CS281B/Stat241B: Advanced Topics in Learning & Decision Making

Introduction

Lecturer: Michael I. Jordan *Scribes: Hao Zhang*

1 Binary Classification

We assume training data

$$\{x_i, y_i\}_{i=1}^N \qquad x_i \in \mathbb{R}^d, y_i \in \{0, 1\}$$

Sometimes, the data are linearly separable as illustrated in figure 1. We'll make that assumption for now.

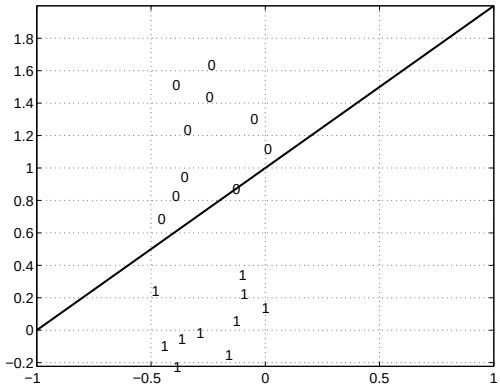

Figure 1: linearly separable data

We'll start off by discussing classifiers of the following kinds:

1. **Perceptron**: given a parameter $\theta \in \mathbb{R}^d$, define an update rule as:

$$\theta_{t+1} = \theta_t + \lambda(y_{i(t)} - \hat{y}_{i(t)})x_{i(t)}$$

where t indicates the iteration of the algorithm, i is the index involved in the t'th iteration. The predictor $\hat{y}_i$ is defined as:

$$\hat{y}_i = \begin{cases} 1 & \theta^T x_i > 0 \\ 0 & \text{o.w.} \end{cases}$$

The intuition behind the update rule is to step in the direction to reduce prediction error, in a "gradient-descent" kind of way. The value of λ specifies the step size. It is guaranteed by the Perceptron Convergence Theorem that if the data are linearly separable then the preceding algorithm will find it in a finite number of steps.

Issues:

(a) The convergence theorem concerns only the training data (data with known label). We have to work harder to bound the test error (error when predicting new cases).

(b) If the data are not linearly separable, the behavior of the procedure is erratic.

(c) It doesn't generalize well to multi-class.

2. **Logistic Regression** (example of a GLIM – general linear model):

The statistical model is assumed to be:

$$p(y_i = 1 \mid x_i) = \frac{1}{1 + e^{-\theta^T x_i}}$$

(aside: for notational convenience, assume the vector x_i always has a leading 1, acting as the constant term).

The model yields $p(y_i = 0 \mid x_i)$ and we can capture both cases with one equation:

$$p(y_i|x_i) = \frac{e^{y_i \theta^T x_i}}{1 + e^{\theta^T x_i}}$$

so that the expression agrees with the model where y_i is binary but also is defined for continuous $y_i \in [0, 1]$. We now express the *likelihood* as the probablity of observing the known data under the assumed model:

$$\mathbf{L}(\theta) = \prod_{i=1}^{N} p(y_i \mid x_i, \theta)$$

The log-likelihood is:

$$\begin{aligned} l(\theta) &= \log(\mathbf{L}(\theta)) = \sum_{i=1}^{N} \{y_i \theta^T x_i - \log(1 + e^{\theta^T x_i})\} \\ \frac{\partial l}{\partial \theta} &= \sum_{i=1}^{N} (y_i x_i - \frac{e^{\theta^T x_i}}{1 + e^{\theta^T x_i}} x_i) \\ &= \sum_{i=1}^{N} (y_i - \hat{y}_i) x_i \qquad \text{where } \hat{y}_i = p(y_i = 1 \mid x_i, \theta) \end{aligned}$$

(The log-likelihood is convex in θ, a very nice property for optimization algorithms) The gradient $\frac{\partial l}{\partial \theta}$, expressed in terms of current predictor value $\hat{y}_i$, enables "gradient ascent" algorithms, including Newton-Raphson, which is generally used for logistic regression. When the data amount is large, however, computing the full gradient based on all data points becomes slow. A faster alternative is "stochastic" gradient algorithm, which updates the parameter value based on one data point at a time:

$$\theta_{t+1} = \theta_t + \lambda(y_{i(t)} - \hat{y}_{i(t)}) x_{i(t)}$$

which is formally the same as the update algorithm of perceptrons, but with a different predictor. Large amount of literature is devoted to the study of stochastic gradient algorithms.

This method generalizes well to multi-class: the so-called "multinomial logit" model.

3. **Linear Discriminant Analysis**:

The goal is to find θ to make the ratio of between-class variance large w.r.t. within-class variance:

The within-class variance is defined as

$$\hat{\Sigma}_w = \frac{1}{n} \left(\sum_{i:y_i=1} (x_i - \hat{\mu}_1)(x_i - \hat{\mu}_1)^T + \sum_{i:y_i=0} (x_i - \hat{\mu}_0)(x_i - \hat{\mu}_0)^T \right)$$

The optimization problem is:

$$\max_{\theta} \frac{\theta^T(\hat{\mu}_1 - \hat{\mu}_0)(\hat{\mu}_1 - \hat{\mu}_0)^T\theta}{\theta^T\hat{\Sigma}_w\theta} \qquad (*)$$

This form of ratio is called the (generalized) *Rayleigh quotient*. Compare this to the following problem:

$$\max_{x} \frac{x^TAx}{x^Tx} \qquad \text{where } A \text{ is real symmetric}$$

One should have learned that the maximum value is the largest eigenvalue of A and x is the corresponding eigenvector. Our problem can be reduced to this probelm by noticing that:

$\hat{\Sigma}_w$ is positive definite almost always (meaning with probability 1). This implies that it has the Cholesky decomposition:

$$\hat{\Sigma}_w = LL^T \qquad \text{for some } L,$$

for lower triangular L. Therefore we express (*) as

$$(*) = \max_{\beta} \frac{\beta^TL^{-1}(\hat{\mu}_1 - \hat{\mu}_0)(\hat{\mu}_1 - \hat{\mu}_0)^TL^{-T}\beta}{\beta^T\beta} \qquad \text{where } \beta = L^T\theta$$

Optimize for β first and then obtain θ.

4. **Maximal Margin Classifier**:

   As seen in figure 1, the separating plane obtained from the perceptron is arbitrary as long it divide the data on the two sides. In addition, a "maximal-margin" classifier also tries to maximize the minimum distance of data points to the decision boundary, as illustrated in figure 2.

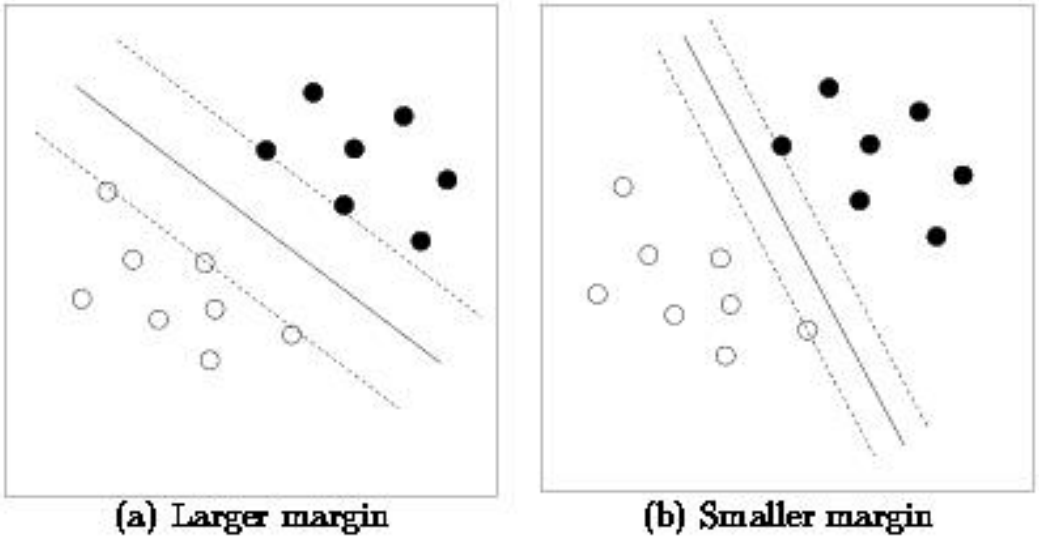

Figure 2: Maximal Margin Classifier (image from *Support Vector Machines Applied to Speech Pattern Classification*, K.K. Chin, Darwin College, Cambridge University, 1999)