Statistical Techniques in Robotics (16-831, F14) Lecture#16 (Tuesday October 28)

Support Vector Machines and Kernel Methods

Lecturer: Drew Bagnell *Scribes: Sibi Venkatesan (sibiv) and John Yao (johnyao)* [1]

1 Support Vector Machines

We begin by considering binary, linear classification. In this problem, we are trying to map elements from our feature space into the set $\{-1, 1\}$. When we have two features, $F_1, F_2 \in \mathbb{R}$, we can think of the problem graphically. In the following figure, we represent the points with a label of 1 as O's, we the points with a label of -1 as X's. We wish to find a linear decision boundary (a straight line) that separates the points from each class. The decision boundary is shown as the dotted line.

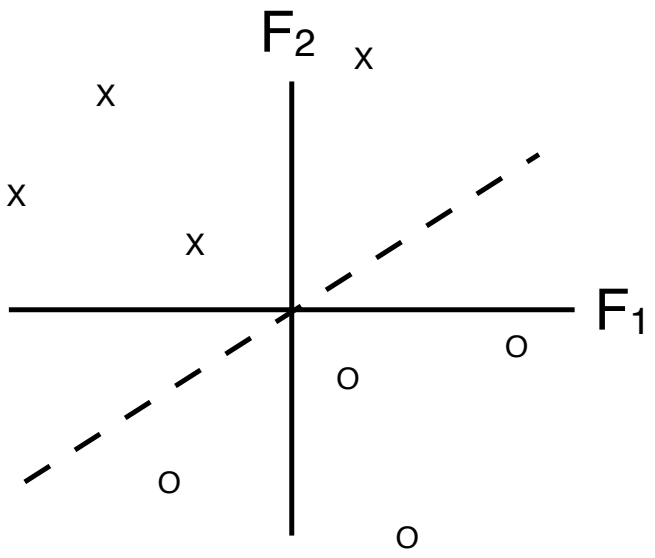

Figure 1: Linear binary classification

To represent this notion mathematically, we denote the i^{th} training point as f_i, and the class of this point as $y_i \in \{-1, +1\}$. The linear classification problem requires us to find a set of weights $w \in \mathbb{R}^n$ such that:

$$\forall i \; y_i w^T f_i \geq 0 \tag{1}$$

One issue with this formulation of the problem is that it has a trivial solution. Specifically, if we set all of our weights to 0, we will have $y_i w^T f_i = 0$ for all points. To address this issue, we modify our constraints to be:

$$\forall i \; y_i w^T f_i \geq margin \tag{2}$$

[1] Some content adapted from previous scribes: Robert Fisher and Liz Cha

This can have the added benefit of improving generalization. Once again, we can represent this graphically. In the following picture, the space between the dotted line and the decision boundary represents the margin.

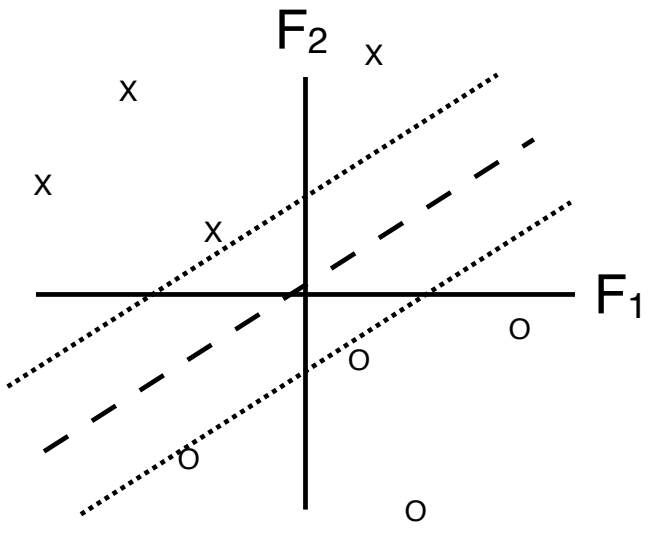

Figure 2: Linear binary classification with margins

Unfortunately, the data may not always be linearly separable. To address this issue, we introduce a slack variable, ξ. The problem becomes:

$$\begin{aligned} \textbf{Minimize} \quad & \tfrac{1}{2}\lambda||w||^2 + \sum_i \xi_i \\ \textbf{Such that} \quad & y_i w^T f_i \geq 1 - \xi_i \\ & \xi_i \geq 0 \\ & margin > 0 \end{aligned}$$

There will be one slack variable ξ_i to correspond to each of the data-points that we are considering.

Next, we observe that $\xi = \max(0, 1 - y_i w^T f_i)$, because the slack variable is 0 if this point is labeled correctly. This allows us to generate the loss function

$$l_t = \frac{1}{2}\lambda||w||^2 + \max(0, 1 - y_t w^T f_t) \tag{3}$$

The subgradient $\nabla l_t(w)$ is as follows:

$$\nabla l_t(w) = \begin{cases} \lambda w & \text{if } y_t w^T f_t \geq 1 \\ \lambda w - y_t f_t & \text{otherwise} \end{cases} \tag{4}$$

We note that the loss function is strongly convex, since we can always lower bound it with a quadratic.

2 Weighted Data Points

Let $\sigma_i \in \mathbb{R}$ denote a scalar weight on the ith data point. How can we incorporate weighted data point into a SVM?

1. Margin Scaling
   - instead of $y_t w^T f_t \geq 1$, use the constraint $y_t w^T f_t \geq \sigma_t$
2. Replicate the more important data points
   - the weights need to be integers
   - this doesn't work well with large variations in weights
   - increasing the number of data points increases computational resource requirement
3. Sub-select
4. Slack Scaling
   - multiply the hinge loss function for the ith data point by the weight σ_i

2.1 SVMs with Multiple Classes

We can represent problems with more than two classes by having a weight vector, w_i for each class.

When we get a classification of a particular example (for example, example i is of class 1), we generate a set of constraints that can be expressed as either

$$\begin{aligned} w_1^T f_i &\geq w_2^T f_i + 1 \\ w_1^T f_i &\geq w_3^T f_i + 1 \\ w_1^T f_i &\geq w_4^T f_i + 1 \\ &\dots \end{aligned} \tag{5}$$

or

$$w_1^T f_i \geq \max_{c \neq i} (w_c^T f_i + 1) \tag{6}$$

In the first method, we have separate slack variables for each pairwise combination of w_1 and every other w_i. This is a linear problem. When we apply the subgradient method, we update the weights of all classes whose contraints were violated.

This stands in contrast to the second method, in which there is only a single slack variable. Consequently, in the subgradient method we only update the weight of the class that violates its constraint the most.

2.2 Non-linear SVMs using Kernels

So far, we've dealt with linear SVMs which use dot-products with weight vectors in order to classify elements. If the data is not linearly-separable, we might want to create a non-linear classifier as opposed to using a soft-margin SVM.

Kernel methods are a type of such techniques which allow us to non-linear classifiers. The algorithm is similar to the ones we saw above, except that we now replace the dot-product with a kernel function.

Kernels are described in more detail in the section below.

3 Kernels and Reproducing Kernel Hilbert Spaces

Kernel and kernel methods are viewed in many contexts:

- Duality - the classic way to introduce kernel methods.
- Function spaces - useful for other techniques like boosters.
- Representer theorem
- Gaussian Processes

We are going to focus on function spaces as we discuss kernels and kernel methods in detail.

3.1 Function spaces

Consider the loss function defined over an input weight vector w:

$$L(w) = \sum_i (y_i - w^T x_i)^2 + \frac{\lambda}{2}||w||^2$$

Now, let us move on to defining loss over functions. Consider the set of all functions $f : \chi \to \mathbb{R}$, where χ is defined to be our feature space.

We can then define a new loss function, which is itself over a function, as follows:

$$L[f] = \sum_i (y_i - f(x_i))^2 + \frac{\lambda}{2}||f||^2$$

Essentially, we have generalized our dot-product with weights into a more general class of functions. We will define the norm of a function in due time.

When we look at function spaces, there are few terms which come in handy. Given a function space $\mathcal{H}$, we define the following:

- Functionals: These are functions which take functions as input and map them to real numbers.

  $$L : \mathcal{H} \to \mathbb{R}$$

  We will look at these in more detail shortly.

- Operators: These are functions which take functions as input and map them to other functions.

  $$O : \mathcal{H} \to \mathcal{H}$$

Conventionally, functionals and operators are written with square brackets, as opposed to paranthesis. Specifically, we write $L[f]$ for a functional L with input f.

3.2 Kernels

Before we talk about the RKHS, let us take a quick look at kernels.

A kernel $K : \chi \times \chi \to \mathbb{R}$ intuitively measures the *correlation* between two vectors x and y. There are a few restrictions on what functions qualify as a kernel.

- Firstly, the kernel K must be symmetric, i.e. $K(x_i, x_j) = K(x_j, x_i)$.
- Secondly, the kernel K must be positive-definite.

  What does that mean? Suppose we have a set of vectors $\{x_1, \ldots, x_n\}$ in the feature space. We can define the matrix $\mathbf{K}$ with entries $\mathbf{K}_{ij} = K(x_i, x_j)$. Then, our matrix $\mathbf{K}$ must be positive-semidefinite, i.e. $x^T \mathbf{K} x \geq 0$, $\forall x \in \chi$.

  For some reason, a kernel K being positive-definite means that the matrix $\mathbf{K}$ must be positive-**semi**definite, not necessarily positive-definite.

An advantage of these restrictions on kernels is the ability to define a norm in the function-space.

A useful way to visualize kernels is something which looks like a bump. $K(x, \cdot)$ can be thought of as a bump centered at x. Some examples of kernels are given below. We'll look at few more later.

- $K(x, y) = e^{-\lambda(x-y)^2}$
- $K(x, y) = x^T y$

It is easy to see why the second kernel function is positive definite; the kernel matrix is just a covariance matrix ($X^T X$) which is positive-semidefinite.

3.3 Reproducing Kernel Hilbert Spaces

Given a symmetric, positive-definte kernel K, the reproducing kernel Hilbert space $\mathcal{H}_K$ is the space of functions of the form

$$f(\cdot) = \sum_{i=1}^{N} \alpha_i K(x_i, \cdot)$$

Basically, a function $f \in \mathcal{H}_K$ is a weighted sum of kernels centered at various locations x_i. This makes it obvious that $\mathcal{H}_K$ is a linear space over functions.

Let us extend this space with a notion of an inner product. We define an inner product over the RKHS $\mathcal{H}_K$ as follows. For two functions $f, g \in \mathcal{H}_K$,

$$\langle f, g \rangle = \sum_i \sum_j \alpha_i \beta_j K(x_i, x_j) = \alpha^T \mathbf{K} \beta$$

where we have the matrix $\mathbf{K}$ with the entries $\mathbf{K}_{ij} = K(x_i, x_j)$, and

$$f = \sum_i \alpha_i K(x_i, \cdot)$$

$$g = \sum_j \beta_j K(x_j, \cdot)$$

This now allows us to define a norm (or seminorm) over $\mathcal{H}_K$ as follows:

$$||f||^2 = \langle f, f \rangle$$

Why are these function spaces called "reproducing?" Stay tuned for the next episode of *Scribe Notes: Statistical Techniques in Robotics* to find out!