# Kernel method

From Wikipedia, the free encyclopedia

In machine learning, kernel methods are a class of algorithms for pattern analysis, whose best known member is the support vector machine (SVM). The general task of pattern analysis is to find and study general types of relations (for example clusters, rankings, principal components, correlations, classifications) in datasets. For many algorithms that solve these tasks, the data in raw representation have to be explicitly transformed into feature vector representations via a user-specified *feature map*: in contrast, kernel methods require only a user-specified *kernel*, i.e., a similarity function over pairs of data points in raw representation.

Kernel methods owe their name to the use of kernel functions, which enable them to operate in a high-dimensional, *implicit* feature space without ever computing the coordinates of the data in that space, but rather by simply computing the inner products between the images of all pairs of data in the feature space. This operation is often computationally cheaper than the explicit computation of the coordinates. This approach is called the "kernel trick". Kernel functions have been introduced for sequence data, graphs, text, images, as well as vectors.

Algorithms capable of operating with kernels include the kernel perceptron, support vector machines (SVM), Gaussian processes, principal components analysis (PCA), canonical correlation analysis, ridge regression, spectral clustering, linear adaptive filters and many others. Any linear model can be turned into a non-linear model by applying the kernel trick to the model: replacing its features (predictors) by a kernel function.

Most kernel algorithms are based on convex optimization or eigenproblems and are statistically well-founded. Typically, their statistical properties are analyzed using statistical learning theory (for example, using Rademacher complexity).

## Contents



## Motivation and informal explanation

Kernel methods can be thought of as instance-based learners: rather than learning some fixed set of parameters corresponding to the features of their inputs, they instead "remember" the $i$-th training example $(\mathbf{x}_i, y_i)$ and learn for it a corresponding weight $w_i$. Prediction for unlabeled inputs, i.e., those not in the training set, is treated by the application of a similarity function $k$, called a kernel, between the unlabeled input $\mathbf{x}'$ and each of the training inputs $\mathbf{x}_i$. For instance, a kernelized binary classifier typically computes a weighted sum of similarities

$$\hat{y} = \operatorname{sgn} \sum_{i=1}^{n} w_i y_i k(\mathbf{x}_i, \mathbf{x}'),$$

where

- $\hat{y} \in \{-1, +1\}$ is the kernelized binary classifier's predicted label for the unlabeled input $\mathbf{x}'$ whose hidden true label $y$ is of interest;
- $k: \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is the kernel function that measures similarity between any pair of inputs $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$;
- the sum ranges over the $n$ labeled examples $\{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$ in the classifier's training set, with $y_i \in \{-1, +1\}$;
- the $w_i \in \mathbb{R}$ are the weights for the training examples, as determined by the learning algorithm;
- the sign function $\operatorname{sgn}$ determines whether the predicted classification $\hat{y}$ comes out positive or negative.

Kernel classifiers were described as early as the 1960s, with the invention of the kernel perceptron.[1] They rose to great prominence with the popularity of the support vector machine (SVM) in the 1990s, when the SVM was found to be competitive with neural networks on tasks such as handwriting recognition.

## Mathematics: the kernel trick

The kernel trick avoids the explicit mapping that is needed to get linear learning algorithms to learn a nonlinear function or decision boundary. For all $\mathbf{x}$ and $\mathbf{x}'$ in the input space $\mathcal{X}$, certain functions $k(\mathbf{x}, \mathbf{x}')$ can be expressed as an inner product in another space $\mathcal{V}$. The function $k: \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is often referred to as a *kernel* or a *kernel function*. The word "kernel" is used in mathematics to denote a weighting function for a weighted sum or integral.

Certain problems in machine learning have additional structure than an arbitrary weighting function $k$. The computation is made much simpler if the kernel can be written in the form of a "feature map" $\varphi: \mathcal{X} \to \mathcal{V}$ which satisfies

$$k(\mathbf{x}, \mathbf{x}') = \langle \varphi(\mathbf{x}), \varphi(\mathbf{x}') \rangle_{\mathcal{V}}.$$

The key restriction is that $\langle \cdot, \cdot \rangle_{\mathcal{V}}$ must be a proper inner product. On the other hand, an explicit representation for $\varphi$ is not necessary, as long as $\mathcal{V}$ is an inner product space. The alternative follows from Mercer's theorem: an implicitly defined function $\varphi$ exists whenever the space $\mathcal{X}$ can be equipped with a suitable measure ensuring the function $k$ satisfies Mercer's condition.

Mercer's theorem is akin to a generalization of the result from linear algebra that associates an inner product to any positive-definite matrix. In fact, Mercer's condition can be reduced to this simpler case. If we choose as our measure the counting measure $\mu(T) = |T|$ for all $T \subset X$, which counts the number of points inside the set $T$, then the integral in Mercer's theorem reduces to a summation

$$\sum_{i=1}^{n} \sum_{j=1}^{n} k(\mathbf{x}_i, \mathbf{x}_j) c_i c_j \geq 0.$$

If this summation holds for all finite sequences of points $(\mathbf{x}_1, \ldots, \mathbf{x}_n)$ in $\mathcal{X}$ and all choices of $n$ real-valued coefficients $(c_1, \ldots, c_n)$ (cf. positive definite kernel), then the function $k$ satisfies Mercer's condition.

Some algorithms that depend on arbitrary relationships in the native space $\mathcal{X}$ would, in fact, have a linear interpretation in a different setting: the range space of $\varphi$. The linear interpretation gives us insight about the algorithm. Furthermore, there is often no need to compute $\varphi$ directly during computation, as is the case with support vector machines. Some cite this running time shortcut as the primary benefit. Researchers also use it to justify the meanings and properties of existing algorithms.

Theoretically, a Gram matrix $\mathbf{K} \in \mathbb{R}^{n \times n}$ with respect to $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ (sometimes also called a "kernel matrix"[2]), where $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$, must be positive semi-definite (PSD).[3] Empirically, for machine learning heuristics, choices of a function $k$ that do not satisfy Mercer's condition may still perform reasonably if $k$ at least approximates the intuitive idea of similarity.[4] Regardless of whether $k$ is a Mercer kernel, $k$ may still be referred to as a "kernel".

If the kernel function $k$ is also a covariance function as used in Gaussian processes, then the Gram matrix $\mathbf{K}$ can also be called a covariance matrix.[5]

Finally, suppose that $\mathbf{K}$ is a square matrix. Then $\mathbf{K}^{\mathrm{T}}\mathbf{K}$ is a positive-semi-definite matrix.

## Applications

Application areas of kernel methods are diverse and include geostatistics,[6] kriging, inverse distance weighting, 3D reconstruction, bioinformatics, chemoinformatics, information extraction and handwriting recognition.

## Popular kernels

- Fisher kernel
- Graph kernels
- Kernel smoother
- Polynomial kernel
- RBF kernel
- String kernels

## See also

- Kernel methods for vector output

## Notes

1. Aizerman, M. A.; Braverman, Emmanuel M.; Rozoner, L. I. (1964). "Theoretical foundations of the potential function method in pattern recognition learning". *Automation and Remote Control*. 25: 821–837. Cited in Guyon, Isabelle; Boser, B.; Vapnik, Vladimir (1993). *Automatic capacity tuning of very large VC-dimension classifiers*. Advances in neural information processing systems. CiteSeerX: 10.1.1.17.7215.
2. Hofmann, Thomas; Scholkopf, Bernhard; Smola, Alexander J. (2008). "Kernel Methods in Machine Learning".
3. Mohri, Mehryar; Rostamizadeh, Afshin; Talwalkar, Ameet (2012). *Foundations of Machine Learning*. The MIT Press. ISBN 9780262018258.
4. http://www.svms.org/mercer/
5. Rasmussen, C. E.; Williams, C. K. I. (2006). "Gaussian Processes for Machine Learning".
6. Honarkhah, M.; Caers, J. (2010). "Stochastic Simulation of Patterns Using Distance-Based Pattern Modeling". *Mathematical Geosciences*. 42: 487–517. doi:10.1007/s11004-010-9276-7.

## References


- Shawe-Taylor, J.; Cristianini, N. (2004). *Kernel Methods for Pattern Analysis*. Cambridge University Press.
- Liu, W.; Principe, J.; Haykin, S. (2010). *Kernel Adaptive Filtering: A Comprehensive Introduction*. Wiley.


## External links

- Kernel-Machines Org (http://www.kernel-machines.org)—community website
- www.support-vector-machines.org (http://www.support-vector-machines.org) *(Literature, Review, Software, Links related to Support Vector Machines - Academic Site)*

- onlineprediction.net Kernel Methods Article (http://onlineprediction.net/?n=Main.KernelMethods)

Retrieved from "https://en.wikipedia.org/w/index.php?title=Kernel_method&oldid=730750764"

Categories: Kernel methods for machine learning | Geostatistics | Classification algorithms

---