Modern Regression

Jean-Luc Bouchot

School of Mathematics and Statistics
Beijing Institute of Technology
jlbouchot@bit.edu.cn

Spring 2021

Outline

Remark 2.1

As seen from the height/weight dataset, it is, if one wishes, possible to fit perfectly the data.

Theorem 2.1 (Lagrange Interpolation - Admitted)

Let $\{(x_i, y_i)\}_{1 \leq i \leq n}$ be n samples of a given phenomenon.
Assuming $x_i \neq x_j$ for all $i \neq j$, then there exists a degree $n-1$ polynomial P_n such that the approximation error is 0: $P_n(x_i) = y_i$ for all $1 \leq i \leq n$.

Remark 2.2

Assume the underlying model is indeed a polynomial one: what happens if the samples are noisy?

Example 2.1

Assume the following data are given

Target	Predictor	Noisy target
-0.5	-2.5	-0.492
1	-1	0.936
2.5	0.5	2.542
4	2	4.011

1. Compute the estimations using polynomial features of degree 0 up to 3 (included)
2. Compute the approximation errors for each of the polynomial features.

Remark 2.3

This gives the following results:

	Degree features	Coef 0	Coef 1	Coef 2	Coef 3	Error with true
Noiseless	$d = 0$	1.75	0	0	0	3.354
	$d = 1$	2	1	0	0	0
	$d = 2$	2	1	0	0	0
	$d = 3$	2	1	0	0	0
Noisy	$d = 0$	1.749	0	0	0	3.354
	$d = 1$	2.001	1.008	0	0	0.026
	$d = 2$	1.989	1.001	0.005	0	0.033
	$d = 3$	2.006	1.079	-0.007	-0.016	0.078

Example 2.2

We reiterate the same idea, with the following data:

Predictor	Noisier target	Noisiest target
-2.5	-0.502	0.013
-1	0.922	1.204
0.5	2.608	2.473
2	3.896	4.220

Remark 2.4

We obtain the following results:

	Degree features	Coef 0	Coef 1	Coef 2	Coef 3	Error with true
Noisy	$d = 0$	1.749	0	0	0	3.354
	$d = 1$	2.001	1.008	0	0	0.026
	$d = 2$	1.989	1.001	0.005	0	0.033
	$d = 3$	2.006	1.079	-0.007	-0.016	0.078
Noisier	$d = 0$	1.731	0	0	0	3.354
	$d = 1$	1.979	0.992	0	0	0.047
	$d = 2$	2.021	0.984	-0.015	0	0.082
	$d = 3$	2.05	1.129	-0.040	-0.033	0.169
Noisiest	$d = 0$	1.978	0	0	0	3.385
	$d = 1$	2.209	0.926	0	0	0.518
	$d = 2$	2.039	0.957	0.062	0	0.588
	$d = 3$	2.017	0.870	0.077	0.020	0.595

Example 2.3

Looking back at the solution we have obtained, we notice the following: let $\beta(d)$ denotes the $(d+1)$-dimensional vector of coefficients obtained in the regression, its norm is

	$\|\beta(0)\|$	$\|\beta(1)\|$	$\|\beta(2)\|$	$\|\beta(3)\|$
Noiseless	1.75	2.236	2.236	2.236
Noisy	1.749	2.241	2.230	2.278
Noisier	1.731	2.214	2.248	2.347
Noisiest	1.977	2.395	2.253	2.196

Example 2.4

Let us try on a bigger training set: 20 sampling points uniformly spaced, the target values are computed from a noisy linear model. We let d vary from 0 to 25.

Definition 2.1

The **ridge regression** *is a regression problem which penalizes heavy coefficients. It is expressed as*

$$\widehat{\beta} := \underset{\beta \in \mathbb{R}^D}{\operatorname{argmin}} \|X\beta - \mathbf{y}\|_2^2 + \lambda \|\beta\|_2^2,$$

where $X \in \mathbb{R}^{n \times D}$ *denotes the data matrix and* $\mathbf{y} \in \mathbb{R}^n$ *the target (dependent) variables.*

Proposition 2.1

The Ridge Regression approach is equivalent to the following constrained optimization problem

$$\widehat{\beta} = \underset{\beta \in \mathbb{R}^D}{\operatorname{argmin}} \|X\beta - \mathbf{y}\|_2^2$$

$$\textit{subject to } \|\beta\|_2^2 \leq \tau,$$

for a certain value of τ which depends on λ.

Proposition 2.2

Let $\lambda > 0$. The solution to the ridge regression problem is given by

$$\widehat{\beta} = \left(X^T X + \lambda I\right)^{-1} X^T \mathbf{y}$$

Remark 2.5

As soon as $\lambda > 0$, the solution is well defined (and unique!). See the homework.

Theorem 2.2 (Spectral theorem)

Let $A \in \mathbb{R}^{n\times n}$ be a symmetric matrix. There exists a unitary matrix $U \in R^{n\times n}$ such that

$$A = U\,\mathrm{diag}(\lambda_1, \cdots, \lambda_n)U^T,$$

where the λ_i's are the eigenvalues of A.

Proposition 2.3

Assume the noise components in the measurements are zero-mean, normally and independently distributed with variance σ^2. The ridge regression estimator is biased and

$$\mathbb{E}\left[\widehat{\beta(\lambda)}\right] - \beta = \left((X^T X + \lambda I)^{-1} - (X^T X)^{-1}\right) X^T X \beta.$$

Proposition 2.4

Assume the noise components in the measurements are zero-mean, normally and independently distributed with variance σ^2. The covariance matrix of the ridge estimator is given by

$$Var(\widehat{\beta(\lambda)}) = \sigma^2 \left(X^T X + \lambda I\right)^{-1} X^T X \left(X^T X + \lambda I\right)^{-1}.$$

Corollary 2.1

The Ridge estimator has less variance than the ordinary least squares.

Proposition 2.5

Under the same hypothesis as above

$$\lim_{\lambda \to \infty} Var(\widehat{\beta(\lambda)}) = 0.$$

Proposition 2.6

Let $y = f(x) + \varepsilon$ *for some* 0 *mean and known variance* σ^2 *random variable* ε. *Assume the noise is independent of a predictor* $\widehat{f}$ *then the* **bias variance compromise** *reads*

$$MSE = \mathbb{E}[(y - \widehat{f})^2] = \textit{Variance} \ + \ \textit{Bias}^{\,2} + \ \textit{Noise}$$

Definition 2.2 (This is an informal definition)

Underfitting *is the learning of a model which has low variance and high bias.*
Overfitting *is the learning of a model which has high variance and low bias.*

Remark 2.6

From now on, we need to add some dependencies regarding coefficients.

- $\mathcal{D} \subset \mathbb{R}^d \times \mathbb{R}$ denotes the dataset of n sampling points and sampling values $(\mathbf{x}_i, y_i)$ for $1 \leq i \leq n$.
- The $\mathbf{x}_i$ are understood as row vector (i.e. $\mathbf{x}_i \in \mathbb{R}^{1\times d}$). It may or may not include the intercept, depending on the model chosen and may or may not represent the feature of the sampling points and not the points themselves – this is usually clear in the context.
- We assume the y_i's to be coming from a linear (with respect to the regression parameters) model, and noisy: $y_i = f_\beta(\mathbf{x}_i) + \varepsilon_i = \mathbf{x}_i\beta + \varepsilon_i$.
- β denotes the regression variables.
- θ denotes the set of hyperparameters (e.g. λ in the ridge regression, the max degree of polynomial)
- $\widehat{\beta}$ in fact depends on
  1. the hyper parameters chosen
  2. the data set used to estimate

  We will therefore write $\widehat{\beta}(\lambda; \mathcal{D})$.
- The prediction depend on the hyperparameter and dataset $\rightarrow \widehat{y}_i = f(\mathbf{x}_i, \lambda; \mathcal{D}) = f_{\widehat{\beta}(\lambda;\mathcal{D})}(\mathbf{x}_i)$.

Definition 2.3

The goal of a learning algorithm is to minimize the **risk***: given a loss function (e.g. squared error or Hamming distance)* $\mathcal{L} : \mathbb{R}^d \times \mathbb{R} \to \mathbb{R}^+$*, we want to minimize*

$$\mathcal{R}(\widehat{f}) = \mathbb{E}\left[\mathcal{L}(\widehat{f}(x), y)\right] = \int \mathcal{L}\left(\widehat{f}(x), y\right) dP(x, y)$$

where $\widehat{f}$ *denotes the tested model and* $(x, y) \sim P(x, y)$*.*

Definition 2.4

The risk is unavailable – $P(x, y)$ is unknown – and we only have access to the **empirical risk**:

$$\mathcal{R}_{\mathcal{D}}(\widehat{f}) = \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(\widehat{f}(x_i), y_i).$$

Remark 2.7

Of course, one needs to be careful about the set of functions on which we are trying to optimize!
$\rightarrow$ high risk of overfitting!

Proposition 2.7

The law of large numbers ensures us that

$$\mathcal{R}_{\mathcal{D}}(\widehat{f}) \to \mathcal{R}(\widehat{f})$$

as the size of $\mathcal{D}$ increases to ∞: $|\mathcal{D}| \to \infty$.

Definition 2.5

Usually, a dataset is split into three parts:

- *A* **training set** *on which the model will be learned (=the regression parameters will be estimated)*
- *A* **validation set** *which does not overlap the training set and serves to evaluate the hyperparameter choices*
- *A* **test set** *which acts as dataset used for the empirical risk. This set is completely independent from the two previous ones.*

Always check what's what ...

Definition 2.6

A k **fold cross validation** *approach is such that the training set is split in k comparable subsets, each of which will be used as validation sets of a model trained on the $k-1$ others.*
The process is repeated k times for each tested parameters / model.

Definition 2.7

The k-fold cross validation with $k = n$ (the number of samples) is called **Leave-one-out cross validation**.

Remark 2.8 (Cross validation in practice)

Assume given

- $\mathcal{D} = \cup_{i=1}^{k} \mathcal{D}_k$ with $\mathcal{D}_i \cap \mathcal{D}_j = \varnothing$ if $i \neq j$, your k folded dataset
- a goodness of fit criteria (loss function / MSE / Huber / Hamming distance ...). We call it $\mathcal{L}$.

For a given choice of hyperparameters, set a global fit to 0 then iterate for $i = 1$ to k

- Learn the regression parameters based on the $k-1$ fold, excluding the i^{th} one.
- Evaluate the fit on the i^{th} fold.
- Combine the i^{th} fit with the global fit so far (for the current choice of hyperparameters).

And repeat this for all potential candidate hyperparameters. Choose the best global fit

Definition 2.8

An estimator in which the ℓ_1 norm of the regression parameters is used as a regularizer is known as the **LASSO** *operator (Least Absolute Shrinkage and Selection Operator). It optimizes the following criteria:*

$$\min \|X\beta - \mathbf{y}\|_2^2 + \lambda \|\beta\|_1.$$

Proposition 2.8

Assume the data matrix is orthonormal: $X^T X = I_d = (X^T X)^{-1}$. *Then the solution to the LASSO problem*

$$\min_{\beta \in \mathbb{R}^d} \frac{1}{2} \|X\beta - \mathbf{y}\|_2^2 + \lambda \|\beta\|_1$$

is given by

$$\widehat{\beta}(\lambda) = S_\lambda(\beta^{\ell_2})$$

where S_λ *is the soft thresholding operation defined component wise as*

$$S_\lambda(t) = \text{sign}(t)[t - \lambda, 0]_+ = \text{sign}(t) \max\{|t| - \lambda, 0\}$$

and where $\beta^{\ell_2} = (X^T X)^{-1} X^T \mathbf{y} = X^T \mathbf{y}$ *denotes the classical regression estimator.*

Proposition 2.9

LASSO with Forward Stagewise algorithm:

1. *set a small* $\eta > 0$
2. *Standardize (*0 *mean and unit variance) your data and center your targets* $\mathbf{y}$.
3. *Set the residual* $r = \mathbf{y}$ *and the predictors* $\beta = 0$.
4. *Find the predictor most correlated with* r*:* $j = \text{argmax}_i \left|\langle X_i^T, r\rangle\right|$
5. *Update* $\beta_j \leftarrow \beta_j + \eta \,\text{sign}(\langle X_j^T r\rangle)$.
6. *Update* $r \leftarrow r - \eta \,\text{sign}(\langle X_j^T r\rangle)$.
7. *Repeat the three previous steps as many times as necessary.*

Remark 2.9

The Forward Stagewise algorithm is not the most efficient for solving the LASSO problem. LARS (Least Angle Regression Shrinkage) is more efficient but

1. It is incredibly easy to implement.
2. It tends to work **really** well in high-dimensions.

Definition 2.9

The estimator which combines both ridge and LASSO estimator is known as the **ElasticNet**.

$$\min \|X\beta - \mathbf{y}\|_2^2 + \lambda R_\alpha(\beta)$$

with $R_\alpha(\beta) = \alpha\|\beta\|_2^2 + (1-\alpha)\|\beta\|_1$ *for some* $\alpha \in [0,1]$.

Remark 2.10

It is important to note the followings:

- $\alpha = 0$ yields the LASSO.
- $\alpha = 1$ gives the ridge estimator.
- we now have two hyper parameters which we need to evaluate $\rightarrow$ Grid Search may be used.