Kevin Jamieson

Regression with Side Information

In linear regression we have seen how penalizing the ℓ_2-norm of the weights (Ridge) and ℓ_1-norm of the weights (Lasso) affect the resulting solutions. Using different loss functions and regularizers to obtain different desired behaviors is very popular in machine learning. First let's generate some data. Let $n = 50$ and $f(x) = 10\sum_{k=1}^{4} \mathbf{1}\{x \geq \frac{k}{5}\}$, noting that $f(x)$ is non-decreasing in x (i.e., $f(x) \geq f(z)$ whenever $x \geq z$). For $i = 1, \ldots, n$ let each $x_i = \frac{i-1}{n-1}$ and $y_i = \mathbf{1}\{i \neq 25\}(f(x_i) + \epsilon_i)$ where $\epsilon_i \sim \mathcal{N}(0, 1)$. The case where $i = 25$ represents an outlier in the data.

In the last homework we solved a problem of the form $\arg\min_\alpha \sum_{i=1}^n \ell(y_i - \sum_{j=1}^n k(x_i, x_j)\alpha_j)$ where $\ell(z) = \ell_{ls}(z) := z^2$ was the least squares loss. Least squares is the MLE for Gaussian noise, but is very sensitive to outliers. A more robust loss is the Huber loss:

$$\ell_{huber}(z) = \begin{cases} z^2 & \text{if } |z| \leq 1 \\ 2|z| - 1 & \text{otherwise} \end{cases}$$

which acts like least squares close to 0 but like the absolute value far from 0. Moreover, define a matrix $D \in \{-1, 0, 1\}^{(n-1)\times n}$

$$D_{i,j} = \begin{cases} -1 & \text{if } i = j \\ 1 & \text{if } i = j - 1 \\ 0 & \text{otherwise} \end{cases}$$

so that for any vector $z \in \mathbb{R}^n$ we have $Dz = (z_2 - z_1, z_3 - z_2, \ldots z_n - z_{n-1})$ In what follows let $k(x, z) = \exp(-\gamma\|x - z\|^2)$ where $\gamma > 0$ is a hyperparameter.

a. As a baseline, let

$$\widehat{\alpha} = \arg\min_\alpha \sum_{i=1}^n \ell_{ls}(y_i - \sum_{j=1}^n K_{i,j}\alpha_j) + \lambda\alpha^T K\alpha \ , \qquad \widehat{f}(x) = \sum_{i=1}^n \widehat{\alpha}_i k(x_i, x)$$

where $K_{i,j} = k(x_i, x_j)$ is a kernel evaluation and λ is the regularization constant. Plot the original data $\{(x_i, y_i)\}_{i=1}^n$, the true $f(x)$, the $\widehat{f}(x)$ found through leave-one-out CV. (Hint: start with the problem on the homepage of `http://www.cvxpy.org/` and modify it as needed.)

b. Now let

$$\widehat{\alpha} = \arg\min_\alpha \sum_{i=1}^n \ell_{huber}(y_i - \sum_{j=1}^n K_{i,j}\alpha_j) + \lambda\alpha^T K\alpha \ , \qquad \widehat{f}(x) = \sum_{i=1}^n \widehat{\alpha}_i k(x_i, x)$$

where $K_{i,j} = k(x_i, x_j)$ is a kernel evaluation and λ is the regularization constant. Plot the original data $\{(x_i, y_i)\}_{i=1}^n$, the true $f(x)$, the $\widehat{f}(x)$ found through leave-one-out CV. (Hint: `huber` is a function in `cvxpy`.)

c. The total variation (TV) of a real-valued function g over $\{1, \ldots, n\}$ is defined as $\int_{i=1}^{n-1} |g_i - g_{i-1}| = ||Dg||_1$ and is a common regularizer for de-noising a function (its two-dimensional counterpart is very popular for image de-noising or filling-in missing/damaged parts of photos). Let

$$\widehat{\alpha} = \arg\min_\alpha ||K\alpha - y||^2 + \lambda_1||DK\alpha||_1 + \lambda_2\alpha^T K\alpha \ , \qquad \widehat{f}(x) = \sum_{i=1}^n \widehat{\alpha}_i k(x_i, x)$$

where $K_{i,j} = k(x_i, x_j)$ is a kernel evaluation and λ_1, λ_2 are regularization constants. For intuition, the penalizer $||DK\alpha||_1$ prefers functions $\widehat{f}$ with sparse jumps in function value while $\alpha^T K \alpha$ perfers functions that are smoothly varying. On your own (not necessary to report plots), plot $\widehat{f}$ for a variety values of $\gamma, \lambda_1, \lambda_2$ to see how they affect the solution. Use leave-one-out cross validation to find a good setting of $\gamma, \lambda_1, \lambda_2$. Plot the original data $\{(x_i, y_i)\}_{i=1}^n$, the true $f(x)$, the $\widehat{f}(x)$ found through leave-one-out CV.

d. We say a function g over $\{1, \ldots, n\}$ is non-decreasing if $g_i - g_{i-1} \geq 0$ for all i, or $Dg \geq 0$ where the inequality applies elementwise. Perhaps due to domain knowledge, we know that the original unnoisy function we are trying to estimate is non-decreasing. Let

$$\widehat{\alpha} = \arg\min_{\alpha} ||K\alpha - y||^2 + \lambda \alpha^T K \alpha \ , \qquad \widehat{f}(x) = \sum_{i=1}^{n} \widehat{\alpha}_i k(x_i, x)$$
$$\text{subject to} \quad DK\alpha \geq 0$$

where $K_{i,j} = k(x_i, x_j)$ is a kernel evaluation and λ is a regularization constant. The above is known as a quadratic program because it can be written as $\arg\min_x \frac{1}{2} x^T Q x + p^T x + c$ subject to $Ax \leq b$. On your own (not necessary to report plots), plot $\widehat{f}$ for a variety values of γ, λ to see how they affect the solution. Use leave-one-out cross validation to find a good setting of γ, λ. Plot the original data $\{(x_i, y_i)\}_{i=1}^n$, the true $f(x)$, the $\widehat{f}(x)$ found through leave-one-out CV. Note that the defined constraint only forces $\widehat{f}$ to be monotonic on the training data, not over all $x \in [0, 1]$, but it is instructive to think about how one might achieve this.