CS281B/Stat241B (Spring 2008) Statistical Learning Theory **Lecture: 4**

Kernel Methods for Pattern Classification

Lecturer: Peter Bartlett *Scribe: Kristal Sauer*

1 The Perceptron Algorithm

Recall the following upper bound on the expected risk of a randomized version of the perceptron algorithm.

Theorem. Suppose that, for some $r, \delta > 0$ and $\theta \in \mathbb{R}$, we have a.s. $\| X \| < r$ and $\frac{Y\theta' X}{\|\theta\|} \geq \delta$. Then, the following randomized variant of the perceptron algorithm returns f_n with

$$\mathbb{E}R(f_n) \leq \frac{r^2}{n\delta^2}$$

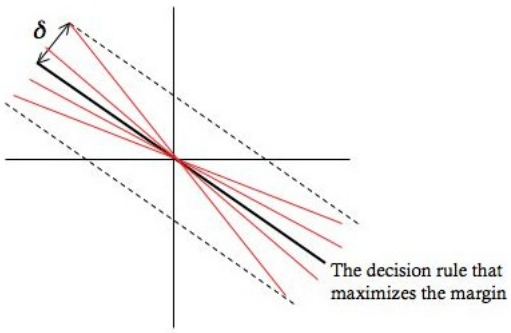

Figure 1. Choose the decision rule that yields the maximum margin.

The following is a converse result: a lower bound on the expected risk of any algorithm.

Theorem. For any decision rule f_n, there is a P on $\mathbb{R}^d \times \{\pm 1\}$ s.t. for some $\theta \in \mathbb{R}^d$, a.s.

$$\frac{\theta' XY}{\| \theta \|} \geq \delta$$

$$\| X \| \leq r$$

but

$$\mathbb{E}R(f_n) \geq \frac{\min\left(d, n, r^2/\delta^2\right) - 1}{2n} \left(1 - \frac{1}{n}\right)^n.$$

The **proof idea** is as follows:

We have support of P_x on the scaled basis vectors,

$$\left\{ re_i : 1 \leq i \leq \left\lfloor \frac{r^2}{\delta^2} \right\rfloor \right\}, \text{where } \left\lfloor \frac{r^2}{\delta^2} \right\rfloor = k$$

$$\theta = \frac{1}{\sqrt{k}} \sum_{i=1}^{k} b_i e_i$$

Then,

$$Y_i \theta' X_i = \frac{r}{\sqrt{k}} \sim \delta \text{ (modulo some factor).}$$

This concludes our discussion of the perceptron algorithm. We now continue to kernel methods.

2 Kernel Methods for Pattern Classification

We have seen that, when there is a large margin linear threshold function, the perceptron algorithm will quickly find a linear threshold function that classifies the data, stopping when it achieves classification of the training examples. Note that it does not choose a large margin linear threshold function.

It is interesting to consider the performance of a method that does maximize the margin, given by $\delta = \frac{Y\theta' X}{\|\theta\|}$. An example of such a method is the *Support Vector Machine* (SVM).

Consider the following optimization problem.

$$\boxed{\max_{\gamma,w} \gamma \text{ s.t. } \frac{y_i w' x_i}{\|w\|} \geq \gamma \; i = 1, \ldots, n}$$

We could add the constraints $\| w \| \leq 1$ or $\| w \| = \frac{1}{\gamma}$ to lower the number of variables, because we only really care about the direction of w.

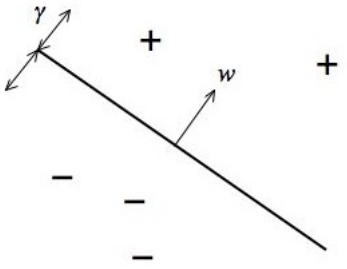

Figure 2. SVM optimizes γ.

We may thus express the optimization problem given as the equivalent optimization problem,

$$\boxed{\min_{w \in \mathbb{R}^d} \| w \|^2 \text{ s.t. } y_i w' x_i \geq 1 \; i = 1, \ldots, n}$$

We call this the *Hard-Margin SVM*. This optimization problem is the *primal* problem. Note that we are maximizing a quadratic criterion subject to linear constraints; thus, we have a *Quadratic Program* (QP).

Recall that for constrained optimization, we use *Lagrange Multipliers*.

We introduce Lagrange Multipliers $\alpha_i \geq 0$. We have one for each $i = 1, \ldots, n$.

Lagrangian:

$$L(w,\alpha) := \frac{1}{2} \parallel w \parallel^2 - \sum_{i=1}^{n} \alpha_i \left(y_i w' x_i - 1\right)$$

(We have added the scaling factor of $\frac{1}{2}$ for convenience—it tidies the constants in what follows.)

We wish to minimize $L(w,\alpha)$ wrt w, the *primal* variables, and maximize it wrt α, the *dual* variables. That is, we wish to find the saddle point of $L(w,\alpha)$.

Suppose we're at a saddle point.

- If $y_i w' x_i < 1$, we could increase α_i to increase L. So at the saddle point, we must have the constraints satisfied.
- If $y_i w' x_i > 1$, then $\alpha_i = 0$.
- In any case, $\alpha_i \left(y_i w' x_i - 1\right) = 0 \forall i$

$\implies$ At the saddle point (w,α), w is primal feasible. $\implies$ At the saddle point, $L(w,\alpha) = \frac{1}{2} \parallel w \parallel^2$.

At the saddle point,

$$\frac{\partial}{\partial w} L(w,\alpha) = 0 \implies w* = \sum_{i=1}^{n} \alpha_i y_i x_i$$

Consider especially points for which $\alpha_i > 0 \implies y_i w' x_i = 1$. Such points are referred to as *support vectors*. Only the support vectors enter into the constraint to determine the solution to the optimization problem.

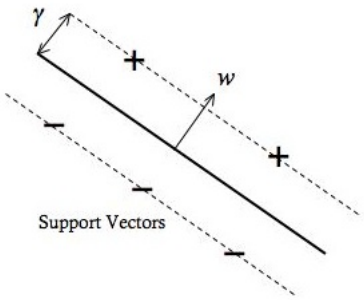

Figure 3. Support vectors enter into constraint.

Let us also consider the dual optimization problem:

Substituting $w*$ into $L(w,\alpha)$:

$$\begin{aligned} g(\alpha) &:= L(w*,\alpha) \\ &= \frac{1}{2} \sum_{i,j} \alpha_i y_i x_i' x_j y_j \alpha_j - \sum_{i,j} \alpha_i y_i x_i' x_j y_j \alpha_j + \sum_i \alpha_i \\ &= \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i y_i x_i' x_j y_j \alpha_j \end{aligned}$$

Dual:

$$\boxed{\max_{\alpha\in\mathbb{R}^d} \sum_{i=1}^n \alpha_i - \frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j x_i' x_j \text{ s.t. } \alpha_i \geq 0 \; i=1,\ldots,n}$$

Notice that the x_is only enter the optimization problem via the inner products.

As with the perceptron algorithm, we can express the solution in terms of a mapping to an inner product space:

$$\begin{aligned} f_n(x) &= \operatorname{sign}\left(\langle w, \Phi(x)\rangle\right) \\ &= \operatorname{sign}\left(\sum_{i=1}^n \alpha_i y_i \langle \Phi(x_i), \Phi(x)\rangle\right) \\ &= \operatorname{sign}\left(\sum_{i=1}^n \alpha_i y_i k(x_i, x)\right) \end{aligned}$$

where $k(x_i, x)$ is the kernel and α is a solution to

$$\boxed{\max_{\alpha\in\mathbb{R}^d} \sum_{i=1}^n \alpha_i - \frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j k(x_i, x_j) \text{ s.t. } \alpha_i \geq 0 \; i=1,\ldots,n}$$

To recap, note the two key ideas of SVMs:

- Maximized margin
- Arbitrary inner product

Let us now consider some examples of kernels.

- Polynomial

$$\begin{aligned} k_2(u,v) &= (u'v)^2 \\ k_2(u,v) &= \left(\sum_{i=1}^d u_i v_i\right)^2 \\ &= \left(u_1^2, \sqrt{2}u_1u_2, u_2^2\right)\begin{pmatrix} v_1^2 \\ \sqrt{2}v_1v_2 \\ v_2^2 \end{pmatrix} \\ &= \Phi_2(u)'\,\Phi_2(v) \;\text{ with } \Phi_2 : \mathbb{R}^2 \mapsto \mathbb{R}^3 \end{aligned}$$

  Note. The feature space might not be unique; eg,

$$\begin{aligned} \psi_2(u) &= \left(u_1^2, u_1u_2, u_2u_1, u_2^2\right) \\ k_2(u,v) &= \psi_2(u)'\,\psi_2(v) \end{aligned}$$

- Gaussian

$$k_G(u,v) = \exp\left(\frac{-\parallel u-v\parallel^2}{2\sigma^2}\right) = \langle \phi_G(u), \phi_G(v)\rangle$$

  Here, u and v are infinite-dimensional vectors.

Above, the kernels are defined on $\mathbb{R}^d$; however, we may talk about any space (e.g., documents) on which we can define an inner product.