SD-TSIA : Ridge / Tikhonov

Joseph Salmon
http://josephsalmon.eu
Télécom ParisTech

Outline

Table of Contents

Reminder

$$\boxed{\mathbf{y} = X\boldsymbol{\theta}^\star + \boldsymbol{\varepsilon}}$$

- $\mathbf{y} \in \mathbb{R}^n$ observations
- $X \in \mathbb{R}^{n \times p}$ is the design matrix (column-wise: features)
- $\boldsymbol{\theta}^\star \in \mathbb{R}^p$ is the **true** model parameter we aim at finding
- $\boldsymbol{\varepsilon} \in \mathbb{R}^n$ is the noise

Rem: possibly a supplementary variable is added for the constants

Singular Value Decomposition (SVD)

Theorem Golub and Van Loan (2013)

For any matrix $X \in \mathbb{R}^{n\times p}$, there exists two orthogonal matrices $U = [\mathbf{u}_1, \dots, \mathbf{u}_n] \in \mathbb{R}^{n\times n}$ and $V = [\mathbf{v}_1, \dots, \mathbf{v}_p] \in \mathbb{R}^{p\times p}$, such that

$$U^\top X V = \operatorname{diag}(s_1, \dots, s_{\operatorname{rg}(X)}) = \Sigma \in \mathbb{R}^{n\times p}$$

with $s_1 \geqslant s_2 \geqslant \cdots \geqslant s_{\operatorname{rg}(X)} > 0$, with $\operatorname{rg}(X) = \operatorname{rang}(X)$.

Rem: $\forall i, i' \in [n], j, j' \in [p]$, one has $\mathbf{u}_i^\top \mathbf{u}_{i'} = \delta_{i,i'}$ and $\mathbf{v}_j^\top \mathbf{v}_{j'} = \delta_{j,j'}$

$$X = U\Sigma V^\top \Leftrightarrow X = \sum_{i=1}^{\operatorname{rg}(X)} s_i \mathbf{u}_i \mathbf{v}_i^\top$$

A least squares solution is then:

$$\hat{\boldsymbol{\theta}}^{\mathrm{OLS}} = X^+ \mathbf{y} = \sum_{i=1}^{\operatorname{rg}(X)} \frac{1}{s_i} \mathbf{v}_i \mathbf{u}_i^\top \mathbf{y}$$

Numerical instabilities

$$\hat{\boldsymbol{\theta}}^{\mathrm{OLS}} = X^{+}\mathbf{y} = \sum_{i=1}^{\mathrm{rg}(X)} \frac{1}{s_i}\mathbf{v}_i\mathbf{u}_i^{\top}\mathbf{y}$$

If the smallest singular values s_i get close to zero then the numerical solution of the SVD is unstable!

Rem: this drawback is common not only for least squares, but also to other inverse problems (also referred to as "ill posed" in numerical analysis and signal processing)

Normal equations

A solution least squares solution $\boldsymbol{\theta}$ need to satisfy:

$$X^\top X\boldsymbol{\theta} = X^\top \mathbf{y} \Leftrightarrow V\Sigma^\top \Sigma V^\top \boldsymbol{\theta} = V\Sigma^\top U^\top \mathbf{y}$$

and if we look for $\boldsymbol{\theta}$ with the following form: $\boldsymbol{\theta} = V\boldsymbol{\beta}$, this is equivalent to

$$\Sigma^\top \Sigma \boldsymbol{\beta} = \Sigma^\top U^\top \mathbf{y}$$

$\Sigma^\top \Sigma$ diagonal with $r = \mathrm{rang}(X)$ non zero elements that are the s_i^2

$$\Sigma^\top \Sigma = \left[\begin{array}{ccc|c} s_1^2 & & 0 & \\ & \ddots & & \Large 0 \\ 0 & & s_r^2 & \\ \hline & \Large 0 & & \Large 0 \end{array}\right] \in \mathbb{R}^{p \times p}$$

Normal equations (continued)

Regularized alternative : solve normal equations where

$$\left[\begin{array}{ccc|c} s_1^2 & & 0 & \\ & \ddots & & \Large 0 \\ 0 & & s_r^2 & \\ \hline & \Large 0 & & \Large 0 \end{array}\right] \text{ replaced by } \left[\begin{array}{ccc|c} s_1^2 & & 0 & \\ & \ddots & & \Large 0 \\ 0 & & s_r^2 & \\ \hline & \Large 0 & & \Large 0 \end{array}\right] + \lambda \operatorname{Id}_p$$

It can be re-written by:

$$(\lambda \operatorname{Id}_p + \Sigma^\top \Sigma)\boldsymbol{\beta} = \Sigma^\top U^\top \mathbf{y}$$

i.e.,we add a small $\lambda > 0$ to all the eigen values of $X^\top X$, λbeing called the **regularization parameter**

$$\boldsymbol{\beta} = (\lambda \operatorname{Id}_p + \Sigma^\top \Sigma)^{-1} \Sigma^\top U^\top \mathbf{y}$$

and hence

$$\boldsymbol{\theta} = V(\lambda \operatorname{Id}_p + \Sigma^\top \Sigma)^{-1} \Sigma^\top U^\top \mathbf{y}$$

Ridge : explicit formulation

With the SVD, the following equation simplifies:

$$\boldsymbol{\theta} = V(\lambda \operatorname{Id}_p + \Sigma^\top \Sigma)^{-1} \Sigma^\top U^\top \mathbf{y}$$

This gives a simple formulation for the *Ridge* estimator

$$\boxed{\hat{\boldsymbol{\theta}}_\lambda^{\mathrm{rdg}} = (\lambda \operatorname{Id}_p + X^\top X)^{-1} X^\top \mathbf{y}}$$

Reminder: under the full rank hypothesis $\hat{\boldsymbol{\theta}}^{\mathrm{OLS}} = (X^\top X)^{-1} X^\top \mathbf{y}$

Rem:

$$\lim_{\lambda \to 0^+} \hat{\boldsymbol{\theta}}_\lambda^{\mathrm{rdg}} = \hat{\boldsymbol{\theta}}^{\mathrm{OLS}}$$

$$\lim_{\lambda \to +\infty} \hat{\boldsymbol{\theta}}_\lambda^{\mathrm{rdg}} = 0 \in \mathbb{R}^p$$

Kernel trick

Kernel trick: According to whether $n > p$ or $n \leqslant p$, it is more suitable to consider the following equivalent formulation of the Ridge estimator:

$$\boxed{X^\top (XX^\top + \lambda \,\mathrm{Id}_n)^{-1} \mathbf{y} = (X^\top X + \lambda \,\mathrm{Id}_p)^{-1} X^\top \mathbf{y}}$$

- LHS: inverse/solve an $n \times n$ matrix
- RHS: inverse/solve an $p \times p$ matrix

Rem: this property is also useful for kernel method such as SVM (*cf.machine learning* course)

Exo: Show the kernel trick using the SVD of X

Ridge / Tikhonov : penalized definition

$$\boxed{\hat{\boldsymbol{\theta}}_{\lambda}^{\mathrm{rdg}} = \mathop{\arg\min}_{\boldsymbol{\theta} \in \mathbb{R}^p} \left(\underbrace{\|\mathbf{y} - X\boldsymbol{\theta}\|_2^2}_{\text{data fitting}} + \underbrace{\lambda \|\boldsymbol{\theta}\|_2^2}_{\text{regularization}} \right)}$$

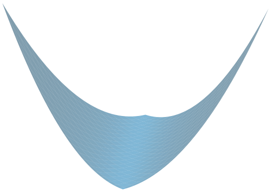
- Note that the *Ridge* estimator is **unique** for any fixed $\lambda > 0$
- We recover the limiting cases:

$$\lim_{\lambda \to 0} \hat{\boldsymbol{\theta}}_{\lambda}^{\mathrm{rdg}} = \hat{\boldsymbol{\theta}}^{\mathrm{OLS}} \text{(solution with smallest } \|\cdot\|_2 \text{ norm)}$$

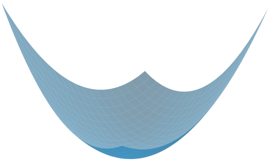
$$\lim_{\lambda \to +\infty} \hat{\boldsymbol{\theta}}_{\lambda}^{\mathrm{rdg}} = 0 \in \mathbb{R}^p$$

- Link between the two formulation thanks to the first order conditions: for $f(\boldsymbol{\theta}) = \|\mathbf{y} - X\boldsymbol{\theta}\|_2^2/2 + \lambda\|\boldsymbol{\theta}\|_2^2/2$
$\nabla f(\boldsymbol{\theta}) = X^\top (X\boldsymbol{\theta} - \mathbf{y}) + \lambda \boldsymbol{\theta} = 0 \Leftrightarrow (X^\top X + \lambda \, \mathrm{Id}_p)\boldsymbol{\theta} = X^\top \mathbf{y}$

Motivation



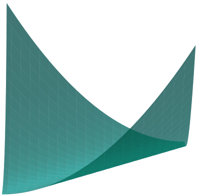
OLS



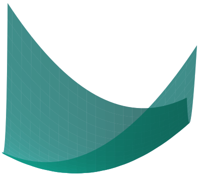
Ridge

Regularize: simplify the problem when ill-conditioned

Motivation



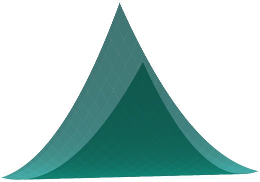
OLS



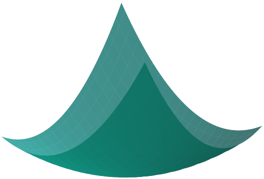
Ridge

Regularize: simplify the problem when ill-conditioned

Motivation



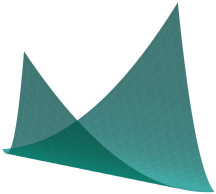
OLS



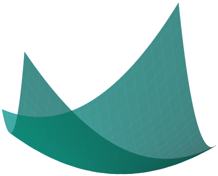
Ridge

Regularize: simplify the problem when ill-conditioned

Motivation



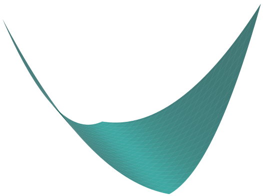
OLS



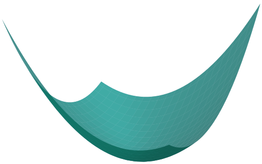
Ridge

Regularize: simplify the problem when ill-conditioned

Motivation



OLS



Ridge

Regularize: simplify the problem when ill-conditioned

Constraint interpretation

A "Lagrangian" formulation is as follows:

$$\boxed{\underset{\boldsymbol{\theta}\in\mathbb{R}^p}{\arg\min} \left(\underbrace{\frac{1}{2}\|\mathbf{y} - X\boldsymbol{\theta}\|_2^2}_{\text{data fitting}} + \underbrace{\frac{\lambda}{2}\|\boldsymbol{\theta}\|_2^2}_{\text{regularization}} \right)}$$

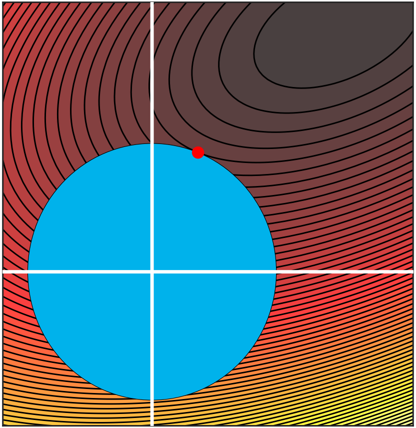
has for a certain $T > 0$ the same solution as:

$$\begin{cases} \underset{\boldsymbol{\theta}\in\mathbb{R}^p}{\arg\min} \|\mathbf{y} - X\boldsymbol{\theta}\|_2^2 \\ \text{s.t. } \|\boldsymbol{\theta}\|_2^2 \leqslant T \end{cases}$$

<u>Rem</u>: the link $T \leftrightarrow \lambda$ is not explicit!

- If $T \to 0$ we recover the null vector: $0 \in \mathbb{R}^p$
- If $T \to \infty$ we recover $\hat{\boldsymbol{\theta}}^{\mathrm{OLS}}$ (un-constrained)

Level lines and and constraints set



Optimization under ℓ_2 constraints

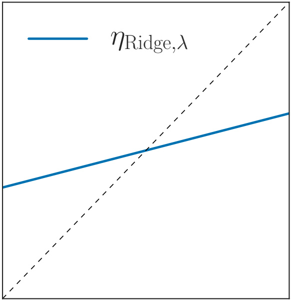
The orthogonal case

Consider the simple case: $X^\top X = \mathrm{Id}_p$

$$\hat{\boldsymbol{\theta}}_\lambda^{\mathrm{rdg}} = (\lambda\, \mathrm{Id}_p + X^\top X)^{-1} X^\top \mathbf{y}$$

$$\hat{\boldsymbol{\theta}}_\lambda^{\mathrm{rdg}} = (\lambda\, \mathrm{Id}_p + \mathrm{Id}_p)^{-1} X^\top \mathbf{y} = \frac{1}{\lambda+1} X^\top \mathbf{y}$$

$$\hat{\mathbf{y}} = \frac{1}{\lambda+1}\mathbf{y} = \left(\eta_{\mathrm{rdg},\lambda}(\mathbf{y}_i)\right)_{i=1,\dots,n}$$



Rem: the real function $\eta_{\mathrm{rdg},\lambda}$ is a linear contraction (shrinkage)

Associated prediction

From the *Ridge* coefficient:

$$\hat{\boldsymbol{\theta}}_\lambda^{\mathrm{rdg}} = (\lambda \operatorname{Id}_p + X^\top X)^{-1} X^\top \mathbf{y}$$

the associated prediction is given by:

$$\hat{\mathbf{y}} = X \hat{\boldsymbol{\theta}}_\lambda^{\mathrm{rdg}} = X(\lambda \operatorname{Id}_p + X^\top X)^{-1} X^\top \mathbf{y}$$

<u>Rem</u>: the estimator $\hat{\mathbf{y}}$ is linear w.r.t. $\mathbf{y}$

<u>Rem</u>: reminding $X = \sum_{i=1}^{\mathrm{rg}(X)} s_i \mathbf{u}_i \mathbf{v}_i^\top$, (SVD) the matrix $H_\lambda := X(\lambda \operatorname{Id}_p + X^\top X)^{-1} X^\top = \sum_{j=1}^{\mathrm{rg}(X)} \frac{s_j^2}{s_j^2 + \lambda} \mathbf{u}_j \mathbf{u}_j^\top$ is the equivalent of the **hat matrix**
If $\lambda \neq 0$, we do not have $H_\lambda^2 = H_\lambda = \sum_{j=1}^{\mathrm{rg}(X)} \mathbf{u}_j \mathbf{u}_j^\top$ anymore, so H_λ is not a projection (in general).

N remarks

Reminder: normalizing the p features the same way is necessary if you want the penalty to be similar for all features:

- center the observation and the features $\Rightarrow$ no coefficient for the constants (hence no constraint on it)
- not centering features $\Rightarrow$ do not put constraint on the constant feature (bias/intercept)

$$\hat{\boldsymbol{\theta}}_\lambda^{\mathrm{rdg}} = \operatorname*{arg\,min}_{\boldsymbol{\theta}\in\mathbb{R}^p} \|\mathbf{y} - X\boldsymbol{\theta} - \theta_0 \mathbf{1}_n\|_2^2 + \lambda \sum_{j=1}^p \theta_j^2$$

Alternative (without normalization): change the penalty in

$$\operatorname*{arg\,min}_{\boldsymbol{\theta}\in\mathbb{R}^p} \|\mathbf{y} - X\boldsymbol{\theta}\|_2^2 + \lambda \sum_{j=1}^p \alpha_j \boldsymbol{\theta}_j^2 \quad (\textit{e.g.},\ \alpha_j = \|\mathbf{x}_j\|_2^2)$$

Rem:for cross validation one can use $\frac{\|\mathbf{y}-X\boldsymbol{\theta}\|_2^2}{2n}$ rather than $\frac{\|\mathbf{y}-X\boldsymbol{\theta}\|_2^2}{2}$ as the data fitting part

Normalization (continued)

Consider the estimator Ridge with variables $X' = [\mathbf{x}'_1, \ldots, \mathbf{x}'_K]$, such that there exist a linear link $\sum_{k=1}^K \mu_k \mathbf{x}'_k = 1$ (*e.g.*,qualitative variable):

$$\hat{\boldsymbol{\theta}}_\lambda^{\mathrm{rdg}} = \mathop{\arg\min}_{\boldsymbol{\theta}\in\mathbb{R}^p, \boldsymbol{\theta}'\in\mathbb{R}^K} \|\mathbf{y} - X\boldsymbol{\theta} - X'\boldsymbol{\theta}' - \theta_0 \mathbf{1}_n\|_2^2 + \lambda \sum_{j=1}^p \theta_j^2 + \lambda \sum_{k=1}^K \theta_k'^2$$

is equivalent to solve :

$$\hat{\boldsymbol{\theta}}_\lambda^{\mathrm{rdg}} = \mathop{\arg\min}_{\substack{\boldsymbol{\theta}\in\mathbb{R}^p, \boldsymbol{\theta}'\in\mathbb{R}^K \\ \sum_{k=1}^K \theta'_k = 0}} \|\mathbf{y} - X\boldsymbol{\theta} - X'\boldsymbol{\theta}' - \theta_0 \mathbf{1}_n\|_2^2 + \lambda \sum_{j=1}^p \theta_j^2 + \lambda \sum_{k=1}^K \theta_k'^2$$

Exo: proof ; help: *cf.*Park (2006), page 35

General form of the bias

Under Additive White Gaussian Noise (AWGN) assumption $\mathbf{y} = X\boldsymbol{\theta}^\star + \boldsymbol{\varepsilon}$ with $\mathbb{E}(\boldsymbol{\varepsilon}) = 0$:

$$\begin{aligned}
\mathbb{E}(\hat{\boldsymbol{\theta}}_\lambda^{\mathrm{rdg}}) =& \mathbb{E}[(\lambda \operatorname{Id}_p + X^\top X)^{-1} X^\top \mathbf{y}] \\
=& \mathbb{E}[(\lambda \operatorname{Id}_p + X^\top X)^{-1} X^\top X \boldsymbol{\theta}^\star + (\lambda \operatorname{Id}_p + X^\top X)^{-1} X^\top \boldsymbol{\varepsilon}] \\
=& (\lambda \operatorname{Id}_p + X^\top X)^{-1} X^\top X \boldsymbol{\theta}^\star \\
=& \sum_{i=1}^{\mathrm{rg}(X)} \frac{s_i^2}{s_i^2 + \lambda} \mathbf{v}_i \mathbf{v}_i^\top \boldsymbol{\theta}^\star
\end{aligned}$$

Rem: one recovers $\mathbb{E}(\hat{\boldsymbol{\theta}}^{\mathrm{OLS}}) = \sum_{i=1}^{\mathrm{rg}(X)} \mathbf{v}_i \mathbf{v}_i^\top \boldsymbol{\theta}^\star$ when $\lambda \to 0$

Rem: the bias is $\mathbb{E}(\hat{\boldsymbol{\theta}}_\lambda^{\mathrm{rdg}}) - \boldsymbol{\theta}^\star = -\lambda (X^\top X + \lambda \operatorname{Id}_p)^{-1} \boldsymbol{\theta}^\star$

Variance in the general case

Under the assumption $\mathbb{E}(\boldsymbol{\varepsilon}) = 0$, and with a homoscedastic model: $\mathbb{E}(\boldsymbol{\varepsilon}\boldsymbol{\varepsilon}^\top) = \sigma^2 \operatorname{Id}_n$

Variance / Covariance

$$V_\lambda^{\mathrm{rdg}} = \mathbb{E}\left((\hat{\boldsymbol{\theta}}_\lambda^{\mathrm{rdg}} - \mathbb{E}(\hat{\boldsymbol{\theta}}_\lambda^{\mathrm{rdg}}))(\hat{\boldsymbol{\theta}}_\lambda^{\mathrm{rdg}} - \mathbb{E}(\hat{\boldsymbol{\theta}}_\lambda^{\mathrm{rdg}})^\top \right)$$

Explicit computation:

$$\begin{aligned}
V_\lambda^{\mathrm{rdg}} =& \mathbb{E}((\lambda \operatorname{Id}_p + X^\top X)^{-1} X^\top \boldsymbol{\varepsilon}\boldsymbol{\varepsilon}^\top X (\lambda \operatorname{Id}_p + X^\top X)^{-1}) \\
&= (\lambda \operatorname{Id}_p + X^\top X)^{-1} X^\top \mathbb{E}(\boldsymbol{\varepsilon}\boldsymbol{\varepsilon}^\top) X (\lambda \operatorname{Id}_p + X^\top X)^{-1} \\
&= \sigma^2 (\lambda \operatorname{Id}_p + X^\top X)^{-2} X^\top X \qquad \text{(matrix commute here)} \\
&= \sum_{i=1}^{\operatorname{rg}(X)} \frac{s_i^2 \sigma^2}{(s_i^2 + \lambda)^2} \mathbf{v}_i \mathbf{v}_i^\top
\end{aligned}$$

<u>Rem</u>: one recovers $V^{\mathrm{OLS}} = \sum_{i=1}^{\operatorname{rg}(X)} \frac{\sigma^2}{s_i^2} \mathbf{v}_i \mathbf{v}_i^\top$ when $\lambda \to 0$

<u>Rem</u>: one find a null variance when $\lambda \to \infty$

Prediction risk

Homoscedastic assumption: $\mathbb{E}(\boldsymbol{\varepsilon}\boldsymbol{\varepsilon}^\top) = \sigma^2 \,\mathrm{Id}_n$

Quadratic prediction risk $\mathbb{E}\|X\boldsymbol{\theta}^\star - X\hat{\boldsymbol{\theta}}_\lambda^{\mathrm{rdg}}\|^2$

Under the Homoscedastic assumption:

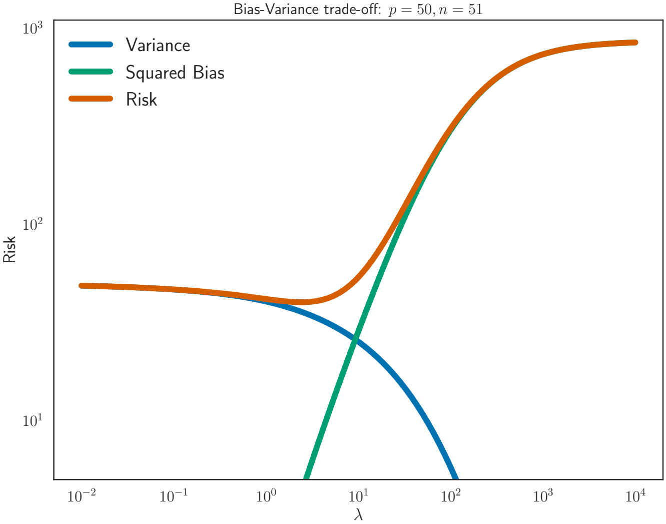
$$R_{\mathrm{pred}}(\boldsymbol{\theta}^\star, \hat{\boldsymbol{\theta}}_\lambda^{\mathrm{rdg}}) = \mathbb{E}\left[(\hat{\boldsymbol{\theta}}_\lambda^{\mathrm{rdg}} - \boldsymbol{\theta}^\star)^\top (X^\top X)(\hat{\boldsymbol{\theta}}_\lambda^{\mathrm{rdg}} - \boldsymbol{\theta}^\star)\right]$$

Explicit computation (begins as for OLS):

$$\begin{aligned} R_{\mathrm{pred}}(\boldsymbol{\theta}^\star, \hat{\boldsymbol{\theta}}_\lambda^{\mathrm{rdg}}) &= \mathbb{E}\left[(\hat{\boldsymbol{\theta}}_\lambda^{\mathrm{rdg}} - \boldsymbol{\theta}^\star)^\top (X^\top X)(\hat{\boldsymbol{\theta}}_\lambda^{\mathrm{rdg}} - \boldsymbol{\theta}^\star)\right] \\ &= \mathbb{E}\left[(X(X^\top X + \lambda\,\mathrm{Id}_p)^{-1} X^\top \boldsymbol{\varepsilon})^\top (X(X^\top X + \lambda \mathrm{Id}_p)^{-1} X^\top \boldsymbol{\varepsilon})\right] \\ &\quad + \lambda^2 \boldsymbol{\theta}^{\star\top} (X^\top X + \lambda\,\mathrm{Id}_p)^{-2} \boldsymbol{\theta}^\star \\ &= \sum_{i=1}^{\mathrm{rg}(X)} \frac{s_i^4 \sigma^2}{(s_i^2 + \lambda)^2} + \lambda^2 \boldsymbol{\theta}^{\star\top} (X^\top X + \lambda\,\mathrm{Id}_p)^{-2} \boldsymbol{\theta}^\star \end{aligned}$$

<u>Rem</u>: $\lim_{\lambda \to 0} R_{\mathrm{pred}}(\boldsymbol{\theta}^\star, \hat{\boldsymbol{\theta}}_\lambda^{\mathrm{rdg}}) = \mathrm{rg}(X)\sigma^2$, $\lim_{\lambda \to \infty} R_{\mathrm{pred}}(\boldsymbol{\theta}^\star, \hat{\boldsymbol{\theta}}_\lambda^{\mathrm{rdg}}) = \|X\boldsymbol{\theta}^\star\|_2^2$

Bias / Variance: simulated example

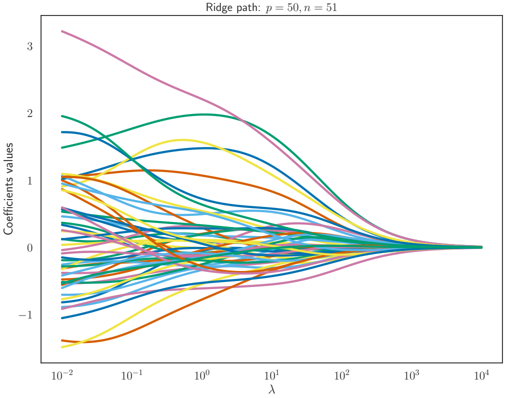


$$X \in \mathbb{R}^{51\times 50}, \boldsymbol{\theta}^\star = (2,2,2,2,2,0,\ldots,0)^\top$$

Table of Contents

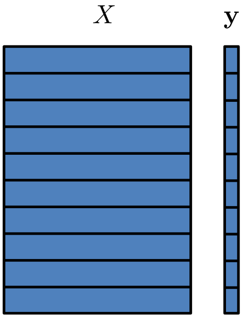
Choosing λ

```
n_features = 50; n_samples = 50
X = np.random.randn(n_samples, n_features)
theta_true = np.zeros([n_features, ])
theta_true[0:5] = 2.
y_true = np.dot(X, theta_true)
y = y_true + 1. * np.random.rand(n_samples,)
```



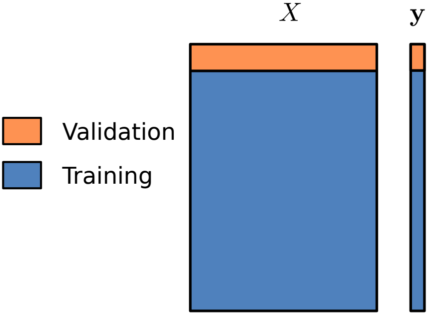
K-fold Cross-Validation ($K = 10$)

- Choose a grid of r λ's to test: $\lambda_1, \ldots, \lambda_r$
- Divide $(X, \mathbf{y})$ into K blocks (sample-wise):



K-fold Cross-Validation ($K = 10$)

- Choose a grid of r λ's to test: $\lambda_1, \ldots, \lambda_r$
- Divide $(X, \mathbf{y})$ into K blocks (sample-wise):

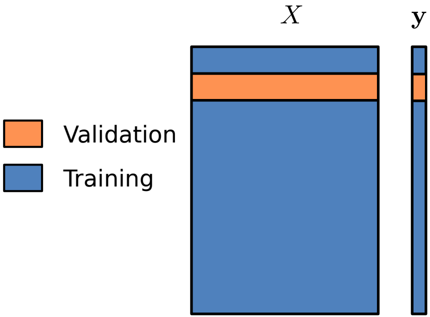


$$k = 1$$

1. Compute with the training part the estimators for $\lambda_1, \ldots, \lambda_r$: $\hat{\boldsymbol{\theta}}^{\lambda_1}, \ldots, \hat{\boldsymbol{\theta}}^{\lambda_r}$
2. Evaluate the (prediction) error $\mathrm{Error}_1^k, \ldots, \mathrm{Error}_r^k$ over the validation part,

K-fold Cross-Validation ($K = 10$)

- Choose a grid of r λ's to test: $\lambda_1, \dots, \lambda_r$
- Divide $(X, \mathbf{y})$ into K blocks (sample-wise):

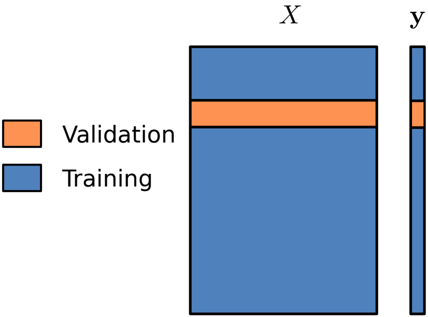


$k = 2$

1. Compute with the training part the estimators for $\lambda_1, \dots, \lambda_r$: $\hat{\boldsymbol{\theta}}^{\lambda_1}, \dots, \hat{\boldsymbol{\theta}}^{\lambda_r}$
2. Evaluate the (prediction) error $\mathrm{Error}_1^k, \dots, \mathrm{Error}_r^k$ over the validation part,

K-fold Cross-Validation ($K = 10$)

- Choose a grid of r λ's to test: $\lambda_1, \ldots, \lambda_r$
- Divide $(X, \mathbf{y})$ into K blocks (sample-wise):

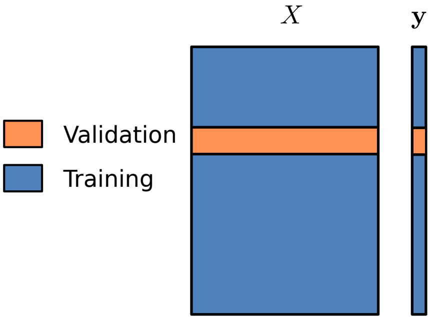


$$k = 3$$

1. Compute with the training part the estimators for $\lambda_1, \ldots, \lambda_r$: $\hat{\boldsymbol{\theta}}^{\lambda_1}, \ldots, \hat{\boldsymbol{\theta}}^{\lambda_r}$
2. Evaluate the (prediction) error $\mathrm{Error}_1^k, \ldots, \mathrm{Error}_r^k$ over the validation part,

K-fold Cross-Validation ($K = 10$)

- Choose a grid of r λ's to test: $\lambda_1, \ldots, \lambda_r$
- Divide $(X, \mathbf{y})$ into K blocks (sample-wise):

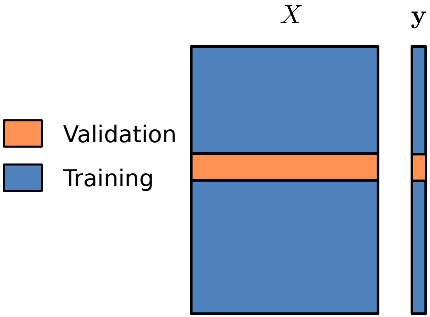


$$k = 4$$

1. Compute with the training part the estimators for $\lambda_1, \ldots, \lambda_r$: $\hat{\boldsymbol{\theta}}^{\lambda_1}, \ldots, \hat{\boldsymbol{\theta}}^{\lambda_r}$
2. Evaluate the (prediction) error $\mathrm{Error}_1^k, \ldots, \mathrm{Error}_r^k$ over the validation part,

K-fold Cross-Validation ($K = 10$)

- Choose a grid of r λ's to test: $\lambda_1, \ldots, \lambda_r$
- Divide $(X, \mathbf{y})$ into K blocks (sample-wise):

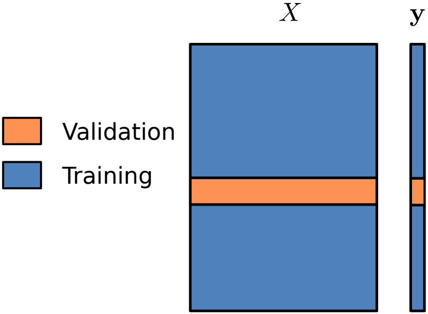


$$k = 5$$

1. Compute with the training part the estimators for $\lambda_1, \ldots, \lambda_r$: $\hat{\boldsymbol{\theta}}^{\lambda_1}, \ldots, \hat{\boldsymbol{\theta}}^{\lambda_r}$
2. Evaluate the (prediction) error $\mathrm{Error}_1^k, \ldots, \mathrm{Error}_r^k$ over the validation part,

K-fold Cross-Validation ($K = 10$)

- Choose a grid of r λ's to test: $\lambda_1, \ldots, \lambda_r$
- Divide $(X, \mathbf{y})$ into K blocks (sample-wise):

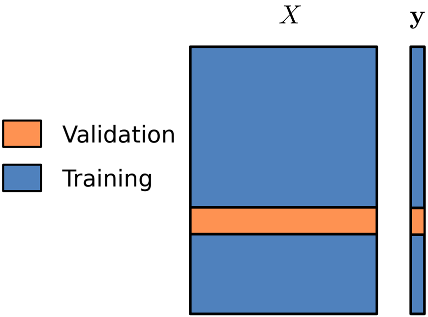


$k = 6$

1. Compute with the training part the estimators for $\lambda_1, \ldots, \lambda_r$: $\hat{\boldsymbol{\theta}}^{\lambda_1}, \ldots, \hat{\boldsymbol{\theta}}^{\lambda_r}$
2. Evaluate the (prediction) error $\mathrm{Error}_1^k, \ldots, \mathrm{Error}_r^k$ over the validation part,

K-fold Cross-Validation ($K = 10$)

- Choose a grid of r λ's to test: $\lambda_1, \dots, \lambda_r$
- Divide $(X, \mathbf{y})$ into K blocks (sample-wise):

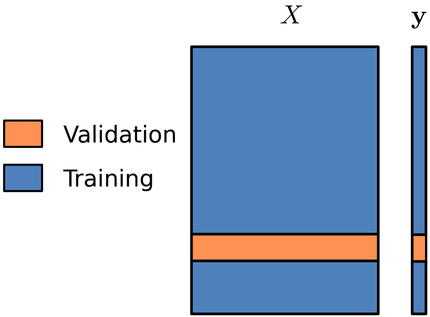


$k = 7$

1. Compute with the training part the estimators for $\lambda_1, \dots, \lambda_r$: $\hat{\boldsymbol{\theta}}^{\lambda_1}, \dots, \hat{\boldsymbol{\theta}}^{\lambda_r}$
2. Evaluate the (prediction) error $\mathrm{Error}_1^k, \dots, \mathrm{Error}_r^k$ over the validation part,

K-fold Cross-Validation ($K = 10$)

- Choose a grid of r λ's to test: $\lambda_1, \ldots, \lambda_r$
- Divide $(X, \mathbf{y})$ into K blocks (sample-wise):

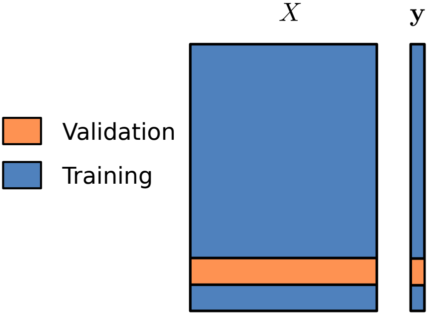


$k = 8$

1. Compute with the training part the estimators for $\lambda_1, \ldots, \lambda_r$: $\hat{\boldsymbol{\theta}}^{\lambda_1}, \ldots, \hat{\boldsymbol{\theta}}^{\lambda_r}$
2. Evaluate the (prediction) error $\mathrm{Error}_1^k, \ldots, \mathrm{Error}_r^k$ over the validation part,

K-fold Cross-Validation ($K = 10$)

- Choose a grid of r λ's to test: $\lambda_1, \dots, \lambda_r$
- Divide $(X, \mathbf{y})$ into K blocks (sample-wise):

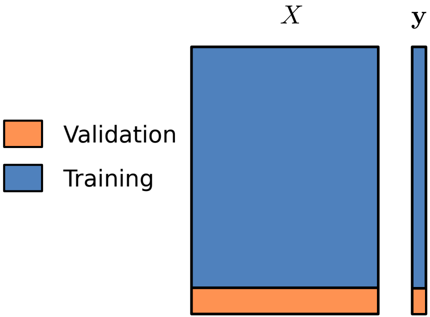


$k = 9$

1. Compute with the training part the estimators for $\lambda_1, \dots, \lambda_r$: $\hat{\boldsymbol{\theta}}^{\lambda_1}, \dots, \hat{\boldsymbol{\theta}}^{\lambda_r}$
2. Evaluate the (prediction) error $\mathrm{Error}_1^k, \dots, \mathrm{Error}_r^k$ over the validation part,

K-fold Cross-Validation ($K = 10$)

- Choose a grid of r λ's to test: $\lambda_1, \dots, \lambda_r$
- Divide $(X, \mathbf{y})$ into K blocks (sample-wise):

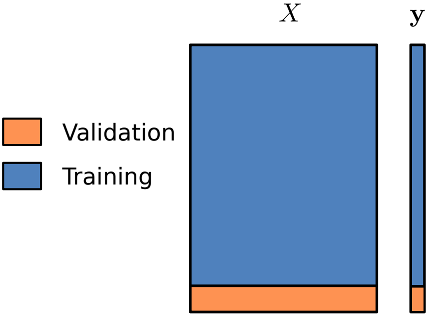


$k = 10$

1. Compute with the training part the estimators for $\lambda_1, \dots, \lambda_r$: $\hat{\boldsymbol{\theta}}^{\lambda_1}, \dots, \hat{\boldsymbol{\theta}}^{\lambda_r}$
2. Evaluate the (prediction) error $\mathrm{Error}_1^k, \dots, \mathrm{Error}_r^k$ over the validation part,

K-fold Cross-Validation ($K = 10$)

- Choose a grid of r λ's to test: $\lambda_1, \dots, \lambda_r$
- Divide $(X, \mathbf{y})$ into K blocks (sample-wise):



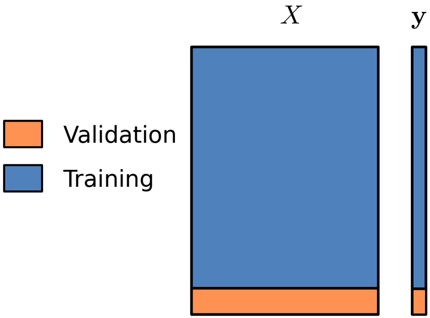
$k = 10$

1. Compute with the training part the estimators for $\lambda_1, \dots, \lambda_r$: $\hat{\boldsymbol{\theta}}^{\lambda_1}, \dots, \hat{\boldsymbol{\theta}}^{\lambda_r}$
2. Evaluate the (prediction) error $\mathrm{Error}_1^k, \dots, \mathrm{Error}_r^k$ over the validation part,

Parameter choice: compute $\widehat{\mathrm{Error}}_1, \dots, \widehat{\mathrm{Error}}_r$, average the errors and choose $\hat{i}^{\mathrm{CV}} \in [\![1, r]\!]$ achieving the smallest one

K-fold Cross-Validation ($K = 10$)

- Choose a grid of r λ's to test: $\lambda_1, \dots, \lambda_r$
- Divide $(X, \mathbf{y})$ into K blocks (sample-wise):



$k = 10$

1. Compute with the training part the estimators for $\lambda_1, \dots, \lambda_r$: $\hat{\boldsymbol{\theta}}^{\lambda_1}, \dots, \hat{\boldsymbol{\theta}}^{\lambda_r}$
2. Evaluate the (prediction) error $\mathrm{Error}_1^k, \dots, \mathrm{Error}_r^k$ over the validation part,

<u>**Parameter choice**</u>: compute $\widehat{\mathrm{Error}}_1, \dots, \widehat{\mathrm{Error}}_r$, average the errors and choose $\hat{i}^{\mathrm{CV}} \in [\![1, r]\!]$ achieving the smallest one

<u>**Re-calibration**</u>: compute $\hat{\boldsymbol{\theta}}^{\lambda_{\hat{i}^{\mathrm{CV}}}}$ this time over the whole sample

CV in practice

Extreme cases of CV

- $K = 1$ impossible, needs $K = 2$
- $K = n$, "*leave-one-out*" strategy (*cf.* **Jackknife**): as many blocks as observations
  Rem: $K = n$ (often) computationally efficient but instable

Practical advice:

- "randomise the sample" : having samples in random order avoid artifacts block (each fold needs to be representative of the whole sample!)
- standard choices: $K = 5, 10$

Alternatives: random partition validation/test, time series variants, etc. http://scikit-learn.org/stable/modules/cross_validation.html

Rem: in prediction the best predictors can be averaged/combined instead of recomputing an estimators over the whole set

CV variants `sklearn`

Crucial points: the structures train/test artificially created should represent faithfully the underlying learning problem

Classical alternatives:

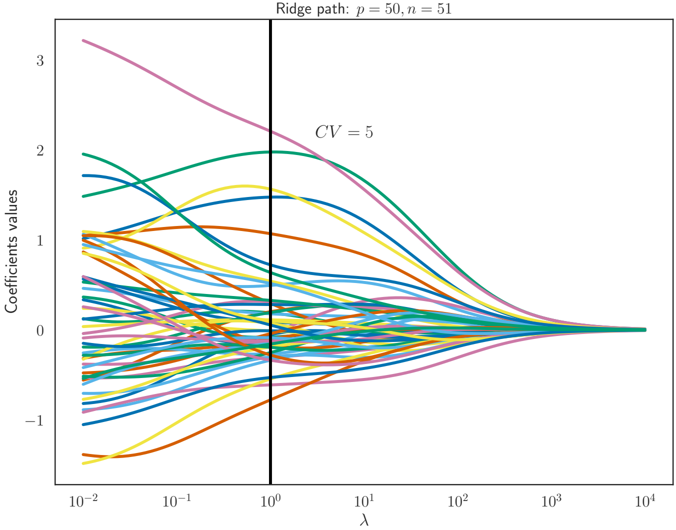
- random partitioning in train/test sets (*cf.*`train_test_split`)
- Time series variant: `TimeSeriesSplit` (never predict the past with future information)
- For classification tasks with unbalanced classes `StratifiedKFold`

Rem: averaging estimators (with weights reflecting their performance) is also relevant for prediction

More details:

http://scikit-learn.org/stable/modules/cross_validation.html

Choosing λ : example with $CV = 5$ (I)



Choosing λ : example with $CV = 5$ (II)

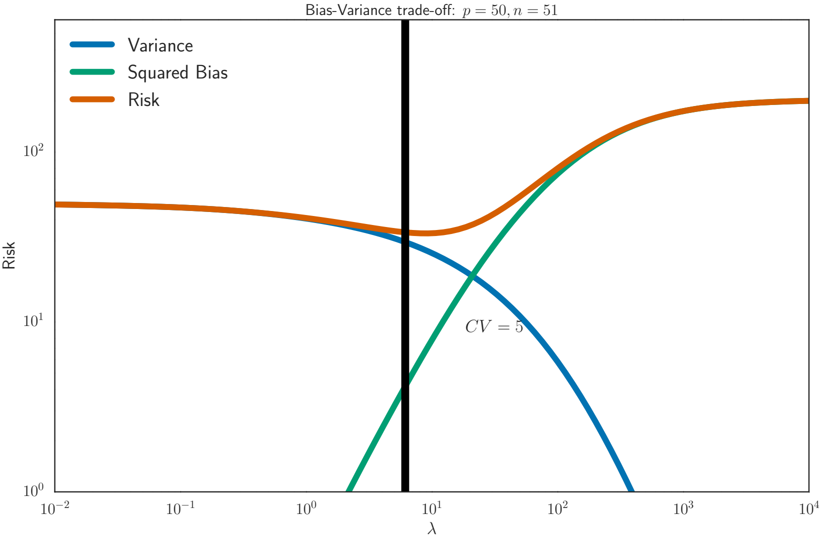


Table of Contents

Algorithms to compute the *Ridge* estimator

- `'svd'`: most stable method, useful for computing many λ's cause the SVD price is paid only once
- `'cholesky'` : matrix decomposition leading to a close form solution `scipy.linalg.solve`
- `'sparse_cg'`: conjugate gradient descent, useful also for sparse cases and high dimension (set `tol`/`max_iter` to a small value)
- stochastic gradient descent approaches : if n is huge

*cf.*the code of `Ridge`, `ridge_path`, `RidgeCV` in the module `linear_model` of `sklearn`

Rem: it is rare to compute the *Ridge* estimator only for one single λ

Rem: crucial issue of computing SVD for huge matrices...

References I

- G. H. Golub and C. F. van Loan.
  Matrix computations.
  Johns Hopkins University Press, Baltimore, MD, fourth edition, 2013.
- M. Y. Park.
  Generalized linear models with regularization.
  PhD thesis, Stanford University, 2006.