Mathematical Statistics

course # 11

Damien Garreau

Université Côte d'Azur

April 16, 2020
Recap (I)

▶ Gaussian linear model with centered data:

\[ Y = X\beta^* + \epsilon \in \mathbb{R}^n, \]

with \( \epsilon \sim \mathcal{N}(0, \sigma^2 I_n) \), \( X \in \mathbb{R}^{n \times d} \), and \( \beta^* \in \mathbb{R}^d \)

▶ Linear regression:

\[ \hat{\beta}_{\text{LS}} \in \arg\min_{\beta \in \mathbb{R}^d} \left\{ \| Y - X\beta \|^2 \right\}. \]

▶ solution given by

\[ \hat{\beta}_{\text{LS}} = (X^T X)^\dagger X^T Y, \]

where \( \dagger \) denotes the generalized inverse

▶ Ridge regression:

\[ \hat{\beta}_{\text{Ridge}} \in \arg\min_{\beta \in \mathbb{R}^d} \left\{ \| Y - X\beta \|^2 + \lambda \| \beta \|^2 \right\}, \]

with \( \lambda > 0 \)
Recap (II)

- Solution given by
  \[ \hat{\beta}_{\text{Ridge}} = (X^\top X + \lambda I_d)^{-1} X^\top Y. \]

- Bias-variance tradeoff:

![Bias-variance trade-off for ridge regression](image)

- Graph showing the bias-variance trade-off for ridge regression with the equation:
  \[ \text{MSE} = \text{bias}^2 + \text{variance} \]
  - Orange line represents \( \text{bias}^2 \)
  - Blue line represents variance
  - Black line represents MSE = \( \text{bias}^2 + \text{variance} \)
  - The graph illustrates that as \( \lambda \) increases, the bias decreases, and the variance increases, leading to a trade-off between bias and variance.
Outline

1. Least Absolute Shrinkage and Selection Operator
2. Introduction to statistical learning
3. Cross-validation
1. Least Absolute Shrinkage and Selection Operator
The Lasso (I)

▶ **Idea:** Least Absolute Shrinkage (Lasso) replaces the $L^2$ norm by the $L^1$ norm (Tibshirani, Regression Shrinkage and Selection via the Lasso, 1986)

▶ namely, for some $\lambda > 0$, find

$$\hat{\beta}_{\text{Lasso}} \in \arg\min_{\beta \in \mathbb{R}^d} \left\{ \| Y - X \beta \|^2 + \lambda \| \beta \|_1 \right\}.$$

▶ recall that $\| x \|_1 = \sum_{i=1}^d |x_i|$: 

![L2 ball](image1.png) ![L1 ball](image2.png)
The Lasso (II)

- recall that we are looking for:
  \[ \hat{\beta}_{\text{Lasso}} \in \arg \min_{\beta \in \mathbb{R}^d} \left\{ \| Y - X\beta \|^2 + \lambda \| \beta \|_1 \right\}. \]  
  \hspace{1cm} (1)

- equivalent formulation (see TD):
  \[ \hat{\beta}_{\text{Lasso}} \in \arg \min_{\beta \in \mathbb{R}^d} \| Y - X\beta \|^2 \]
  subject to \( \| \beta \|_1 \leq t \).

- **Question:** how do we find \( \hat{\beta}_{\text{Lasso}} \)?

- **Problem:** (1) is convex, but **not differentiable**

- several possibilities:
  - subgradient methods
  - Least-angle regression: also gives the regularization path (for the same computational cost)
The Lasso (III)

Lasso regularization path

\[ \hat{\beta}_i \]

\[ t \]

\[ 0 \quad 2 \quad 4 \quad 6 \quad 8 \quad 10 \]

\[ 0 \quad 1 \quad 2 \quad 3 \quad 4 \]

\[ -3 \quad -2 \quad -1 \quad 0 \quad 1 \quad 2 \quad 3 \quad 4 \]

\[ 0 \quad 2 \quad 4 \quad 6 \quad 8 \quad 10 \]
The Lasso (IV)

- **Intuition:** $L^1$ norm promotes sparsity (sparse = “parcimonieux”)

![Graph showing $L^2$ and $L^1$ balls](image)

- in red, the constraint $||\beta|| \leq t$
- in black the level sets of $||Y - X\beta||^2$
- $L^1$ ball has many facets and edges: often the solution has many zero coordinates!
The Lasso (IV)

▶ let us define
\[ \|\beta\|_0 = |\{i \text{ s.t. } \beta_i \neq 0\}|, \]
the number of non-zero coordinates of vector $\beta$

▶ **Definition:** when
\[ \|\beta\|_0 \ll d, \]
we say that $\beta$ is *sparse* ("parcimonieux")

▶ **Bet on sparsity:** assume that the groundtruth is sparse
  ▶ if it is, we do well with the Lasso
  ▶ if not, then no method is going to perform well

▶ also important to have a **variable selection** method

▶ quickly identify which covariates are important
The Lasso (V)

- let us try to understand why in the orthonormal case \((X^\top X = I_d)\)
- we write

\[
R(\beta) = \|Y - X\beta\|^2 + \lambda \|\beta\|_1 \\
= (Y - X\beta)^\top (Y - X\beta) + \lambda \|\beta\|_1 \\
R(\beta) = Y^\top Y - 2\beta^\top X^\top Y + \beta^\top X^\top X\beta + \lambda \|\beta\|_1
\]

- recall that \(\hat{\beta}^{LS} = (X^\top X)^{-1}X^\top Y = X^\top Y\):

\[
R(\beta) = -2\beta^\top \hat{\beta}^{LS} + \beta^\top \beta + \lambda \|\beta\|_1 + \text{cst} \\
= \sum_{j=1}^{d} \left\{-2\beta_j\hat{\beta}_j^{LS} + \beta_j^2 + \lambda |\beta_j|\right\} + \text{cst}
\]
The Lasso (VI)

that is,

\[ R(\beta) = \sum_{j=1}^{d} R_j(\beta_j) + \text{cst}, \]

with \( R_j(x) = -2\hat{\beta}_j^{LS}x + x^2 + \lambda |x| \)

let us look at a coordinate such that \( \hat{\beta}_j^{\text{Lasso}} \neq 0 \)

then we can differentiate:

\[ \frac{\partial R(\beta)}{\partial \beta_j} = \frac{\partial R_j(\beta_j)}{\partial \beta_j} = -2\hat{\beta}_j^{LS} + 2\beta_j + \lambda \text{sign} (\beta_j) \]

\( \hat{\beta}_j^{\text{Lasso}} \) solves

\[ -2\hat{\beta}_j^{LS} + 2x + \lambda \text{sign} (x) = 0. \]
The Lasso (VII)

- a nonzero solution must satisfy, depending on its sign,
  \[-2\hat{\beta}_j^{LS} + 2x + \lambda = 0 \quad \text{or} \quad -2\hat{\beta}_j^{LS} + 2x - \lambda = 0.\]

- that is,
  \[x = \hat{\beta}_j^{LS} - \frac{\lambda}{2} \quad \text{if} \quad x > 0 \quad \text{or} \quad x = \hat{\beta}_j^{LS} + \frac{\lambda}{2} \quad \text{if} \quad x > 0.\]

- we deduce the three possibilities:
  \[
  \begin{cases}
  \hat{\beta}_j^{LS} - \frac{\lambda}{2} > 0 & \Rightarrow \hat{\beta}_j^{Lasso} = \hat{\beta}_j^{LS} - \frac{\lambda}{2} \\
  \hat{\beta}_j^{LS} + \frac{\lambda}{2} < 0 & \Rightarrow \hat{\beta}_j^{Lasso} = \hat{\beta}_j^{LS} + \frac{\lambda}{2} \\
  \hat{\beta}_j^{LS} \in \left(-\frac{\lambda}{2}, \frac{\lambda}{2}\right) & \Rightarrow \hat{\beta}_j^{Lasso} = 0
  \end{cases}
  \]

In short,

\[
\hat{\beta}_j^{Lasso} = \text{sign} (\hat{\beta}_j^{LS}) \cdot \left(\left|\hat{\beta}_j^{LS}\right| - \frac{\lambda}{2}\right)^+.
\]
The Lasso (VIII)

Soft thresholding operator
2. Introduction to statistical learning
we are given an i.i.d. sample \((X_i, Y_i)_{1 \leq i \leq n}\) distributed according to a distribution \(P\) on \(\mathbb{R}^d \times \mathbb{R}\)

**Goal:** construct a good predictor \(\hat{f} : \mathbb{R}^d \rightarrow \mathbb{R}\)

**More precisely:** construct \(\hat{f}\) from the observations \(X_1, \ldots, X_n\) such that, for any new observation \(x_{\text{new}}\), the prediction \(\hat{Y} = \hat{f}(x_{\text{new}})\) is close to the groundtruth \(y_{\text{new}}\), where \((x_{\text{new}}, y_{\text{new}}) \sim P\)

**Question:** what does “close” mean?

we quantify the quality of prediction with a loss function

\[
\ell : \mathbb{R} \times \mathbb{R} \rightarrow \mathbb{R}_+
\]

\[(y, \hat{y}) \mapsto \ell(y, \hat{y})\]

generally, we require

\[
\ell(y, y) = 0
\]

\[
\ell \geq 0
\]

\[
\ell(\cdot, \cdot) \text{ is symmetric}
\]
a typical choices for regression is the **squared loss**: 
\[ \ell(y, \hat{y}) = (y - \hat{y})^2 \]

**Classical losses for regression**

- **squared loss**
- **absolute deviation**
- **Huber loss**
given a loss function $\ell$, we can look at the quality of our predictor

given a realization $(x_i, y_i)$, we first define the **generalization error** (also **test error**) of $\hat{f} = \hat{f}(x_1, \ldots, x_n)$ by

$$E_{\text{test}} = \mathbb{E} \left[ \ell(Y, \hat{f}(X)) \mid X_1 = x_1, \ldots, X_n = x_n \right].$$

in the previous display, only the new observation $(X, Y)$ is random

**Intuition:** how well our predictor does on *new, unseen* data

this is truly what we want to minimize!

unfortunately, we do not have access to $E_{\text{test}}$

best we can do: **empirical risk** (or **train error**)

$$E_{\text{train}} = \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, \hat{f}(x_i)).$$
Statistical learning (IV)

- classical strategy: **empirical risk minimization** (ERM) and variations thereof:

\[ \hat{f} \in \arg\min_{h \in \mathcal{H}} \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, h(x_i)), \]

where \( \mathcal{H} \) is a function space

- this is the second restriction: we cannot choose the best predictor of all possible functions!

- **Example (i)**: ordinary least squares, we pick

\[ \hat{\beta}^{\text{LS}} \in \arg\min_{\beta \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} (y_i - \beta^\top x_i)^2, \]

which is ERM with \( \ell = \) squared loss and \( \mathcal{H} = \) linear functions
Example (ii): ridge regression, for $t > 0$ we pick

$$\hat{\beta}_t^{\text{Ridge}} \in \arg \min_{\beta \in \mathbb{R}^d, \|\beta\| \leq t} \frac{1}{n} \sum_{i=1}^{n} (y_i - \beta^T x_i)^2,$$

which is ERM with $\ell = \text{squared loss}$ and $\mathcal{H} = \text{linear functions of prescribed } L^2 \text{ norm}$

Example (iii): Lasso, for $t > 0$ we pick

$$\hat{\beta}_t^{\text{Lasso}} \in \arg \min_{\beta \in \mathbb{R}^d, \|\beta\|_1 \leq t} \frac{1}{n} \sum_{i=1}^{n} (y_i - \beta^T x_i)^2,$$

which is ERM with $\ell = \text{squared loss}$ and $\mathcal{H} = \text{linear functions of prescribed } L^1 \text{ norm}$
Statistical learning (VI)

- Slightly different from statistics!
- Recall the Gaussian linear model

\[ Y = X\beta^* + \varepsilon, \]

with \( \varepsilon \sim \mathcal{N}(0, \sigma^2 I_n) \)

- **Statistics** is about inference: we want to find \( \hat{\beta} \) that is as close as possible to \( \beta^* \), and the key quantity is **mean squared error**

\[ \text{MSE}(\hat{\beta}) = \mathbb{E} \left[ ||\hat{\beta} - \beta^*||^2 \right]. \]

- **Statistical learning** is about generalization: we want to find \( \hat{\beta} \) that predicts well on unseen data, and the key quantity is **test error**

\[ E_{\text{test}} = \mathbb{E} \left[ \ell(Y, \hat{f}(X)) \mid X_1 = x_1, \ldots, X_n = x_n \right]. \]
But what is the problem with ERM?

train error consistently decrease with model complexity!

thus if we just minimize $E_{\text{train}}$, **we will always choose the model with highest complexity**

generally, this is bad in terms of test error (**overfitting**)

let us first see several example to understand model complexity

**Example (i):** 1D polynomial regression, we assume that

$$ y_i = \beta_0 + \beta_1 x_i + \cdots + \beta_{d-1} x_i^{d-1} + \epsilon_i, $$

with $\epsilon_i$ i.i.d. $\mathcal{N}(0, \sigma^2)$

in this example, $d$ is a good measure of the model complexity: higher $d$ gives higher degree polynomials that **can fit any data**
Statistical learning (VIII)

Polynomial regression in dimension 1

- groundtruth
- degree = 0
- degree = 1
- degree = 2
- degree = 10

---

23
Example (ii): ridge regression

Intuition: take the trace of the hat matrix

in linear regression, this is the rank of \( X \), \( \approx \) number of independent fixed predictors (prove this!)

in the case of ridge regression,

\[
\text{trace}(H_\lambda) = \text{trace}\left( X(X^\top X + \lambda I_d)^{-1}X^\top \right) \\
= \text{trace}\left( U\Sigma(\Sigma \Sigma^\top + \lambda I_d)^{-1}\Sigma^\top U^\top \right) \\
= \sum_{j=1}^{d} \frac{\sigma_j^2}{(\sigma_j^2 + \lambda)^2} \left( = \frac{d}{(\lambda + 1)^2} \text{ for orthonormal data} \right)
\]

we call it degrees of freedom and write

\[
df(\lambda) = \sum_{j=1}^{d} \frac{\sigma_j^2}{(\sigma_j^2 + \lambda)^2}.
\]
Train and test error for ridge regression

- **Train error**
- **Test error (est.)**
Example (iii): the Lasso

possible to define a similar notion:

$\text{df}(\hat{y}_i) = \frac{1}{\sigma^2} \sum_{i=1}^{n} \text{Cov}(\hat{y}_i, y_i)$.

similar picture (figure from Hastie, Tibshirani, Friedman, The Elements of Statistical Learning, 2011)
Statistical learning (XII)

- short summary: only relying on train error can lead to overfitting, what we really want to optimize is the test error
- **Disclaimer!** this is not the whole picture
- some models are able to generalize well even when the complexity is over the roof
- especially true when they have some inductive bias, allowing to choose the “best” solution among many
- conceptualized recently as the **double descent** paradigm (Belkin, Hsu, Ma, Mandal, Reconciling modern machine learning practice and the bias-variance trade-off, 2018)
- what is always true: if the model complexity is too low, then underfitting (high bias)
Statistical learning (XIII)

- double descent behavior for random features with ReLU activation

(figure from Mei and Montanari, The generalization error of random features regression: Precise asymptotics and double descent curve, 2019)
3. Cross-validation
Cross-validation (I)

- model $\hat{f}$ trained on
  \[ x_{\text{train}} = \{(x_i, y_i) \in \mathbb{R}^d \times \mathbb{R}, 1 \leq i \leq n\} . \]

- we assume that our model $\hat{f}_\lambda$ depends on a parameter $\lambda$
- we want to find $\lambda$ that minimizes the test error
- **Idea:** estimate the test error by creating a test set from the train set
- we call this new set the **validation set**
- **Ideally,** with enough data, the picture is the following:
Cross-validation (II)

- **Problem:** in general, data is scarce
- we want to use all the available data
- **Idea:** $K$-fold cross-validation ("validation croisée"):  
  - split the training set in $K$ parts  
  - for each $i \in \{1, \ldots, K\}$, train on $K - 1$ parts  
  - and compute the test error on the remaining part  
  - aggregate the errors  
- typical choice: $K = 5, 10$ or $n$ (*leave-one-out*)  
- schematically:
Cross-validation (III)

- more details: \( \kappa : \{1, \ldots, n\} \rightarrow \{1, \ldots, K\} \) indexing function
- \( \kappa(i) \) tells us in which box observation \( i \) belongs
- we define \( \hat{f}^{-k} \) the model trained on all observations not in box \( k \)
- then the cross-validation estimate of the prediction error is

\[
CV(\hat{f}) = \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, \hat{f}^{-\kappa(i)}(x_i)).
\]

- when our model depends on a parameter \( \lambda \), we define similarly

\[
CV(\hat{f}, \lambda) = \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, \hat{f}_{\lambda}^{-\kappa(i)}(x_i)).
\]

- then \( \lambda \mapsto CV(\hat{f}, \lambda) \) is an estimate of the test error curve, and we can choose \( \hat{\lambda} \) minimizing \( CV(\hat{f}, \lambda) \)
Cross-validation (IV)

- **Computational cost of cross-validation:** $K \times$ cost of training our model on $(1 - 1/K)n$ points + $n \times$ cost of prediction
- this can be **huge** and generally constrains $K$ to small values
- but for small $K$, we are estimating the **averaged test error**

$$
\overline{E}_{\text{test}} = \mathbb{E}_{X_1, \ldots, X_n}[E_{\text{test}}],
$$

since the training sets for $\hat{f}^{-k}$ are quite different
- generally we are interested in $E_{\text{test}}$, pushing for larger $K$
- $K = 5$ or $10$ is a good in-between (empirically)
- **Remark:** in certain cases, it is possible to compute CV faster (see TD)