César Souza

Kernel Functions for Machine Learning Applications

MARCH 17, 2010 / CESARSOUZA / 49 COMMENTS

In recent years, Kernel methods have received major attention, particularly due to the increased popularity of the Support Vector Machines. Kernel functions can be used in many applications as they provide a simple bridge from linearity to non-linearity for algorithms which can be expressed in terms of dot products. In this article, we will list a few kernel functions and some of their properties.

- Check the source code for all kernel functions here.

Many of these functions have been incorporated in Accord.NET, a framework for creating machine learning, statistics, and computer vision applications.

Contents

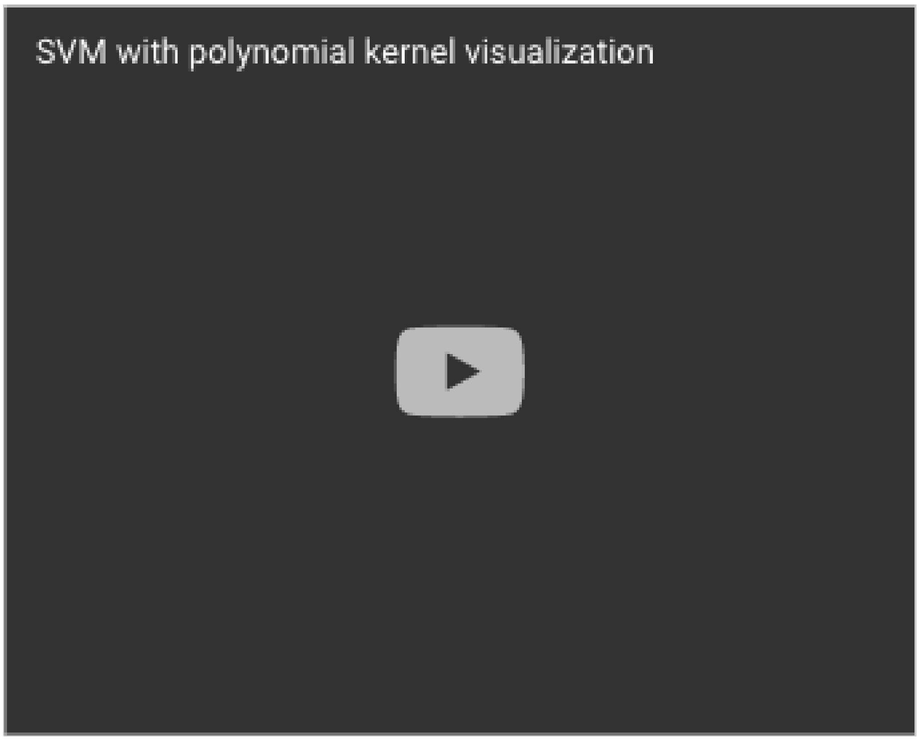
Kernel Methods

Kernel methods are a class of algorithms for pattern analysis or recognition, whose best known element is the support vector machine (SVM). The general task of pattern analysis is to find and study general types of relations (such as clusters, rankings, principal components, correlations, classifications) in general types of data (such as sequences, text documents, sets of points, vectors, images, graphs, etc) (Wikipedia, 2010a).

The main characteristic of Kernel Methods, however, is their distinct approach to this problem. Kernel methods map the data into higher dimensional spaces in the hope that in this higher-dimensional space the data could become more easily separated or better structured. There are also no constraints on the form of this mapping, which could even lead to infinite-dimensional spaces. This mapping function, however, hardly needs to be computed because of a tool called the kernel trick.

The Kernel trick

The Kernel trick is a very interesting and powerful tool. It is powerful because it provides a bridge from linearity to non-linearity to any algorithm that can expressed solely on terms of dot products between two vectors. It comes from the fact that, if we first map our input data into a higher-dimensional space, a linear algorithm operating in this space will behave non-linearly in the original input space.



Now, the Kernel trick is really interesting because that mapping does not need to be ever computed. If our algorithm can be expressed only in terms of a inner product between two vectors, all we need is replace this inner product with the inner product from some other suitable space. That is where resides the “trick”: wherever a dot product is used, it is replaced with a Kernel function. The kernel function denotes an inner product in feature space and is usually denoted as:

$$K(x,y) = <\phi(x),\phi(y)>$$

Using the Kernel function, the algorithm can then be carried into a higher-dimension space without explicitly mapping the input points into this space. This is highly desirable, as sometimes our higher-dimensional feature space could even be infinite-dimensional and thus unfeasible to compute.

Kernel Properties

Kernel functions must be continuous, symmetric, and most preferably should have a positive (semi-) definite Gram matrix. Kernels which are said to satisfy the Mercer's theorem are positive semi-definite, meaning their kernel matrices have only non-negative Eigen values. The use of a positive definite kernel insures that the optimization problem will be convex and solution will be unique.

However, many kernel functions which aren't strictly positive definite also have been shown to perform very well in practice. An example is the Sigmoid kernel, which, despite its wide use, it is not positive semi-definite for certain values of its parameters. Boughorbel (2005) also experimentally demonstrated that Kernels which are only conditionally positive definite can possibly outperform most classical kernels in some applications.

Kernels also can be classified as anisotropic stationary, isotropic stationary, compactly supported, locally stationary, nonstationary or separable nonstationary. Moreover, kernels can also be labeled scale-invariant or scale-dependant, which is an interesting property as scale-invariant kernels drive the training process invariant to a scaling of the data.

Choosing the Right Kernel

Choosing the most appropriate kernel highly depends on the problem at hand – and fine tuning its parameters can easily become a tedious and cumbersome task. Automatic kernel selection is possible and is discussed in the works by Tom Howley and Michael Madden.

The choice of a Kernel depends on the problem at hand because it depends on what we are trying to model. A polynomial kernel, for example, allows us to model feature conjunctions up to the order of the polynomial. Radial basis functions allows to pick out circles (or hyperspheres) – in constrast with the Linear kernel, which allows only to pick out lines (or hyperplanes).

The motivation behind the choice of a particular kernel can be very intuitive and straightforward depending on what kind of information we are expecting to extract about the data. Please see the final notes on this topic from Introduction to Information Retrieval, by Manning, Raghavan and Schütze for a better explanation on the subject.

Kernel Functions

Below is a list of some kernel functions available from the existing literature. As was the case with previous articles, every LaTeX notation for the formulas below are readily available from their alternate text html tag. I can not guarantee all of them are perfectly correct, thus use them at your own risk. Most of them have links to articles where they have been originally used or proposed.

1. Linear Kernel

- Linear kernel documentation – linear kernel source code – how to create SVMs in .NET with Accord.NET

The Linear kernel is the simplest kernel function. It is given by the inner product <x,y> plus an optional constant **c**. Kernel algorithms using a linear kernel are often equivalent to their non-kernel counterparts, i.e. KPCA with linear kernel is the same as standard PCA.

$$k(x, y) = x^T y + c$$

2. Polynomial Kernel

The Polynomial kernel is a non-stationary kernel. Polynomial kernels are well suited for problems where all the training data is normalized.

$$k(x, y) = (\alpha x^T y + c)^d$$

Adjustable parameters are the slope **alpha**, the constant term **c** and the polynomial degree **d**.

3. Gaussian Kernel

The Gaussian kernel is an example of radial basis function kernel.

$$k(x, y) = \exp\left(-\frac{||x - y||^2}{2\sigma^2}\right)$$

Alternatively, it could also be implemented using

$$k(x, y) = \exp\left(-\gamma ||x - y||^2\right)$$

The adjustable parameter **sigma** plays a major role in the performance of the kernel, and should be carefully tuned to the problem at hand. If overestimated, the exponential will behave almost linearly and the higher-dimensional projection will start to lose its non-linear power. In the other hand, if underestimated, the function will lack regularization and the decision boundary will be highly sensitive to noise in training data.

4. Exponential Kernel

The exponential kernel is closely related to the Gaussian kernel, with only the square of the norm left out. It is also a radial basis function kernel.

$$k(x, y) = \exp\left(-\frac{||x - y||}{2\sigma^2}\right)$$

5. Laplacian Kernel

The Laplace Kernel is completely equivalent to the exponential kernel, except for being less sensitive for changes in the sigma parameter. Being equivalent, it is also a radial basis function kernel.

$$k(x, y) = \exp\left(-\frac{||x - y||}{\sigma}\right)$$

It is important to note that the observations made about the sigma parameter for the Gaussian kernel also apply to the Exponential and Laplacian kernels.

6. ANOVA Kernel

The ANOVA kernel is also a radial basis function kernel, just as the Gaussian and Laplacian kernels. It is said to perform well in multidimensional regression problems (Hofmann, 2008).

$$k(x, y) = \sum_{k=1}^{n} \exp(-\sigma(x^k - y^k)^2)^d$$

7. Hyperbolic Tangent (Sigmoid) Kernel

The Hyperbolic Tangent Kernel is also known as the Sigmoid Kernel and as the Multilayer Perceptron (MLP) kernel. The Sigmoid Kernel comes from the Neural Networks field, where the bipolar sigmoid function is often used as an activation function for artificial neurons.

$$k(x, y) = \tanh(\alpha x^T y + c)$$

It is interesting to note that a SVM model using a sigmoid kernel function is equivalent to a two-layer, perceptron neural network. This kernel was quite popular for support vector machines due to its origin from neural network theory. Also, despite being only conditionally positive definite, it has been found to perform well in practice.

There are two adjustable parameters in the sigmoid kernel, the slope **alpha** and the intercept constant **c**. A common value for alpha is 1/N, where N is the data dimension. A more detailed study on sigmoid kernels can be found in the works by Hsuan-Tien and Chih-Jen.

8. Rational Quadratic Kernel

The Rational Quadratic kernel is less computationally intensive than the Gaussian kernel and can be used as an alternative when using the Gaussian becomes too expensive.

$$k(x, y) = 1 - \frac{||x - y||^2}{||x - y||^2 + c}$$

9. Multiquadric Kernel

The Multiquadric kernel can be used in the same situations as the Rational Quadratic kernel. As is the case with the Sigmoid kernel, it is also an example of an non-positive definite kernel.

$$k(x, y) = \sqrt{||x - y||^2 + c^2}$$

10. Inverse Multiquadric Kernel

The Inverse Multi Quadric kernel. As with the Gaussian kernel, it results in a kernel matrix with full rank (Micchelli, 1986) and thus forms a infinite dimension feature space.

$$k(x, y) = \frac{1}{\sqrt{||x - y||^2 + c^2}}$$

11. Circular Kernel

The circular kernel is used in geostatic applications. It is an example of an isotropic stationary kernel and is positive definite in *R*^{2}.

$$k(x,y) = \frac{2}{\pi} \arccos(-\frac{||x-y||}{\sigma}) - \frac{2}{\pi} \frac{||x-y||}{\sigma} \sqrt{1 - \left(\frac{||x-y||}{\sigma}\right)^2}$$

$$\text{if } ||x-y|| < \sigma \text{, zero otherwise}$$

12. Spherical Kernel

The spherical kernel is similar to the circular kernel, but is positive definite in R^3.

$$k(x,y) = 1 - \frac{3}{2} \frac{||x-y||}{\sigma} + \frac{1}{2} \left(\frac{||x-y||}{\sigma}\right)^3$$

$$\text{if } ||x-y|| < \sigma \text{, zero otherwise}$$

13. Wave Kernel

The Wave kernel is also symmetric positive semi-definite (Huang, 2008).

Invalid Equation

14. Power Kernel

The Power kernel is also known as the (unrectified) triangular kernel. It is an example of scale-invariant kernel (Sahbi and Fleuret, 2004) and is also only conditionally positive definite.

$$k(x,y) = -||x-y||^d$$

15. Log Kernel

The Log kernel seems to be particularly interesting for images, but is only conditionally positive definite.

$$k(x,y) = -log(||x-y||^d + 1)$$

16. Spline Kernel

The Spline kernel is given as a piece-wise cubic polynomial, as derived in the works by Gunn (1998).

$$k(x,y) = 1 + xy + xy \; min(x,y) - \frac{x+y}{2} \; min(x,y)^2 + \frac{1}{3} \min(x,y)^3$$

However, what it actually mean is:

$$k(x,y) = \prod_{i=1}^{d} 1 + x_i y_i + x_i y_i \min(x_i,y_i) - \frac{x_i+y_i}{2} \min(x_i,y_i)^2 + \frac{\min(x_i,y_i)^3}{3}$$

With

$$x, y \in R^d$$

17. B-Spline (Radial Basis Function) Kernel

The B-Spline kernel is defined on the interval [−1, 1]. It is given by the recursive formula:

$$k(x, y) = B_{2p+1}(x - y)$$

$$\text{where } p \in N \text{ with } B_{i+1} := B_i \otimes B_0.$$

In the work by Bart Hamers it is given by:

$$k(x, y) = \prod_{p=1}^{d} B_{2n+1}(x_p - y_p)$$

Alternatively, B_n can be computed using the explicit expression (Fomel, 2000):

$$B_n(x) = \frac{1}{n!} \sum_{k=0}^{n+1} \binom{n+1}{k} (-1)^k (x + \frac{n+1}{2} - k)_+^n$$

Where x_+ is defined as the truncated power function:

$$x_+^d = \begin{cases} x^d, & \text{if } x > 0 \\ 0, & \text{otherwise} \end{cases}$$

18. Bessel Kernel

The Bessel kernel is well known in the theory of function spaces of fractional smoothness. It is given by:

$$k(x, y) = \frac{J_{v+1}(\sigma ||x - y||)}{||x - y||^{-n(v+1)}}$$

where J is the Bessel function of first kind. However, in the Kernlab for R documentation, the Bessel kernel is said to be:

$$k(x, x') = -Bessel_{(nu+1)}^{n}(\sigma |x - x'|^2)$$

19. Cauchy Kernel

The Cauchy kernel comes from the Cauchy distribution (Basak, 2008). It is a long-tailed kernel and can be used to give long-range influence and sensitivity over the high dimension space.

$$k(x, y) = \frac{1}{1 + \frac{||x-y||^2}{\sigma^2}}$$

20. Chi-Square Kernel

The Chi-Square kernel comes from the Chi-Square distribution:

$$k(x, y) = 1 - \sum_{i=1}^{n} \frac{(x_i - y_i)^2}{\frac{1}{2}(x_i + y_i)}$$

However, as noted by commenter Alexis Mignon, this version of the kernel is only conditionally positive-definite (CPD). A positive-definite version of this kernel is given in (Vedaldi and Zisserman, 2011) as

$$k(x, y) = \sum_{i=1}^{n} \frac{2x_i y_i}{(x_i + y_i)}$$

and is suitable to be used by methods other than support vector machines.

21. Histogram Intersection Kernel

The Histogram Intersection Kernel is also known as the Min Kernel and has been proven useful in image classification.

$$k(x, y) = \sum_{i=1}^{n} \min(x_i, y_i)$$

22. Generalized Histogram Intersection

The Generalized Histogram Intersection kernel is built based on the Histogram Intersection Kernel for image classification but applies in a much larger variety of contexts (Boughorbel, 2005). It is given by:

$$k(x, y) = \sum_{i=1}^{m} \min(|x_i|^{\alpha}, |y_i|^{\beta})$$

23. Generalized T-Student Kernel

The Generalized T-Student Kernel has been proven to be a Mercel Kernel, thus having a positive semi-definite Kernel matrix (Boughorbel, 2004). It is given by:

$$k(x, y) = \frac{1}{1 + ||x - y||^d}$$

24. Bayesian Kernel

The Bayesian kernel could be given as:

$$k(x, y) = \prod_{l=1}^{N} \kappa_l(x_l, y_l)$$

where

$$\kappa_l(a,b) = \sum_{c\in\{0;1\}} P(Y=c \mid X_l = a)\ P(Y=c \mid X_l = b)$$

However, it really depends on the problem being modeled. For more information, please see the work by Alashwal, Deris and Othman, in which they used a SVM with Bayesian kernels in the prediction of protein-protein interactions.

25. Wavelet Kernel

The Wavelet kernel (Zhang et al, 2004) comes from Wavelet theory and is given as:

$$k(x,y) = \prod_{i=1}^{N} h(\frac{x_i - c}{a})\ h(\frac{y_i - c}{a})$$

Where **a** and **c** are the wavelet dilation and translation coefficients, respectively (the form presented above is a simplification, please see the original paper for details). A translation-invariant version of this kernel can be given as:

$$k(x,y) = \prod_{i=1}^{N} h(\frac{x_i - y_i}{a})$$

Where in both h(x) denotes a mother wavelet function. In the paper by Li Zhang, Weida Zhou, and Licheng Jiao, the authors suggests a possible h(x) as:

$$h(x) = cos(1.75x)exp(-\frac{x^2}{2})$$

Which they also prove as an admissible kernel function.

Source Code

The latest version of the source code for almost all of the kernels listed above is available in the Accord.NET Framework. Some are also available in the sequel of this article, Kernel Support Vector Machines for Classification and Regression in C#. They are provided together with a comprehensive and simple implementation of SVMs (Support Vector Machines) in C#. However, for the latest sources, which may contain bug fixes and other enhancements, please download the most recent version available of Accord.NET.

See also

- Kernel Support Vector Machines (kSVMs)
- Principal Component Analysis (PCA)
- Kernel Principal Component Analysis (KPCA)
- Linear Discriminant Analysis (LDA)
- Non-Linear Discriminant Analysis with Kernels (KDA)

- Logistic Regression Analysis in C#
- The Accord.NET Framework: Scientific Computing in .NET
- Haar-feature object detection in C# (The Viola-Jones Classifier)
- Handwriting Recognition using Kernel Discriminant Analysis
- Handwriting Recognition Revisited: Kernel Support Vector Machines
- Logistic Regression Analysis

References

- On-Line Prediction Wiki Contributors. "Kernel Methods." On-Line Prediction Wiki. http://onlineprediction.net/?n=Main.KernelMethods (accessed March 3, 2010).
- Genton, Marc G. "Classes of Kernels for Machine Learning: A Statistics Perspective." Journal of Machine Learning Research 2 (2001) 299-312.
- Hofmann, T., B. Schölkopf, and A. J. Smola. "Kernel methods in machine learning." Ann. Statist. Volume 36, Number 3 (2008), 1171-1220.
- Gunn, S. R. (1998, May). "Support vector machines for classification and regression." Technical report, Faculty of Engineering, Science and Mathematics School of Electronics and Computer Science.
- Karatzoglou, A., Smola, A., Hornik, K. and Zeileis, A. "Kernlab – an R package for kernel Learning." (2004).
- Karatzoglou, A., Smola, A., Hornik, K. and Zeileis, A. "Kernlab – an S4 package for kernel methods in R." J. Statistical Software, 11, 9 (2004).
- Karatzoglou, A., Smola, A., Hornik, K. and Zeileis, A. "R: Kernel Functions." Documentation for package 'kernlab' version 0.9-5. http://rss.acs.unt.edu/Rdoc/library/kernlab/html/dots.html (accessed March 3, 2010).
- Howley, T. and Madden, M.G. "The genetic kernel support vector machine: Description and evaluation". Artificial Intelligence Review. Volume 24, Number 3 (2005), 379-395.
- Shawkat Ali and Kate A. Smith. "Kernel Width Selection for SVM Classification: A Meta-Learning Approach." International Journal of Data Warehousing & Mining, 1(4), 78-97, October-December 2005.
- Hsuan-Tien Lin and Chih-Jen Lin. "A study on sigmoid kernels for SVM and the training of non-PSD kernels by SMO-type methods." Technical report, Department of Computer Science, National Taiwan University, 2003.
- Boughorbel, S., Jean-Philippe Tarel, and Nozha Boujemaa. "Project-Imedia: Object Recognition." INRIA – INRIA Activity Reports – RalyX. http://ralyx.inria.fr/2004/Raweb/imedia/uid84.html (accessed March 3, 2010).
- Huang, Lingkang. "Variable Selection in Multi-class Support Vector Machine and Applications in Genomic Data Analysis." *PhD Thesis*, 2008.
- Manning, Christopher D., Prabhakar Raghavan, and Hinrich Schütze. "Nonlinear SVMs." The Stanford NLP (Natural Language Processing) Group. http://nlp.stanford.edu/IR-book/html/htmledition/nonlinear-svms-1.html (accessed March 3, 2010).
- Fomel, Sergey. "Inverse B-spline interpolation." *Stanford Exploration Project*, 2000. http://sepwww.stanford.edu/public/docs/sep105/sergey2/paper_html/node5.html (accessed March 3, 2010).
- Basak, Jayanta. "A least square kernel machine with box constraints." *International Conference on Pattern Recognition 2008* 1 (2008): 1-4.
- Alashwal, H., Safaai Deris, and Razib M. Othman. "A Bayesian Kernel for the Prediction of Protein – Protein Interactions." *International Journal of Computational Intelligence* 5, no. 2 (2009): 119-124.
- Hichem Sahbi and François Fleuret. "Kernel methods and scale invariance using the triangular kernel". INRIA Research Report, N-5143, March 2004.
- Sabri Boughorbel, Jean-Philippe Tarel, and Nozha Boujemaa. "Generalized histogram intersection kernel for image recognition". *Proceedings of the 2005 Conference on Image Processing*, volume 3, pages 161-164, 2005.
- Micchelli, Charles. Interpolation of scattered data: Distance matrices and conditionally positive definite functions. *Constructive Approximation* 2, no. 1 (1986): 11-22.
- Wikipedia contributors, "Kernel methods," *Wikipedia, The Free Encyclopedia,* http://en.wikipedia.org/w/index.php?title=Kernel_methods&oldid=340911970 (accessed March 3, 2010).
- Wikipedia contributors, "Kernel trick," *Wikipedia, The Free Encyclopedia,* http://en.wikipedia.org/w/index.php?title=Kernel_trick&oldid=269422477 (accessed March 3, 2010).

- Weisstein, Eric W. "Positive Semidefinite Matrix." From *MathWorld*–A Wolfram Web Resource. http://mathworld.wolfram.com/PositiveSemidefiniteMatrix.html
- Hamers B. "Kernel Models for Large Scale Applications", *Ph.D.* , Katholieke Universiteit Leuven, Belgium, 2004.
- Li Zhang, Weida Zhou, Licheng Jiao. Wavelet Support Vector Machine. IEEE Transactions on System, Man, and Cybernetics, Part B, 2004, 34(1): 34-39.
- Vedaldi, A. and Zisserman, A. Efficient Additive Kernels via Explicit Feature Maps. IEEE Transactions on Pattern Recognition and Machine Intelligence, Vol. XX, No. XX, June, 2011.

Citing this work

If you would like, please cite this work as: Souza, César R. "Kernel Functions for Machine Learning Applications." 17 Mar. 2010. Web. <http://crsouza.blogspot.com/2010/03/kernel-functions-for-machine-learning.html>.

C#, Mathematics, Statistics

49 Comments

mwiley63
APRIL 25, 2010 AT 4:10 AM

Great article!

I am just learning about perceptrons and kernels, and found this information very helpful.

When will you be able to post some source code? I am trying to program a kernel perceptron right now, but am unable to figure out how to build the kernel function. I am having trouble connecting my example to the theory of the kernel function.

César Souza
APRIL 26, 2010 AT 11:48 AM

Hi,

Soon I'm going to post a new article about (Kernel) Support Vector Machines, which will also contain the code for most Kernels presented here.

However, I just updated my Kernel Discriminant Analysis code and also included some of them there. You could try checking it, if you wish.

Good luck,
César

Uday

MAY 29, 2010 AT 10:45 AM

César
Thanks for nice article. I wanted to find the parameter range for each like sigma in Cauchy, Laplace etc do they vary in all the real number range? To optimize it.
Also theta mentioned does it vary from 0 to 360?
Thanks in advance
Uday

César Souza
MAY 29, 2010 AT 1:40 PM

Hi Uday,

In the paper Practical Selection of SVM Parameters and Noise Estimation for SVM Regression the authors have taken sigma values in the range (0.2~0.5)*range(x) for the Gaussian kernel, x being their input data. If the input data was normalized to be in the [0,1] range, then perhaps good choices for sigma would lie in the [0.2,0.5] range.

But this is for the Gaussian kernel. I have linked the original sources for each kernel in the post, perhaps you may found additional information about those kernels in their original paper.

The theta values in the Multiquadric kernels are just positive constants. Perhaps I should change the symbols to something else. For the Wave kernel, perhaps good values would be in the 0~2*pi range.

In most cases, a grid search would be required to find the most suitable Kernel parameters. In this page you can find more directions for parameter tuning in SVMs.

Regards,
César

Uday
MAY 30, 2010 AT 2:57 PM

Thanks César for link and suggestions. For Quadratic, Inverse Quadratic and MutiQuadratic the constant will be like shifting the axis? Do you think moving it in real range 0 to RealMax would have impact on learning?
Uday

César Souza
MAY 31, 2010 AT 5:13 PM

I am not very sure of the exact interpretation of the constant parameter in those Kernels, but its value will probably affect learning. That is why a grid search or a pattern search is advisable in this case.

Regards,
César

Zhao
JULY 13, 2010 AT 8:19 AM

Nice article, very comprehensive for the available kernel functions! Suggestion: the wavelet kernel may be added in the list.

Zhao
JULY 15, 2010 AT 11:24 PM

I am dubious about the form of the ANOVA kernel mentioned in the above article. The summation is right?

César Souza
AUGUST 18, 2010 AT 12:49 AM

Just cleaning some old comments to avoid confusion. There was a problem with the ANOVA kernel and it has been corrected, thanks to Zhao. A preliminary version of the Wavelet kernel will also be available soon.

Cheers!
César

Anonymous
JANUARY 21, 2011 AT 5:54 PM

Thanks for the tool; a great work in fact. Can you please add the RBF kernel? it seems to be widely used.
Thanks

César Souza
JANUARY 21, 2011 AT 9:25 PM

Hi Anonymous,

The RBF is not exactly a kernel but a family of kernels sharing the same common form: k(x,y) = exp(-gamma*||x-y||^2), with gamma being a positive constant. It is the same form depicted in the alternate form of the Gaussian kernel in this post. The most popular member of the RBF family of kernels is the Gaussian kernel, which can be derived from the general RBF form by taking gamma = 1/(sigma^2).

Best regards,
César

Abhinav Maurya

APRIL 11, 2011 AT 5:55 AM

The formulation of Circulation Kernel is slightly incorrect. Particularly the square inside the square-root applies to numerator and denominator both.

Anonymous
APRIL 14, 2011 AT 3:40 AM

u done great job. awesome. do you have source code for genetic kernel for support vector machine in java or Simple SMO in java

César Souza
APRIL 16, 2011 AT 4:50 PM

Hi Abhinav,

Thanks for noticing. I have updated the article with the correction.

@Anonymous: No, unfortunately I do not have any code in Java. Perhaps you could try taking a look on Weka.

Best Regards,
César

Anonymous
SEPTEMBER 27, 2011 AT 1:45 PM

I have a question about the multiquadratic kernel. Usually a kernel value is larger, if the vectors are more similar to each other. But with the multiquadratic kernel, it is just the opposite. Can this work? Do you have to switch the classification value of the SVM?

César Souza
SEPTEMBER 28, 2011 AT 4:30 PM

Hi Anonymous,

Indeed, the Multiquadric kernel is not positive definite. Actually, it is a negative-definite kernel. I have included it for theoretical interest in presenting the Inverse Multiquadric kernel, but I have not applied it to any dataset to see how it performs.

However, the interesting thing to note is that some non-positive (semi-)definite kernels have been used with relative success in practice, albeit it isn't so clear in which situations they work. In some cases, an special or modified version of the learning algorithm which can deal with non-PSD kernels is required.

For further information, perhaps you can try taking a look on the work by Hsuan-Tien Lin and Chih-Jen Lin for some detailed discussion on the use of the (non-PSD) sigmoid kernel.

All the best,
César

Anonymous
OCTOBER 11, 2011 AT 6:11 PM

Very nice and helpful article!!
Thanks.

Android apps development
NOVEMBER 19, 2011 AT 4:39 AM

Thanks for sharing your info. I really appreciate your efforts and I will be waiting for your further write ups thanks once again.
Android app development| Android app developer|

Anonymous
DECEMBER 3, 2011 AT 8:55 PM

Hi César Souza,

Thanks for this great article. Can you please upload the MATLAB codes for SVM Training & classification with Gaussian Kernel for imagr datasets. I want to classify the images into different emotions.

madhuri
JANUARY 30, 2012 AT 5:16 AM

Once in my college studies, I got to know about kernel function, and now thru ur blog I remember that and even got to know more about it.Cheers !!
Web Shopping Cart | Free Ecommerce Software

François-Xavier Thomas
MARCH 5, 2012 AT 4:30 PM

Just one word : wow!
I have a course on kernel methods in college, but we were lacking this list of known useful kernels, so thanks for taking the time to post this!

Cool boy
AUGUST 24, 2012 AT 8:18 AM

I'm surely coming again to read these articles and blogs
injection molding machines

sinan ABO ALCHAMLAT
FEBRUARY 6, 2013 AT 9:35 AM

Nice article, you done great job, I have a question how i can compare between two function Kernel (Gaussian Kernel, IBS Kernel)

andera
SEPTEMBER 9, 2013 AT 10:48 AM

Thank you a lot for this contribution.
I have to minor remarks however.

The expression for the Cauchy kernel should have the squared sigma at the denominator
(It doesn't change the shape of the kernel, just for correctness).

In Basak (2008) the authors says:
"Note that, it is not possible to use Cauchy and Gaussian+Exponential kernels in SVM since these kernels are not Mercer kernels."

So, apparently, Cauchy is not a Mercer kernel.

César Souza
OCTOBER 20, 2013 AT 9:21 PM

This is really interesting, as even if it is not a Mercer's kernel, the Cauchy often works well within a SVM. Thanks for sharing this information!

Also, I've added the squared sigma in the formula. Thanks!

Regards,
Cesar

blogBJblog
FEBRUARY 11, 2014 AT 4:35 PM

Do you have any more information about the Log Kernel? I'm currently using it to segment color images and finding it to be very powerful but I have only found one source (S Boughorbel et. al. 2005).

Anonymous
AUGUST 10, 2014 AT 10:02 AM

Nice work

New Tech Show
NOVEMBER 24, 2014 AT 4:41 PM

very deep on kernels, if you can share more about how to use kernel functions on real life problems like plagiarism check or spam filtering

---

Khalid Ibrahim
FEBRUARY 10, 2015 AT 9:43 AM

very special. easy to understand…. taking things from root .. stay blessed

---

Alexis Mignon
FEBRUARY 20, 2015 AT 8:56 AM

Note that the chi2 kernel you propose is most probably not positive definite (PD) but only conditionally positive definite (CPD), while for a number of algorithms (including SVM) this has no consequence, since only the induced distance metric is important, for algorithms relying the PD property (like KernelPCA) this may be problematic. A PD kernel inducing the chi2 distance metrics is:

k(x, y) = sum_i 2 x_i y_i / (x_i + y_i)

For some explanation about the link between PD and CPD kernels see for instance:
Hilbertian metrics and Positive definite kernels on probability measures. M. Hein and O. Bousquet. Techincal Report. Max Plank Institute

For an example of use of this chi2 PD kernel see:
Efficient Additive Kernels via Explicit Feature Maps

César Souza
FEBRUARY 20, 2015 AT 2:10 PM

Hi Alexis,

Thanks! I will correct the post with this addition!

Taylor Smith
MARCH 2, 2015 AT 4:15 PM

Hi César, wonderful article. I wonder (specifically for the Gaussian and related Kernels) what norm is being used? L2, L1 or is that denotation for vector length? Thanks!

César Souza
MARCH 2, 2015 AT 5:58 PM

Hi Taylor!

Unless otherwise noted, all the functions are using the L2 norm (or simply, the Euclidean norm).

Best regards,
Cesar

Anonymous
MARCH 6, 2015 AT 2:12 AM

Nice post but I think you have a typo under Kernel Properties? you said "Kernels which are said to satisfy the Mercer's theorem are positive semi-definite, meaning their kernel matrices have no non-negative Eigen values."
I think you mean …their kernel matrices have only non-negative eigenvalues..i.e. all eigenvalues >= 0, not "no non-negative eigenvalues" which would imply all negative eigenvalues.

César Souza
MARCH 6, 2015 AT 7:01 AM

Thanks! It was indeed a typo, thanks for noticing and reporting it!

Pingback: Liblinear algorithms in C# | ~/cesarsouza/blog

Pingback: Partial Least Squares Analysis and Regression in C# | ~/cesarsouza/blog

Pan Chao
APRIL 7, 2015 AT 2:56 PM

Hi Cesar,

Thank you for this excellent post. I learned a lot. I wonder which file in [your github framework repository](https://github.com/accord-net/framework) contains the codes for those kennel functions?

Thanks,
Pan

cesarsouza@gmail.com ★
APRIL 7, 2015 AT 6:42 PM

Hi Pan,

I corrected the links in the post. I have included a link to the Kernels directory right on the top of this post. In any case, the direct link is:
https://github.com/accord-net/framework/tree/development/Sources/Accord.Statistics/Kernels

Hope it helps!

Best regards,
Cesar

Pingback: Quora

Pingback: Just Another Kernel Cookbook (in Python) | Just Another Blog

---

Pingback: Text Classification and Sentiment Analysis | Ahmet Taspinar

---

Kaan Akşit
DECEMBER 17, 2015 AT 6:48 PM

Excellent read! Helped a lot!

Thank you,
Kaan

Pingback: Text Classification & Sentiment Analysis tutorial / blog | Make Data Work

---

Pritesh Jain
FEBRUARY 23, 2016 AT 12:08 PM

Visit for great iOS android source code . you can also sell your app templates or code here ”
mobile app development and keep working on freedom251

Gunjan Naik
MAY 1, 2016 AT 5:39 AM

Does these kernels same as kernel which are used in image processing? In image processing,Gaussian kernel is used for blurring of image.

How to find kernel matrix from a kernel equation?

MOVIECHARGE
SEPTEMBER 29, 2016 AT 12:00 AM

I want mathlat code

Pablo
JANUARY 19, 2017 AT 9:35 AM

First of all, Thank you very much for your work. It have been helping me to understand a lot of things about machine learning. I have a question:
After you get the model from SVM, how do you get the equation of the model, for example, if the machine is created using a polynomial kernel (degree = 1), how do you get the line equation (mx + b) of this model. Is there any method in the svm class to get the model equation?

Gopinath V
MARCH 17, 2017 AT 6:38 PM

Is this possible to extend this to identify hand written Alphabets ?