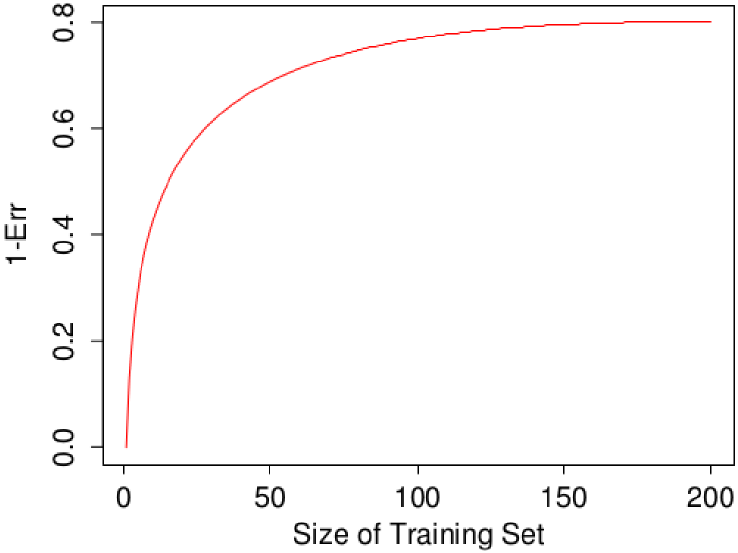
(figure from `http://qingkaikong.blogspot.com/2017/02/machine-learning-9-more-on-artificial.html`)

Cross-Validation: choice of K

How to choose K?

- there is no a clear solution;
- bias-variance trade-off:
    - smaller the K, smaller the variance (but larger bias);
    - larger the K, smaller the bias (but larger variance);
    - extreme cases:
        - $K = 2$, half observations for training, half for testing;
        - $K = N$, leave-one-out cross-validation (LOOCV);
    - LOOCV estimates the expected test error approximatively unbiased;
    - LOOCV has very large variance (the "training sets" are very similar to one another);
- usual choices are $K = 5$ and $K = 10$.

Cross-Validation: choice of K



Cross-Validation: further aspects

If we want to select a tuning parameter (e.g., no. of neighbours)

- train $\hat{f}^{-k}(X, \alpha)$ for each α;
- compute $CV(\hat{f}, \alpha) = \frac{1}{K}\sum_{k=1}^{K} \frac{1}{|\mathcal{F}_k|}\sum_{i \in \mathcal{F}_k} L(y_i, \hat{f}^{-k}(x_i, \alpha))$;
- obtain $\hat{\alpha} = \text{argmin}_\alpha CV(\hat{f}, \alpha)$.

The generalized cross-validation (GCV),

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

- is a convenient approximation of LOOCV for linear fitting under square loss;
- has computational advantages.

Cross-Validation: the wrong and the right way to do cross-validation

Consider the following procedure:

1. find a subset of good (= most correlated with the outcome) predictors;
2. use the selected predictors to build a classifier;
3. use cross-validation to compute the prediction error.

Practical example (see R file):

- generated X, an $[N = 50] \times [p = 5000]$ data matrix;
- generate independently y_i, $i = 1, \dots, 50$, $y_i \in \{0, 1\}$;
- the true error test is 0.50;
- implementing the procedure above. What does it happen?

Cross-Validation: the wrong and the right way to do cross-validation

Why it is not correct?

- Training and test sets are **NOT** independent!
- observations on the test sets are used twice.

Correct way to proceed:

- divide the sample in K folds;
- both perform variable selection and build the classifier using observations from $K - 1$ folds;
  - possible choice of the tuning parameter included;
- compute the prediction error on the remaining fold.

Bootstrap Methods: bootstrap

IDEA: generate pseudo-samples from the empirical distribution function computed on the original sample;

- by sampling with replacement from the original dataset;
- mimic new experiments.

Suppose $Z = \{\underbrace{(x_1, y_1)}_{z_1}, \dots, \underbrace{(y_N, x_N)}_{z_N}\}$ be the training set:

- by sampling with replacement, $Z_1^* = \{\underbrace{(y_1^*, x_1^*)}_{z_1^*}, \dots, \underbrace{(y_N^*, x_N^*)}_{z_N^*}\}$;
- ... ...
- by sampling with replacement, $Z_B^* = \{\underbrace{(y_1^*, x_1^*)}_{z_1^*}, \dots, \underbrace{(y_N^*, x_N^*)}_{z_N^*}\}$;
- use the B bootstrap samples $Z_1^*, \dots, Z_B^*$ to estimate any aspect of the distribution of a map $S(Z)$.

Bootstrap Methods: bootstrap

For example, to estimate the variance of $S(Z)$,

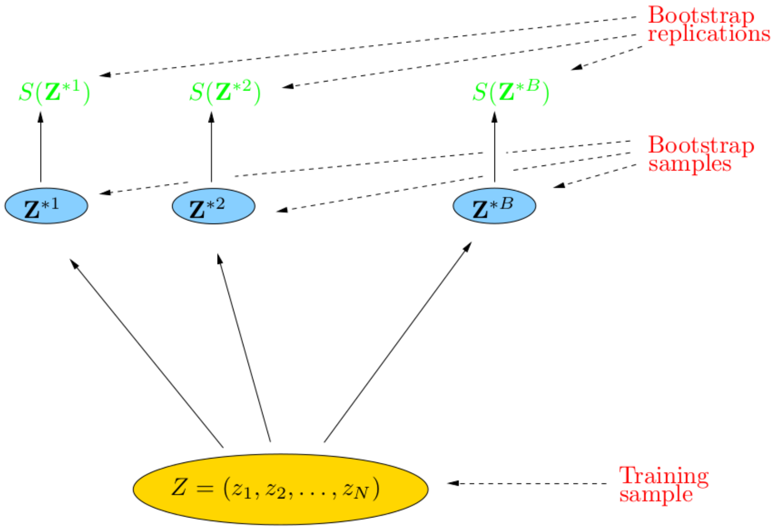
$$\widehat{\text{Var}}[S(Z)] = \frac{1}{B-1}\sum_{b=1}^{B}(S(Z_b^*) - \bar{S}^*)^2$$

where $\bar{S}^* = \frac{1}{B}\sum_{b=1}^{B} S(Z_b^*)$.

Note that:

- $\widehat{\text{Var}}[S(Z)]$ is the Monte Carlo estimate of $\text{Var}[S(Z)]$ under sampling from the empirical distribution $\hat{F}$.

Bootstrap Methods: bootstrap



Bootstrap Methods: estimate prediction error

Very simple:

- generate B bootstrap samples $Z_1^*, \dots, Z_B^*$;
- apply the prediction rule to each bootstrap sample to derive the predictions $\hat{f}_b^*(x_i)$, $b = 1, \dots, B$;
- compute the error for each point, and take the average,

$$\widehat{\text{Err}}_{\text{boot}} = \frac{1}{B} \sum_{b=1}^{B} \frac{1}{N} \sum_{i=1}^{N} L(y_i, \hat{f}_b^*(x_i)).$$

Is it correct? **NO!!!**

Again, training and test set are **NOT** independent!

Bootstrap Methods: example

Consider a classification problem:

- two classes with the same number of observations;
- predictors and class label independent $\Rightarrow \text{Err} = 0.5$.

Using the 1-nearest neighbour:

- if $y_i \in Z_b^* \rightarrow \hat{\text{Err}} = 0$;
- if $y_i \notin Z_b^* \rightarrow \hat{\text{Err}} = 0.5$;

Therefore,

$$\widehat{\text{Err}}_{\text{boot}} = 0 \times \Pr[Y_i \in Z_b^*] + 0.5 \times \underbrace{\Pr[Y_i \notin Z_b^*]}_{0.368} = 0.184$$

Bootstrap Methods: why 0.368

$\Pr[$observation i does not belong to the boostrap sample $b] = 0.368$

Since

$$\Pr[Z^*_{b[j]} \neq y_i] = \frac{N-1}{N},$$

is true for each position $[j]$, then

$$\Pr[Y_i \notin Z^*_b] = \left(\frac{N-1}{N}\right)^N \xrightarrow{N\to\infty} e^{-1} \approx 0.368,$$

Consequently,

$$\Pr[\text{observation } i \text{ is in the boostrap sample } b] \approx 0.632.$$

Bootstrap Methods: correct estimate prediction error

Note:

- each bootstrap sample has N observations;
- some of the original observations are included more than once;
- some of them (in average, $0.368N$) are not included at all;
  - these are not used to compute the predictions;
  - they can be used as a test set,

$$\widehat{\text{Err}}^{(1)} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{|C_{[-i]}|}\sum_{b\in C_{[-i]}} L(y_i, \hat{f}_b^*(x_i))$$

where $C_{[-i]}$ is the set of indeces of the bootstrap samples which do not contain the observation i and $|C_{[-i]}|$ denotes its cardinality.

Bootstrap Methods: 0.632 bootstrap

Issue:

- the average number of unique observations in the bootstrap sample is $0.632N \rightarrow$ not so far from $0.5N$ of 2-fold CV;
- similar bias issues of 2-fold CV;
- $\widehat{\text{Err}}^{(1)}$ slightly overestimates the prediction error.

To solve this, the 0.632 bootstrap estimator has been developed,

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

- in practice it works well;
- in case of strong overfit, it can break down;
  - consider again the previous classification problem example;
  - with 1-nearest neighbour, $\overline{\text{err}} = 0$;
  - $\widehat{\text{Err}}^{(0.632)} = 0.632\ \widehat{\text{Err}}^{(1)} = 0.632 \times 0.5 = 0.316 \neq 0.5.$

Bootstrap Methods: 0.632+ bootstrap

Further improvement, 0.632+ bootstrap:

- based on the no-information error rate γ;
- γ takes into account the amount of overfitting;
- γ is the error rate if predictors and response were independent;
- computed by considering all combinations of x_i and y_i,

$$\hat{\gamma} = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{N} \sum_{i'=1}^{N} L(y_i, \hat{f}(x_{i'})).$$

Bootstrap Methods: 0.632+ bootstrap

The quantity $\hat{\gamma}$ is used to estimate the relative overfitting rate,

$$\hat{R} = \frac{\widehat{\text{Err}}^{(1)} - \overline{\text{err}}}{\hat{\gamma} - \overline{\text{err}}},$$

which is then use in the 0.632+ bootstrap estimator,

$$\widehat{\text{Err}}^{(0.632+)} = (1 - \hat{w})\ \overline{\text{err}} + \hat{w}\ \widehat{\text{Err}}^{(1)},$$

where

$$\hat{w} = \frac{0.632}{1 - 0.368\ \hat{R}}.$$

Methods using Derived Input Directions: summary

- Principal Components Regression
- Partial Least Squares

Principal Component Regression: singular value decomposition

Consider the singular value decomposition (SVD) of the $N \times p$ (standardized) input matrix X,

$$X = UDV^T$$

where:

- U is the $N \times p$ orthogonal matrix whose columns span the column space of X;
- D is a $p \times p$ diagonal matrix, whose diagonal entries $d_1 \geqslant d_2 \geqslant \cdots \geqslant d_p \geqslant 0$ are the singular values of X;
- V is the $p \times p$ orthogonal matrix whose columns span the row space of X.

Principal Component Regression: principal components

Simple algebra leads to

$$X^T X = V D^2 V^T,$$

the eigen decomposition of $X^T X$ (and, up to a constant N, of the sample covariance matrix $S = X^T X/N$).

Using the eigenvectors v_j (columns of V), we can define the principal components of X,

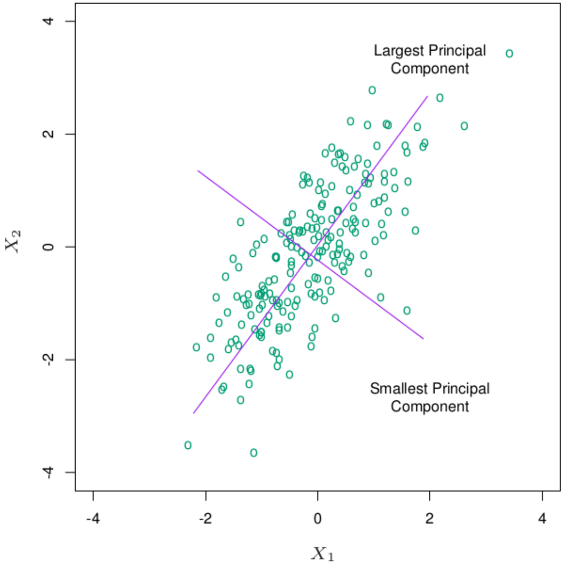
$$z_j = X v_j.$$

- the first principal component z_1 has the largest sample variance (among all linear combinations of the columns of X);

$$\text{Var}(z_1) = \text{Var}(X v_1) = \frac{d_1^2}{N}$$

- since $d_1 \geqslant \dots \geqslant d_p \geqslant 0$, then $\text{Var}(z_1) \geqslant \dots \geqslant \text{Var}(z_p)$.

Principal Component Regression: principal components



Principal Component Regression: principal components

Principal component regression (PCR):

- use $M \leqslant p$ principal components as input;
- regress y on $z_1, \dots, z_M$;
- since the principal components are orthogonal,

$$\hat{y}_{\mathsf{pcr}}(M) = \bar{y} + \sum_{m=1}^{M} \hat{\theta}_m z_m,$$

where $\hat{\theta}_m = \langle z_m, y\rangle / \langle z_m, z_m\rangle$.

Since z_m are linear combinations of x_j,

$$\hat{\beta}_{\mathsf{pcr}}(M) = \sum_{m=1}^{M} \hat{\theta}_m v_m.$$

Principal Component Regression: remarks

Note that:

- PCR can be used in high-dimensions, as long as $M < n$;
- idea: remove the directions with less information;
- if $M = N$, $\hat{\beta}_{\text{pcr}}(M) = \hat{\beta}_{\text{OLS}}$;
- M is a tuning parameter, may be chosen via cross-validation;
- shrinkage effect (clearer later);

- principal component are scale dependent, it is important to standardize X!

Partial Least Squares: idea

Partial least square (PLS) is based on an idea similar to PCR:

- construct a set of linear combinations of X;
- PCR only uses X, ignoring y;
- in PLS we want to also consider the information on y;
- as for PCR, it is important to first standardize X.

Partial Least Squares: algorithm

1. standardize each x_j, set $\hat{y}^{[0]} = \bar{y}$ and $x_j^{[0]} = x_j$;
2. For $m = 1, 2, \dots, p$,
   (a) $z_m = \sum_{j=1}^{p} \hat{\varphi}_{mj} x_j^{[m-1]}$, with $\hat{\varphi}_{mj} = \langle x_j^{[m-1]}, y \rangle$;
   (b) $\hat{\theta}_m = \langle z_m, y \rangle / \langle z_m, z_m \rangle$;
   (c) $\hat{y}^{[m]} = \hat{y}^{[m-1]} + \hat{\theta} z_m$;
   (d) orthogonalize each $x_j^{[m-1]}$ with respect to z_m,

$$x_j^{[m]} = x_j^{[m-1]} - \left(\frac{\langle z_m, x_j^{[m-1]} \rangle}{\langle z_m, z_m \rangle} \right) z_m, \quad j = 1, \dots, p;$$

3. output the sequence of fitted vectors $\{\hat{y}^{[m]}\}_1^p$.

Partial Least Squares: step by step

First step:

(a) compute the first PLS direction, $z_1 = \sum_{j=1}^{p} \hat{\varphi}_{1j} x_j$,
  - based on the relation between each x_j and y, $\hat{\varphi}_1 = \langle x_j, y \rangle$;

(b) estimate the related regression coefficient, $\hat{\theta}_1 = \frac{\langle z_1, y \rangle}{\langle z_1, z_1 \rangle} = \frac{\overline{z_1 y}}{\overline{z_1^2}}$;

(c) model after the first iteration: $\hat{y}^{[1]} = \bar{y} + \hat{\theta}_1 z_1$;

(d) orthogonalize $x_1, \dots, x_p$ w.r.t. z_1, $x_j^{[2]} = x_j - \left(\frac{\langle z_1, x_j \rangle}{\langle z_1, z_1 \rangle} \right) z_1$;

We are now ready for the second step . . .

Partial Least Squares: step by step

…using $x_j^{[2]}$ instead of x_j:

(a) compute the second PLS direction, $z_2 = \sum_{j=1}^p \hat{\varphi}_{2j} x_j^{[2]}$,
   - based on the relation between each $x_j^{[2]}$ and y, $\hat{\varphi}_2 = \langle x_j^{[2]}, y\rangle$;

(b) estimate the related regression coefficient, $\hat{\theta}_2 = \frac{\langle z_2, y\rangle}{\langle z_2, z_2\rangle}$;

(c) model after the second iteration: $\hat{y}^{[2]} = \bar{y} + \hat{\theta}_1 z_1 + \hat{\theta}_2 z_2$;

(d) orthogonalize $x_1^{[2]}, \dots, x_p^{[2]}$ w.r.t. z_2,
$$x_j^{[2]} = x_j^{[2]} - \left(\frac{\langle z_2, x_j^{[2]}\rangle}{\langle z_2, z_2\rangle}\right) z_2;$$

and so on, until the $M \leqslant p$ step $\rightarrow$ M derived inputs.

Partial Least Squares: PLS versus PCR

Differences:

PCR the derived input directions are the principal components of X, constructed by looking at the variability of X;

PLS the input directions take into consideration both the variability of X and the correlation between X and y.

Mathematically:

PCR $\max_\alpha \text{Var}(X\alpha)$, s.t.

- $||\alpha|| = 1$ and $\alpha^T S v_\ell = 0$, $\ell = 1, \dots, M-1$;

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

- $||\alpha|| = 1$ and $\alpha^T S \varphi_\ell = 0$, $\forall \ell < M$.

In practice, the variance tends to dominate $\rightarrow$ similar results!

Ridge Regression: historical notes

When two predictors are strongly correlated $\rightarrow$ collinearity;

- in the extreme case of linear dependency $\rightarrow$ super-collinearity;
- in the case of super-collinearity, $X^T X$ is not invertible (not full rank);

Hoerl & Kennard (1970): $X^T X \rightarrow X^T X + \lambda I_p$, where $\lambda > 0$ and

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

With $\lambda > 0$, $(X^T X + \lambda I_p)^{-1}$ exists.

Ridge Regression: estimator

Substituting $X^T X$ with $X^T X + \lambda I_p$ in the LS estimator,

$$\hat{\beta}_{\text{ridge}}(\lambda) = (X^T X + \lambda I_p)^{-1} X^T y.$$

Alternatively, the ridge estimator can be seen as the minimizer of

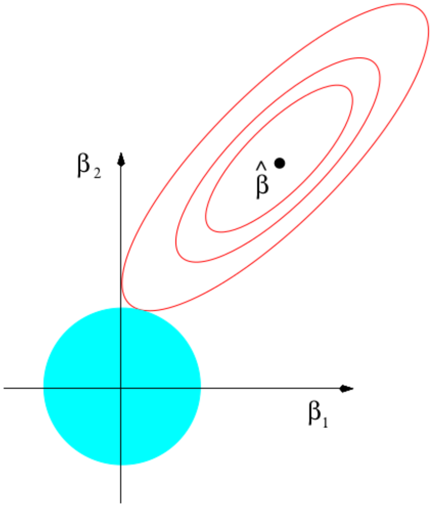
$$\sum_{i=1}^{N} (y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij})^2,$$

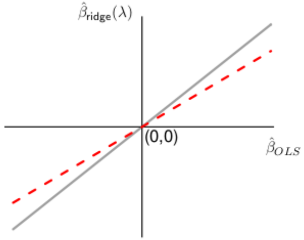
subject to $\sum_{j=1}^{p} \beta_j^2 \leqslant t$.

Which is the same as

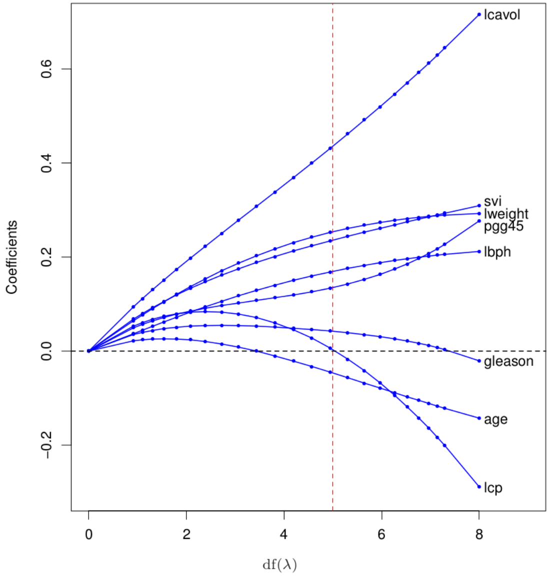
$$\hat{\beta}_{\text{ridge}}(\lambda) = \text{argmin}_{\beta} \left\{ \sum_{i=1}^{N} (y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij})^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \right\}.$$

Ridge Regression: visually





Ridge Regression: visually



Ridge Regression: remarks

Note:

- ridge solution is not equivariant under scaling $\rightarrow X$ must be standardized before applying the minimizer;
- the intercept is not involved in the penalization;
- Bayesian interpretation:
  - $Y_i \sim N(\beta_0 + x_i^T \beta, \sigma^2)$;
  - $\beta \sim N(0, \tau^2)$;
  - $\lambda = \sigma^2 / \tau^2$;
  - $\hat{\beta}_{\text{ridge}}(\lambda)$ as the posterior mean.

Ridge Regression: bias

$$\begin{aligned}
E[\hat\beta_{\text{ridge}}(\lambda)] &= E[(X^TX + \lambda I_p)^{-1}X^Ty] \\
&= E[(I_p + \lambda(X^TX)^{-1})^{-1}\underbrace{(X^TX)^{-1}X^Ty}_{\hat\beta_{\text{LS}}}] \\
&= \underbrace{(I_p + \lambda(X^TX)^{-1})^{-1}}_{w_\lambda}E[\hat\beta_{\text{LS}}] \\
&= w_\lambda\beta \implies E[\hat\beta_{\text{ridge}}(\lambda)] \neq \beta \text{ for } \lambda > 0.
\end{aligned}$$

- $\lambda \to 0$, $E[\hat\beta_{\text{ridge}}(\lambda)] \to \beta$;
- $\lambda \to \infty$, $E[\hat\beta_{\text{ridge}}(\lambda)] \to 0$ (without intercept);
- due to correlation, $\lambda_a > \lambda_b \nRightarrow |\hat\beta_{\text{ridge}}(\lambda)| > |\hat\beta_{\text{ridge}}(\lambda)|$.

Ridge Regression: variance

Consider the variance of the ridge estimator,

$$\begin{aligned}\mathsf{Var}[\hat\beta_{\mathsf{ridge}}(\lambda)] &= \mathsf{Var}[w_\lambda \hat\beta_{\mathsf{LS}}]\\ &= w_\lambda \mathsf{Var}[\hat\beta_{\mathsf{LS}}] w_\lambda^T\\ &= \sigma^2 w_\lambda (X^TX)^{-1} w_\lambda^T.\end{aligned}$$

Then,

$$\begin{aligned}&\mathsf{Var}[\hat\beta_{\mathsf{LS}}] - \mathsf{Var}[\hat\beta_{\mathsf{ridge}}(\lambda)] = \sigma^2\left[(X^TX)^{-1} - w_\lambda (X^TX)^{-1} w_\lambda^T\right]\\ &\quad= \sigma^2 w_\lambda \left[(I_p + \lambda(X^TX)^{-1})(X^TX)^{-1}(I_p + \lambda(X^TX)^{-1})^T - (X^TX)^{-1}\right] w_\lambda^T\\ &\quad= \sigma^2 w_\lambda \left[((X^TX)^{-1} + 2\lambda(X^TX)^{-2} + \lambda^2(X^TX)^{-3}) - (X^TX)^{-1}\right] w_\lambda^T\\ &\quad= \sigma^2 w_\lambda \left[2\lambda(X^TX)^{-2} + \lambda^2(X^TX)^{-3})\right] w_\lambda^T > 0\end{aligned}$$

(since all terms are quadratic and therefore positive)

$$\implies \mathsf{Var}[\hat\beta_{\mathsf{ridge}}(\lambda)] \preceq \mathsf{Var}[\hat\beta_{\mathsf{LS}}]$$

Ridge Regression: degrees of freedom

Note that the ridge solution is a linear combination of y, as the least squares one:

- $\hat{y}_{\text{LS}} = \underbrace{X(X^T X)^{-1} X^T}_{H} y \longrightarrow df = \text{trace}(H) = p$;
- $\hat{y}_{\text{ridge}} = \underbrace{X(X^T X + \lambda I_p)^{-1} X^T}_{H_\lambda} y \longrightarrow df(\lambda) = \text{trace}(H_\lambda)$;
    - $\text{trace}(H_\lambda) = \sum_{i=1}^p \frac{d_j^2}{d_j^2 + \lambda}$;
    - d_j is the diagonal element of D in the SVD of X;
    - $\lambda \to 0$, $df(\lambda) \to p$;
    - $\lambda \to \infty$, $df(\lambda) \to 0$.

Ridge Regression: more about shrinkage

Recall the SVD decomposition $X = UDV^T$, and the properties

$$U^TU = I_p = V^TV.$$

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

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

Ridge Regression: more about shrinkage

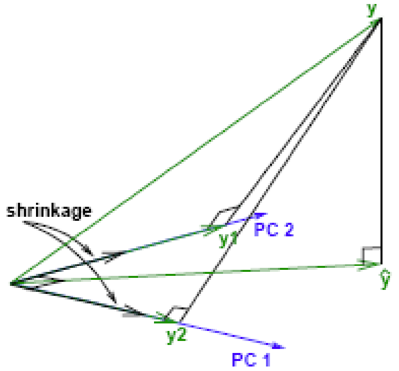
$$\begin{aligned}\hat{\beta}_{\text{ridge}} &= (X^TX + \lambda I_p)^{-1}X^Ty\\ &= (VDU^TUDV^T + \lambda I_p)^{-1}VDU^Ty\\ &= (VD^2V^T + \lambda VV^T)^{-1}VDU^Ty\\ &= V(D^2 + \lambda I_p)^{-1}V^TVDU^Ty\\ &= V(D^2 + \lambda I_p)^{-1}U^Ty\end{aligned}$$

$$\begin{aligned}\hat{y}_{\text{ridge}} &= X\hat{\beta}_{\text{ridge}}\\ &=UDV^TV(D^2 + \lambda I_p)^{-1}U^Ty\\ &=UV^TVD^2(D^2 + \lambda I_p)^{-1}U^Ty\\ &=U\underbrace{D^2(D^2 + \lambda I_p)^{-1}}_{\displaystyle\Big\downarrow\ \sum_{j=1}^{p}\frac{d_j^2}{d_j^2+\lambda}}U^Ty\end{aligned}$$

So:

- small singular values d_j correspond to directions of the column space of X with low variance;
- ridge regression penalizes the most these directions.

Ridge Regression: more about shrinkage



N-dimensional sample space
based on principle components

$\beta_1 = ||y1|| / ||pc1||$ $\qquad \beta_1^{ridge} = \dfrac{||pc1||^2 \beta_1}{||pc1||^2 + \lambda}$

$\beta_2 = ||y2|| / ||pc2||$ $\qquad \beta_2^{ridge} = \dfrac{||pc2||^2 \beta_2}{||pc2||^2 + \lambda}$

(picture from `https://onlinecourses.science.psu.edu/stat857/node/155/`)

References I

Hoerl, A. E. & Kennard, R. W. (1970). Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics* **12**, 55–67.