Functions as Features and "Kernels"

Stephen Vardeman

Analytics Iowa LLC

ISU Statistics and IMSE

Nonlinear Maps and Function Spaces

Often, what P encodes about a relationship between $\mathbf{x}$ and y is very complicated and "non-linear." Standard (and almost all tractable) mathematics relies on "linear" operations: additions of vectors, multiplication of vectors by scalars, inner products (and associated norms and distances), etc. "Ordinary" creation of features can be thought of as a way to map a feature space $\Re^p$ (non-linearly) to a higher-dimensional (Euclidean and therefore linear) feature space $\Re^q$. But that can be ineffective because q large enough to allow for good prediction based on linear operations is so large as to make an appropriate transform $T : \Re^p \to \Re^q$ impossible to identify and/or use.

A very clever and practically powerful development in machine learning has been the realization that for some purposes, it is not necessary to map from $\Re^p$ to a Euclidean space, but that mapping to a linear space **of functions** may be helpful.

Mapping to predictors from function spaces

If $\mathcal{A}$ is an abstract feature space of functions (that is an inner product space) one might think of mapping

$$T : \Re^p \to \mathcal{A}$$

and using linear operations and relationships in $\mathcal{A}$ to make (relationships and/or) predictors based on **a**s in $\mathcal{A}$ (and then defining corresponding ones for **x**s in $\Re^p$ by simply applying T to **x**s of interest to make **a**s and corresponding predictions). After all, functions are really just high-dimensional vectors, and if transforming $\Re^p \to \Re^q$ with $p < q$ is often useful, so also might be transforming $\Re^p \to \mathcal{A}$.

This line of argument has especially been taken advantage of through the use of so-called "kernel functions." (Be careful. There are many different usages of the word "kernel" in the machine learning world.)

Non-negative definite functions

Suppose that a symmetric function $\mathcal{K}(\mathbf{x}, \mathbf{z})$ with domain $\Re^p \times \Re^p$ is non-negative definite in the sense that for any training set $\mathbf{T}$ the (symmetric) $N \times N$ so-called **Gram matrix**

$$\mathbf{K} = \left(\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j)\right)_{\substack{i=1,\ldots,N \\ j=1,\ldots,N}}$$

is non-negative definite. Then the space of functions that are finite linear combinations of "slices" of $\mathcal{K}(\mathbf{x}, \mathbf{z})$, i.e. functions of $\mathbf{x}$ of the form

$$\sum_{j=1}^{M} c_j \mathcal{K}(\mathbf{x}, \mathbf{z}_j)$$

for $M > 0$ real numbers $c_1, c_2, \ldots, c_M$, and elements $\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_M$ of $\Re^p$ form a linear space. Call it $\mathcal{A}$.

Matched inner product for a kernel space

It is possible to define a very convenient inner product on $\mathcal{A}$ starting from

$$\langle \mathcal{K}(\cdot, \mathbf{z}_1), \mathcal{K}(\cdot, \mathbf{z}_2) \rangle_{\mathcal{A}} \equiv \mathcal{K}(\mathbf{z}_1, \mathbf{z}_2)$$

This relationship and the bilinearity of any inner product of necessity imply that

$$\begin{aligned} \left\langle \sum_{j=1}^{M} c_{1j}\mathcal{K}(\cdot, \mathbf{z}_j), \sum_{j=1}^{M} c_{2j}\mathcal{K}(\cdot, \mathbf{z}_j) \right\rangle_{\mathcal{A}} &= \sum_{j=1}^{M}\sum_{j'=1}^{M} c_{1j}c_{2j'}\mathcal{K}(\mathbf{z}_j, \mathbf{z}_{j'}) \\ &= \mathbf{c}_1'\mathbf{K}\mathbf{c}_2 \end{aligned}$$

for $\mathbf{c}_1' = (c_{11}, c_{12}, \ldots, c_{1M})$, $\mathbf{c}_2' = (c_{21}, c_{22}, \ldots, c_{2M})$, and $\mathbf{K}$ the (non-negative definite) $M \times M$ matrix with entries $\mathcal{K}(\mathbf{z}_i, \mathbf{z}_j)$. The special case of $\mathbf{c} = \mathbf{c}_1 = \mathbf{c}_2$ further provides the simple form

$$\left\| \sum_{j=1}^{M} c_j\mathcal{K}(\mathbf{x}, \mathbf{z}_j) \right\|_{\mathcal{A}}^2 = \mathbf{c}'\mathbf{K}\mathbf{c}$$

Distance in the function space and terminology

Of course, since $\mathcal{K}$ defines the inner product in $\mathcal{A}$ it also defines the distance between $\sum_{j=1}^{M} c_{1j}\mathcal{K}(\cdot, \mathbf{z}_j)$ and $\sum_{j=1}^{M} c_{2j}\mathcal{K}(\cdot, \mathbf{z}_j)$

$$d_{\mathcal{A}}\left(\sum_{j=1}^{M} c_{1j}\mathcal{K}(\cdot, \mathbf{z}_j), \sum_{j=1}^{M} c_{2j}\mathcal{K}(\cdot, \mathbf{z}_j)\right) = \sqrt{(\mathbf{c}_1 - \mathbf{c}_2)' \mathbf{K} (\mathbf{c}_1 - \mathbf{c}_2)}$$

(with $\mathbf{c}_1, \mathbf{c}_2$, and $\mathbf{K}$ as on the previous slide).

Here $\mathcal{K}$ serves as a **reproducing kernel**. It both defines the linear space of functions of interest and provides the inner product for the space. Under some conditions, the space $\mathcal{A}$ (whose elements are functions $\Re^p \rightarrow \Re$) can be extended to include *limits* of finite linear combinations of slices of the kernel function $\mathcal{K}(\cdot, \cdot)$ and the resulting construct is termed a Reproducing Kernel (Hilbert) Space (**RKHS**) of functions.

Standard transform to the function space

The (standard non-linear) transform $T : \Re^p \rightarrow \mathcal{A}$ is defined by

$$T(\mathbf{x})(\cdot) = \mathcal{K}(\mathbf{x}, \cdot)$$

(remember here that $T(\mathbf{x})(\cdot)$ is a function of "$\cdot$"). The inner product in $\mathcal{A}$ of two images of elements of $\Re^p$ is

$$\langle T(\mathbf{x}), T(\mathbf{z}) \rangle_{\mathcal{A}} = \mathcal{K}(\mathbf{x}, \mathbf{z})$$

and for a training set with inputs $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N$ the span of $\{T(\mathbf{x}_i)\}_{i=1,\ldots,N}$ is a linear subspace of $\mathcal{A}$.

Gaussian kernels

Probably the most used kernel function in machine learning is the "Gaussian kernel"

$$\mathcal{K}(\mathbf{x}, \mathbf{z}) = \exp\left(-\gamma \|\mathbf{x} - \mathbf{z}\|^2\right)$$

that produces

$$T(\mathbf{x})(\cdot) = \exp\left(-\gamma \|\mathbf{x} - \cdot\|^2\right)$$

that are radially symmetric p-variate Normal density functions located at $\mathbf{x}$. The function space $\mathcal{A}$ consists of linear combinations of such functions (and limits of them) and the abstract inner product of $T(\mathbf{x})$ and $T(\mathbf{z})$ is $\exp\left(-\gamma \|\mathbf{x} - \mathbf{z}\|^2\right)$.

Other kernels

One can give up requiring that the domain of a kernel function $\mathcal{K}(\mathbf{x}, \mathbf{z})$ is a subset of $\Re^p \times \Re^p$, replacing it with arbitrary $\mathcal{X} \times \mathcal{X}$ and requiring *only* that the Gram matrix be non-negative definite for any set of $\{\mathbf{x}_i\}_{i=1}^n$, $\mathbf{x}_i \in \mathcal{X}$. It is in this context that the "string kernels" of "text processing" can be called "kernels" and the balance of Section 1.4.3 of Notes IV details ways of making kernels.