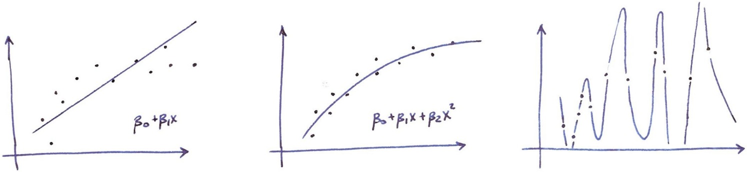
Overfitting



	$M = 0$	$M = 1$	$M = 3$	$M = 9$
$w_0^\star$	0.19	0.82	0.31	0.35
$w_1^\star$		-1.27	7.99	232.37
$w_2^\star$			-25.43	-5321.83
$w_3^\star$			17.37	48568.31
$w_4^\star$				-231639.30
$w_5^\star$				640042.26
$w_6^\star$				-1061800.52
$w_7^\star$				1042400.18
$w_8^\star$				-557682.99
$w_9^\star$				125201.43

Bishop, *Pattern Recognition and Machine Learning*

Regularization

Idea: *penalize* large coefficients. (Occam's razor!)

$$\mathcal{L} = \frac{1}{n}\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2 + \lambda R(\boldsymbol{\beta})$$

Here $\lambda R(\boldsymbol{\beta})$ is a *penalty* term and λ is called *regularization parameter*.

Some common choices are (all convex):

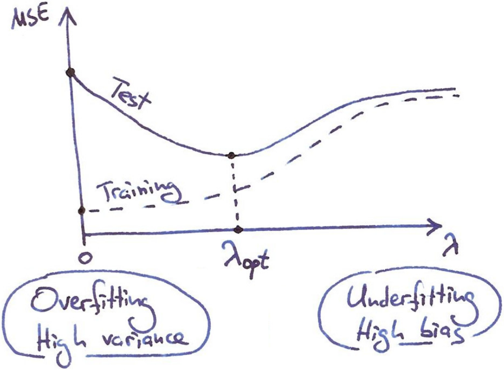
$$R(\boldsymbol{\beta}) = \|\boldsymbol{\beta}\|^2 = \sum_i \beta_i^2 \qquad \text{ridge}$$

$$R(\boldsymbol{\beta}) = \|\boldsymbol{\beta}\|_1 = \sum_i |\beta_i| \qquad \text{lasso}$$

$$R(\boldsymbol{\beta}) = \lambda_1\|\boldsymbol{\beta}\|_1 + \lambda_2\|\boldsymbol{\beta}\|_2^2 \qquad \text{elastic net}$$

Bias–variance tradeoff

Loss function: $\mathcal{L} = \frac{1}{n}\|\mathbf{y} - \mathbf{X}\beta\|^2 + \lambda R(\beta)$.



Ridge regression

Ridge regression

Loss function:

$$\mathcal{L} = \frac{1}{n}\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2 + \lambda\|\boldsymbol{\beta}\|^2.$$

Gradient:

$$\nabla\mathcal{L} = -\frac{2}{n}\mathbf{X}^\top(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}) + 2\lambda\boldsymbol{\beta}.$$

Gradient descent:

$$\begin{aligned}\boldsymbol{\beta} \leftarrow \boldsymbol{\beta} - \eta\nabla\mathcal{L} &= \boldsymbol{\beta} + \eta\frac{2}{n}\mathbf{X}^\top(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}) - 2\eta\lambda\boldsymbol{\beta} = \\ &= \underbrace{(1 - 2\eta\lambda)}_{\text{“weight decay”}}\boldsymbol{\beta} + \eta\frac{2}{n}\mathbf{X}^\top(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}).\end{aligned}$$

Analytic solution for ridge regression

Gradient is equal to zero at the minimum:

$$-\frac{2}{n}\mathbf{X}^\top(\mathbf{y}-\mathbf{X}\hat{\beta}) + 2\lambda\hat{\beta} = 0$$

$$\mathbf{X}^\top\mathbf{X}\hat{\beta} + n\lambda\hat{\beta} = \mathbf{X}^\top\mathbf{y}$$

$$(\mathbf{X}^\top\mathbf{X} + n\lambda\mathbf{I})\hat{\beta} = \mathbf{X}^\top\mathbf{y}$$

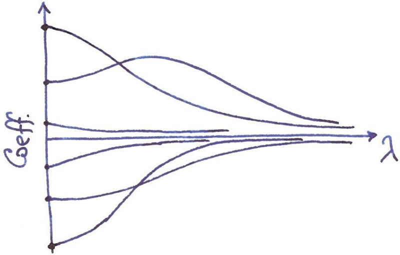
$$\boxed{\hat{\beta} = (\mathbf{X}^\top\mathbf{X} + n\lambda\mathbf{I})^{-1}\mathbf{X}^\top\mathbf{y}}$$

This is an example of a *shrinkage* estimator.

One can prove that if $\mathbf{X}^\top\mathbf{X}$ has full rank, then $\lambda_{\mathrm{opt}} > 0$ (Hoerl and Kennard, 1970).

Shrinkage in action

Ridge estimator: $\hat{\beta}_\lambda = (\mathbf{X}^\top \mathbf{X} + n\lambda \mathbf{I})^{-1} \mathbf{X}^\top \mathbf{y}$.



A note on not penalizing the intercept

Loss function:

$$\mathcal{L} = \frac{1}{n}\|\mathbf{y} - \mathbf{X}\beta\|^2 + \lambda\|\boldsymbol{\beta}\|^2$$

What happens with $\hat{y}$ when $\lambda \to \infty$? It is convenient if $\hat{y}_i \to \bar{y}$ and not to 0. This will be the case if both $\mathbf{X}$ and $\mathbf{y}$ have been centered (and $\mathbf{X}$ does not contain $\mathbf{x}_0$). Otherwise we need to write explicitly

$$\mathcal{L} = \frac{1}{n}\sum_{i=1}^{n}\Big(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j X_{ij}\Big)^2 + \lambda\sum_{j=1}^{p}\beta_j^2.$$

Another note: there may be no $\frac{1}{n}$ factor in some implementations.

SVD perspective

Consider singular value decomposition $\mathbf{X} = \mathbf{USV}^\top$.

We previously showed that in OLS regression

$$\hat{\mathbf{y}} = \mathbf{UU}^\top\mathbf{y}.$$

In ridge regression,

$$\begin{aligned}
\hat{\mathbf{y}} = \mathbf{X}\hat{\beta} &= \underbrace{\mathbf{X}(\mathbf{X}^\top\mathbf{X} + n\lambda\mathbf{I})^{-1}\mathbf{X}^\top}_{\text{hat matrix}}\mathbf{y} \\
&= \mathbf{USV}^\top(\mathbf{VS}^2\mathbf{V}^\top + n\lambda\mathbf{VV}^\top)^{-1}\mathbf{VSU}^\top\mathbf{y} \\
&= \mathbf{U}\,\mathrm{diag}\left\{\frac{s_i^2}{s_i^2 + n\lambda}\right\}\mathbf{U}^\top\mathbf{y}.
\end{aligned}$$

I.e. ridge regression stronger affects small singular values.

Bayesian perspective

Previously we showed that $\hat{\boldsymbol{\beta}}_{\mathrm{OLS}}$ is the maximum likelihood solution of

$$y = \boldsymbol{\beta}^\top \mathbf{x} + \epsilon,$$
$$\epsilon \sim \mathcal{N}(0, \sigma^2).$$

This treats $\boldsymbol{\beta}$ as fixed. What if we treat it as random and assume a *prior* distribution $\boldsymbol{\beta} \sim \mathcal{N}(\mathbf{0}, \tau^2\mathbf{I})$?..

It turns out that $\hat{\boldsymbol{\beta}}_\lambda$ with $\lambda = \sigma^2/(n\tau^2)$ is the mean of the *posterior* distribution, i.e. it is a *maximum a posteriori (MAP)* estimator.

Bayes theorem, prior, and posterior

Joint and conditional probabilities:

$$P(A, B) = P(A \mid B)P(B) = P(B \mid A)P(A).$$

$\Rightarrow$ Bayes theorem:

$$P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)}.$$

Example:

$$P(\text{pandemics} \mid \text{mask}) = \frac{P(\text{mask} \mid \text{pandemics})P(\text{pandemics})}{P(\text{mask})} =$$

$$= \frac{P(\text{mask} \mid \text{pandemics})P(\text{pandemics})}{P(\text{mask} \mid \text{pand.})P(\text{pand.}) + P(\text{mask} \mid \neg\text{pand.})P(\neg\text{pand.})}$$

Bayes theorem, prior, and posterior

$$P(\text{pandemics} \mid \text{mask}) = \frac{P(\text{mask} \mid \text{pandemics})P(\text{pandemics})}{P(\text{mask})}$$

$$P(\text{Halloween} \mid \text{mask}) = \frac{P(\text{mask} \mid \text{Halloween})P(\text{Halloween})}{P(\text{mask})}$$

$$P(\text{diving} \mid \text{mask}) = \frac{P(\text{mask} \mid \text{diving})P(\text{diving})}{P(\text{mask})}$$

So when there are many options, it is often enough to write

$$P(A \mid B) \sim P(B \mid A)P(A).$$

Prior, posterior, and likelihood

For continuous random variables:

$$p(x \mid y) \sim p(y \mid x)p(x).$$

In our case of a generative model:

$$\underbrace{p(\text{params} \mid \text{data})}_{\text{posterior}} \sim \underbrace{p(\text{data} \mid \text{params})}_{\text{likelihood}} \underbrace{p(\text{params})}_{\text{prior}}.$$

Taking the logarithm:

$$\underbrace{\log p(\text{params} \mid \text{data})}_{\text{log-posterior}} \sim \underbrace{\log p(\text{data} \mid \text{params})}_{\text{log-likelihood}} + \underbrace{\log p(\text{params})}_{\text{log-prior}}.$$

Bayesian linear regression

Probabilistic model and prior:

$$\begin{aligned} y &= \boldsymbol{\beta}^\top \mathbf{x} + \epsilon, \\ \epsilon &\sim \mathcal{N}(0, \sigma^2), \\ \boldsymbol{\beta} &\sim \mathcal{N}(0, \tau^2 \mathbf{I}). \end{aligned}$$

Log-likelihood (last lecture):

$$-\frac{n}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2.$$

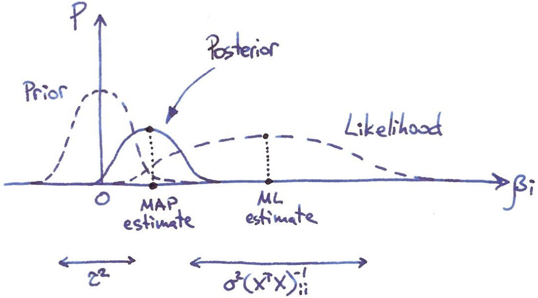
Log-prior:

$$-\frac{p}{2}\log(2\pi\tau^2) - \frac{1}{2\tau^2}\|\boldsymbol{\beta}\|^2.$$

Bayesian linear regression

Hence negative log-posterior:

$$\ldots + \frac{1}{2\sigma^2}\|\mathbf{y} - \mathbf{X}\beta\|^2 + \frac{1}{2\tau^2}\|\beta\|^2 \sim \frac{1}{n}\|\mathbf{y} - \mathbf{X}\beta\|^2 + \underbrace{\frac{\sigma^2}{n\tau^2}}_{\text{effective } \lambda}\|\beta\|^2.$$



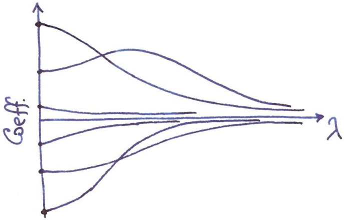
Exercise: product of two Gaussians is a Gaussian.

Lasso regression

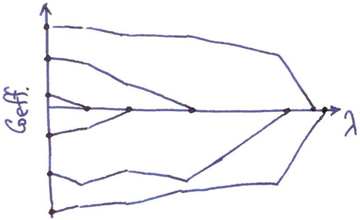
Lasso regression

$$\mathcal{L} = \frac{1}{n}\|\mathbf{y} - \mathbf{X}\beta\|^2 + \lambda\|\beta\|_1.$$

No analytic solution. But one can show that solutions are *sparse*.



Ridge



Lasso

Lagrange multipliers

Theorem: minimizing a loss $\mathcal{L}(\mathbf{w})$ subject to constraints $C(\mathbf{w}) = 0$ is equivalent to minimizing $\mathcal{L}(\mathbf{w}) + \lambda C(\mathbf{w})$ over $\mathbf{w}$ and λ. Here λ is called *Lagrange multiplier*.

The same is true for inequality constraints $C(\mathbf{w}) \leq 0$ (with some extra conditions that I omit here for simplicity).

This means that the following two formulations are equivalent:

$$\mathcal{L} = \frac{1}{n}\|\mathbf{y} - \mathbf{X}\beta\|^2 + \lambda\|\boldsymbol{\beta}\|_1,$$
$$\mathcal{L} = \frac{1}{n}\|\mathbf{y} - \mathbf{X}\beta\|^2 \;\; \text{s.t.} \;\; \|\boldsymbol{\beta}\|_1 \leq t.$$

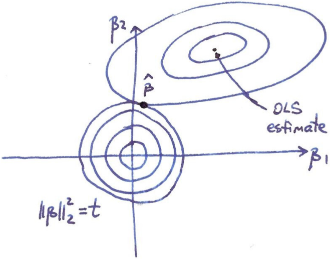
And the same is true for ridge regression with $\lambda\|\boldsymbol{\beta}\|_2^2$.

EBERHARD KARLS UNIVERSITÄT TÜBINGEN

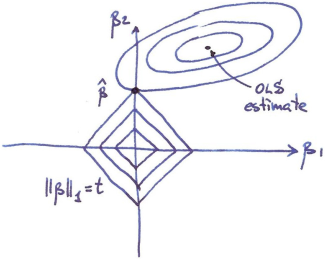
Ridge vs. lasso

$$\text{Ridge:} \quad \mathcal{L} = \frac{1}{n}\|\mathbf{y} - \mathbf{X}\beta\|^2 \ \text{ s.t. } \ \|\beta\|_2^2 \leq t.$$

$$\text{Lasso:} \quad \mathcal{L} = \frac{1}{n}\|\mathbf{y} - \mathbf{X}\beta\|^2 \ \text{ s.t. } \ \|\beta\|_1 \leq t.$$



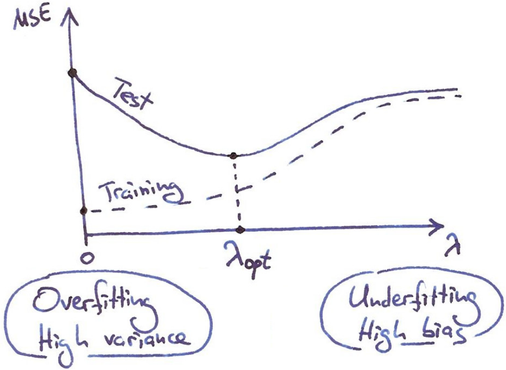
Ridge



Lasso

Model selection

Bias–variance tradeoff



Training, test, and validation sets

Split the dataset into:

- Training set: used for model fitting;
- Test set: used for model evaluation.

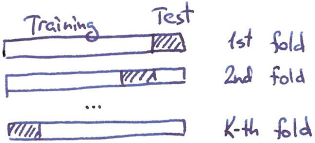
Note: there is a tradeoff between training and test set sizes. Rule of thumb: $\sim$90% training, $\sim$10% test.

If we need to tune some hyper-parameters, e.g. λ, it is more appropriate to use three sets:

- Training set: used for model fitting;
- Validation set: used for hyper-parameter tuning;
- Test set: used for final model evaluation.

Cross-validation

Often the dataset is not large enough for a reliable training/test split. Then one could use *cross-validation* (CV):



K-fold cross-validation. n-fold CV is called *leave-one-out* CV (LOOCV). Rule of thumb: $K = 10$.

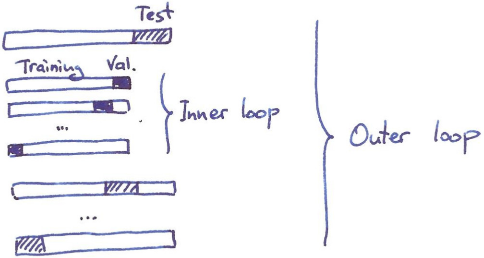
Note that cross-validation measures the performance not of a given model, but of a model building procedure.

If you need a final model (for production or for inspection), then afterwards fit the model using the chosen λ on all of the available data.

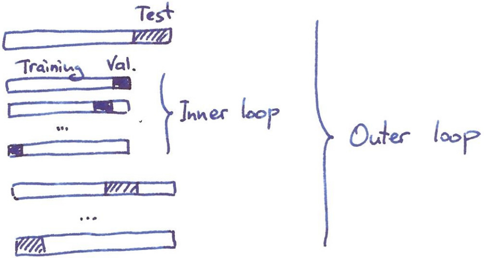
Nested cross-validation

What if we need training, validation, and test? Nested cross-validation!

- Outer loop: puts aside a test set.
- Inner loop: puts aside a validation test.
- After each inner loop: fit the model with chosen λ on all 'inner' data.



Nested cross-validation

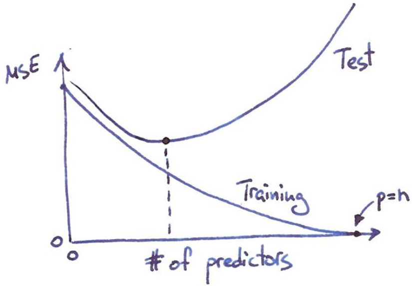


If you need a final model (for production or for inspection), then fit the model using the "inner loop" on the entire dataset.

Exercise: if you use $K = 10$ for the outer loop, $K = 5$ for the inner loop, and 100 values of λ as your grid search, how many models will be built?

Beyond the interpolation threshold

Back to polynomial regression



If $p > n$, the regression problem is *undertermined*: there are infinitely many β values yielding zero loss $\mathcal{L}(\beta) = 0$.

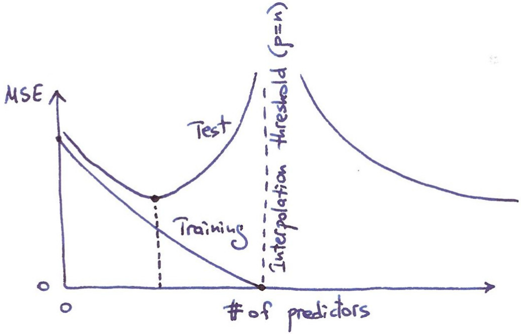
Minimum-norm solution

Using $\mathbf{X} = \mathbf{U}\mathbf{S}\mathbf{V}^\top$, we previously obtained $\hat{\beta}_{\mathrm{OLS}} = \mathbf{V}\mathbf{S}^{-1}\mathbf{U}^\top\mathbf{y}$.

This formula still makes sense if $p > n$ (now $\mathbf{S}$ is $n \times n$, not $p \times p$) and yields the minimum-norm $\hat{\beta}$ among all possible ones satisfying $\mathcal{L}(\beta) = 0$.

Implicit regularization

Here is what can happen if we use the minimum-norm solution beyond the *interpolation threshold*:



Implicit regularization.