Regression and SVM

October 27, 2015

Abstract

Previously, we discussed various methods for polynomial regression and minimizing error in our objective function, and explored famous examples, namely ridge regression and other norms (i.e. LAD for the L1 norm). Here, we will extend those concepts to a few classification problems, through logistic regression (a natural adaptation of the polynomial regression we've been doing) and then more formally through the concept of support vector machines (SVMs).

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

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

We can estimate w numerically using stochastic gradient descent; if we let $\lambda = 0$, then we have no regularization and we end up with this data, as well as figures 1-8:

Table 1: $\lambda = 0$ (step = .001, threshold = .008)

	`stdev1`		`stdev2`		`stdev4`		`data_nonsep`	
w_{init}	#it	Loss	#it	Loss	#it	Loss	#it	Loss
$[1, 1, 1]$	255	0	91	0.09	49	0.32	37	0.48
$[4, 4, 4]$	466	0	259	0.12	91	0.36	64	0.44
$[1, 10, 0]$	408	0	354	0.11	39	0.33	32	0.53
$[-5, -5, -5]$	299	0	199	0.1	89	0.33	73	0.49

However, if we vary λ, then we obtain more stable solutions with respect to w, and the solutions are more constant regardless of the starting paramters. Figure 9-12 show the decision bounday for $\lambda = 20$.

Figures 1-4 show the training data, and figures 5-8 show the validation data for $\lambda = 0$. The predictor is reasonably effective and seems to seek to minimize the loss. When we increase λ, however, we instead increase the loss as we try to minimize the $\lambda w^\intercal w$ term in the error function. This means that the model is more inaccurate in this case, which is because with two parameters, w wasn't likely going to overfit to the data anyways. But note that the effect is not too large, even for larger λ ($\lambda = 40$) - our main goal is still to minimize the total error.

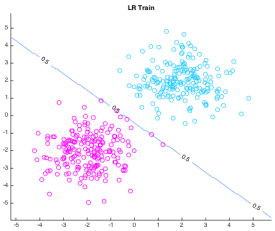

Figure 1: `stdev = 1`

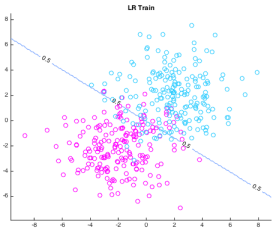

Figure 2: `stdev = 2`

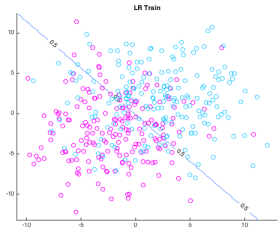

Figure 3: `stdev = 4`

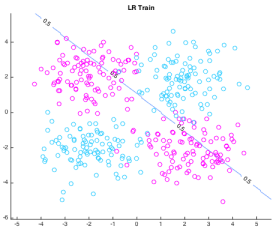

Figure 4: `unseparable`

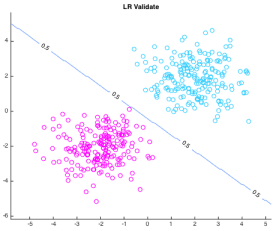

Figure 5: `stdev = 1`

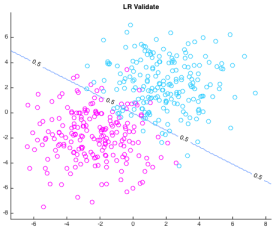

Figure 6: `stdev = 2`

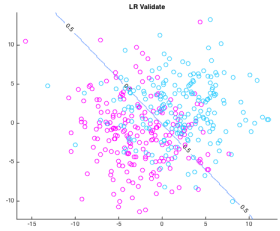

Figure 7: `stdev = 4`

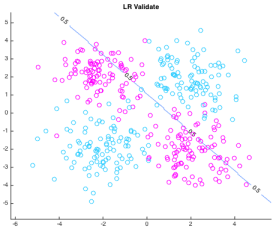

Figure 8: `unseparable`

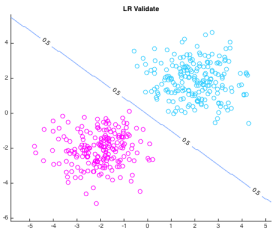

Figure 9: `stdev = 1`

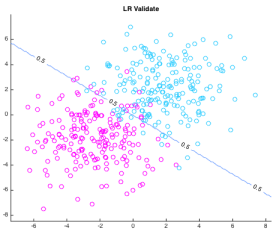

Figure 10: `stdev = 2`

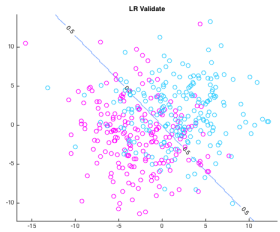

Figure 11: `stdev = 4`

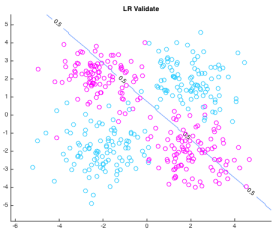

Figure 12: `unseparable`

2 Support Vector Machines

2.1 Overview and Formulation

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**:

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

$$s.t. \;\; 0 \leq \alpha_i \leq C \tag{6}$$

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

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. In particular, for small C, the margin is very large, and vice versa.. 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 and constraints (taking the opportunity to insert a kernel function, K):

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

$$H_{i,j} = y^{(i)} y^{(j)} K(x^{(i)}, x^{(j)}) \tag{11}$$

$$f = [-1\ \ -1\ \ \ldots\ \ -1] \text{ (of length } n) \tag{12}$$

$$s.t. \;\; Y^\intercal \alpha = 0 \tag{13}$$

$$s.t. \;\; 0 \leq \alpha_i \leq C, \forall i \in n \tag{14}$$

2.2 Testing Implementation

Applying our implementation of the soft-SVM to the 4 datasets from part 1, setting $C = 1$, we obtain the classifiers in 13 - 16. We note that the classifiers are reasonable for the three separable data sets, and that the margin grows as the standard deviation of the test sets increases. This makes sense - we are less sure about the classification for these, so the margin grows. Our validation errors were 0.0822, 68.67, 227.0, and 404.7, respectively. The training errors were 1.16, 88.56, 240.9, and 396.6, respectively. Again, these errors are logical and match what we expect - as the data becomes less easily separable, the error increases.

2.3 Kernel Extension

As was briefly mentioned in 2.1, we can easily insert a kernel function into the dual form of the SVM. In this particular case, we choose to insert the Gaussian Radial Basis Function (RBF):

$$K(x_1, x_2) = \exp\left(-\frac{||x_1 - x_2||^2}{2\sigma^2} \right) \tag{15}$$

This is particularly useful, because instead of looking at the dot product of x_1 and x_2, we instead may examine the distance between them, allowing us to find radially clustered groups. In this formulation, σ is a scalar, and so all

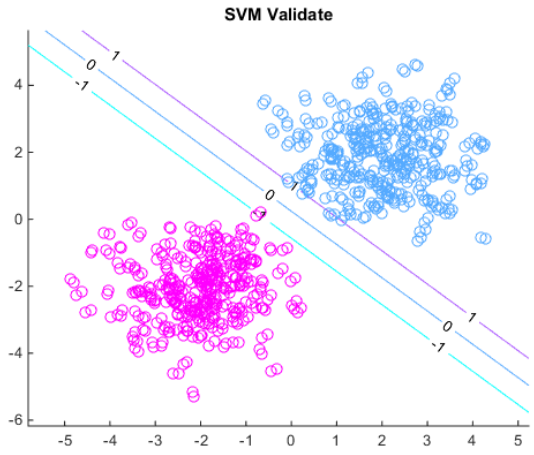

Figure 13: `stdev1`

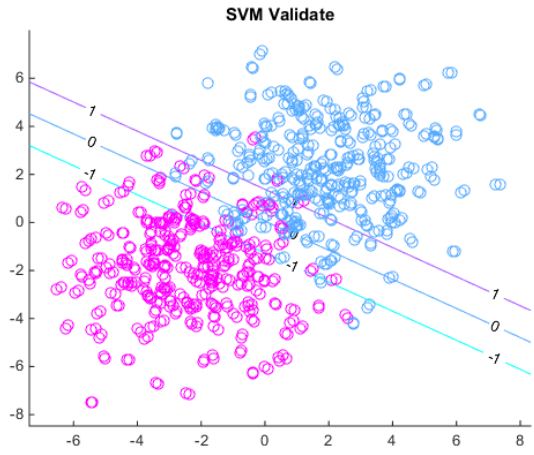

Figure 14: `stdev2`

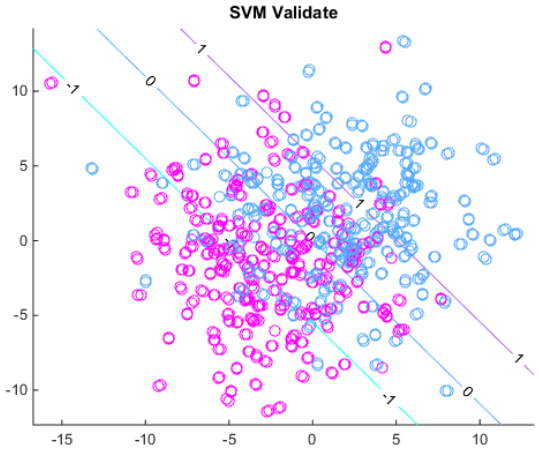

Figure 15: `stdev4`

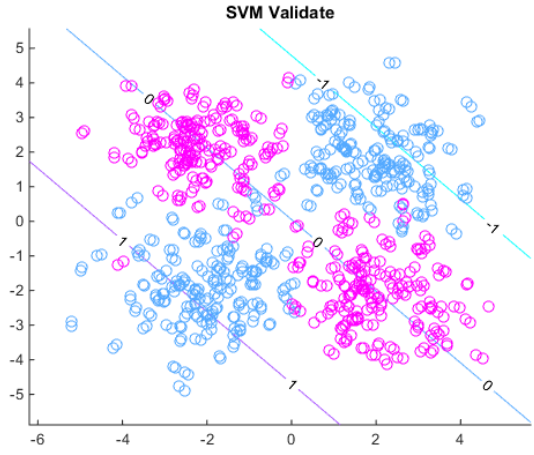

Figure 16: `nonsep`

dimensions must be normalized with respect to their std deviation in order to obtain meaningful results. Alternative formulations exist that have vector σ to account for variances in std deviation. Applying the RBF kernel to the `nonsep` dataset, setting $\sigma = 1$ and varying C, we obtain the classifiers in figures 17 - 21.

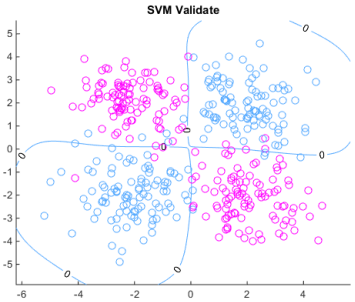

Figure 17: $C = 0.01, \sigma = 1$

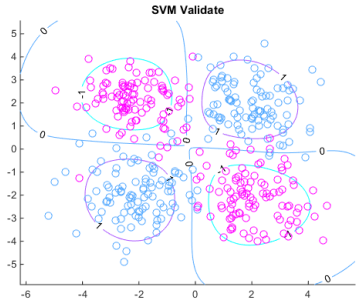

Figure 18: $C = 0.1, \sigma = 1$

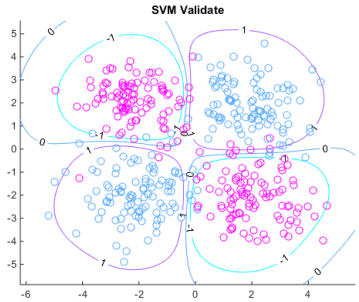

Figure 19: $C = 1, \sigma = 1$

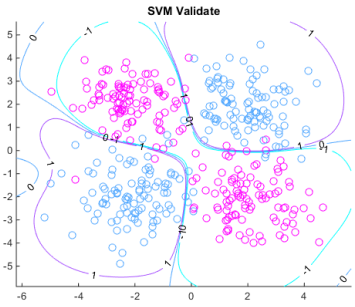

Figure 20: $C = 10, \sigma = 1$

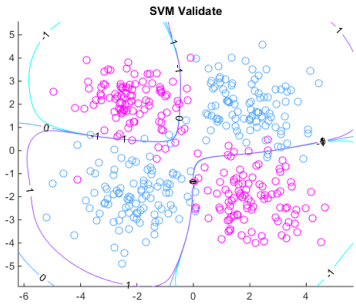

Figure 21: $C = 100, \sigma = 1$

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 $\mathcal{M} = (400, 324, 133, 68, 59)$. Varying σ, we notice that the "clusters" the classifier finds are composed of many smaller Gaussians for small σ and composed of fewer larger Gaussians for larger σ. Basically, the size of the "clumps" the classifier looks for proportional to σ.

As per the usual procedure we used in regression on 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

3.1 LR

3.2 SVM

Now, we apply the SVM to the Titanic dataset. We apply the standard method to select C (ie, calculate the loss on the validation set for the classifiers trained with the training set for various values of C.) Applying this method, we get $C = 0.1$, which leads to a validation error of 81.1067. The classifier this creates, using a linear kernel, is summarized below:

Table 2: θ_i for $C = 0.1$.

Column	Meaning	θ_i
1	Passenger Class 1	2.02e-04
2	Passenger Class 2	3.87e-04
3	Passenger Class 3	-5.89e-04
4	Sex	2.00
5	Age	-1.10e-05
6	Number of Siblings/Spouses Aboard	-5.64e-05
7	Number of Parents/Children Aboard	3.10e-04
8	Passenger Fare	2.77e-06
9	Embark from Southampton	-3.20e-04
10	Embark from Cherbourg	1.68e-04
11	Embark from Queenstown	1.52e-04

We can clearly see from θ_i that sex is by far the most important factor towards determining y. Plotting, for example, sex on the x-axis and age on the y-axis, as well as the θ generated when considering just these two factors, we see that the examples are split fairly well by sex, but age is well-mixed. In the plot, we add a scattering factor of some random number in $[-.15, .15]$ to the plotted points (θ is trained on the points before the scattering factor).

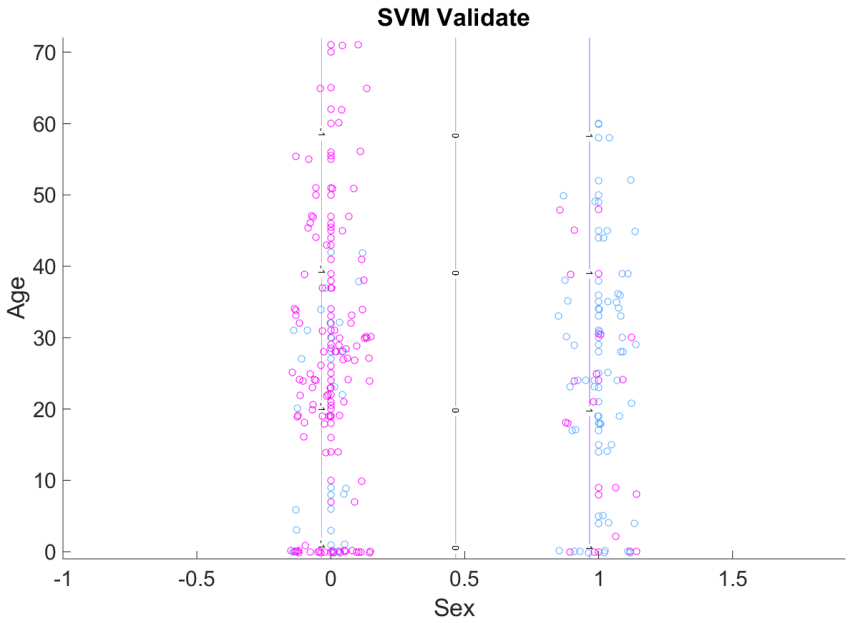

Figure 22: Decision boundary considering only age and sex

3.3 Comparing and Contrasting LR and SVM