Regression and SVM

October 27, 2015

Abstract

1 Logistic Regression

In past discussions, we've often encountered the objective function of our MLE estimate, or of some other estimate that we've established for some parameter θ that we'd like to minimize so as to minimize the value of our error function. This is called **regression**. Here we will discuss **logistic regression**, which involves fitting the data to a curve of the form

$$\sigma(x) = \frac{1}{1+e^{-x}}$$

This is known as the **sigmoid** function, and it's often used when the data we would like to fit is well-modeled as a classification problem (with labels +1 and -1). Given d-dimensional data $x^{(1)}, x^{(2)}, ..., x^{(n)}$ and corresponding scalar labels $y^{(1)}, y^{(2)}, ..., y^{(n)}$, we can define our LR objection function as:

$$\text{NLL}(w) = \sum_{i=0}^{n} \log(1 + e^{-y^{(i)}(x^{(i)}\cdot w + w_0)})$$

given the parameters w, w_0. To reduce overfitting, we can apply an $L2$ regularization term to our equation, as in ridge regression. As a result, we are simply minimizing

$$E_{LR}(w) = \text{NLL}(w) + \lambda w^\intercal w$$

so that

$$w^* = \text{argmin}_w \left[\sum_{i=0}^{n} \log(1 + e^{-y^{(i)}(x^{(i)}\cdot w + w_0)}) + \lambda w^\intercal w\right]$$

2 Support Vector Machines

Support Vector Machines are a machine-learning technique that is used to classify binary sets of data. We define θ as the normal to the decision hyperplane, and classify data points by $\text{sgn}(\theta^\intercal x + \theta_0)$. In order to train the classifier, we initially set up the minimization problem (known as **Hard-SVM**):

$$\max_{\theta,\theta_0} \frac{1}{||\theta||} \min_{1\leq i\leq n} y^{(i)}(\theta^\intercal x^{(i)} + \theta_0) \tag{1}$$

However, we cannot satisfy this minimization if the training set is not linearly separable. Therefore, we add a "slack variable" to each constraint, which is a measure of the "wrongness" of the decision boundary when classifying that point. We call these slack variables ξ_i. We then desire to minimize

$$\min_{\theta,\theta_0,\xi} \frac{\lambda}{2}||\theta||^2 + \frac{1}{n}\sum_{i=1}^{n}\xi_i, \tag{2}$$

$$s.t. \quad y^{(i)}(\theta^\intercal x^{(i)} + \theta_0) \geq 1 - \xi_i, \tag{3}$$

$$\xi_i \geq 0, i \in n \tag{4}$$

However, we find it more computationally efficient to solve the dual form of the **Soft-SVM**:

$$\max_{\alpha\in\mathbb{R}^n}\left[\sum_{i=1}^{n}\alpha_i-\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y^{(i)}y^{(j)}(x^{(i)})^\intercal x^{(j)}\right] \tag{5}$$

$$s.t.\quad 0\le\alpha_i\le C \tag{6}$$

$$\sum_i \alpha_i y^{(i)}=0 \tag{7}$$

To provide some intuition: α_i is the weight of each training point on the final decision boundary. It is 0 for all $x^{(i)}$ that are farther from the decision boundary than the margin, so that only the "important" $x^{(i)}$ are "support vectors". C is the maximum value of α, and determines the size of the margin. **ADD SOMETHING ABOUT WHICH WAY THAT WORKS HERE**. To return from α to the more familiar θ and θ_0, we plug in:

$$\theta=\sum_{i=1}^{n}\alpha_i y^{(i)}x^{(i)} \tag{8}$$

$$\theta_0=\frac{1}{\mathcal{M}}\left[\sum_{j\in\mathcal{M}}\left(y^{(j)}-\sum_{i\in\mathcal{S}}\alpha_i y^{(i)}(x^{(j)})^\intercal x^{(i)}\right)\right] \tag{9}$$

Where $\mathcal{M}=\{i:0<\alpha_i<C\}$ and $\mathcal{S}=\{i:0<\alpha_i\}$.

For example, given the data $X=[1,2;\ 2,2;\ 0,0;\ -2,3], Y=[1;\ 1;\ -1;\ -1]$, we generate the following objective function:

$$\min_{\alpha\in\mathbb{R}^n}\left[\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y^{(i)}y^{(j)}(x^{(i)})^\intercal x^{(j)}-\sum_{i=1}^{n}\alpha_i\right] \tag{10}$$

This can be reformulated to be plugged into a standard quadratic programming solver, with **blah blah blah**

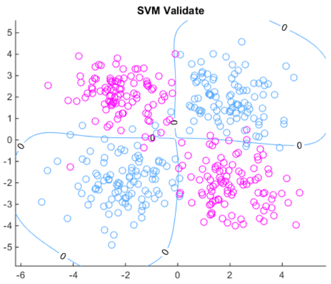

Figure 1: $C=0.01,\sigma=1$

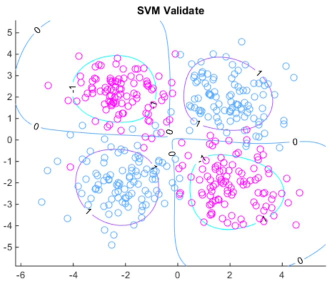

Figure 2: $C=0.1,\sigma=1$

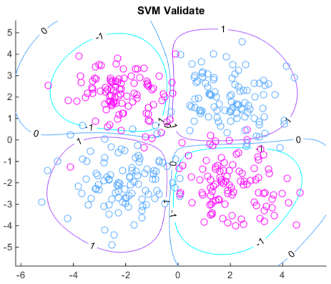

Figure 3: $C=1,\sigma=1$

We also note that as C increases, the number of boundary points decreases. Thus, the number of support vectors decrease. For the nonseparable example, for $C=(.01,.1,1,10,100)$, we see that $\mathcal{M}=(400,324,133,68,59)$.

As per the usual procedure with the approximately equivalent λ, to select the correct C, we first compute α for various values of C using a training set, then determine the validation error using a validation set. We select the C which produces the lowest validation error in order to maximize the generality of our solution.

3 Titanic Data

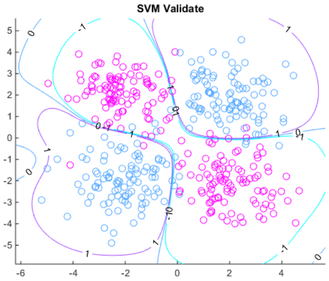

Figure 4: $C = 10, \sigma = 1$

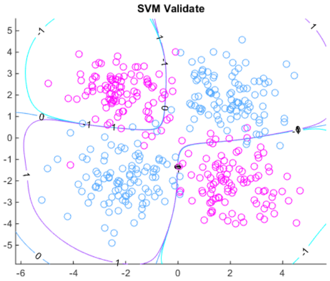

Figure 5: $C = 100, \sigma = 1$