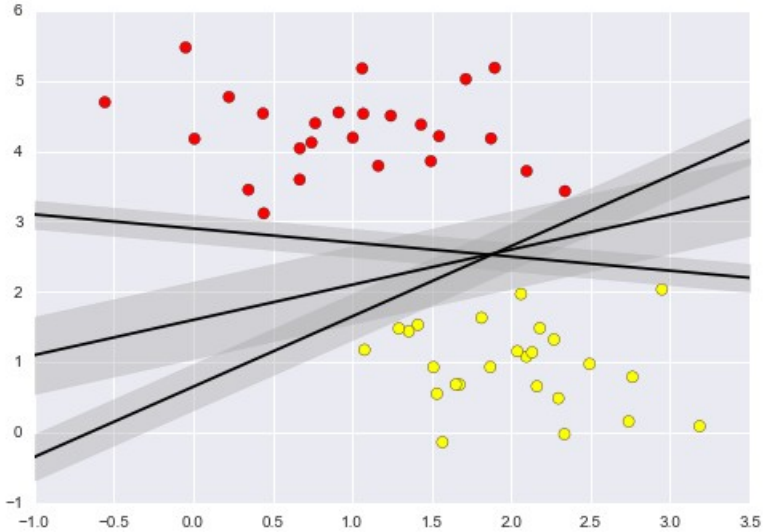
Kernel algorithms

Vlad Gladkikh

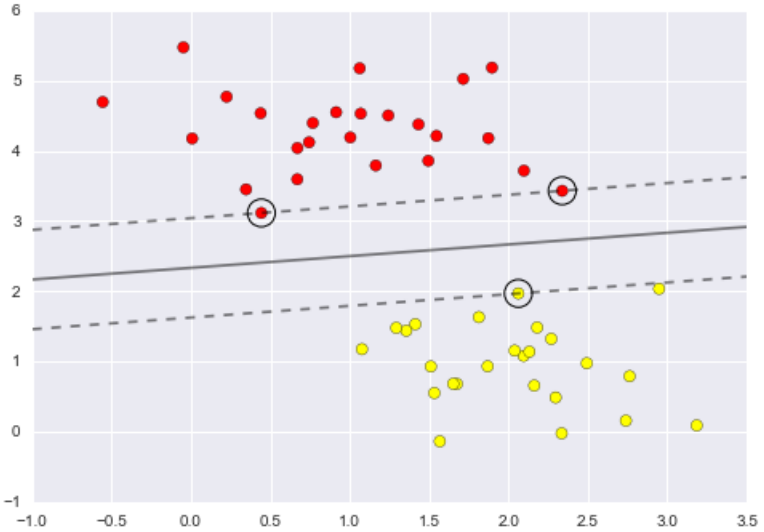
IBS CMCM

Support vector machines (SVMs)



Classification

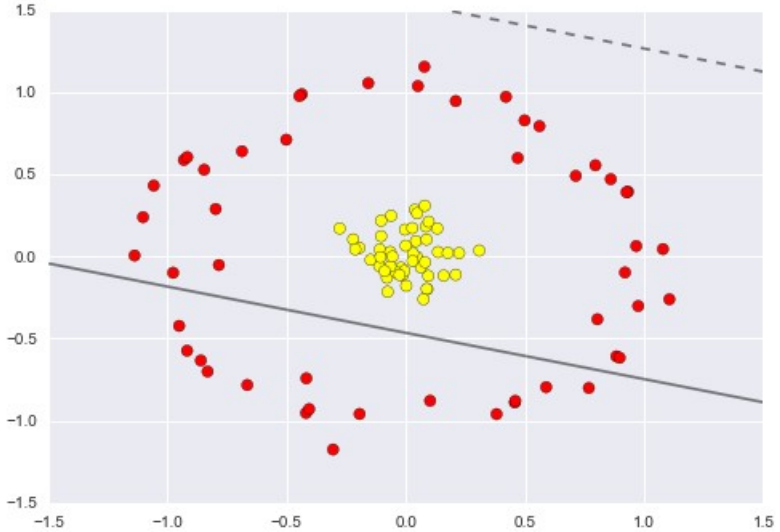
Maximizing the margin

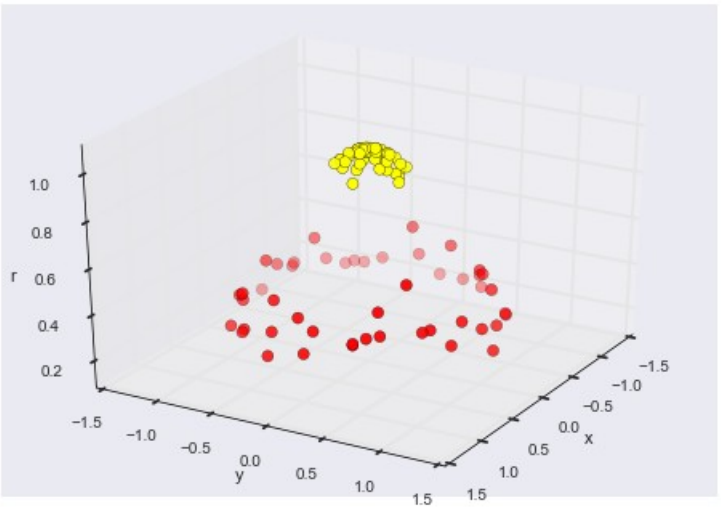


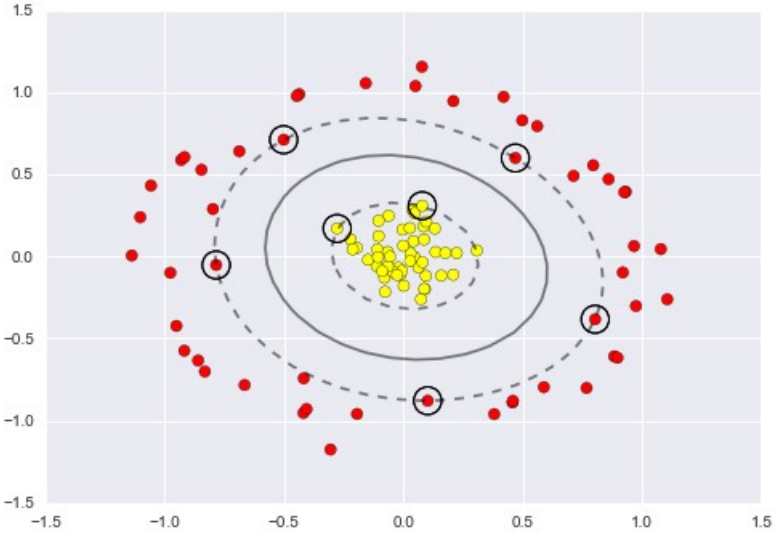
Support vectors: points that touch the margin

https://jakevdp.github.io/PythonDataScienceHandbook/05.07-support-vector-machines.html

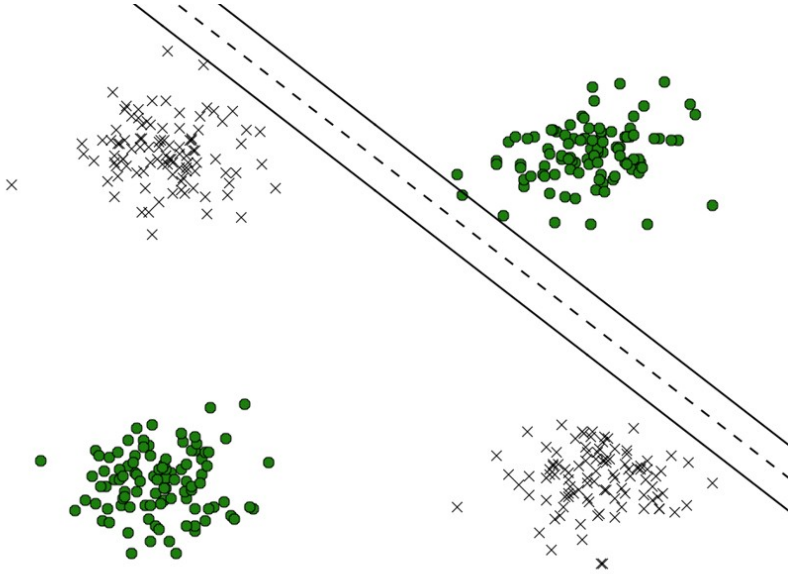
https://towardsdatascience.com/an-introduction-to-support-vector-regression-svr-a3ebc1672c2





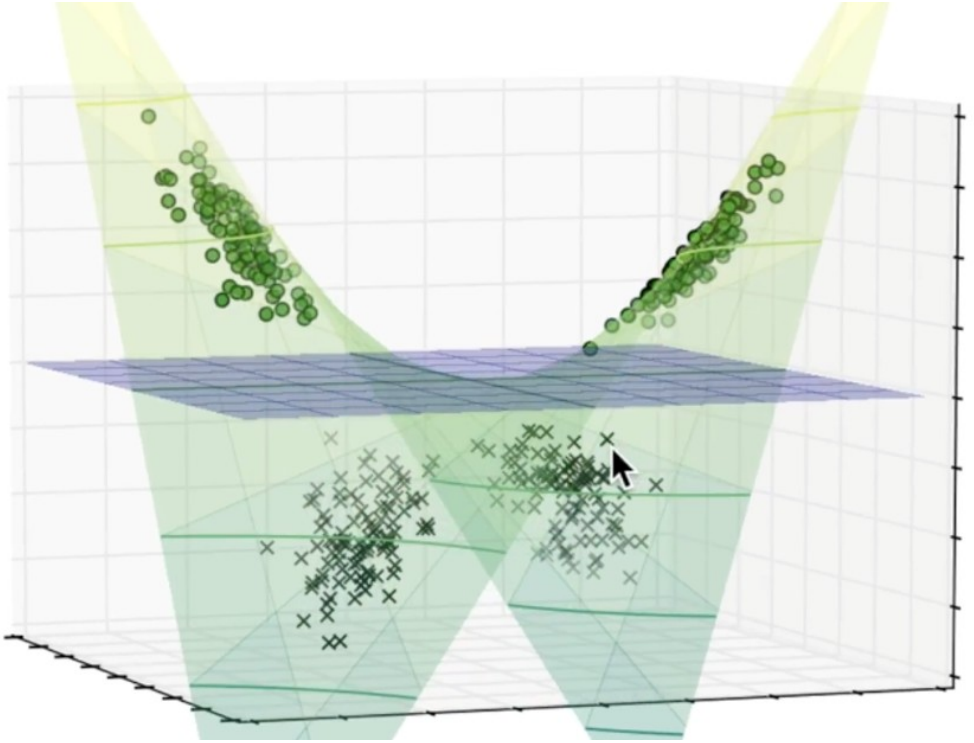


https://jakevdp.github.io/PythonDataScienceHandbook/05.07-support-vector-machines.html



Learning with kernels:

Mapping the inputs into a higher- dimensional space and applying the linear algorithm there.

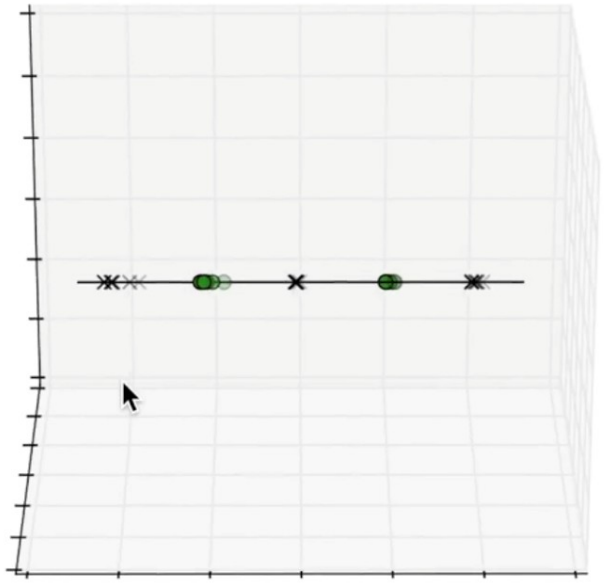


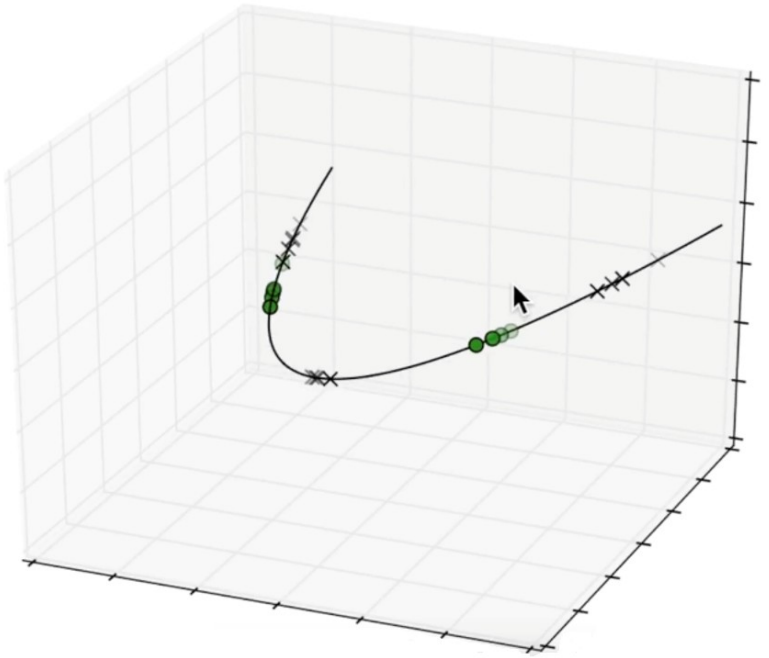
Problems:

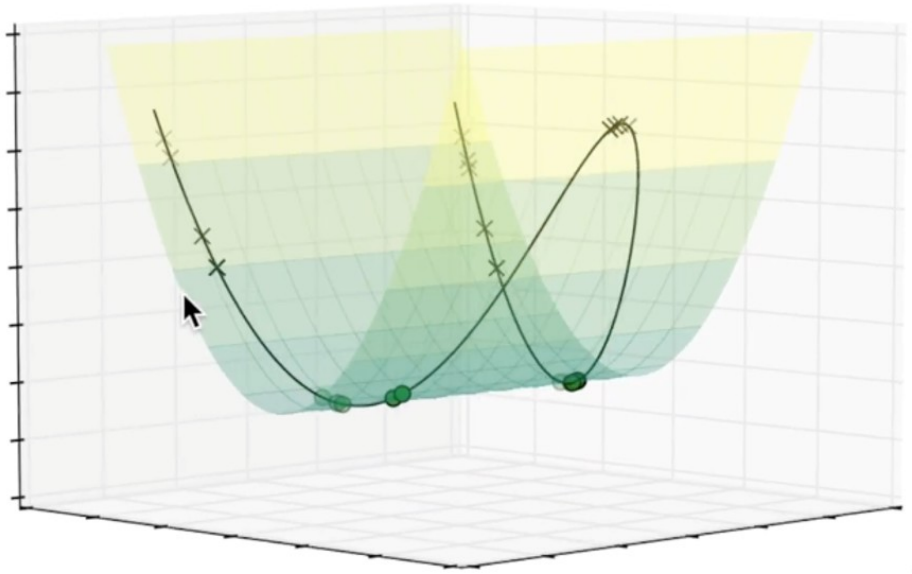
– computational complexity

– how to find the right mapping

https://e2eml.school/open_the_black_box.html







https://e2eml.school/open_the_black_box.html

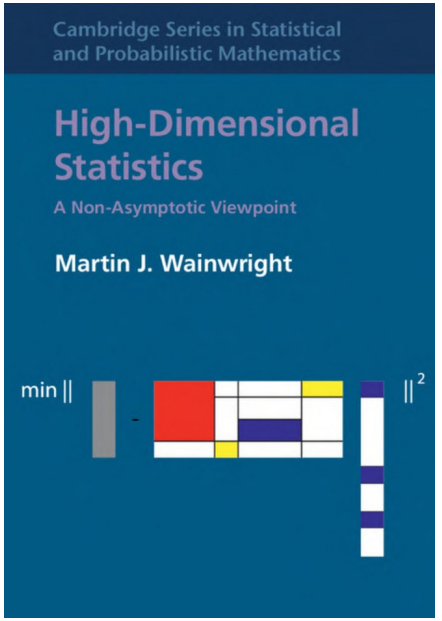
I. Steinwart, D. Hush, C. Scovel. An Explicit Description of the Reproducing Kernel Hilbert Spaces of Gaussian RBF Kernels (2006) IEEE Trans. Inform. Theory 2006, 52, 4635 DOI: 10.1109/TIT.2006.881713 https://ieeexplore.ieee.org/document/1705021

Matthias Rupp, Machine learning for quantum mechanics in a nutshell https://doi.org/10.1002/qua.24954

K.-R. Muller et al. An introduction to kernel-based learning algorithms
https://ieeexplore.ieee.org/document/914517

N. Aronszajn, Theory of Reproducing Kernels. Trans. Am. Math. Soc. 1950, 68, 337
https://doi.org/10.1090/S0002-9947-1950-0051437-7

Vu et al. Understanding kernel ridge regression: Common behaviors from simple functions to density functionals. https://onlinelibrary.wiley.com/doi/full/10.1002/qua.24939



Chapter 12

Many linear ML algorithms can be rewritten to use only inner products between inputs.

– e.g. info about norms, angles, distances, i.e., about relations between inputs

This reduces the problem of arbitrary computations with feature space vectors to computing inner products between them.

One can replace evaluation of inner products in feature space by evaluations of a kernel function in input space.

Kernels operate on input space vectors, but yield the same results as inner product evaluations in feature space.

Riesz representation theorem

$$\exists M < \infty \ : \ |L(f)| \le M \|f\|_H \ \forall \ f \in H$$

$$L : H \rightarrow R$$

Let L be a bounded linear functional on a Hilbert space H.

$$L(f + \alpha g) = L(f) + \alpha L(g) \ \ \forall \ f, g \in H \ \forall \ \alpha \in R$$

Then there exists a unique $g \in H$ such that $L(f) = \langle f, g \rangle_H$ for all $f \in H$

g is called the **representer** of the functional L

A **kernel** is a function that corresponds to an inner product in some feature space.

$$\forall\, x_1, x_2 \in X : k(x_1, x_2) = \langle \phi(x_1), \phi(x_2) \rangle$$

It is not necessary to know ϕ explicitly, their existence is sufficient.

$$f(x) = \sum_{i=1}^{n} \alpha_i \langle \phi(x_i), \phi(x) \rangle = \sum_{i=1}^{n} \alpha_i k(x_i, x)$$

α_i – regression coefficients
x_i – training inputs

Examples of kernels

Linear kernel $k(x_1, x_2) = \langle x_1, x_2 \rangle$

– identical input and feature space, $\phi(x) = x$

Gaussian kernel $k(x_1, x_2) = \exp\left(-\frac{\|x_1 - x_2\|_2^2}{2\sigma^2}\right)$

Laplacian kernel: $k(x_1, x_2) = \exp\left(-\frac{\|x_1 - x_2\|_1}{\sigma}\right)$

Sinusoidal Fourier basis functions $\phi_j(x) = \sin\left(\frac{(2j-1)\pi x}{2}\right), \quad j \in \{1,2,...\}$

$$\langle \phi_j, \phi_k \rangle_{L^2[0,1]} = \int_0^1 \phi_j(x)\phi_k(x)\,dx = \begin{cases} 1 & if\ j=k \\ 0 & otherwise \end{cases}$$

Given some sequence $(\mu)_{j=1}^{\infty}, \quad \mu_j \geq 0, \quad \sum_{j=1}^{\infty} \mu_j < \infty$ let us define the feature map

$$\Phi(x) = \left(\sqrt{\mu_1}\phi_1(x), \sqrt{\mu_2}\phi_2(x), ...\right)$$

$$\Phi(x) \in l^2(N) = \left\{ (\theta_j)_{j=1}^{\infty} \mid \sum_{j=1}^{\infty} \theta_j^2 < \infty \right\}$$

This feature map defines a kernel $k(x,z) = \langle \Phi(x), \Phi(z) \rangle_{l^2(N)} = \sum_{j=1}^{\infty} \mu_j \phi_j(x)\phi_j(z)$

A very broad class of **positive semidefinite** (**PSD**) kernel functions can be generated in this way.

$\forall\, x_1, x_2 \in X : k(x_1, x_2) = \langle \phi(x_1), \phi(x_2) \rangle$

$\phi(x)$ may not be unique

E.g., consider the polynomial kernel $k(u, v) = \langle u, v \rangle^d$

For $u, v \in R^2$ and $d = 2$, both

$$\phi(u_1, u_2) = (u_1^2, u_1 u_2, u_2 u_1, u_2^2)$$

and

$$\tilde{\phi}(u_1, u_2) = (u_1^2, \sqrt{2}\, u_1 u_2, u_2^2)$$

satisfy the equation $k(u, v) = \langle \phi(u), \phi(v) \rangle = \langle \tilde{\phi}(u), \tilde{\phi}(v) \rangle$

$$k(u, v) = \langle u, v \rangle^2 = (u_1 v_1 + u_2 v_2)^2 = u_1^2 v_1^2 + 2\, u_1 v_1 u_2 v_2 + u_2^2 v_2^2$$

Given a kernel neither the feature map nor the feature space are uniquely determined.

However, one can always construct a canonical feature space, namely, the **Reproducing Kernel Hilbert Space** (**RKHS**).

Any positive semidefinite kernel function $k : X \times X \rightarrow R$ can be used to construct a particular Hilbert space of functions on X.

This Hilbert space is unique, and has the following special property:

for any $x \in X$ the function $k(\cdot, x)$ belongs to H, and satisfies the relation

$$\langle f, k(\cdot, x) \rangle_H = f(x) \quad \forall f \in H$$

This property is known as the **kernel reproducing property** for the Hilbert space.

It allows us to think of the kernel itself as defining a feature map $x \rightarrow k(\cdot, x) \in H$

Inner products in the embedded space reduce to kernel evaluations, since the reproducing property ensures that

$$\langle k(\cdot, x), k(\cdot, z) \rangle_H = k(x, z) \quad \forall x, z \in X$$

Theorem

Given any positive semidefinite kernel function k, there is a unique Hilbert space H in which the kernel satisfies the reproducing property.

It is known as the **Reproducing Kernel Hilbert Space** (**RKHS**) associated with k.

How to construct a RKHS

To define a Hilbert space, we need

1) to form a vector space of functions

2) to endow it with an appropriate inner product

Consider functions of the form $f(\cdot)=\sum_j \alpha_j k(\cdot, x_j), \quad x_j \in X, \alpha \in R^n$

Define their inner product as $\langle f, \bar{f} \rangle = \sum_j \sum_{k=1} \alpha_j \bar{\alpha}_k k(x_j, \bar{x}_k)$

This proposed inner product does satisfy the kernel reproducing property, since

$$\langle f, k(\cdot, x) \rangle = \sum_j \alpha_j k(x_j, x) = f(x)$$

E.g. The space $L_2[0,1]$ is not an RKHS

$f \in L_2[0,1], \quad f:[0,1] \rightarrow R$ that is Lebesgue-integrable, and $\|f\|_{L^2[0.1]} = \int_0^1 f^2(x)dx < \infty$

Inner product: $\langle f, g \rangle_{L^2[0.1]} = \int_0^1 f(x)g(x)dx$

$L_2[0,1]$ is a Hilbert space isomorphic to $l^2(N) = \left\{ (\theta_j)_{j=1}^{\infty} \mid \sum_{j=1}^{\infty} \theta_j^2 < \infty \right\}$

But it is not an RKHS

$$\int_0^1 f(y)R_x(y)dy = f(x) \quad \forall f \in L^2[0,1]$$

$$\Rightarrow \quad R_x(y) = \delta(x-y) \notin L^2[0,1]$$

Kernel Ridge Regression (KRR)

$$f(x) = \sum_{i=1}^{n} \alpha_i k(x_i, x)$$

$$f(x_i) = \sum_{j=1}^{n} \alpha_j k(x_i, x_j) = K\alpha$$

$$\underset{\alpha \in R^n}{argmin} \sum_{i=1}^{n} \left(f(x_i) - y_i \right)^2 + \lambda \| f \|^2$$

$$\underset{\alpha \in R^n}{argmin} \| K\alpha - y \|^2 + \lambda \alpha^T K \alpha$$

$$\nabla_\alpha \left(\| K\alpha - y \|^2 + \lambda \alpha^T K \alpha \right) = 0$$

$$-Ky + K^2 \alpha + \lambda K \alpha = 0$$

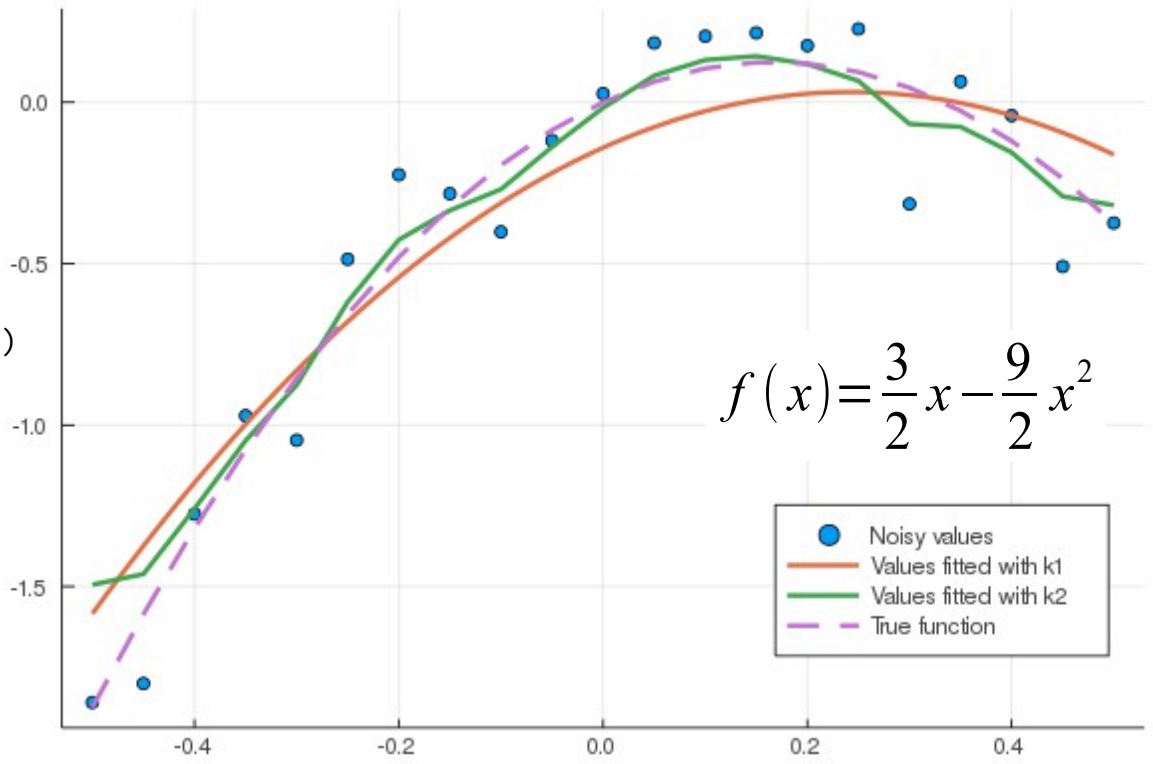
$$Ky = K(K + \lambda I)\alpha$$

The regression coefficients α are obtained by solving the linear system $(K + \lambda I)\alpha = y$

$$k_1(x,z) = (1 + xz)^2$$
$$k_2(x,z) = (1 + min(x,z))$$

```
k1(x,z) = (1.0 + x*z)^2.0;
k2(x,z) = 1.0 + min(x,z);
```

```
function k_matrix(k, x::Array{Float64,1})
    n = length(x)
    K = Array{Float64}(undef,(n,n))
    for i ∈ 1:n
        for j ∈ 1:n
            K[i,j] = k(x[i], x[j])
        end
    end
    return K
end;
```



$$f(x) = \frac{3}{2}x - \frac{9}{2}x^2$$

```
function krr_predict(k,
                     x_pred::Array{Float64,1},
                     x_train::Array{Float64,1},
                     α::Array{Float64,1})
    y = similar(x_pred)
    n_pred = length(y)
    n_train = length(x_train)
    for i ∈ 1:n_pred
        y[i] = 0.0
        for j ∈ 1:n_train
            y[i] += α[j] * k(x[j], x_pred[i])
        end
    end
    return y
end;
```

$$f(x) = \sum_{i=1}^{n} \alpha_i k(x_i, x)$$

$$(K + \lambda I)\alpha = y$$

```
K1 = k_matrix(k1, x);
K2 = k_matrix(k2, x);

α1 = (K1 + Matrix(λ*I, n, n) )\y;
α2 = (K2 + Matrix(λ*I, n, n) )\y;

yreg1 = krr_predict(k1, x_pred, x, α1);
yreg2 = krr_predict(k2, x_pred, x, α2);
```

$$H^1[0,1]=\left\{ f:[0,1]\to R \;\middle|\; f(0)=0,\ f \text{ is ab. cts},\ f' \in L^2[0,1] \right\}$$

– the first-order Sobolev space

f is absolutely continuous (or ab.cts. for short) if f' exists almost everywhere and is Lebesgue-integrable, and

$$f(x)=f(0)+\int_0^1 f'(z)\,dz \quad \forall\, x\in[0,1]$$

Inner product: $\langle f, g\rangle_{H^1[0,1]}=\int_0^1 f'(z)\,g'(z)\,dz$

$H^1[0,1]$ is an RKHS with $k(x,z)=min(x,z)$

$$\langle f, R_x\rangle_{H^1[0,1]}=\int_0^1 f'(z)\,R'_x(z)\,dz=\int_0^x f'(z)\,dz=f(x)$$

Theorem

Suppose that H_1 and H_2 are both RKHSs with kernels k_1 and k_2, respectively.

Then the space $H = H_1 + H_2$ with norm

$$\|f\|_H^2 = \min_{f = f_1 + f_2,\ f \in H_1,\ f_2 \in H_2} \{\|f_1\|_{H_1}^2 + \|f_2\|_{H_2}^2\}$$

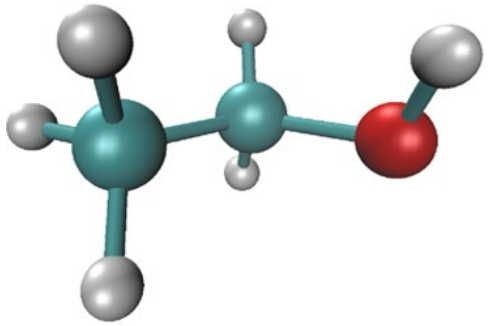
is an RKHS with kernel $k = k_1 + k_2$

$k_2(x, z) = (1 + min(x, z))$ is the kernel in $H = H_1 + H_2$ where

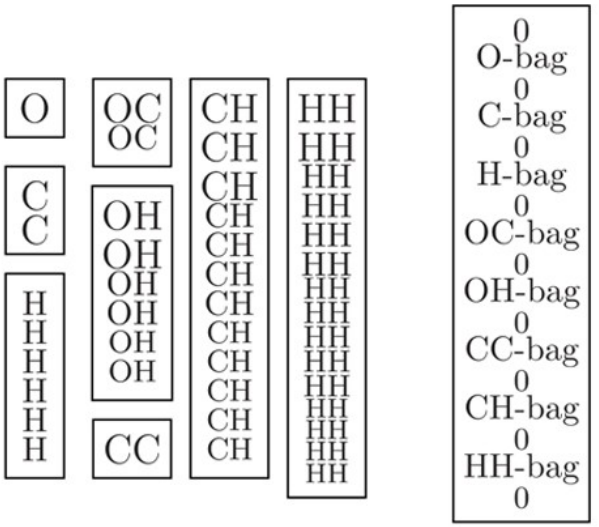
$H_1 = span\{1\}$ is the set of all constant functions

H_2 – the first-order Sobolev space

K. Hansen et al. Assessment and Validation of Machine Learning Methods for Predicting Molecular Atomization Energies.
dx.doi.org/10.1021/ct400195d | J. Chem. Theory Comput. 2013, 9, 3404−3419



	O	C	C	H	H	H	H	H	H
O	O	OC	OC	OH	OH	OH	OH	OH	OH
C	OC	C	CC	CH	CH	CH	CH	CH	CH
C	OC	CC	C	CH	CH	CH	CH	CH	CH
H	OH	CH	CH	H	HH	HH	HH	HH	HH
H	OH	CH	CH	HH	H	HH	HH	HH	HH
H	OH	CH	CH	HH	HH	H	HH	HH	HH
H	OH	CH	CH	HH	HH	HH	H	HH	HH
H	OH	CH	CH	HH	HH	HH	HH	H	HH
H	OH	CH	CH	HH	HH	HH	HH	HH	H



$$C_{ij} = \begin{cases} 0.5\, Z_i^{2.4} & \forall i = j \\ \frac{Z_i Z_j}{|\mathbf{R}_i - \mathbf{R}_j|} & \forall i \neq j. \end{cases}$$

Bag of Bonds descriptor

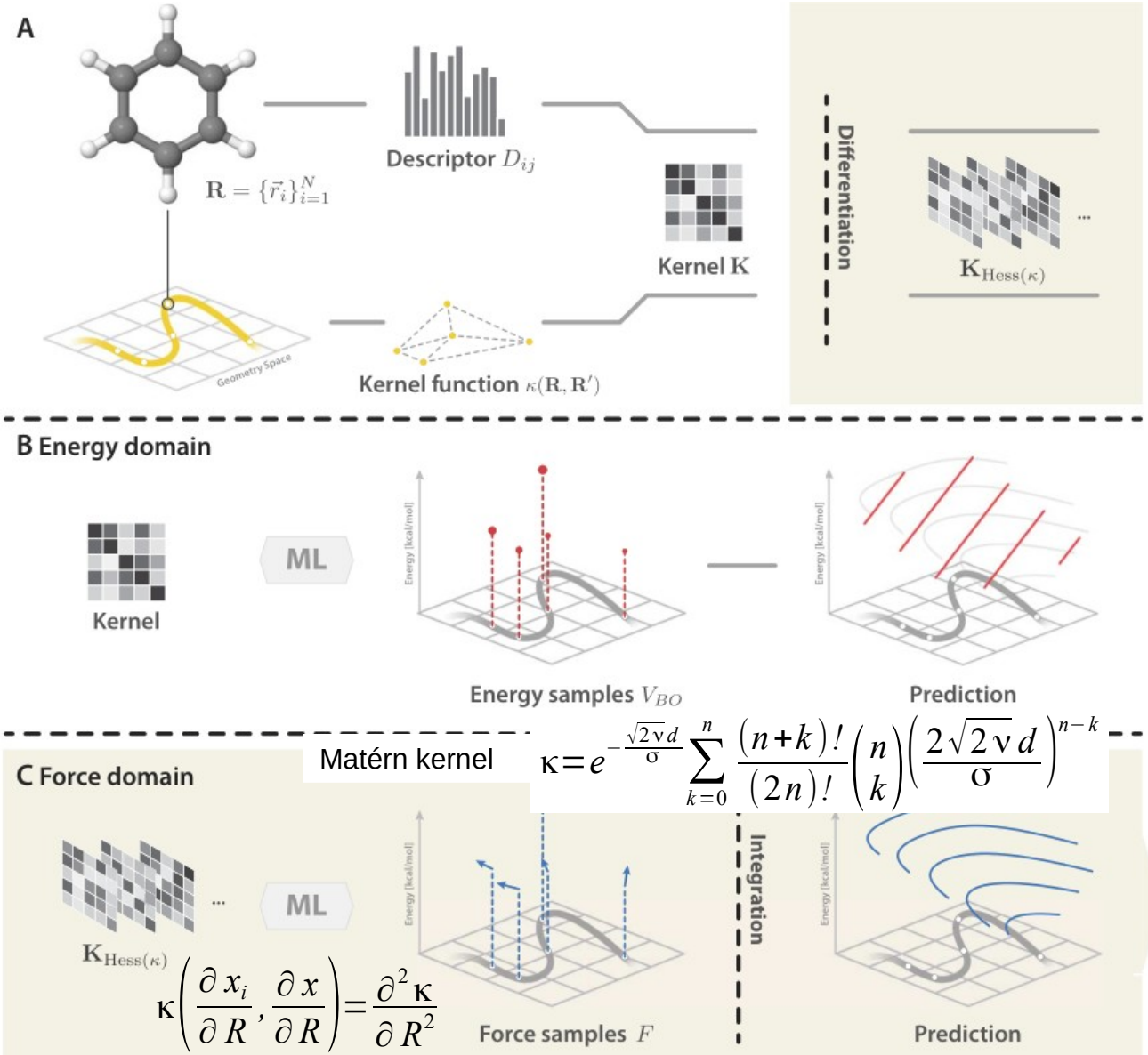
$$E_{BoB} = \sum_{i=1}^{N} \alpha_i \, e^{-d(M, M_i)/\sigma}$$

Table 1. Performance of Different Models Evaluated out-of-Sample in Five-Fold Cross-Validation on the GDB-7 Database[a]

model	MAE [kcal/mol]
dressed atoms	15.1
sum-overbonds	9.9
Lennard-Jones potential	8.7
polynomial pot. (n = 6)	5.6
polynomial pot. (n = 10)	3.9
polynomial pot. (n = 18)	3.0
Bag of Bonds (p = 2, Gaussian)	4.5
Bag of Bonds (p = 1, Laplacian)	1.5
Coulomb matrix (p = 2, Gaussian)[17]	10.0
Coulomb matrix (p = 1, Laplacian)[16]	4.3

Chmiela et al. Machine learning of accurate energy-conserving molecular force fields (2017) Science Advances, Vol. 3, no. 5, e1603015, DOI: 10.1126/sciadv.1603015

They constructed molecular force fields using a restricted number of samples from ab initio molecular dynamics (AIMD) trajectories.



Descriptor encodes molecular structure.

$$D_{ij} = \begin{cases} \|\vec{r}_i - \vec{r}_j\|^{-1} & \text{for } i > j \\ 0 & \text{for } i \leq j \end{cases}$$

Kernel function measures the similarity between pairs of inputs.

$$\kappa : \langle \phi(\mathbf{D}), \phi(\mathbf{D}') \rangle_{\mathcal{H}}$$

Problem:
Energy-based model lacks detail in under-sampled regions.

Solution: $\left(K_{Hess(\kappa)} + \lambda I\right)\vec{\alpha} = -\vec{F}$
Training in the force domain accurately reproduces PES topology.

$$\hat{f}_F(\vec{x}) = \sum_{i=1}^{M} \sum_{j=1}^{3N} (\vec{\alpha}_i)_j \frac{\partial}{\partial x_j} \nabla\kappa(\vec{x}, \vec{x}_i)$$

$$\hat{f}_E(\vec{x}) = \sum_{i=1}^{M} \sum_{j=1}^{3N} (\vec{\alpha}_i)_j \frac{\partial}{\partial x_j} \kappa(\vec{x}, \vec{x}_i)$$

More links

Charles A. Micchelli, Massimiliano Pontil;
On learning vector-valued functions. (2005)
Neural Comput. 17(1):177-204. doi: 10.1162/0899766052530802.

Michiel Hermans, Benjamin Schrauwen;
Recurrent kernel machines: computing with infinite echo state networks. (2012)
Neural Comput. 24(1):104-33. doi: 10.1162/NECO_a_00200.

James D. B. Nelson, Robert I. Damper, Steve R. Gunn, Baofeng Guo;
A signal theory approach to support vector classification: the sinc kernel. (2009)
Neural Netw. 22(1):49-57. doi: 10.1016/j.neunet.2008.09.016.

Vikas Sindhwani, Ha Quang Minh, Aurélie C. Lozano;
Scalable matrix-valued kernel learning for high-dimensional nonlinear multivariate regression and Granger Causality. (2013)
UAI'13: Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence, 586–595

Yoshua Bengio;
On the challenge of learning complex functions. (2007)
Prog Brain Res. 165:521-34. doi: 10.1016/S0079-6123(06)65033-4

Wei-Feng Zhang, Dao-Qing Dai, Hong Yan;
Framelet kernels with applications to support vector regression and regularization networks. (2010)
IEEE Trans Syst Man Cybern B Cybern. 40(4):1128-44. doi: 10.1109/TSMCB.2009.2034993

Shaohua Kevin Zhou, Rama Chellappa; From sample similarity to ensemble similarity: probabilistic distance measures in reproducing kernel Hilbert space. (2006) IEEE Trans. Pattern Anal. Mach. Intell. 28(6):917-29. doi: 10.1109/TPAMI.2006.120.