CS142: Machine Learning Spring 2017

Lecture 5

Instructor: Pedro Felzenszwalb *Scribes: Dan Xiang, Tyler Dae Devlin*

Robust Regression and Linear Programming

Robust Regression

Let's recall the setup for least-squares regression. We are given a set of training data $T = \{(x_1, y_1), \ldots, (x_N, y_N)\}$, where $x_i \in \mathbb{R}^D$ and $y_i \in \mathbb{R}$. Depending on the specific application, we devise an appropriate feature map $\phi : \mathbb{R}^D \to \mathbb{R}^M$. In least-squares regression, we attempt to minimize the sum-of-squares error function given by

$$E(w) = \sum_{i=1}^{N} (w^T \phi(x_i) - y_i)^2.$$

We noted that because we are using squared errors, our model is particularly sensitive to outliers. To avoid this problem, we might use an alternative error function,

$$Q(w) = \sum_{i=1}^{N} |w^T \phi(x_i) - y_i|,$$

which uses absolute — rather than squared — differences. Minimizing Q leads us to the least absolute deviations (LAD) regression method.

Just as least-squares regression could be interpreted in a maximum likelihood framework under the assumption of Gaussian errors, LAD regression is equivalent to maximum likelihood estimation of w assuming the errors e_i are distributed according to a Laplace$(0, b)$ distribution.

Linear Programming

Although LAD regression is more robust to outliers, minimizing the sum of absolute differences is a more difficult optimization problem which requires the use of *linear programming*. Here we briefly introduce the basics of linear programming.

A linear programming problem consists of the following components.

1. A set of variables,
$$\xi = (\xi_1, \ldots, \xi_n), \quad \xi_i \in \mathbb{R}.$$
2. A linear objective function to minimize,
$$f(\xi) = C^T \xi, \quad C \in \mathbb{R}^n.$$
3. A set of linear constraints of the form
$$a^T \xi \geq b \quad \text{or} \quad a^T \xi \leq b \quad \text{or} \quad a^T \xi = b,$$
wher $a \in \mathbb{R}^n$ and $b \in \mathbb{R}$.

In MATLAB, we would write $\xi = \texttt{linprog}(C, A, b)$ to solve
$$\xi^* = \operatorname*{argmin}_{\xi} C^T \xi$$
subject to the constraints $A\xi \leq b$, where $C \in \mathbb{R}^n$, $A \in \mathbb{R}^{m \times n}$, and $b \in \mathbb{R}^m$.

Notation: For two vectors $u, v \in \mathbb{R}^n$, we say that $u \leq v$ if $u_i \leq v_i$ for all $i \in \{1, \ldots, n\}$.

There are three things that can happen in a linear programming problem.

1. We find an optimal solution.
2. There is no feasible ξ that satisfies the constraints.
3. The linear program may be unbounded, i.e. there is no minimum solution even though it is possible to satisfy the constraints.

We now construct a linear program to minimize the absolute deviations error function $Q(w)$.

1. First we define our variables. Let $\xi = (w_1, \ldots, w_M, e_1, \ldots, e_N) \in \mathbb{R}^{N+M}$.
2. Next we define the objective function that we would like to minimize. In the context of LAD regression, we want to minimize the sum of absolute differences, $\sum_{i=1}^N e_i = C^T \xi$ where $C = (0, \ldots, 0, 1, \ldots, 1) \in \mathbb{R}^{M+N}$. (The first M elements of C are 0, and the last N elements are 1.)

3. Finally we have constraint that for each $i \in \{1, \dots, N\}$,

$$e_i = |w^T \phi(x_i) - y_i|.$$

It turns out that it is sufficient to require that $|w^T \phi(x_i) - y_i| \leq e_i$. We can rewrite this constraint equivalently as the two constraints,

$$-e_i \leq w^T \phi(x_i) - y_i \quad \text{and} \quad w^T \phi(x_i) - y_i \leq e_i.$$

(Recall that $|a| \leq b \Rightarrow -b \leq a \leq b$.)

Now for each i we have two inequality constraints:

$$\begin{aligned} e_i &\geq w^T \phi(x_i) - y_i, & (1) \\ e_i &\geq -(w^T \phi(x_i) - y_i). & (2) \end{aligned}$$

We can write constraint (1) as

$$\begin{bmatrix} \phi(x_i) & 0 & \dots & 0 & -1 & 0 & \dots & 0 \end{bmatrix} \cdot \xi \leq y_i,$$

where the -1 is in the $(M+i)^{th}$ entry of the vector. Note that $\phi(x_i) \in \mathbb{R}^M$ and takes up the first M entries of the vector.

By the same token, we can write constraint (2) as

$$\begin{bmatrix} -\phi(x_i) & 0 & \dots & 0 & -1 & 0 & \dots & 0) \end{bmatrix} \cdot \xi \leq -y_i.$$

We want to minimize $C^T \xi$ subject to $A\xi \leq b$, where $A \in \mathbb{R}^{2N \times (N+M)}$ is

$$A = \begin{bmatrix} \phi(x_1) & -1 & 0 & 0 & \dots & 0 \\ -\phi(x_1) & -1 & 0 & 0 & \dots & 0 \\ \phi(x_2) & 0 & -1 & 0 & \dots & 0 \\ -\phi(x_2) & 0 & -1 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \end{bmatrix}$$

and $b \in \mathbb{R}^{2N}$ is

$$b = \begin{bmatrix} y_1 \\ -y_1 \\ y_2 \\ -y_2 \\ \vdots \\ y_N \\ -y_N \end{bmatrix}$$

This is a big linear programming problem.

Regularization

Regularization is a way to combat against overfitting. The idea is that since overfitting can occur when the model is to complex for the amount of data available, we should penalize our estimates of the target funciton in proportion to the complexity of those estimates. In the context of linear models, we use a regularizer that keeps the weight values from becoming too large. More precisely, define a new error function E' by

$$E'(w) = \lambda w^T w + \frac{1}{2} \sum_{i=1}^{N} (w^T \phi(x_i) - y_i)^2, \tag{3}$$

where $\lambda w^T w$ is the regularization term with $\lambda \in \mathbb{R}$. As before, we estimate the target function by minimizing $E'(w)$. Adjusting the value of λ changes the relative importance of the regularization term. Large values of λ result in less extreme estimates of the target function, at the cost of larger in-sample sum-of-squares errors.

It can be shown that minimizing $E(w)$ is equivalent to maximizing $P(T|w)$, where T is the training set. Maximum a posteriori (MAP) estimation is a framework that seeks to maximize

$$P(w|T) = \frac{P(T|w)P(w)}{P(T)}.$$

Minimizing (3) gives the MAP estimate of w.