Lecture 10: Kernel Methods

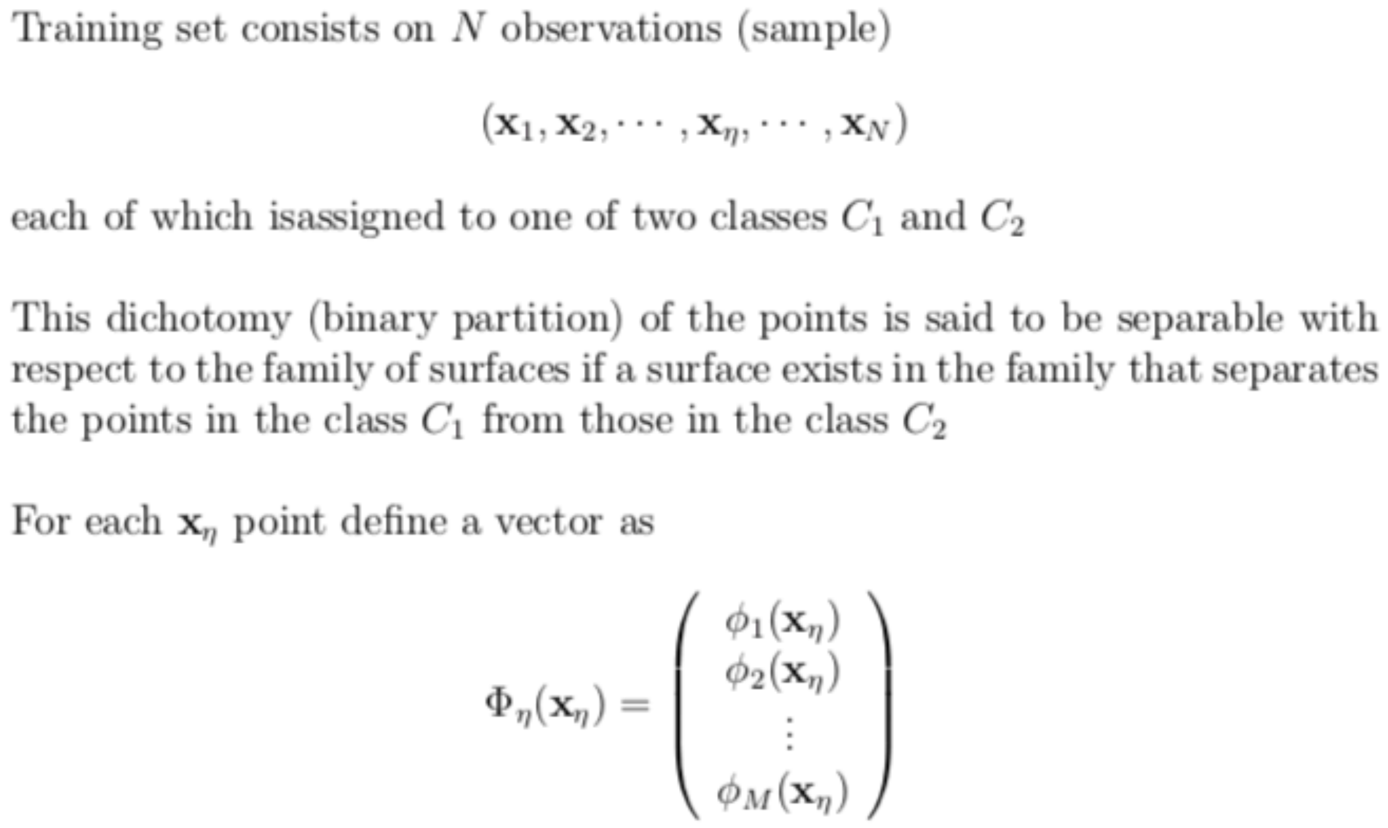
Andreas Wichert
Department of Computer Science and Engineering
Técnico Lisboa

Cover’s theorem

- A complex pattern-classification problem, cast in a high-dimensional space nonlinearly, is more likely to be linearly separable than in a low-dimensional space, provided that the space is not densely populated. (Cover, 1965)

- Once we have linearly separable patterns, the classification problem can be solved.

Cover's theorem



Training set consists on N observations (sample)

$$(\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_\eta, \cdots, \mathbf{x}_N)$$

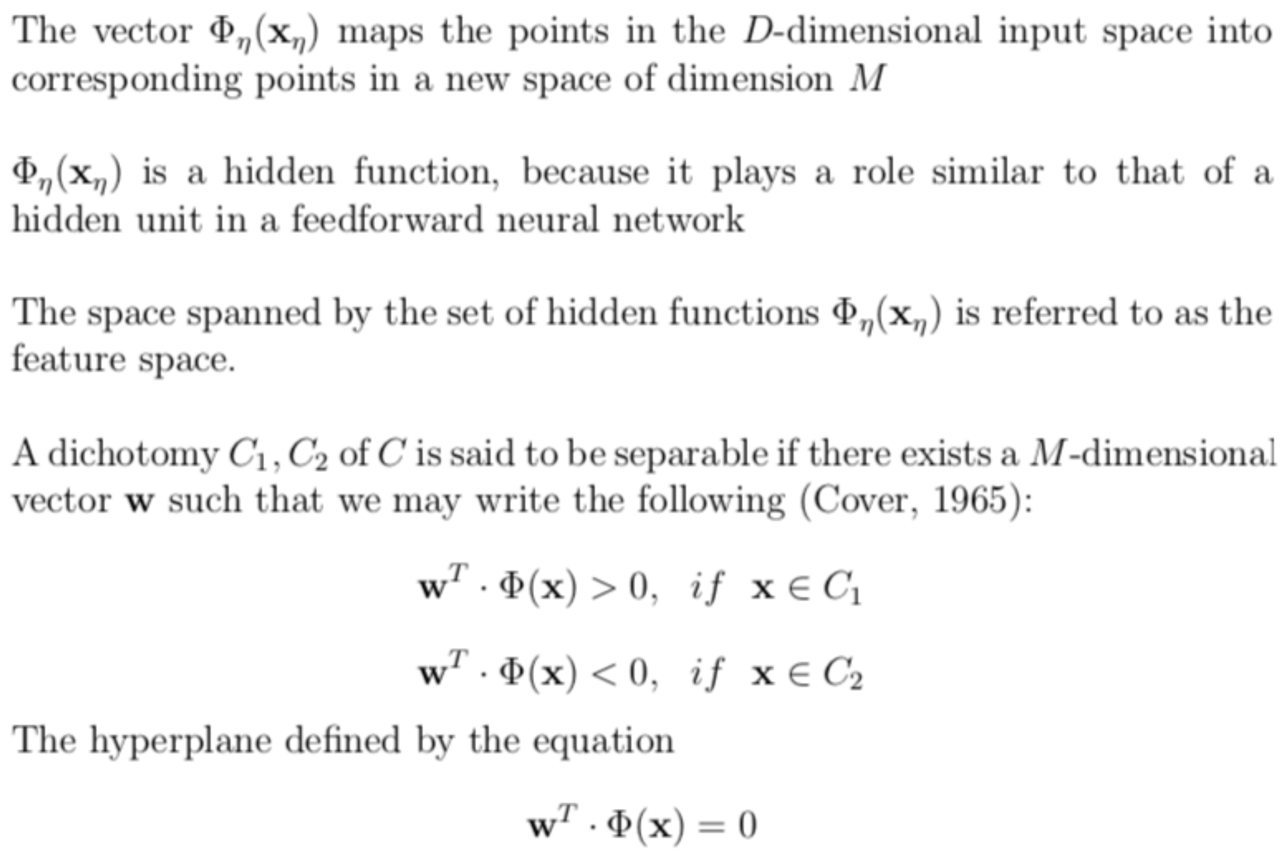
each of which isassigned to one of two classes C_1 and C_2

This dichotomy (binary partition) of the points is said to be separable with respect to the family of surfaces if a surface exists in the family that separates the points in the class C_1 from those in the class C_2

For each $\mathbf{x}_\eta$ point define a vector as

$$\Phi_\eta(\mathbf{x}_\eta) = \begin{pmatrix} \phi_1(\mathbf{x}_\eta) \\ \phi_2(\mathbf{x}_\eta) \\ \vdots \\ \phi_M(\mathbf{x}_\eta) \end{pmatrix}$$

Cover’s theorem



The vector $\Phi_\eta(\mathbf{x}_\eta)$ maps the points in the D-dimensional input space into corresponding points in a new space of dimension M

$\Phi_\eta(\mathbf{x}_\eta)$ is a hidden function, because it plays a role similar to that of a hidden unit in a feedforward neural network

The space spanned by the set of hidden functions $\Phi_\eta(\mathbf{x}_\eta)$ is referred to as the feature space.

A dichotomy C_1, C_2 of C is said to be separable if there exists a M-dimensional vector $\mathbf{w}$ such that we may write the following (Cover, 1965):

$$\mathbf{w}^T \cdot \Phi(\mathbf{x}) > 0, \quad if \;\; \mathbf{x} \in C_1$$

$$\mathbf{w}^T \cdot \Phi(\mathbf{x}) < 0, \quad if \;\; \mathbf{x} \in C_2$$

The hyperplane defined by the equation

$$\mathbf{w}^T \cdot \Phi(\mathbf{x}) = 0$$

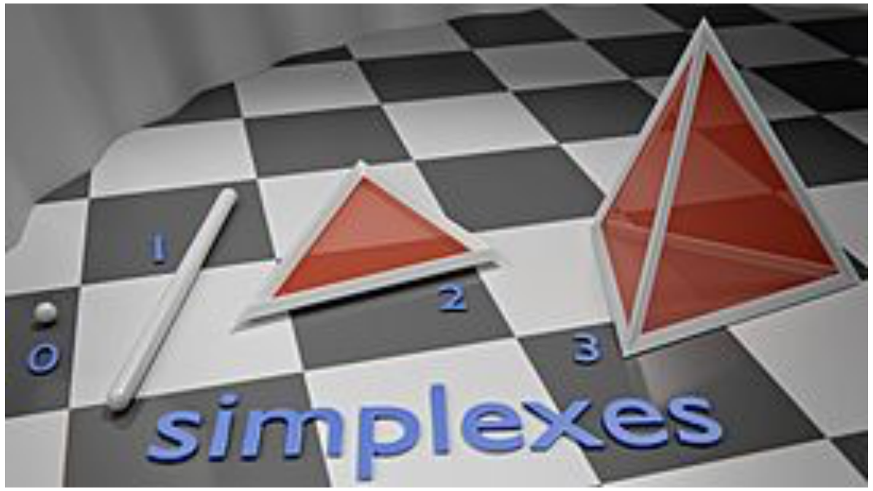
Cover’s theorem

- Cover’s theorem on the separability (1965)
- Given a set of training data that is not linearly separable, one can with high probability transform it into a training set that is linearly separable by projecting it into a **higher**-dimensional space via some non-linear transformation $\Phi_\eta(\mathbf{x}_\eta)$

- Lift N samples onto the vertices of the simplex in the $N - 1$ dimensional real space. Every partition of the samples into two sets is separable by a linear separator. (VC-dimension of a Perceptron)

Simplex

- A simplex is a generalization of the notion of a triangle or tetrahedron to arbitrary dimensions.
- A k-simplex is a k-dimensional polytope which is the convex hull of its k + 1 vertices.



XOR Problem

A pair of Gaussian hidden functions is defined as:

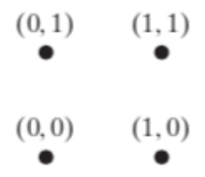
$$\phi_1(\mathbf{x}) = \exp(-\|\mathbf{x} - \mathbf{t}_1\|^2), \quad \mathbf{t}_1 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

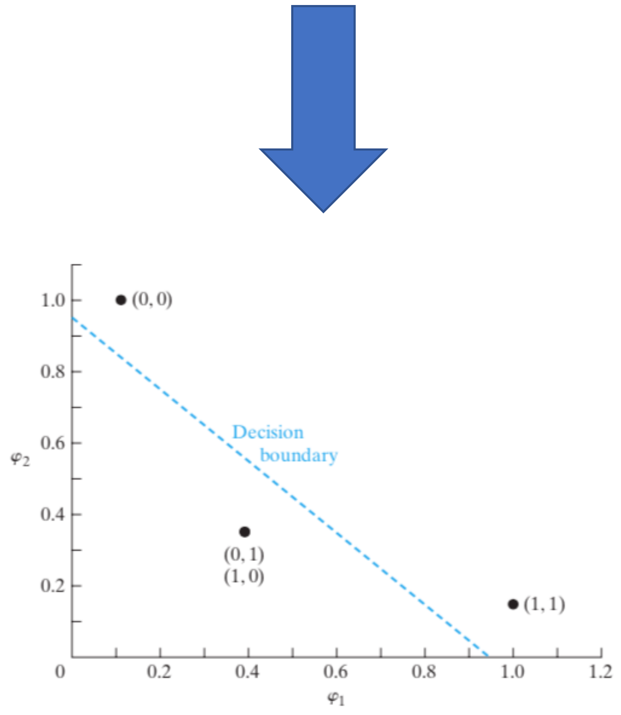
$$\phi_2(\mathbf{x}) = \exp(-\|\mathbf{x} - \mathbf{t}_2\|^2), \quad \mathbf{t}_2 = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

$$\Phi_\eta(\mathbf{x}_\eta) = \begin{pmatrix} \phi_1(\mathbf{x}_\eta) \\ \phi_2(\mathbf{x}_\eta) \end{pmatrix}$$

There is no increase in the dimensionality of the hidden space compared with the input space.

Nonlinearity exemplified by the use of Gaussian hidden functions is sufficient to transform the XOR problem into a linearly separable one.



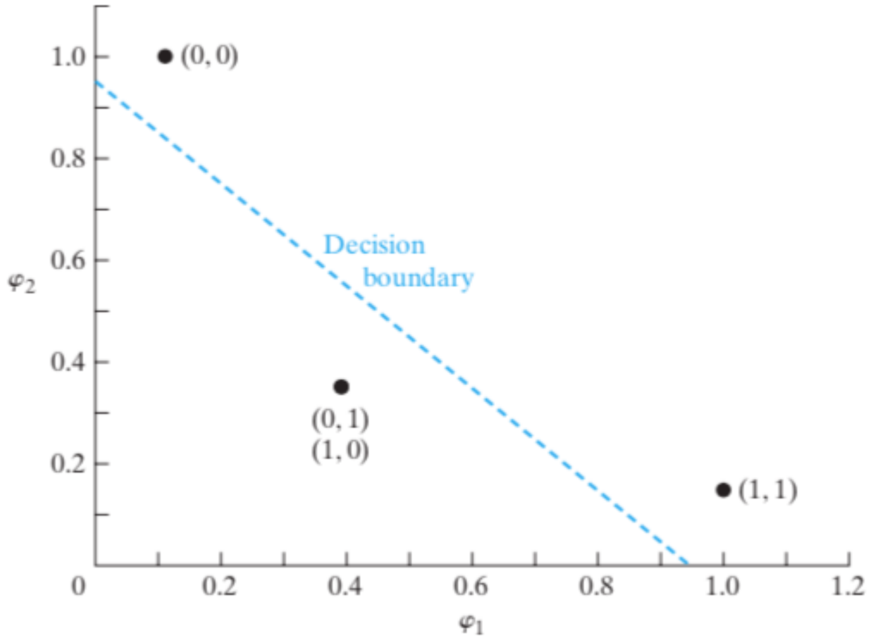


XOR Problem

$$\phi_1(\mathbf{x}) = \exp(-\|\mathbf{x} - \mathbf{t}_1\|^2), \quad \mathbf{t}_1 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

$$\phi_2(\mathbf{x}) = \exp(-\|\mathbf{x} - \mathbf{t}_2\|^2), \quad \mathbf{t}_2 = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

$$\Phi_\eta(\mathbf{x}_\eta) = \begin{pmatrix} \phi_1(\mathbf{x}_\eta) \\ \phi_2(\mathbf{x}_\eta) \end{pmatrix}$$



Input Pattern $\mathbf{x}$	First Hidden Function $\varphi_1(\mathbf{x})$	Second Hidden Function $\varphi_2(\mathbf{x})$
(1,1)	1	0.1353
(0,1)	0.3678	0.3678
(0,0)	0.1353	1
(1,0)	0.3678	0.3678

Interpolation Problem

$$s : \mathbb{R}^D \to \mathbb{R}^1$$

The maps s defines a $hypersurface$ (graph) $\Gamma \subset \mathbb{R}^{D+1}$ in the same the mapping $f(x) = x^2$ draws a parabola in $\mathbb{R}^2$

The interpolation problem:

Given a set of N different points $\mathbf{x}_\eta \in \mathbb{R}^D$ and a set of real numbers $d_\eta \in \mathbb{R}^1$ with $\eta = 1, 2, \cdots, N$, find a function $F : \mathbb{R}^N \to \mathbb{R}^1$ that satisfies the interpolation condition:

$$F(\mathbf{x}_\eta) = d_\eta, \quad \eta = 1, 2, \cdots, N$$

The interpolating surface is constrained to pass through all the training data points

Interpolation Problem

- The radial-basis-functions (RBF) technique consists of choosing a function *F* that has the form

$$F(\mathbf{x}) = \sum_{\eta=1}^{N} w_\eta \cdot \phi(\|\mathbf{x} - \mathbf{x}_\eta\|)$$

with $\phi(\|\mathbf{x} - \mathbf{x}_\eta\|)$ being a set of N arbitrary (generally nonlinear) functions called radial-basis functions.

The N known data points $\mathbf{x}_\eta$ are taken to be the centers of the radial-basis functions.

Interpolation Problem

The N known data points $\mathbf{x}_\eta$ are taken to be the centers of the radial-basis functions.

We get a set of linear equations

$$\begin{pmatrix} \phi_{11} & \phi_{12} & \phi_{13} & \cdots & \phi_{1N} \\ \phi_{21} & \phi_{22} & \phi_{23} & \cdots & \phi_{2N} \\ \vdots & \vdots & \ddots & \vdots & \\ \phi_{N1} & \phi_{N2} & \phi_{N3} & \cdots & \phi_{NN} \end{pmatrix} \cdot \begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_N \end{pmatrix} = \begin{pmatrix} d_1 \\ d_2 \\ \vdots \\ d_N \end{pmatrix}$$

with

$$\phi_{ij} = \phi(\|\mathbf{x}_i - \mathbf{x}_j\|)$$

We can simplify with

$$\mathbf{w} = \begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_N \end{pmatrix}, \quad \mathbf{d} = \begin{pmatrix} d_1 \\ d_2 \\ \vdots \\ d_N \end{pmatrix}$$

with the interpolation matrix

$$\Phi = \{\phi_{ij}\}_{i,j}^{N}$$

The compact notation is given by

$$\Phi \cdot \mathbf{w} = \mathbf{d}$$

Assuming that Φ is nonsingular

$$\mathbf{w} = \Phi^{-1} \cdot \mathbf{d}$$

How can we be sure that the interpolation matrix Φ is nonsingular?

Micchelli’s Theorem

Let set of N different points $\mathbf{x}_\eta \in \mathbb{R}^D$. Then the $N \times N$ interpolation matrix Φ, whose ij-th element is $\phi_{ij} = \phi(\|\mathbf{x}_i - \mathbf{x}_j\|)$ is nonsingular.

Functions that are covered by Micchelli’s Theorem with $r = \|\mathbf{x}_i - \mathbf{x}_j\|$

1. Multiquadratics:

$$\phi(r) = \sqrt{r^2 + c^2}, \qquad c > 0$$

2. Inverse Multiquadratics:

$$\phi(r) = \frac{1}{\sqrt{r^2 + c^2}}, \qquad c > 0$$

3. Gaussian functions:

$$\phi(r) = \exp\left(-\frac{r^2}{2 \cdot \sigma^2}\right), \qquad \sigma > 0$$

Radial Basis Function Networks

1. Input Layer: Dimension D of the input vector $\mathbf{x}$

2. Hidden Layer: Same number N of units as the size of the training sample

$$\phi_\eta = \phi(\|\mathbf{x} - \mathbf{x}_\eta\|), \quad \eta = 1, 2, \cdots, N$$

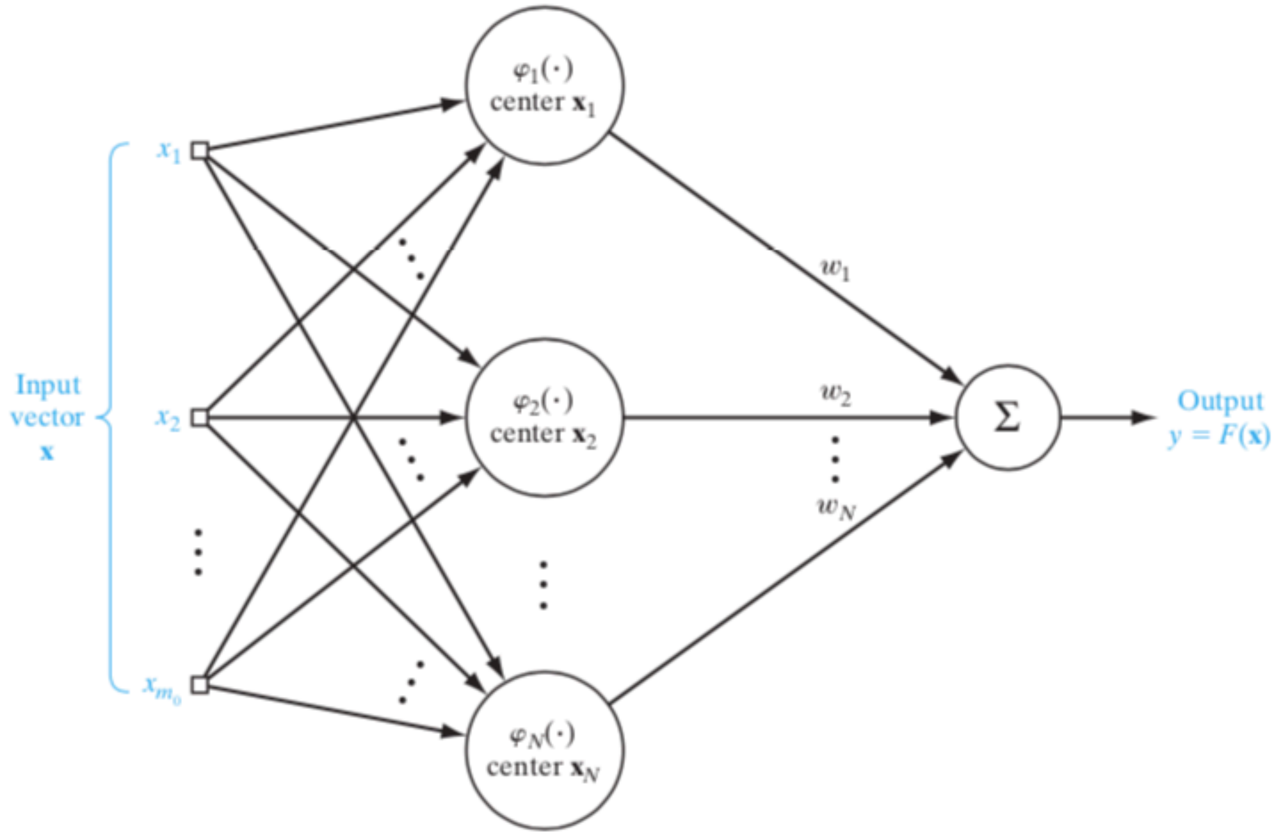
   $\mathbf{x}_\eta$ defines the centre of the radial-basis function.

3. Output Layer: Consists of single computational unit. However there is no constraint on the size.

Usually we use

$$\phi_\eta = \phi(\|\mathbf{x} - \mathbf{x}_\eta\|) = \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}_\eta\|^2}{2 \cdot \sigma^2}\right), \quad \eta = 1, 2, \cdots, N$$

Radial Basis Function Networks


Input vector $\mathbf{x}$
x_1
x_2
x_{m_0}
$\varphi_1(\cdot)$ center $\mathbf{x}_1$
$\varphi_2(\cdot)$ center $\mathbf{x}_2$
$\varphi_N(\cdot)$ center $\mathbf{x}_N$
w_1
w_2
w_N
Σ
Output $y = F(\mathbf{x})$

Modifications of Radial Basis Function Networks

Hidden Layer: Same number N of units as the size of the training sample

Reduce the number of hidden units to $K < N$ using K means clustering on

$$F(\mathbf{x}) = \sum_{k=1}^{K} w_k \cdot \phi(\|\mathbf{x} - \mathbf{c}_k\|)$$

Training set consists on N observations (sample)

$$X = (\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_\eta, \cdots, \mathbf{x}_N)$$

each cluster set is defined as the set of points with where

$$k = 1, ..., K$$

$$C_k - \{\mathbf{x} | d_2(\mathbf{x}, \mathbf{c}_k) - \min_j d_2(\mathbf{x}, \mathbf{c}_j)\}.$$

Each cluster C_k contains the points that are closest to the centroid $\mathbf{c}_k$. The centroid $\mathbf{c}_k$ is represented by the mean value of all the points of C_k

$$\mathbf{c}_k = \frac{1}{|C_k|} \cdot \sum_{x \in C_k} \mathbf{x}.$$

and with the same width

$$\sigma = \frac{d_{max}}{\sqrt{2 \cdot K}}$$

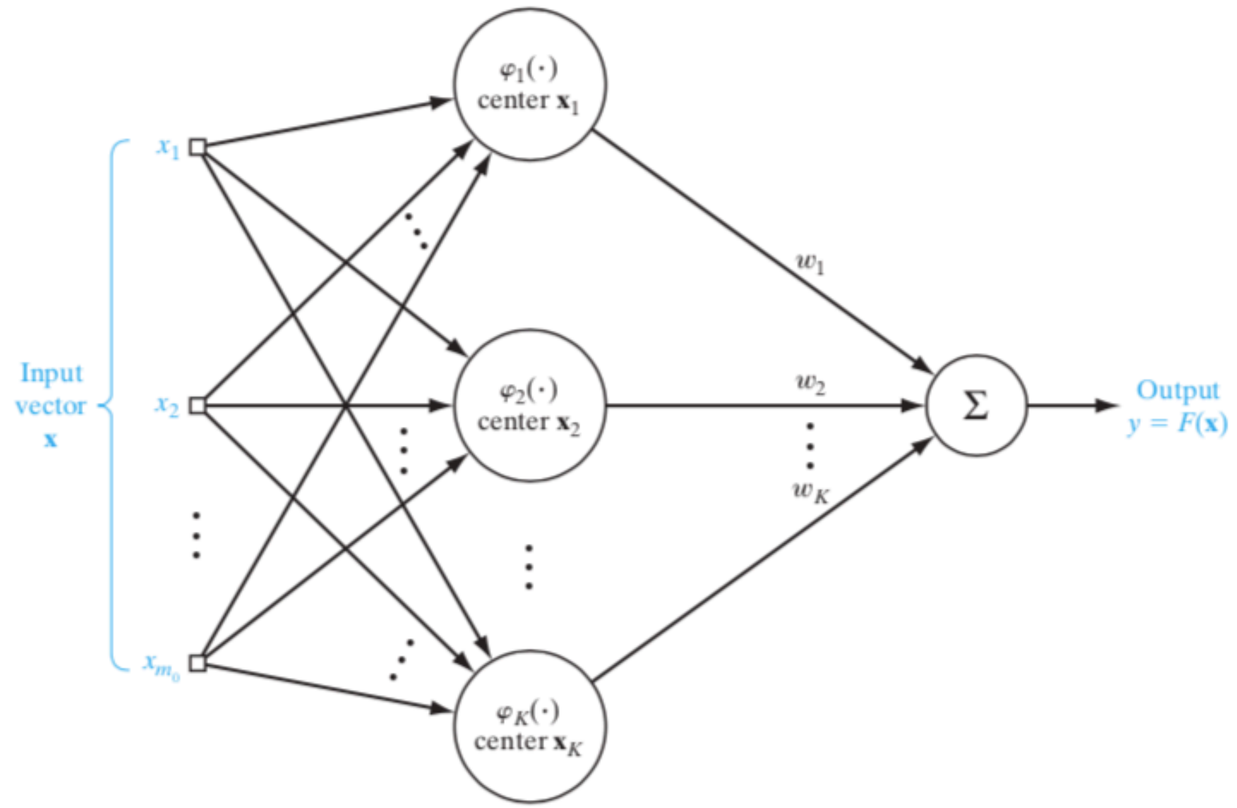
where K is the number of centres and d_{max} is the maximum distance between them (Lowe, 1989).

Ensures that the individual Gaussian units are not too peaked or too flat; both extreme conditions should be avoided

$$\phi_k = \phi(\|\mathbf{x} - \mathbf{c}_k\|) = \exp\left(-\frac{\|\mathbf{x} - \mathbf{c}_k\|^2}{2 \cdot \sigma^2}\right), \quad k = 1, 2, \cdots, K$$

$$\phi_k = \phi(\|\mathbf{x} - \mathbf{c}_k\|) = \exp\left(-\frac{\|\mathbf{x} - \mathbf{c}_k\|^2 \cdot K}{d_{max}^2}\right), \quad k = 1, 2, \cdots, K$$

Radial Basis Function Networks with K-Means



Input vector $\mathbf{x}$
x_1
x_2
x_{m_0}
$\varphi_1(\cdot)$ center $\mathbf{x}_1$
$\varphi_2(\cdot)$ center $\mathbf{x}_2$
$\varphi_K(\cdot)$ center $\mathbf{x}_K$
w_1
w_2
w_K
Σ
Output $y = F(\mathbf{x})$

EM-Clustering

Reduce the number of hidden units to $K < N$ using EM clustering on

$$F(\mathbf{x}) = \sum_{k=1}^{K} w_k \cdot \phi(\|\mathbf{x} - \boldsymbol{\mu}_k\|)$$

with

$$\phi_k = \phi(\|\mathbf{x}-\boldsymbol{\mu}_k\|) = \exp\left(-\frac{1}{2} \cdot (\mathbf{x}_\eta - \boldsymbol{\mu}_k)^T \Sigma_k^{-1} \cdot (\mathbf{x}_\eta - \boldsymbol{\mu}_k)\right), \quad k = 1, 2, \cdots, K$$

Em_Clustering

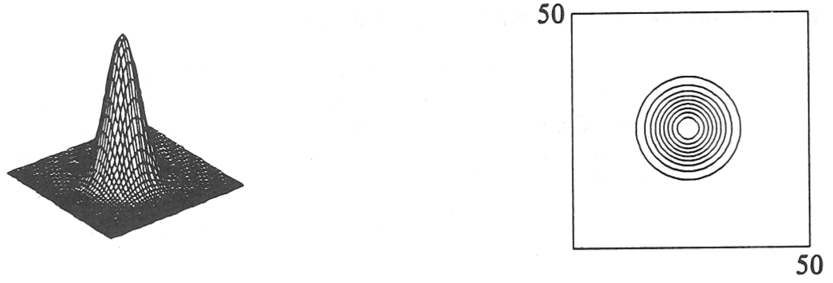


Figure 8-8(a). Radially symmetric basis function.

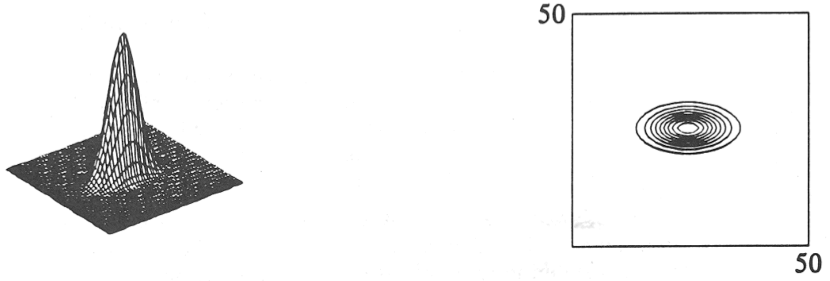


Figure 8-8(b). Diagonal inverse covariance matrix—equal components.

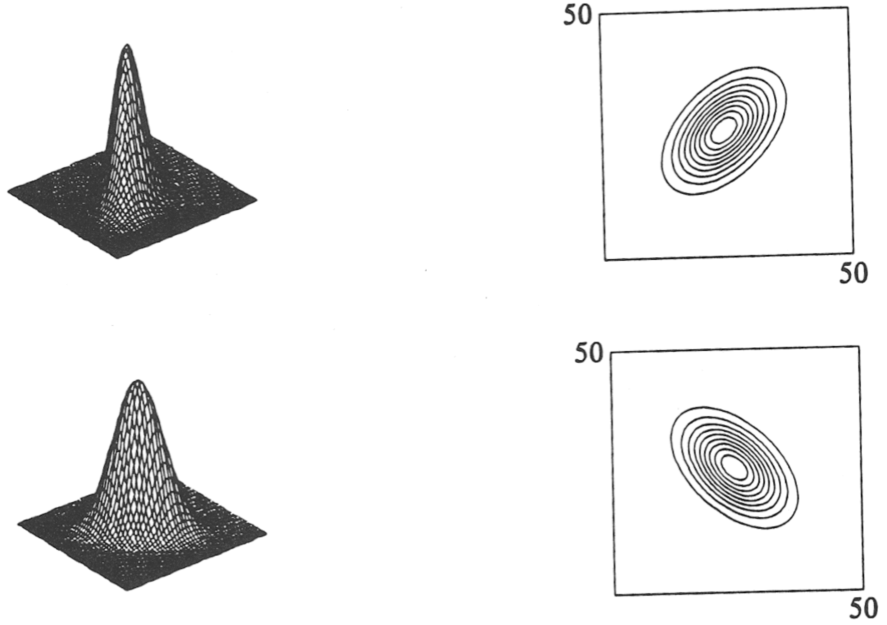


Figure 8-8(c) and (d). Nondiagonal inverse covariance matrices.

Output Layer

Use Perceptron

Use Logistic Regression with

$$\boldsymbol{\phi} = \begin{pmatrix} \phi_1 \\ \phi_2 \\ \vdots \\ \phi_K \end{pmatrix},$$

$$p(C_1|\boldsymbol{\phi}) = \sigma(net) = \frac{1}{1 + e^{(-net)}} = \frac{e^{(net)}}{1 + e^{(net)}}$$

with $\phi_0 = 1$ bias,

$$p(C_1|\boldsymbol{\phi}) = \sigma\left(\sum_{k=0}^{K} w_k \cdot \phi_k\right) = \sigma\left(\mathbf{w}^T \cdot \boldsymbol{\phi}\right)$$

$$p(C_1|\boldsymbol{\phi}) = \sigma\left(\sum_{k=0}^{K} w_k \cdot \phi_k\right) = \sigma\left(\mathbf{w}^T \cdot \boldsymbol{\phi}\right)$$

Error function is defined by negative logarithm of the likelihood which leads to the update rule where the target t_k can be only one or zero (a constraint)

The update rule for gradient decent is given for target $t_\eta \in \{0, 1\}$

$$\Delta w_k = \eta \cdot \sum_{\eta - 1}^{N} (t_\eta - o_\eta) \cdot \phi_{\eta,k}.$$

With L output units use softmax function

$$\sigma(net_s) = \frac{\exp(net_s)}{\sum_{t=1}^{L} \exp(net_t)}$$

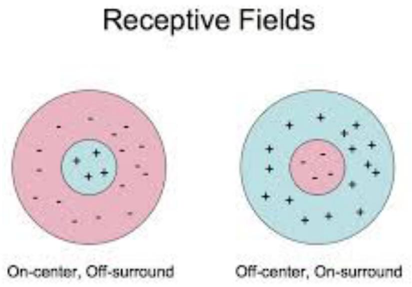
The update rule for gradient decent is given for target $t_{\eta s} \in \{0,1\}$ with $s = 1, 2, \cdots .L$

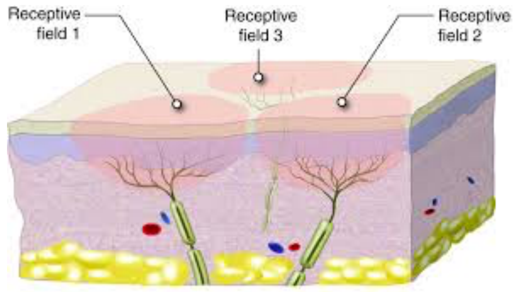
$$\mathbf{t} = \begin{pmatrix} t_1 \\ t_2 \\ \vdots \\ t_L \end{pmatrix}$$

$$\Delta w_{ks} = \eta \cdot \sum_{\eta=1}^{N} (t_{\eta s} - o_{\eta s}) \cdot \phi_{\eta,k}.$$

Usually the training process performed on the RBF network was significantly faster than that performed on the MLP.

Interpretation of Hidden Units





Neurobiology

In a neurobiological context, a receptive field is defined as that region of a sensory field from which an adequate sensory stimulus will elicit a response (Churchland and Sejnowski, 1992)

Relation of the receptive field in ϕ_k, the size is described by σ or Σ in ϕ_k and its centre by $\mathbf{c}_k$ or $\boldsymbol{\mu}_k$.

Interpretation of Hidden Units

Statistics

$$\phi(\mathbf{x}, \mathbf{x}_k) = \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}_k\|^2}{2 \cdot \sigma^2}\right), \quad k = 1, 2, \cdots, K$$

is called Gaussian function.

Formulation of a function as a kernel, denoted by $k(\mathbf{x})$, is that the function has properties similar to those associated with the probability density function of a random variable:

Property 1. The kernel $k(\mathbf{x})$ is a continuous, bounded, and real function of $\mathbf{x}$ and symmetric about the origin, where it attains its maximum value.

Interpretation of Hidden Units Statistics

Property 2. The total volume under the surface of the kernel $k(\mathbf{x})$ is unity; that is, for an D-dimensional vector $\mathbf{x}$, we have

$$\int k(\mathbf{x})d\mathbf{x} = 1$$

Except for a scaling factor, the Gaussian function $\phi(\mathbf{x}, \mathbf{x}_k)$ satisfies both of these properties for the centre , $\mathbf{x}_k$ located at the origin

Because of the interpretation of the Gaussian function as a kernel the term kernel methods is used

Kernel Regression

Consider a nonlinear regression model

$$y_\eta = f(\mathbf{x}_\eta) + \epsilon_\eta, \quad \eta = 1, 2, \cdots, N$$

As a reasonable estimate of the unknown regression function $f(\mathbf{x})$ is the mean of observables, values of the model output y) near a point $\mathbf{x}$.

The local average should be confined to observations in a small neighborhood (receptive field) around the point $\mathbf{x}$.

The unknown function $f(\mathbf{x})$ is equal to the conditional mean of the observable y given the regressor $\mathbf{x}$.

$$f(\mathbf{x}) = \mathbb{E}(y|\mathbf{x}) = \int_{-\infty}^{\infty} y \cdot p_{y|\mathbf{x}}(y|\mathbf{x}) dy$$

with $p_{Y|\mathbf{X}}(y|\mathbf{x})$ conditional probability density function (pdf) of the random variable Y given that the random vector $\mathbf{X}$ is assigned the value $\mathbf{x}$

Cumulative distribution function that maps a variable y into a probability density function $p_Y(y) \in [0,1]$ like for example sigmoid function $\sigma(y)$

Cumulative distribution function that maps a m dimensional vector $\mathbf{y}$ into a probability density function $p_Y(\mathbf{y}) \in [0,1]$ like for example Gaussian over m dimensional space.

$$p_{Y|\mathbf{X}}(y|\mathbf{x}) = \frac{p_{Y,\mathbf{X}}(y,\mathbf{x})}{p_{\mathbf{X}}(\mathbf{x})}$$

$$f(\mathbf{x}) = \frac{\int_{-\infty}^{\infty} y \cdot p_{Y,\mathbf{X}}(y,\mathbf{x}) dy}{p_{\mathbf{X}}(\mathbf{x})}$$

We may use a nonparametric estimator known as the Parzen-Rosenblatt density estimator (Rosenblatt, 1956, 1970; Parzen, 1962) with a kernel $k(\mathbf{x})$

Parzen-Rosenblatt density estimate of $f_{\mathbf{X}}(\mathbf{x})$ as

$$\hat{p}_{\mathbf{X}}(\mathbf{x}) = \frac{1}{N \cdot h^D} \cdot \sum_{\eta=1}^{N} k\left(\frac{\mathbf{x} - \mathbf{x}_\eta}{h}\right)$$

h is a positive number called bandwidth, it h controls the size of the kernel

$$\hat{p}_{\mathbf{X},Y}(\mathbf{x}, y) = \frac{1}{N \cdot h^{D+1}} \cdot \sum_{\eta=1}^{N} k\left(\frac{\mathbf{x} - \mathbf{x}_\eta}{h}\right) \cdot k\left(\frac{y - y_\eta}{h}\right)$$

$$\int_{-\infty}^{\infty} y \cdot \hat{p}_{\mathbf{X},Y}(\mathbf{x}, y) dy = \frac{1}{N \cdot h^D} \cdot \sum_{\eta=1}^{N} k\left(\frac{\mathbf{x} - \mathbf{x}_\eta}{h}\right) \int_{-\infty}^{\infty} y \cdot k\left(\frac{y - y_\eta}{h}\right) dy$$

By integrating and changing the variable of integration we get (See Haykin, 2008)

$$\int_{-\infty}^{\infty} y \cdot \hat{p}_{\mathbf{X},Y}(\mathbf{x}, y) dy = \frac{1}{N \cdot h^D} \cdot \sum_{\eta=1}^{N} y_\eta \cdot k\left(\frac{\mathbf{x} - \mathbf{x}_\eta}{h}\right)$$

After canceling common terms we get

$$F(\mathbf{x}) = \hat{f}(\mathbf{x}) = \frac{\sum_{\eta=1}^{N} y_\eta \cdot k\left(\frac{\mathbf{x} - \mathbf{x}_\eta}{h}\right)}{\sum_{\eta=1}^{N} k\left(\frac{\mathbf{x} - \mathbf{x}_\eta}{h}\right)}$$

There are two ways in which the approximating function $F(\mathbf{x})$ may be viewed, either as Nadaraya-Watson regression estimator or Normalised RBF network.

Nadaraya-Watson regression estimator

$$W_{N,\eta}(\mathbf{x}) = \frac{k\left(\frac{\mathbf{x}-\mathbf{x}_\eta}{h}\right)}{\sum_{\eta=1}^{N} k\left(\frac{\mathbf{x}-\mathbf{x}_\eta}{h}\right)}$$

with

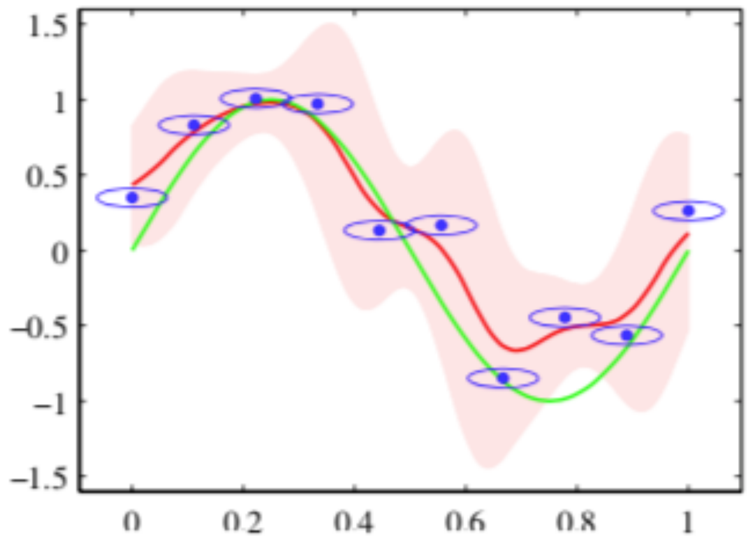
$$\sum_{\eta=1}^{N} W_{N,\eta}(\mathbf{x}) = 1$$

$$F(\mathbf{x}) = \sum_{\eta=1}^{N} W_{N,\eta}(\mathbf{x}) \cdot y_\eta$$

$F(\mathbf{x})$ = is a weighted average of the observable y_η. Proposed by Nadaraya (1964) and Watson (1964)

Nadaraya-Watson regression estimator

- The Nadaraya-Watson kernel regression model using isotropic Gaussian kernels, for the sinusoidal data set.
- The original sine function is shown by the green curve, the data points are shown in blue, and each is the centre of an isotropic Gaussian kernel
- The blue ellipse around each data point shows one standard deviation contour for the corresponding kernel. These appear noncircular due to the different scales on the horizontal and vertical axes.



Normalised RBF network

We set the kernel $k(\mathbf{x})$

$$k\left(\frac{\mathbf{x}-\mathbf{x}_\eta}{h}\right) = k\left(\frac{\|\mathbf{x}-\mathbf{x}_\eta\|}{h}\right)$$

and define the Normalised RBF

$$\psi_N(\mathbf{x},\mathbf{x}_\eta) = \frac{k\left(\frac{\|\mathbf{x}-\mathbf{x}_\eta\|}{h}\right)}{\sum_{\eta=1}^{N} k\left(\frac{\|\mathbf{x}-\mathbf{x}_\eta\|}{h}\right)}$$

with

$$\sum_{\eta=1}^{N} \psi_N(\mathbf{x},\mathbf{x}_\eta) = 1$$

The linear weights w_η applied to the basic functions $\psi_N(\mathbf{x},\mathbf{x}_\eta)$ are the observables, y_η of the regression model for the input data , $\mathbf{x}_\eta$

The linear weights w_η applied to the basic functions $\psi_N(\mathbf{x}, \mathbf{x}_\eta)$ are the observables, y_η of the regression model for the input data , $\mathbf{x}_\eta$

$$y_\eta = w_\eta$$

$$F(\mathbf{x}) = \sum_{\eta=1}^{N} w_\eta \cdot \psi_N(\mathbf{x}, \mathbf{x}_\eta)$$

$\psi_N(\mathbf{x}, \mathbf{x}_\eta)$ may be interpreted as the probability of an event described by the input vector $\mathbf{x}$, conditional on $\mathbf{x}_\eta$

The difference between the normalised radial-basis function and an ordinary radial-basis function is a denominator term that constitutes the normalisation factor.

Constructing Kernels

The kernel function is defined by

$$k(x, x') = \phi(x)^T \cdot \phi(x') = \sum_{i=1}^{M} \phi_i(x) \cdot \phi_i(x')$$

with basis function $\phi_i(x)$

Consider a kernel function n a two dimensional space given by

$$k(\mathbf{x},\mathbf{z}) = (\mathbf{x}^T \cdot \mathbf{z})^2 = (x_1 \cdot z_1 + x_2 \cdot z_2)^2 = x_1^2 \cdot z_1^2 + 2 \cdot x_1 \cdot x_2 \cdot z_1 \cdot z_2 + x_2^2 \cdot z_2^2$$

$$k(\mathbf{x},\mathbf{z}) = (\mathbf{x}^T \cdot \mathbf{z})^2 = (x_1^2, \sqrt{2} \cdot x_1 \cdot x_2, x_2^2) \cdot (z_1^2, \sqrt{2} \cdot z_1 \cdot z_2, z_2^2)^T$$

$$k(\mathbf{x},\mathbf{z}) = (\mathbf{x}^T \cdot \mathbf{z})^2 = \phi(\mathbf{x})^T \cdot \phi(\mathbf{z})$$

with

$$\phi(\mathbf{x}) = (x_1^2, \sqrt{2} \cdot x_1 \cdot x_2, x_2^2)^T$$

A simple way to test whether a function constitutes a valid kernel without having to construct the function $\phi(\mathbf{x})$ explicitly.

A necessary and sufficient condition for a function $k(\mathbf{x}, \mathbf{x'})$ to be a valid kernel (Shawe- Taylor and Cristianini, 2004) is that the Gram matrix K, whose elements are given by $k(\mathbf{x}_n, \mathbf{x}_m)$, should be positive semidefinite for all possible choices of the set $\{\mathbf{x}_n\}$

$$\mathbf{x}^T \cdot K \cdot \mathbf{x} > 0$$

for every non-zero column vector $\mathbf{x}$

Constructing new kernels by building them out of simpler kernels as building blocks

- *f(·)* is any function, *q(·)* is a polynomial with nonnegative coefficients,
- *A* is a symmetric positive semidefinite matrix, x_a and x_b are variables (not necessarily disjoint) with $x = (x_a, x_b)$, and k_a and k_b are valid kernel functions over their respective spaces.

$$
\begin{aligned}
k(\mathbf{x},\mathbf{x}') &= ck_1(\mathbf{x},\mathbf{x}') \\
k(\mathbf{x},\mathbf{x}') &= f(\mathbf{x})k_1(\mathbf{x},\mathbf{x}')f(\mathbf{x}') \\
k(\mathbf{x},\mathbf{x}') &= q\left(k_1(\mathbf{x},\mathbf{x}')\right) \\
k(\mathbf{x},\mathbf{x}') &= \exp\left(k_1(\mathbf{x},\mathbf{x}')\right) \\
k(\mathbf{x},\mathbf{x}') &= k_1(\mathbf{x},\mathbf{x}') + k_2(\mathbf{x},\mathbf{x}') \\
k(\mathbf{x},\mathbf{x}') &= k_1(\mathbf{x},\mathbf{x}')k_2(\mathbf{x},\mathbf{x}') \\
k(\mathbf{x},\mathbf{x}') &= k_3\left(\boldsymbol{\phi}(\mathbf{x}),\boldsymbol{\phi}(\mathbf{x}')\right) \\
k(\mathbf{x},\mathbf{x}') &= \mathbf{x}^{\mathrm{T}}\mathbf{A}\mathbf{x}' \\
k(\mathbf{x},\mathbf{x}') &= k_a(\mathbf{x}_a,\mathbf{x}_a') + k_b(\mathbf{x}_b,\mathbf{x}_b') \\
k(\mathbf{x},\mathbf{x}') &= k_a(\mathbf{x}_a,\mathbf{x}_a')k_b(\mathbf{x}_b,\mathbf{x}_b')
\end{aligned}
$$

Gaussian Kernel

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

with

$$\|\mathbf{x} - \mathbf{x'}\|^2 = \mathbf{x}^T \cdot \mathbf{x} + (\mathbf{x'})^T \cdot \mathbf{x'} - 2 \cdot \mathbf{x}^T \cdot \mathbf{x'}$$

and we get

$$k(\mathbf{x}, \mathbf{x'}) = \exp\left(\frac{\mathbf{x}^T \cdot \mathbf{x}}{2 \cdot \sigma^2}\right) \cdot \exp\left(\frac{\mathbf{x}^T \cdot \mathbf{x'}}{\sigma^2}\right) \cdot \exp\left(\frac{\mathbf{x'}^T \cdot \mathbf{x'}}{2 \cdot \sigma^2}\right)$$

- Since:

$$\begin{aligned} k(\mathbf{x}, \mathbf{x}') &= f(\mathbf{x}) k_1(\mathbf{x}, \mathbf{x}') f(\mathbf{x}') \\ k(\mathbf{x}, \mathbf{x}') &= \exp(k_1(\mathbf{x}, \mathbf{x}')) \end{aligned}$$

- The feature vector that corresponds to the Gaussian kernel has *infinite dimensionality*

Generative mode Kernels

Given a generative model

$$p(\mathbf{x})$$

we can define a kernel by

$$k(\mathbf{x}, \mathbf{x'}) = p(\mathbf{x}) \cdot p(\mathbf{x'})$$

Two inputs $\mathbf{x}$ and $\mathbf{x'}$ are similar if they both have high probabilities.

With positive weighting coefficients $p(i)$

$$k(\mathbf{x}, \mathbf{x'}) = \sum_i p(\mathbf{x}|i) \cdot p(\mathbf{x'}|i) \cdot p(i)$$

with the index i representing a latent variable

$$k(\mathbf{x}, \mathbf{x'}) = \sum_{\mathbf{z}} p(\mathbf{x}|\mathbf{z}) \cdot p(\mathbf{x'}|\mathbf{z}) \cdot p(\mathbf{z})$$

$\mathbf{z}$ is a latent variable

Sigmoidal kernel

A kernel function is the sigmoidal kernel given by

$$k(\mathbf{x}, \mathbf{x'}) = \tanh(a \cdot \mathbf{x}^T \cdot \mathbf{x'} + b)$$

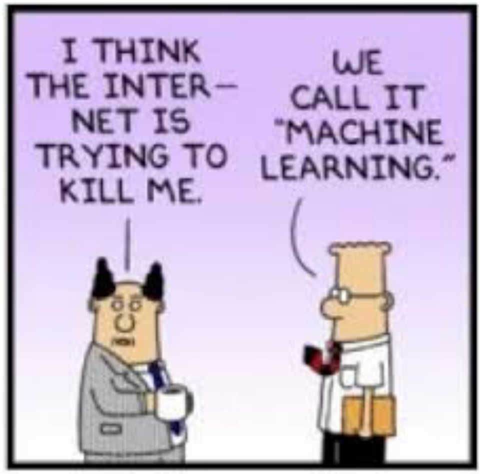
whose Gram matrix in general is not positive semidefinite.

This form of kernel has, however, been used in practice (Vapnik, 1995), because it gives kernel expansions such as the support vector machine and it is similar to the one used in neural network models.

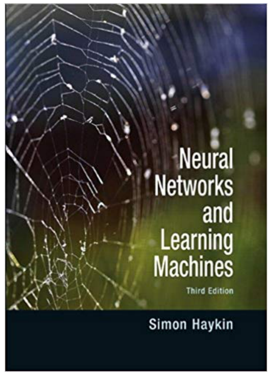
Next:

!!!
depositphotos

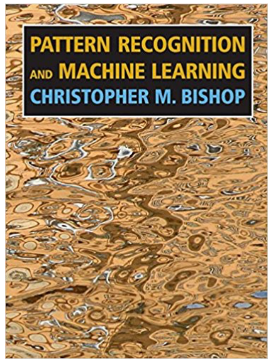
i support
vector machines


I THINK THE INTER-NET IS TRYING TO KILL ME.
WE CALL IT "MACHINE LEARNING."

Literature

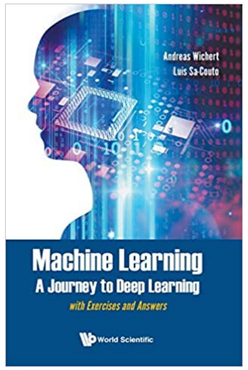


- Simon O. Haykin, Neural Networks and Learning Machine, (3rd Edition), Pearson 2008
  - Chapter 5



- Christopher M. Bishop, Pattern Recognition and Machine Learning (Information Science and Statistics), Springer 2006
  - Chapter 6

Literature



- Machine Learning - A Journey to Deep Learning, A. Wichert, Luis Sa-Couto, World Scientific, 2021
  - Chapter 10