Class Notes
Introduction to Kernel Methods

Miguel D. Dussán

The kernel methods were developed 30 years ago in Rusia by Vapnick, but his work was note in the west only in 90's, and applied for solving classification problems. In the 80's main machine learning's technique was Neural Networks, but it will be noted that this approach is not suitable for all kind of problems.

1. Problems

- Separate two classes with a linear function, when it is not clear linear boundary, i.e., an elipsis.
- Symbolic regression, wich has its practical applications, i.e., DNA sequences, where $\Re$ might be a physical or chemical property. That specific problem might be solved using a neural network but, for each letter in the alphabet, it would be required an input. So, for a real problem, its inputs would be millions (an unfeasible approach).

2. Kernel Method Overview

Suppose that we have the problem of finding a homogeneous real-valued linear function that interpolates a collection of points $S = \{(\mathtt{x}_1, y_1), \ldots, (\mathtt{x}_l, y_l)\}$ where $\mathtt{x}_i$ belongs to $X \subseteq \Re^n$, its pair y_i is a label, $Y \subseteq \Re$, and $\mathtt{x}=(x_1, x_2, \ldots, x_n)$ is the n-dimensional input vectors. An approach would be:

$$g(\mathtt{x}) = \langle \mathtt{w}, \mathtt{x} \rangle = \mathtt{w}'\mathtt{x} = \sum_{i=1}^{n} w_i x_i$$

where $\mathtt{w}$' denotes the transpose of vector $\mathtt{w} \in \Re^n$. That new pattern function (that relates g of the featuring $\mathtt{x}$ with the labels y) should be approximately equal to zero, that is:

$$f((\mathtt{x}, y)) = |y - g(\mathtt{x})| = |y - \langle \mathtt{w}, \mathtt{x} \rangle\, | \approx 0$$

which is known as linear interpolation. When there are $n = l$ linearly independent points, the problem might be solved through the system of linear equations

$$\mathtt{X}\mathtt{w} = \mathtt{y}$$

where $\mathtt{X}$ is the matrix whose rows are $x'_1, \ldots, x'_l$ and $\mathtt{y}$ is the vector $(y_1, \ldots, y_l)'$. But, when $n \neq l$, a different approach is employed because there are more than one $\mathtt{w}$ that fullfill the requirements.

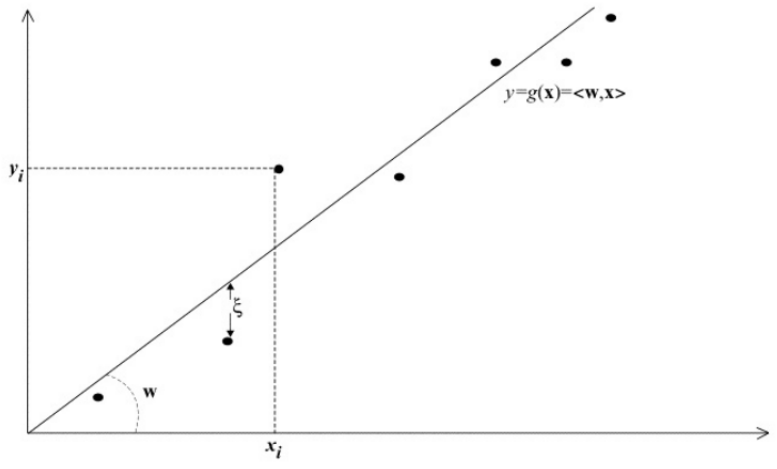

Figura 1: Graphic representation of one dimension linear regression problem. x_i coordinate represents the input vector, and y_i its labels.

If our case is the second one ($n \neq l$), the criterion used to choose the best $\mathtt{w}$ is to pick up the one with the minimum norm. But sometimes there is noise in the process of generate the samples, so there is required also an approximation criterion. The conclusion here is to use a mixed method that integrates both strategies: to find a $\mathtt{w}$ that has small error and small norm. The putative patter function

$$f((\mathtt{x}, y)) = |y - g(\mathtt{x})| = |\xi|$$

generates all the ξ for each one of the training examples, that is, $\xi = (y - g(\mathtt{x}))$.

How to find out the minimum training errors? Here is introduced something called a *loss function*, that is, $L(f, S)$ is the collective loss of a function f on a training set S:

$$L(g, S) = L(\mathtt{w}, S) = \sum_{i=1}^{l} (y_i - g(\mathtt{x}_i))^2 = \sum_{i=1}^{l} \xi_i^2 = \sum_{i=1}^{l} L(g, (\mathtt{x}_i, y_i))$$

where $L(g, (\mathtt{x}_i, y_i)) = \xi_i^2$ is the squared error or loss of g on example $(\mathtt{x}_i, y_i)$. In figure 1 is graphically represented some definitions mentioned earlier.

2.1. Learning schemes

Classical Learning scheme:

Data → Learning Algorithm → Model

Kernel Method Learning scheme:

Data → **Kernel Function** → **Kernel Matrix** → Learning Algorithm → Model

Added an additional step between data and learning: kernel function and kernel matrix.

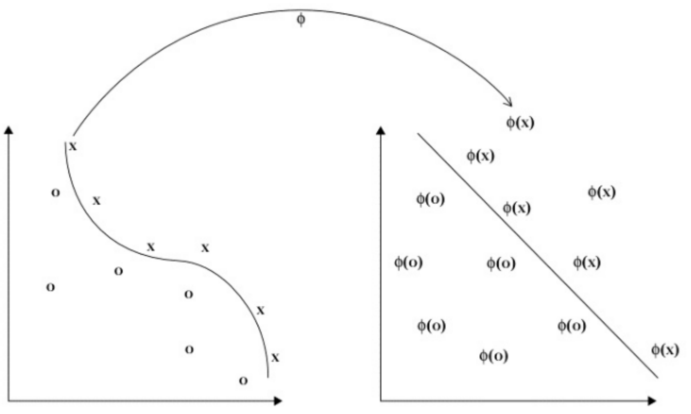

Figura 2: Graphic representation of problem (left) and feature (right) spaces.

3. Problem and Feature space

In kernel methods, there are two spaces: the **problem space**, where the samples originally belongs to, and the feature space, that is, the space where the points are transformed using a kernel function.

In figure 2, the problem and feature spaces are shown. Note that a transformation ϕ is required to go from the first to the second one, and all its points are reflected in function of ϕ. Also, see that a non-linear classification boundary in the problem space becomes a linear one in the feature space.

3.1. Change of space

An example:

$$\begin{array}{lll} f : \Re^3 & \rightarrow & \Re^6 \\ (x, y, z) & \rightarrow & (x^2, y^2, z^2, xy, xz, yz) \end{array}$$

In general, for changing from one to another space: $\Re^n \rightarrow \Re^{\frac{n(n+1)}{2} = \binom{n+1}{2}}$

3.2. Dot product

In Kernel methods, the dot product is calculated in the problem space, and then brought to the feature space. Usually, the dot product is understood as:

$$\langle (x_1, x_2, \dots), (y_1, y_2, \dots) \rangle = \sum_i x_i y_i$$

What if we have a problem with a hundred dimensions?:

$$\langle \phi(x_1, \dots, x_{100}), \phi(y_1, \dots, y_{100}) \rangle = \langle (x_1, \dots, x_{100}), (y_1, \dots, y_{100}) \rangle^2$$

The multiplications needed for this operations rounds about a hundred, too. So, here is where the kernel function help us with the task of calculating the dot product in a kernel space $k : X \times X \rightarrow \Re$ such that $k(x, z) = \langle \phi(x), \phi(y) \rangle$, is called a kernel.

As an example, suppose that we wish to build a kernel function that maps:

$$\phi : \mathtt{x} = (x_1, x_2) \rightarrow \phi(\mathtt{x}) = (x_1^2, x_2^2, \sqrt{2}x_1x_2) \in F = \Re^2$$

So, we have something from $X \subseteq \Re^2$ to $\Re^3$. How can we get a kernel, and benefit from the kernel trick without explicitly evaluate its coordinates in the feature space (that is, in $\Re^3$)? Let's calculate $\langle \phi(\mathtt{x}), \phi(\mathtt{z}) \rangle$:

$$\langle \phi(\mathtt{x}), \phi(\mathtt{z}) \rangle = \langle (x_1^2, x_2^2, \sqrt{2}x_1x_2), (z_1^2, z_2^2, \sqrt{2}z_1z_2) \rangle$$
$$= x_1^2z_1^2 + x_2^2z_2^2 + 2x_1x_2z_1z_2 = (x_1z_1 + x_2z_2)^2 = \langle \mathtt{x}, \mathtt{z} \rangle^2$$
$$= k(\mathtt{x}, \mathtt{z}) = \langle \mathtt{x}, \mathtt{z} \rangle^2$$

That is, if we have two vectors in the problem space, to calculate its ϕ is only required to execute its dot product and elevate the result of the power of two.