Regularized Regression

David M. Blei
Columbia University

December 15, 2015

Modern regression problems are high dimensional, which means that the number of covariates p is large. In practice statisticians *regularize* their models, veering away from the MLE solution to one where the coefficients have smaller magnitude. This lecture is about regularization. It draws on the ideas and treatment in Hastie et al. (2009) (referred to below as ESL).

1 The bias-variance trade off

We first discuss an important concept, the *bias-variance trade off*. In this discussion we will take a frequentist perspective.

Consider a set of random responses drawn from a linear regression with "true" parameter β^*,

$$Y_n \mid x_n, \beta^* \sim N(\beta^* x_n, \sigma^2). \tag{1}$$

The data are $D = \{(x_n, Y_n)\}$. Note that we are holding the covariates x_n fixed; only the responses are random. (We are also assuming x_n is a single covariate; in general, it is p-dimensional and we replace $\beta^* x_n$ with $\beta^{*\top} x_n$.)

With this data set, the maximum likelihood estimate is a *random variable* whose distribution is governed by the distribution of the data $\hat{\beta}(D)$. Recall that β^* is the true parameter that generated the responses. How close to we expect $\hat{\beta}(D)$ to be to β^*?

We can answer this question in a couple of ways. First, suppose we observe a new data input x. We consider the *mean squared error* of our estimate of $\mathbb{E}_{\hat{\beta}}[y \mid x] = \hat{\beta} x$. This is the difference between our predicted expectation of the response and the true expectation of the response,

$$\text{MSE} = \mathbb{E}_{\beta^*}\left[(\hat{\beta}(D)^\top x - \beta^{*\top} x)^2\right]. \tag{2}$$

It is important to keep track of which variables are random. The coefficient β^* is not random;

it is the true parameter that generated the data. The coefficient $\hat{\beta}(D)$ is random; it depends on the randomly generated data set D. The expectation in this equation is with respect to the randomly generated data set. (For simplicity, we will sometimes supress this notation below.)

The MSE decomposes in an interesting way,

$$\begin{aligned} \text{MSE} &= \mathbb{E}\left[(\hat{\beta}x)^2\right] - 2\mathbb{E}\left[\hat{\beta}x\right]\beta^* x + (\beta^* x)^2 \\ &= \mathbb{E}\left[(\hat{\beta}x)^2\right] - 2\mathbb{E}\left[(\hat{\beta}x)\right](\beta^* x) + (\beta^* x)^2 + \mathbb{E}\left[(\hat{\beta}x)\right]^2 - \mathbb{E}\left[(\hat{\beta}x)\right]^2 \\ &= \left(\mathbb{E}\left[(\hat{\beta}x)^2\right] - \mathbb{E}\left[\hat{\beta}x\right]^2\right) + \left(\mathbb{E}\left[\hat{\beta}x\right] - \beta^* x\right)^2 \end{aligned} \tag{3}$$

The second term is the squared *bias*,

$$\text{bias} = \mathbb{E}\left[\hat{\beta}x\right] - \beta^* x. \tag{4}$$

An estimate for which this term is zero is an *unbiased estimate*. The first term is the *variance*,

$$\text{variance} = \mathbb{E}\left[(\hat{\beta}x)^2\right] - \mathbb{E}\left[\hat{\beta}x\right]^2. \tag{5}$$

This reflects the spread of the estimates we might find on account of the randomness inherent in the data. Note that the decomposition holds for any linear function of the coefficients.

A famous result in statistics is the *Gauss-Markov theorem*. Recall that the MLE $\hat{\beta}$ is an unbiased estimate. The theorem states that the MLE is the unbiased estimate with the smallest variance. If you insist on unbiasedness, and you care about the MSE, then you can do no better than the MLE.

Often we care about *expected prediction error*. Suppose we observe a new input x. How wrong will we be on average when we predict the true $y \mid x$ with $\mathbb{E}[y \mid x]$ from a fitted regression?

The expected squared prediction error is

$$\mathbb{E}_{\mathcal{D}}\left[\mathbb{E}_Y\left[(\hat{\beta}x - Y)^2\right]\right]$$

The first expectation is taken for the randomness of $\hat{\beta}$, which is a function of the data. The

second is taken for the randomness of Y given x, which comes from the true model. This decomposes as follows,

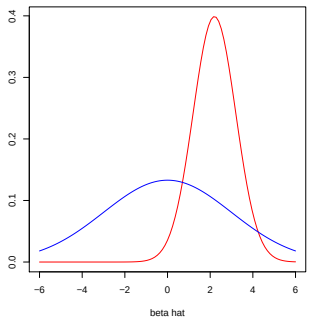
$$\mathbb{E}_{\mathcal{D}}\left[\mathbb{E}_{Y}\left[(\hat{\beta}x - Y)^2\right]\right] = \text{Var}(Y) + \text{MSE}(\hat{\beta}x) \quad (6)$$

$$= \sigma^2 + \text{Bias}^2(\hat{\beta}x) + \text{Var}(\hat{\beta}x). \quad (7)$$

The first term is the inherent uncertainty around the true mean; the second two terms are the bias variance decomposition of the estimator. We cannot do anything about the inherent uncertainty; thus reducing the MSE also reduces expected prediction error.

Classical statistics cared only about unbiased estimators. Modern statistics has explored the trade-off, where it may be worth accepting some bias for a reduction in variance. This can reduce the MSE and, consequently, the expected prediction error on future data.

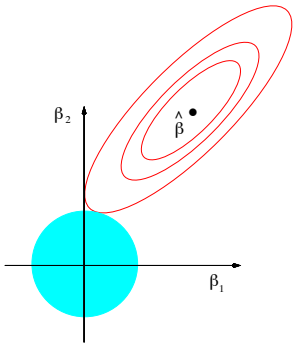
Here a simple picture to illustrate why:



It may be that the MSE is smaller for the biased estimator, because it nevers veers as far away from the truth as the unbiased estimator does.

2 Ridge regression

Regularization. In regression, we can make this trade-off with *regularization*, which means placing constraints on the coefficients β. Here is a picture from ESL for our first example.



In this picture, contours represent values of β with equal RSS (or, equivalently, likelihood). Our procedure finds the best value that is within the blue circle.

This reduces the variance because it limits the space that the parameter vector β can live in. If the true MLE of β lives outside that space, then the resulting estimate *must* be biased because of the Gauss-Markov theorem.

The picture also shows how regularization encourages smaller and perhaps "simpler" models. Simpler models are more robust to *overfitting*, generalizing pooly because of a close match to the training data. Simpler models can also be more *interpretable*, which is another goal of regression. (This is particularly true for the lasso, which we will talk about later.)

Ridge regression. Let's discuss the details of ridge regression. We optimize the RSS subject to a constraint on the sum of squares of the coefficients,

$$\begin{aligned} &\text{minimize} && \textstyle\sum_{n=1}^{N} \frac{1}{2}(y_n - \beta x_n)^2 \\ &\text{subject to} && \textstyle\sum_{i=1}^{p} \beta_i^2 \leq s \end{aligned} \tag{8}$$

This constrains the coefficients to live within a sphere of radius s. (See the picture.) Question: What happens as the radius increases? Answer: Variance goes up; bias goes down.

With some calculus, the ridge regression estimate can also be expressed as

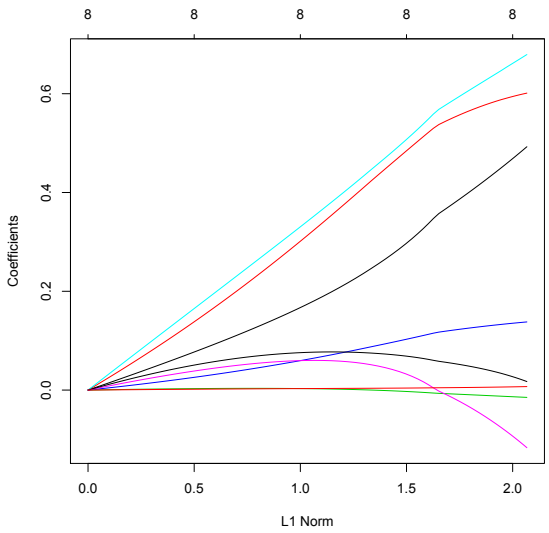
$$\hat{\beta}^{\text{ridge}} = \arg\min_{\beta} \sum_{n=1}^{N} \frac{1}{2}(y_n - \beta x_n)^2 + \lambda \sum_{i=1}^{p} \beta_i^2 \tag{9}$$

This is nice because the problem is convex. Further, it has an analytic solution. (See the reading.) Question: Is it sensitive to scaling? Answer: Yes, in practice we center and scale

the covariates.

There is a 1-1 mapping between the radius s and *complexity parameter* λ. Either of these parameters trades off an increase in bias for a decrease in variance.

From ESL:



How do we choose λ? As we see, the value of the complexity parameter affects our estimate. Question: What would happen if we used training error as the criterion? (Look at the picture to see the answer.)

In practice, we choose λ by *cross validation*. This is an attempt to minimize expected test error. (But later on we will discuss hierarchical models. This can be another way to choose the regularization parameter.)

Here is how it works:

- Divide the data into K folds (e.g., $K = 10$).
- Decide on candidate values of λ (e.g., a grid between 0 and 1)
- For each fold k and value of λ,
  - Estimate $\hat{\beta}_k^{\text{ridge}}$ on the *out-of-fold* samples.
  - For each x_n assigned to fold k, compute its squared error

$$\epsilon_n = (\hat{y}_n - y_n)^2, \tag{10}$$

where $\hat{y}_n = \mathbb{E}_{\hat{\beta}_k^{\text{ridge}}}[Y \mid x_n]$. Note that this estimate of the coefficients did not use (x_n, y_n) as part of its training data.

- We now aggregate the individual errors. The score for λ is

$$\text{MSE}(\lambda) = \frac{1}{N}\sum_{n=1}^{N} \epsilon_n. \tag{11}$$

This is an estimate of the test error. Choose λ that minimizes this score.

Aside: Connection to Bayesian statistics. We have motivated regularized regression via frequentist thinking, i.e., the bias-variance trade-off and an appeal to the true model. Regularized regression, in general, has connections to Bayesian modeling.

We have discussed two common ways of using the posterior to obtain an estimate. The first is *maximum a posteriori (MAP)* estimation,

$$\theta^{\text{MAP}} = \arg\max_{\theta} p(\theta \mid y_1, \ldots, y_N, \alpha) \tag{12}$$

The second is the posterior mean,

$$\theta^{\text{mean}} = \mathbb{E}[\theta \mid y_1, \ldots, y_N, \alpha] \tag{13}$$

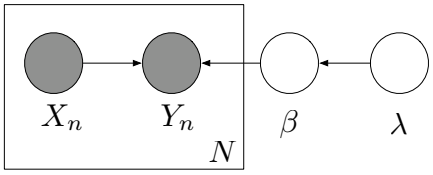
Question: How are these different from the MLE?

Ridge regression and Bayesian methods. Ridge regression corresponds to MAP estimation in the following model:

$$\beta_i \sim N(0, 1/\lambda) \tag{14}$$

$$y_n \mid x_n, \beta \sim N(\beta^\top x_n, \sigma^2) \tag{15}$$

Here is the corresponding graphical model



[This isn't quite right; λ should be a small dot.]

We will derive the relationship. First, note that

$$p(\beta_i \mid \lambda) = \frac{1}{\sqrt{2\pi(1/\lambda)}} \exp\{\lambda \beta_i^2\} \tag{16}$$

We now compute the MAP estimate of β,

$$\begin{aligned}
\max_{\beta} \; p(\beta \mid D, \lambda) &= \max_{\beta} \log p(\beta \mid y_{1:N}, x_{1:N}, \lambda) && (17)\\
&= \max_{\beta} \log p(\beta, y_{1:N} \mid x_{1:N}, \lambda) && (18)\\
&= \max_{\beta} \log \left(p(y_{1:N} \mid x_{1:N}, \beta) \prod_{i=1}^{p} p(\beta_i \mid \lambda) \right) && (19)\\
&= \max_{\beta} -\text{RSS}(\beta; D) - \sum_{i=1}^{p} \lambda \beta_i^2. && (20)
\end{aligned}$$

Ridge regression is equivalent to MAP estimation in the model.

Observe that the hyperparameter λ controls how far away the estimate will be from the MLE. A small hyperparameter (large variance) will choose the MLE; the data totally determine the estimate. As the hyperparameter gets larger, the estimate moves further from the MLE; the prior ($\mathbb{E}[\beta] = 0$) becomes more influential. This matches our recurring theme in Bayesian estimation; both the data and the prior influence the answer.

Finally, note that a "true" Bayesian would not set the hyperparameter by cross-validation. This uses the data to set the prior. However, I think it is a good idea. It is an instance of a more general principle called "Empirical Bayes".

Summary of ridge regression.

1. We constrain β to be in a hypersphere around 0.

2. This is equivalent to minimizing the RSS plus a regularization term.

3. We no longer find the $\hat{\beta}$ that minimizes the RSS. (Contours illustrate constant RSS.)

4. Ridge regression is a kind of *shrinkage*, so called because it reduces the components to be close to 0 and close to each other.

5. Ridge estimates trade off bias for variance.

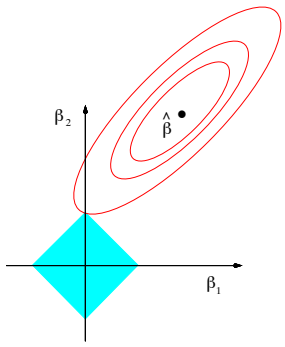
3 The lasso

A closely related regularization method is called *the lasso*. The lasso optimizes the RSS subject to a different constraint,

$$\begin{aligned} &\text{minimize} && \textstyle\sum_{n=1}^{N} \frac{1}{2}(y_n - \beta x_n)^2 \\ &\text{subject to} && \textstyle\sum_{i=1}^{p} |\beta_i| \leq s \end{aligned} \tag{21}$$

This small change yields very different estimates. Here is the picture of the constraint:

From ESL:



Question: What happens as s increases? Question: Where is the solution going to lie with s fixed?

It's a fact: unless it chooses $\hat{\beta}$, the lasso (with p large) will set some of the coefficients to exactly zero. The intuitions come from ESL: "Unlike the disk, the diamond has corners; if the solution occurs at a corner, then it has one parameter β_j equal to zero. When $p > 2$, the diamond becomes a rhomboid, and has many corners, flat edges and faces; there are many more opporunities for the estimated parameters to be zero." (p 90).

In a sense, the lasso is a form of *feature selection*, identifying a relevant subset of the covariates with which to predict. Like ridge regression, it trades off an increase in bias with a decrease in variance. Further, by zeroing out some of the covariates, it provides interpretable (as in, sparse) models.

Sparse models can also be important in real systems that *might* depend on many inputs. Once the sparse solution is found, we need only measure a few of the inputs in order to make predictions. This speeds up the performance of the system.

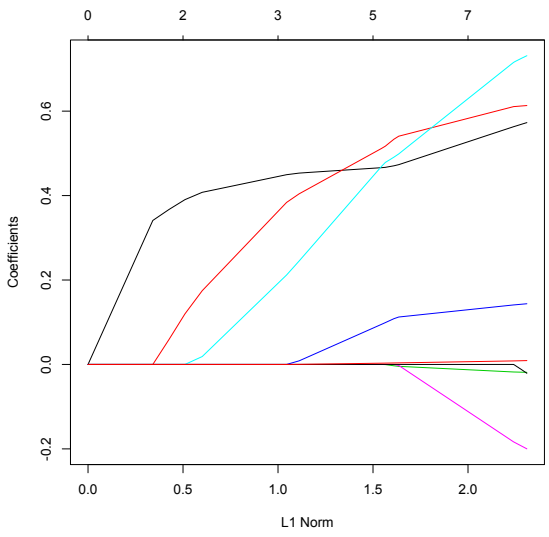
The lasso is equivalent to

$$\hat{\beta}^{\text{lasso}} = \arg\min_{\beta} \sum_{n=1}^{N} \frac{1}{2}(y_n - \beta x_n)^2 + \lambda \sum_{i=1}^{p} |\beta_i| \tag{22}$$

Again, there is a 1-1 mapping between λ and s. This objective, though it does not have an analytic solution, is still convex.

Why is the lasso exciting? Prior to the lasso, the only "sparse" method was *subset selection*, finding the best subset of features with which to model the data. But subset selection has problems: searching over all subsets (of a fixed size) is computationally expensive. In contrast, the lasso efficiently finds a sparse solution by using convex optimization. In a sense, it is akin to a "smooth version" of subset selection. Note the lasso won't consider all possible subsets.

From ESL:



The Bayesian interpretation of the lasso. Like ridge regression, lasso regression corresponds to MAP estimation in a Bayesian model. For the lasso, the model is:

$$\beta_i \sim \text{Laplace}(\lambda) \tag{23}$$

$$Y_n \mid x_n, \beta \sim N(\beta^\top x_n, \sigma^2). \tag{24}$$

Here the coefficients come from a *Laplace distribution*,

$$p(\beta_i \mid \lambda) = \frac{1}{2} \exp\{-\lambda|\beta_i|\}. \tag{25}$$

The lasso, and the general idea of L1 penalized models, has become a cottage industry in modern statistics and machine learning. The reason is that we often want sparse solutions to high-dimensional problems, and we want convex objective functions when analyzing data. L1 penalized methods give us both. Recent research indicates that they have good theoretical properties to boot.

4 (Optional) Generalized regularization

In general, regularization can be seen as minimizing the RSS with a constraint on a q-norm,

$$\text{minimize} \quad \textstyle\sum_{n=1}^{N} \frac{1}{2}(y_n - \beta x_n)^2$$

$$\text{subject to} \quad ||\beta||_q \leq s$$

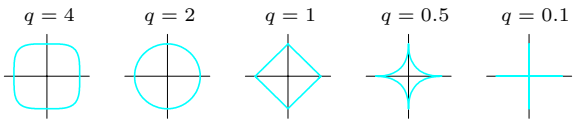
, where the penalty is

$$||\beta||_q = \left(\sum_{i=1}^{p} |\beta_i|^q \right)^{1/q}$$

The methods we discussed so far are

- $q = 2$: ridge regression
- $q = 1$: lasso
- $q = 0$: subset selection

Here is the picture from ESL:



- This brings us away from the minimum RSS solution, but might provide better test prediction via the bias/variance trade-off.

- Complex models have less bias; simpler models have less variance. Regularization encourages simpler models.

Note that each of these methods correspond to a Bayesian solution with a different choice of prior.

$$\hat{\beta}^{\text{ridge}} = \arg\min_{\beta} \sum_{n=1}^{N} \frac{1}{2}(y_n - \beta x_n)^2 + \lambda ||\beta||_q$$

The complexity parameter λ can be chosen with cross validation.

Lasso ($q = 1$) is the only norm that provides sparsity and convexity.

And there are other variants, useful in the literature. Of note:

- The elastic net is a convex combination of L_1 and L_2.
- The grouped lasso finds sparse groups of covariates to include.

Finally, the `glmnet` package in `R` is amazing. It efficiently computes models for a regularization path using L_2 or L_1 penalization. It uses the same model syntax as `lm` or `glm`.

References

Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The Elements of Statistical Learning*. Springer, 2 edition.