Statistical Machine Learning

Christian Walder

Machine Learning Research Group
CSIRO Data61

and

College of Engineering and Computer Science
The Australian National University

Canberra
Semester One, 2020.

(Many figures from C. M. Bishop, "Pattern Recognition and Machine Learning")

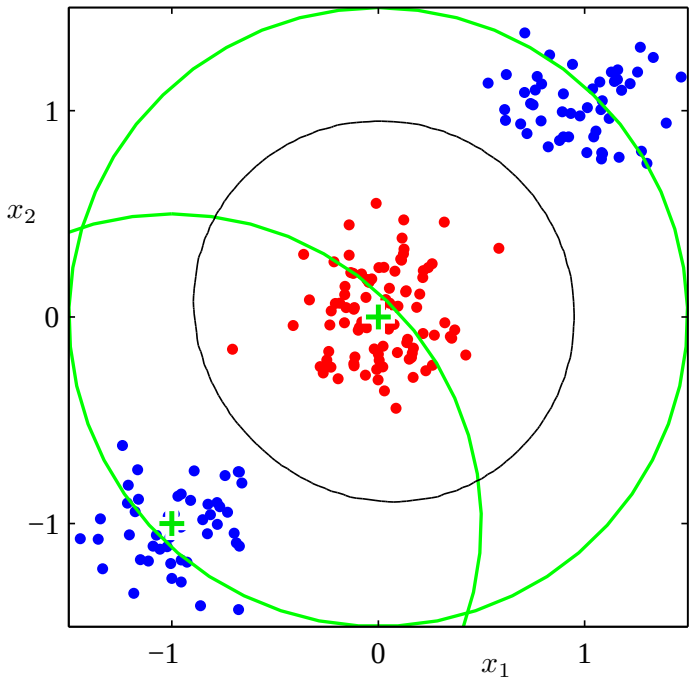
Outlines

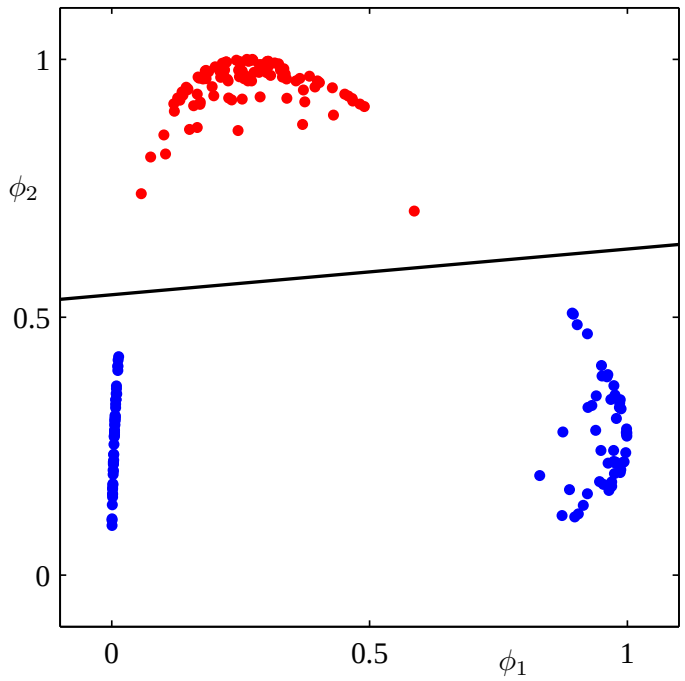
Part XIII

Kernel Methods

Original Input versus Feature Space

- Basic linear regression models direct input $\mathbf{x}$.
- These models are trivially extended to work on a fixed nonlinear transformation of the inputs using a vector of basis functions $\boldsymbol{\phi}(\mathbf{x})$.
- Example: Use two Gaussian basis functions centered at the green crosses in the input space.

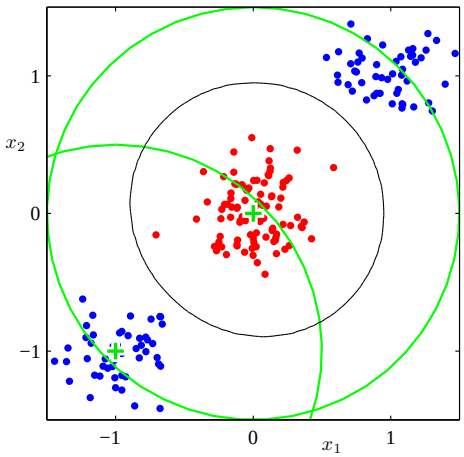


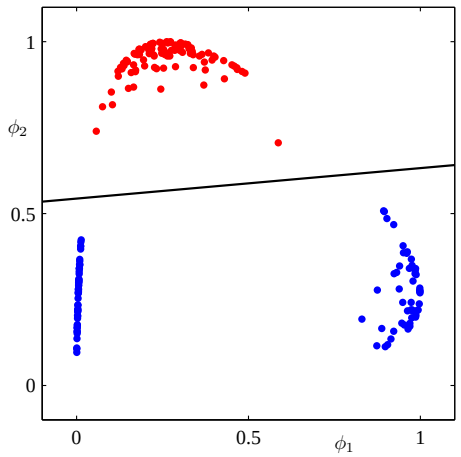


Ong & Walder & Webers
Data61 | CSIRO
The Australian National University

Original Input versus Feature Space

- Linear decision boundaries in the feature space correspond to nonlinear decision boundaries in the input space.
- Classes which are NOT linearly separable in the input space can become linearly separable in the feature space.
- BUT: If classes overlap in input space, they will also overlap in feature space.
- Nonlinear features $\phi(\mathbf{x})$ cannot remove the overlap.





Ong & Walder & Webers
Data61 | CSIRO
The Australian National University

Where have we been?

- Basis function models (regression, classification)
- Flexible basis function models (neural networks)

Ong & Walder & Webers
Data61 | CSIRO
The Australian National University

Where are we going?

- Why not use all training data to make predictions for the test inputs?
- Basic ideas:
  - Continuity : Mostly targets don't change abruptly.
  - Similarity : Each training pair (input, target) tells us something about the possible targets in the neighbourhood of the input.
- Kernels formalise those ideas.
- Nonparametric methods: do not rely on a fixed number of parameters, but rather usually on storing the entire training set (various loose definitions are used here).
- Often leading to Shallow (*c.f. Deep*) learning.

Ong & Walder & Webers
Data61 | CSIRO
The Australian National University

How are we going there?

- 0) Histograms
- 1) Simple density estimation kernels (must only be non-negative integrable)
- 2) Implicit feature mapping kernels (must be positive definite)

Warning:

- The term kernel is highly overloaded.
- Even today we consider two different types of kernel:
  1. Smoothing kernel / density estimation kernel / Parzen window estimator kernel / Nadaraya-Watson kernel
  2. Positive semidefinite kernel / reproducing kernel hilbert space kernel / implicit feature map kernel / support vector machine kernel / $\approx$ Gaussian process covariance function or kernel

Kernel Density Estimation

Data61 | CSIRO
The Australian National University

Density Estimation

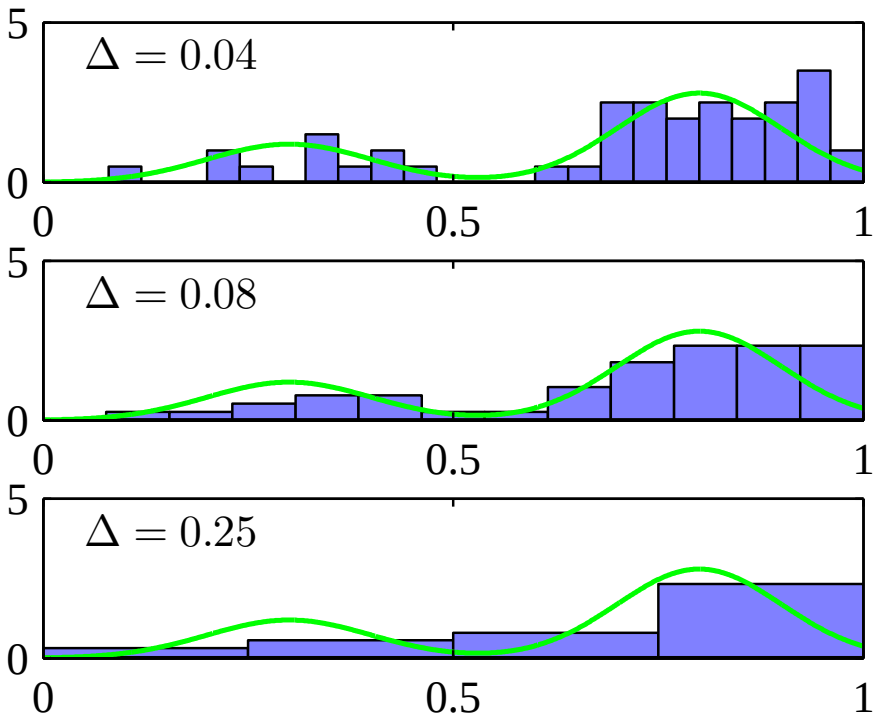
- Suppose we observe data points $\{\mathbf{x}_n\}_{n=1}^N$
  - e.g. just N real numbers
- Suppose we believe these are drawn independently from some distribution $p(\mathbf{x})$
  - e.g. $p(\mathbf{x})$ is Gaussian with unknown mean and variance
- **Density estimation problem**: Estimate $p(\mathbf{x})$ from data

Data61 | CSIRO
The Australian National University

Nonparametric Density Estimation – Histogram

- Partition the space into bins of width Δ_i.
- Count the number n_i of samples falling into each bin i.
- Normalise.

$$p_i = \frac{n_i}{N\Delta_i}$$



Histogram of 50 data points generated from the distribution shown by the green curve for varying common bin width Δ

Ong & Walder & Webers
Data61 | CSIRO
The Australian National University

Nonparametric Density Estimation – Histogram

Advantages:

- Data can be discarded after calculating the p_i (therefore seems to *not* be non-parametric, though often described as such).
- Algorithm can be applied to sequentially arriving data.

Disadvantages:

- Dependency on bin width Δ_i.
- Discontinuities due to the bin edges.
- Exponential scaling with the dimensionality D of the data. Need M^D bins for D dimensions and M bins per dimension.

Ong & Walder & Webers
Data61 | CSIRO
The Australian National University

Nonparametric Density Estimation - Refined

- Draw data from some unknown probability distribution $p(\mathbf{x})$ in a D-dimensional space.
- Consider a small region $\mathcal{R}$ containing $\mathbf{x}$. Probability mass associated with this region

$$P = \int_{\mathcal{R}} p(\mathbf{x}) \, \mathrm{d}\mathbf{x}$$

- Data set of N observations drawn from $p(\mathbf{x})$. Total number K of points found inside of $\mathcal{R}$ is distributed according to the binomial distribution

$$\mathrm{Bin}(K \,|\, N, P) = \frac{N!}{K!(N-K)!} P^K (1-P)^{N-K}$$

- Expectation of K : $\mathbb{E}\left[K/N\right] = P$
- Variance of K : $\mathrm{var}[K/N] = P(1-P)/N$

Nonparametric Density Estimation - Refined

- Expectation of K : $\mathbb{E}[K/N] = P$
- Variance of K : $\mathrm{var}[K/N] = P(1-P)/N$
- For large N, the distribution will be sharply peaked and therefore

$$P \approx K/N$$

- Assuming also that the region has volume V and the region is small enough for $p(\mathbf{x})$ to be roughly constant, then

$$P \approx p(\mathbf{x})V$$

- Combining two contradictory assumptions
  - Region $\mathcal{R}$ is small enough for $p(\mathbf{x})$ to be roughly constant.
  - Region $\mathcal{R}$ is large enough to have enough K points falling into it to get a sharp peak for the binomial distribution.

$$p(\mathbf{x}) \approx \frac{K}{NV}$$

Nonparametric Density Estimation - Refined

- Two ways to exploit

$$p(\mathbf{x}) \approx \frac{K}{NV}$$

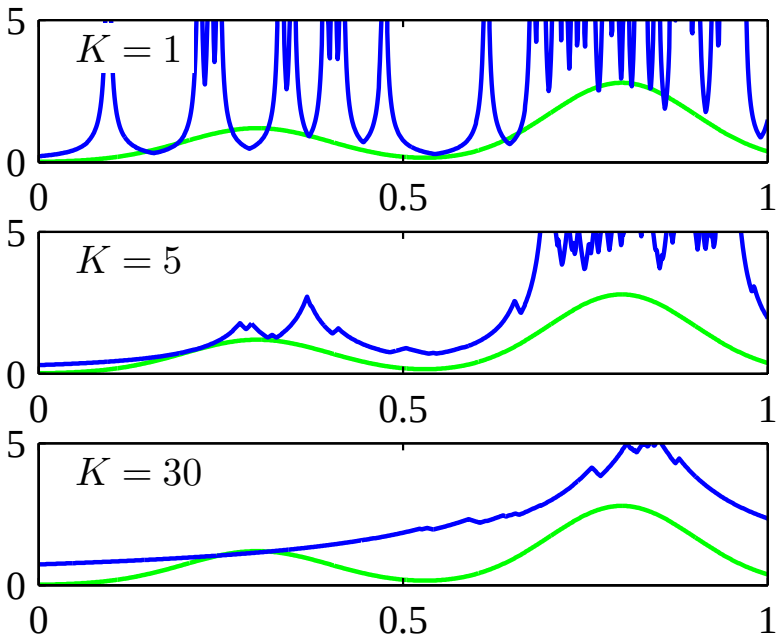
1. Fix K and determine the volume V from the data : K-nearest-neighbours density estimation
2. Fix V and determine K from the data : kernel density estimation

Data61 | CSIRO
The Australian National University

Nonparametric Estimation – Nearest Neighbour

- Fix K and find an appropriate value for V.
- Consider a small sphere around $\mathbf{x}$ and then allow the radius to increase until it contains exactly K data points.
- Calculate the probability by

$$p(\mathbf{x}) \approx \frac{K}{NV}$$



Nearest neighbour density model for different K.

Ong & Walder & Webers
Data61 | CSIRO
The Australian National University

Nonparametric Estimation – Parzen Estimator

- Define region $\mathcal{R}$ to be a small hypercube around $\mathbf{x}$
- Define Parzen window (kernel function)

$$k(\mathbf{u}) = \begin{cases} 1, & |u_i| \leq 1/2, \qquad i = 1, \ldots, D \\ 0, & \text{otherwise} \end{cases}$$

- Total number of data points inside of the hypercube centered at $\mathbf{x}$ with lengths h:

$$K = \sum_{n=1}^{N} k\left(\frac{\mathbf{x} - \mathbf{x}_n}{h}\right)$$

- Density estimate for $p(\mathbf{x})$

$$p(\mathbf{x}) \approx \frac{K}{NV} = \frac{1}{N} \sum_{n=1}^{N} \frac{1}{h^D} k\left(\frac{\mathbf{x} - \mathbf{x}_n}{h}\right)$$

- Interpret as sum over N cubes centered at each of the $\mathbf{x}_n$.

Nonparametric Estimation – Parzen Estimator

- Solved problem with Histogram of fixed discretisation of input space.
- Remaining problem: Discontinuities because of the hypercube (either in or out).
- Choose a smoother kernel function (and normalise correctly).
- Common choice : Gaussian kernel

$$k(\mathbf{x}) = \mathcal{N}(\mathbf{x}|\mathbf{0}, h^2 I) = \frac{1}{(2\pi h^2)^{D/2}} \exp\left\{-\frac{\|\mathbf{x}\|^2}{2h^2}\right\}$$

- Can choose any other kernel function $k(\mathbf{u})$ obeying

$$k(\mathbf{u}) \geq 0,$$

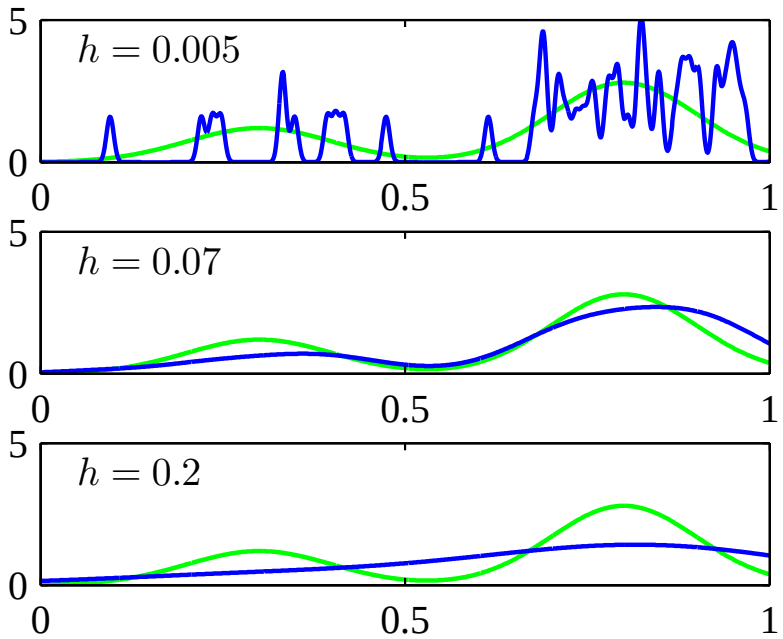
$$\int k(\mathbf{u})\, \mathrm{d}\mathbf{u} = 1$$

Nonparametric Estimation – Parzen Estimator

- Gaussian *Kernel Density Estimate* (KDE):

$$p(\mathbf{x}) = \frac{1}{N}\sum_{n=1}^{N}\frac{1}{(2\pi h^2)^{D/2}}\exp\left\{-\frac{\|\mathbf{x}-\mathbf{x}_n\|^2}{2h^2}\right\}$$

- h controls the trade-off between sensitivity to noise and over-smoothing.



Kernel density model with Gaussian kernel for different h.

Data61 | CSIRO
The Australian National University

Kernel Methods for Classification

Ong & Walder & Webers
Data61 | CSIRO
The Australian National University

The Role of Training Data

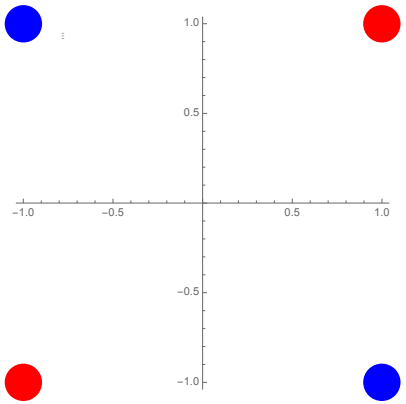
- Parametric methods
  - Learn the model parameter $\mathbf{w}$ from the training data $\mathbf{t}$.
  - Discard the training data $\mathbf{t}$.
- Nonparametric methods
  - Use training data directly for prediction
  - k-nearest neighbours : use k-closest data from the 'training' set for classification
- Kernel methods
  - Base prediction on linear combination of kernel functions evaluated at the training data.

Features

- A feature is a measurable property of a phenomenon being observed or any derived property thereof
  - raw features: the original data
  - derived features: mappings of the original features to some other space (possibly high- or infinite dimensional, e.g., basis functions)
- Feature selection: which features matter for the problem at hand?
  - redundant features
  - problem dependent
- Feature extraction: can we combine the important features to a smaller set of new features?
  - compact representation versus ability to explain to a human

Very simple example - XOR

x_1	x_2	$y = x_1 \text{xor} x_2$
-1	-1	1
-1	1	-1
1	-1	-1
1	1	1

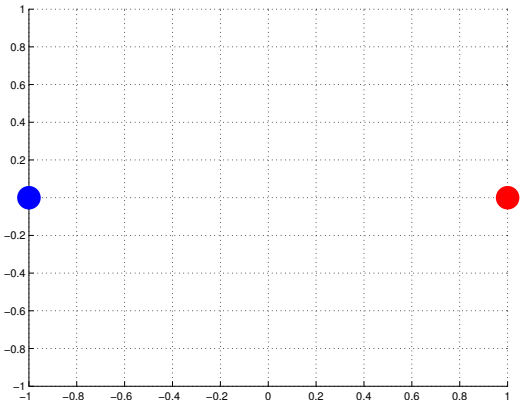


- not linearly separable (why?)
- raw features $\{(-1,-1),(-1,1),(1,-1),(1,1)\}$

Ong & Walder & Webers
Data61 | CSIRO
The Australian National University

Very simple example - XOR

x_1	x_2	$x_{\text{new}} = x_1 \cdot x_2$	$y = x_1 \text{xor} x_2$
-1	-1	1	1
-1	1	-1	-1
1	-1	-1	-1
1	1	1	1



- feature extraction: $x_{\text{new}} = x_1 \cdot x_2$
- data is now separable!
- All classification algorithms work also if we first apply a fixed nonlinear transformation of the inputs using a vector of basis functions $\phi(\mathbf{x})$.

Ong & Walder & Webers
Data61 | CSIRO
The Australian National University

Kernel methods in one slide

- Consider a labelled training set $\{\mathbf{x}_i, t_i\}_{i=1}^N$
- On a new point $\mathbf{x}$, we will predict

$$y(\mathbf{x}) = \sum_{i=1}^{N} a_i \cdot K(\mathbf{x}, \mathbf{x}_i)$$

  where $\{a_i\}_{i=1}^N$ are weights to be determined based on our training set, and $K(\cdot, \cdot)$ is a kernel function
- This is a major departure from the linear models considered previously!
- The kernel function measures the similarity between any two examples
  - Prediction is a weighted average of the training targets
  - Weights depend on the similarity of $\mathbf{x}$ to each training example

From Feature Functions to Kernels (1)

- Suppose we perform linear regression with a feature matrix $\mathbf{\Phi}$ and target vector $\mathbf{t}$, where

$$\mathbf{\Phi} = \begin{bmatrix} \phi_0(\mathbf{x}_1) & \phi_1(\mathbf{x}_1) & \dots & \phi_{M-1}(\mathbf{x}_1) \\ \phi_0(\mathbf{x}_2) & \phi_1(\mathbf{x}_2) & \dots & \phi_{M-1}(\mathbf{x}_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_0(\mathbf{x}_N) & \phi_1(\mathbf{x}_N) & \dots & \phi_{M-1}(\mathbf{x}_N) \end{bmatrix} = \begin{bmatrix} \boldsymbol{\phi}(\mathbf{x}_1)^\top \\ \boldsymbol{\phi}(\mathbf{x}_2)^\top \\ \dots \\ \boldsymbol{\phi}(\mathbf{x}_N)^\top \end{bmatrix}$$

- Recall that the optimal (regularised) $\mathbf{w}^\star$ is

$$\mathbf{w}^\star = (\lambda \mathbf{I} + \mathbf{\Phi}^\top \mathbf{\Phi})^{-1} \mathbf{\Phi}^\top \mathbf{t}$$

- Thus, the prediction for feature vector of new point $\mathbf{x}$ with

$$y(\mathbf{x}) = \boldsymbol{\phi}(\mathbf{x})^\top \mathbf{w}^\star = \boldsymbol{\phi}(\mathbf{x})^\top (\lambda \mathbf{I} + \mathbf{\Phi}^\top \mathbf{\Phi})^{-1} \mathbf{\Phi}^\top \mathbf{t}$$

From Feature Functions to Kernels (2)

- Prediction with optimal (regularised) $\mathbf{w}^\star$

$$y(\mathbf{x}) = \boldsymbol{\phi}(\mathbf{x})^\top \mathbf{w}^\star = \boldsymbol{\phi}(\mathbf{x})^\top (\lambda \mathbf{I} + \boldsymbol{\Phi}^\top \boldsymbol{\Phi})^{-1} \boldsymbol{\Phi}^\top \mathbf{t}$$

- Suppose that M is very large. Then, the inverse of an $M \times M$ matrix above will be expensive to compute.
- Consider however the following trick:

$$\boldsymbol{\phi}(\mathbf{x})^\top (\lambda \mathbf{I} + \boldsymbol{\Phi}^\top \boldsymbol{\Phi})^{-1} \boldsymbol{\Phi}^\top \mathbf{t} = \boldsymbol{\phi}(\mathbf{x})^\top \boldsymbol{\Phi}^\top (\lambda \mathbf{I} + \boldsymbol{\Phi}\boldsymbol{\Phi}^\top)^{-1} \mathbf{t}$$

(this is a useful idea, worth verifying)

From Feature Functions to Kernels (3)

- We have thus written the prediction as

$$y(\mathbf{x}) = \phi(\mathbf{x})^\top \mathbf{\Phi}^\top (\lambda \mathbf{I} + \mathbf{\Phi}\mathbf{\Phi}^\top)^{-1}\mathbf{t}$$
$$= \sum_{i=1}^{N} a_i \cdot K(\mathbf{x}, \mathbf{x}_i)$$

  as before (check by identifying the a_i and K in the above).
- Now, our prediction is determined by an $N \times N$ rather than $M \times M$ matrix
- $\mathbf{\Phi}\mathbf{\Phi}^\top$ is known as the kernel matrix of the training data
  - In some sense, measures the similarities between the training instances
  - For example, for normalised vectors (the case for some implicit kernel mappings), the inner product between two points is a measure of similarity:

$$\arg\max_{\mathbf{v}:\|\mathbf{v}\|=\|\mathbf{u}\|} \langle \mathbf{u}, \mathbf{v} \rangle = \mathbf{u}.$$

- NB: the $\phi(\mathbf{x}_m)^\top$ are never explicitly needed, but only the inner products $\phi(\mathbf{x}_m)^\top \phi(\mathbf{x}_n) \equiv K(\mathbf{x}_m, \mathbf{x}_n)$. (!)

Dual Representations

- Consider a linear regression model with regularised sum-of-squares error

$$J(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}(\mathbf{w}^\top \boldsymbol{\phi}(\mathbf{x}_n) - t_n)^2 + \frac{\lambda}{2}\mathbf{w}^\top\mathbf{w}$$

  where $\lambda \geq 0$.
- We could also write this in more compact form as

$$J(\mathbf{w}) = \frac{1}{2}(\mathbf{t} - \boldsymbol{\Phi}\mathbf{w})^\top(\mathbf{t} - \boldsymbol{\Phi}\mathbf{w}) + \frac{\lambda}{2}\mathbf{w}^\top\mathbf{w}$$

  with the target vector $\mathbf{t} = (t_1, \ldots, t_N)^\top$, and the design matrix

$$\boldsymbol{\Phi} = \begin{bmatrix} \phi_0(\mathbf{x}_1) & \phi_1(\mathbf{x}_1) & \ldots & \phi_{M-1}(\mathbf{x}_1) \\ \phi_0(\mathbf{x}_2) & \phi_1(\mathbf{x}_2) & \ldots & \phi_{M-1}(\mathbf{x}_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_0(\mathbf{x}_N) & \phi_1(\mathbf{x}_N) & \ldots & \phi_{M-1}(\mathbf{x}_N) \end{bmatrix}.$$

Dual Representations

- Critical points for $J(\mathbf{w})$

$$J(\mathbf{w}) = \frac{1}{2}(\mathbf{t} - \mathbf{\Phi}\mathbf{w})^\top(\mathbf{t} - \mathbf{\Phi}\mathbf{w}) + \frac{\lambda}{2}\mathbf{w}^\top\mathbf{w}$$

satisfy

$$\begin{aligned}(\mathbf{\Phi}^\top\mathbf{\Phi} + \lambda\mathbf{I})\mathbf{w} &= \mathbf{\Phi}^\top\mathbf{t} \\ \lambda\mathbf{w} &= \mathbf{\Phi}^\top(\mathbf{t} - \mathbf{\Phi}\mathbf{w}) \\ \mathbf{w} &= \mathbf{\Phi}^\top\mathbf{a} \\ &= \sum_{n=1}^{N}\phi(\mathbf{x}_n)a_n\end{aligned}$$

where $\mathbf{a} = (a_1, \ldots, a_N)^\top$ with components

$$a_n = -\frac{1}{\lambda}\left\{\mathbf{w}^\top\phi(\mathbf{x}_n) - t_n\right\}$$

Ong & Walder & Webers
Data61 | CSIRO
The Australian National University

Dual Representations

- Now express $J(\mathbf{w})$ as a function of this new variable $\mathbf{a}$ instead of $\mathbf{w}$ via the relation $\mathbf{w} = \boldsymbol{\Phi}^\top \mathbf{a}$

$$J(\mathbf{a}) = \frac{1}{2}\mathbf{a}^\top \boldsymbol{\Phi}\boldsymbol{\Phi}^\top \boldsymbol{\Phi}\boldsymbol{\Phi}^\top \mathbf{a} - \mathbf{a}^\top \boldsymbol{\Phi}\boldsymbol{\Phi}^\top \mathbf{t} + \frac{1}{2}\mathbf{t}^\top \mathbf{t} + \frac{\lambda}{2}\mathbf{a}^\top \boldsymbol{\Phi}\boldsymbol{\Phi}^\top \mathbf{a}$$

  where again $\mathbf{t} = (t_1, \ldots, t_N)^\top$.
  - Known as the dual representation
- Define the $N \times N$ Gram matrix $\mathbf{K} = \boldsymbol{\Phi}\boldsymbol{\Phi}^\top$ with elements

$$K_{nm} = \phi(\mathbf{x}_n)^\top \phi(\mathbf{x}_m) = k(\mathbf{x}_n, \mathbf{x}_m).$$

- Express $J(\mathbf{a})$ now as

$$J(\mathbf{a}) = \frac{1}{2}\mathbf{a}^\top \mathbf{K}\mathbf{K}\mathbf{a} - \mathbf{a}^\top \mathbf{K}\mathbf{t} + \frac{1}{2}\mathbf{t}^\top \mathbf{t} + \frac{\lambda}{2}\mathbf{a}^\top \mathbf{K}\mathbf{a}.$$

Stationary Point of $J(\mathbf{a})$

- Let's calculate the stationary point for

$$J(\mathbf{a}) = \frac{1}{2}\mathbf{a}^\top \mathbf{K}\mathbf{K}\mathbf{a} - \mathbf{a}^\top \mathbf{K}\mathbf{t} + \frac{1}{2}\mathbf{t}^\top \mathbf{t} + \frac{\lambda}{2}\mathbf{a}^\top \mathbf{K}\mathbf{a}.$$

- Gradient condition

$$\nabla_{\mathbf{a}} J(\mathbf{a}) = \mathbf{0} = \mathbf{K}\mathbf{K}\mathbf{a} - \mathbf{K}\mathbf{t} + \lambda \mathbf{K}\mathbf{a}$$

- Therefore

$$\mathbf{a}^\star = (\mathbf{K} + \lambda \mathbf{I}_N)^{-1}\mathbf{t}.$$

- Hessian is positive semi-definite

$$H_a J(\mathbf{a}) = \mathbf{K}\mathbf{K} + \lambda \mathbf{K} \succeq \mathbf{0}$$

because $\boldsymbol{\xi}^\top K \boldsymbol{\xi} = \boldsymbol{\xi}^\top \boldsymbol{\Phi}\boldsymbol{\Phi}^\top \boldsymbol{\xi} = \left\| \boldsymbol{\Phi}^\top \boldsymbol{\xi} \right\|^2 \geq 0, \forall \boldsymbol{\xi}$, and similarly for the first term.

- Hence $\mathbf{a}^\star$ minimises $J(\mathbf{a})$.

Prediction for the Linear Regression Model

- Putting $\mathbf{a}^\star$ which minimises the error $J(\mathbf{a})$ into the prediction model for the linear regression, we get for the prediction

$$\begin{aligned} y(\mathbf{x}) &= \mathbf{w}^\top \boldsymbol{\phi}(\mathbf{x}) = \mathbf{a}^{\star\top} \boldsymbol{\Phi} \boldsymbol{\phi}(\mathbf{x}) = (\boldsymbol{\Phi}\boldsymbol{\phi}(\mathbf{x}))^\top \mathbf{a}^\star \\ &= \mathbf{k}(\mathbf{x})^\top (\mathbf{K} + \lambda \mathbf{I}_N)^{-1} \mathbf{t} \end{aligned}$$

  where we defined the vector $\mathbf{k}(\mathbf{x})$ with elements $k_n(\mathbf{x}) = k(\mathbf{x}_n, \mathbf{x}) = \boldsymbol{\phi}(\mathbf{x}_n)^\top \boldsymbol{\phi}(\mathbf{x})$.
- The prediction $y(\mathbf{x})$ can be expressed entirely in terms of the kernel function $k(\mathbf{x}, \mathbf{x}')$ evaluated at the training and test data.
- Looks familiar? See Bayesian Linear Regression.

The Kernel Function

- The kernel function is defined over two points, $\mathbf{x}$ and $\mathbf{x}'$, of the input space

$$k(\mathbf{x}, \mathbf{x}') = \boldsymbol{\phi}(\mathbf{x})^\top \boldsymbol{\phi}(\mathbf{x}').$$

- $k(\mathbf{x}, \mathbf{x}')$ is symmetric.
- It is an inner product of two vectors of basis functions

$$k(\mathbf{x}, \mathbf{x}') = \langle \boldsymbol{\phi}(\mathbf{x}), \boldsymbol{\phi}(\mathbf{x}') \rangle.$$

- For prediction, the kernel function will be evaluated at the training data points. (See next slides.)

Dual Representation

- What have we gained by the dual representation?
- Need to invert an $N \times N$ matrix now, where N is the number of data points. Can be large!
  - In the parameter space formulation, we "only" needed to invert an $M \times M$ matrix, where M was the number of basis functions.
  - But, a kernel corresponds to an inner product of basis functions. So we can use a large number of basis functions, even infinitely many.
- We can construct new valid kernels directly from given ones (whatever the corresponding basis functions of the new kernel might be).
- As a kernel defines a kind of 'similarity' between two points in the input space, we can define kernels over graphs, sets, strings, and text documents.

Kernel Construction

Kernels from Basis Functions

1. Choose a set of basis functions

$$\{\phi_1, \ldots, \phi_M\}$$

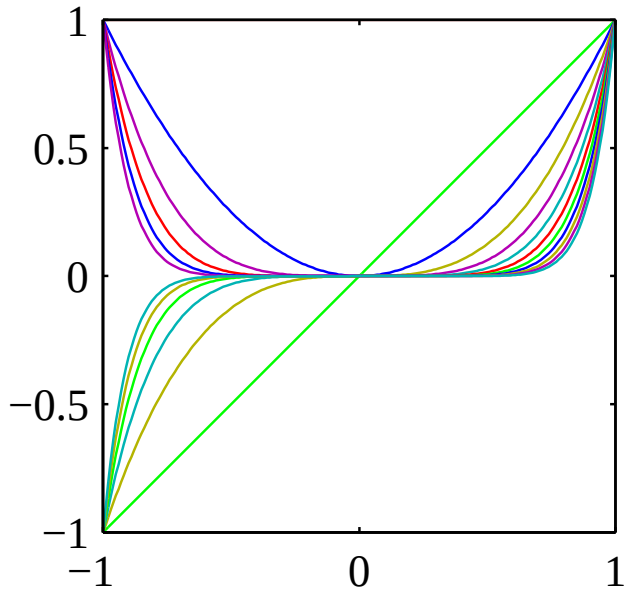
2. Find a new kernel as an inner product between vectors of basis functions evaluated at x and x'

$$k(x, x') = \boldsymbol{\phi}(x)^\top \boldsymbol{\phi}(x) = \sum_{i=1}^{M} \phi_i(x)\phi_i(x')$$

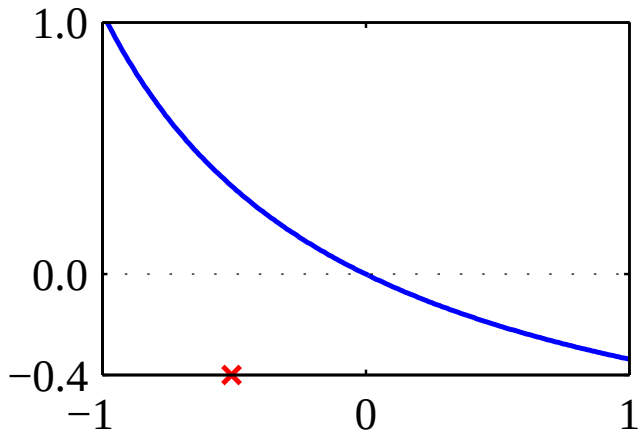
Ong & Walder & Webers
Data61 | CSIRO
The Australian National University

Kernels from Basis Functions

Polynomial basis functions



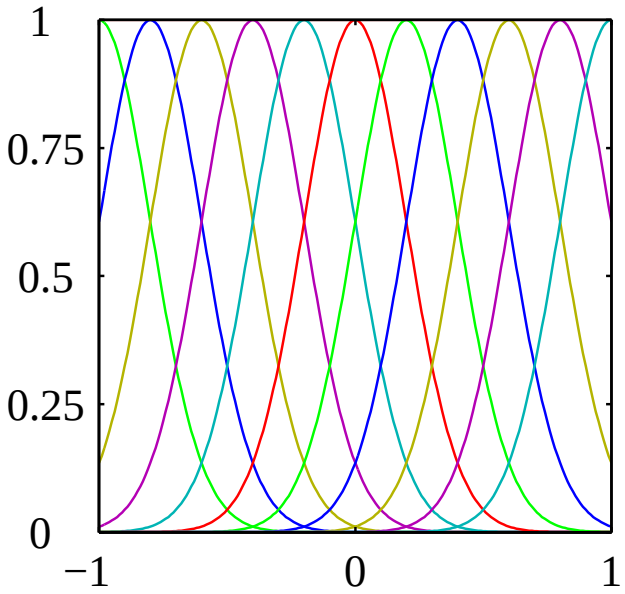
Corresponding kernel $k(x, x')$ as function of x for $x' = -0.5$ (red cross).



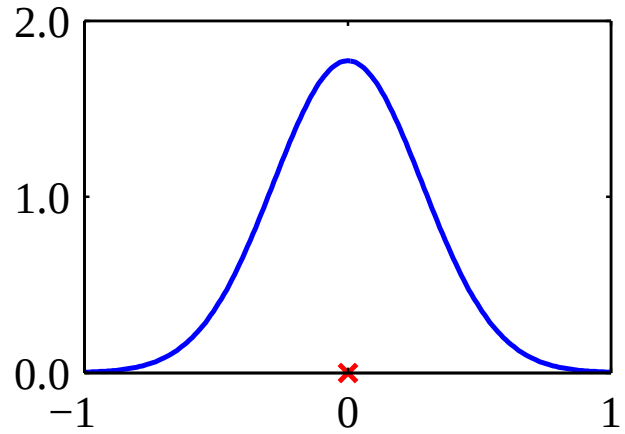
Data61 | CSIRO
The Australian National University

Kernels from Basis Functions

Gaussian basis functions



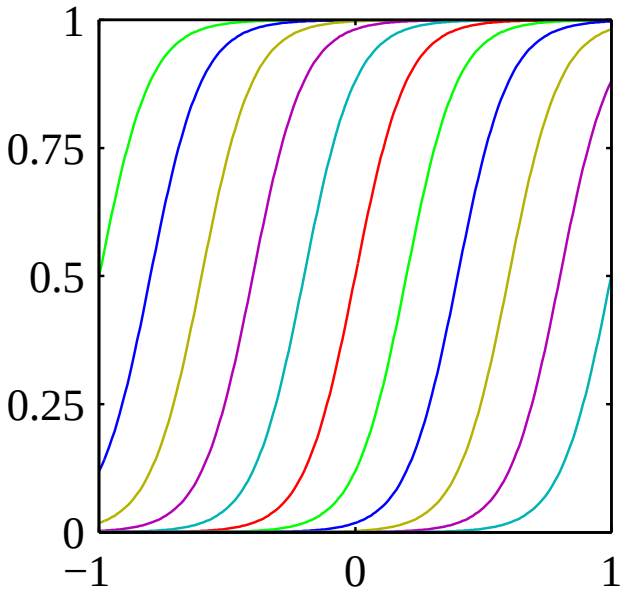
Corresponding kernel $k(x, x')$ as function of x for $x' = 0.0$ (red cross).



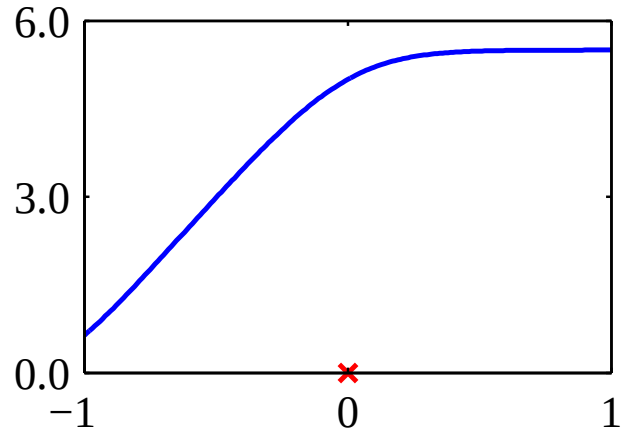
Ong & Walder & Webers
Data61 | CSIRO
The Australian National University

Kernels from Basis Functions

Logistic Sigmoid basis functions



Corresponding kernel $k(x, x')$ as function of x for $x' = 0.0$ (red cross).



Ong & Walder & Webers
Data61 | CSIRO
The Australian National University

Kernels by Guessing a Kernel Function

1. Choose a mapping from two points of the input space to a real number, which is symmetric in its arguments, e.g.

$$k(\mathbf{x}, \mathbf{z}) = (\mathbf{x}^\top \mathbf{z})^2 = k(\mathbf{z}, \mathbf{x})$$

2. Try to write this as an inner product of a vector valued function evaluated at the arguments $\mathbf{x}$ and $\mathbf{z}$, e.g.

$$\begin{aligned} k(\mathbf{x}, \mathbf{z}) &= (\mathbf{x}^\top \mathbf{z})^2 = (x_1 z_1 + x_2 z_2)^2 \\ &= x_1^2 z_1^2 + 2 x_1 z_2 x_2 z_2 + x_2^2 z_2^2 \\ &= (x_1^2, \sqrt{2} x_1 x_2, x_2^2)(z_1^2, \sqrt{2} z_1 z_2, z_2^2)^\top \\ &= \boldsymbol{\phi}(\mathbf{x})^\top \boldsymbol{\phi}(\mathbf{z}) \end{aligned}$$

with the feature mapping $\boldsymbol{\phi}(\mathbf{x}) = (x_1^2, \sqrt{2} x_1 x_2, x_2^2)^\top$.

New Kernels From Theory

- A necessary and sufficient condition for $k(\mathbf{x}, \mathbf{x}')$ to be a valid kernel is that the kernel matrix $\mathbf{K}$, whose elements are $k(\mathbf{x}_n, \mathbf{x}_m)$, should be positive semidefinite for all possible choices of the set $\{\mathbf{x}_n\}$.
- Previously, we constructed $\mathbf{K} = \mathbf{\Phi}\mathbf{\Phi}^\top$, which is automatically positive semidefinite (why?)
  - If we can explicitly construct the kernel via basis functions, we are good
  - Even if we cannot find the basis functions easily, we may be able to deduce $k(\mathbf{x}, \mathbf{x}')$ is a valid kernel

Ong & Walder & Webers
Data61 | CSIRO
The Australian National University

New Kernels From Other Kernels

Given valid kernels $k_1(\mathbf{x}, \mathbf{x}')$ and $k_2(\mathbf{x}, \mathbf{x}')$, the following kernels are also valid:

$$k(\mathbf{x}, \mathbf{x}') = c\, k_1(\mathbf{x}, \mathbf{x}')$$
$$k(\mathbf{x}, \mathbf{x}') = f(\mathbf{x})\, k_1(\mathbf{x}, \mathbf{x}') f(\mathbf{x}')$$
$$k(\mathbf{x}, \mathbf{x}') = q(k_1(\mathbf{x}, \mathbf{x}'))$$
$$k(\mathbf{x}, \mathbf{x}') = \exp(k_1(\mathbf{x}, \mathbf{x}'))$$
$$k(\mathbf{x}, \mathbf{x}') = k_1(\mathbf{x}, \mathbf{x}') + k_2(\mathbf{x}, \mathbf{x}')$$
$$k(\mathbf{x}, \mathbf{x}') = k_1(\mathbf{x}, \mathbf{x}')\, k_2(\mathbf{x}, \mathbf{x}')$$
$$k(\mathbf{x}, \mathbf{x}') = k_3(\boldsymbol{\phi}(\mathbf{x}), \boldsymbol{\phi}(\mathbf{x}'))$$
$$k(\mathbf{x}, \mathbf{x}') = \mathbf{x}^\top \mathbf{A} \mathbf{x}'$$
$$k(\mathbf{x}, \mathbf{x}') = k_a(\mathbf{x}_a, \mathbf{x}'_a) + k_b(\mathbf{x}_b, \mathbf{x}'_b)$$
$$k(\mathbf{x}, \mathbf{x}') = k_a(\mathbf{x}_a, \mathbf{x}'_a)\, k_b(\mathbf{x}_b, \mathbf{x}'_b)$$

$c > 0$ constant

$f(\cdot)$ any function

$q(\cdot)$ polynomial with nonneg. coeff.

$\boldsymbol{\phi}(\mathbf{x})$ any function to $\mathbb{R}^M$

$k_3(\cdot, \cdot)$ valid kernel in $\mathbb{R}^M$

$\mathbf{A} = \mathbf{A}^\top, \mathbf{A} \succeq 0$

$\mathbf{x} = (\mathbf{x}_a, \mathbf{x}_b)$

New Kernels From Other Kernels

Further examples of kernels

$$k(\mathbf{x}, \mathbf{x}') = (\mathbf{x}^\top \mathbf{x}')^M \qquad \text{only terms of degree } M$$

$$k(\mathbf{x}, \mathbf{x}') = (\mathbf{x}^\top \mathbf{x}' + c)^M \qquad \text{all terms up to degree } M$$

$$k(\mathbf{x}, \mathbf{x}') = \exp\left(-\|\mathbf{x} - \mathbf{x}'\|^2 / 2\sigma^2\right) \qquad \text{Gaussian kernel}$$

$$k(\mathbf{x}, \mathbf{x}') = \tanh\left(a\, \mathbf{x}^\top \mathbf{x}' + b\right) \qquad \text{Sigmoidal kernel (invalid)}$$

Generally, we call

$$k(\mathbf{x}, \mathbf{x}') = \mathbf{x}^\top \mathbf{x}' \qquad \text{linear kernel}$$

$$k(\mathbf{x}, \mathbf{x}') = k(\mathbf{x} - \mathbf{x}') \qquad \text{stationary kernel}$$

$$k(\mathbf{x}, \mathbf{x}') = k(\|\mathbf{x} - \mathbf{x}'\|) \qquad \text{homogeneous kernel}$$

Kernels over Graphs, Sets, Strings, Texts

- We 'only' need an appropriate similarity measure $k(\mathbf{x}, \mathbf{x}')$ which is a kernel.
- Example: Given a set $\mathcal{A}$ and the set of all subsets of $\mathcal{A}$, called the power set $\mathcal{P}(\mathcal{A})$.
- For two subsets $\mathcal{A}_1, \mathcal{A}_2 \in \mathcal{P}(\mathcal{A})$, denote the number of elements of the intersection of $\mathcal{A}_1$ and $\mathcal{A}_2$ by $|\mathcal{A}_1 \cap \mathcal{A}_2|$.
- Then it can be shown that

$$k(\mathcal{A}_1, \mathcal{A}_2) = 2^{|\mathcal{A}_1 \cap \mathcal{A}_2|}$$

corresponds to an inner product in a feature space. Therefore, $k(\mathcal{A}_1, \mathcal{A}_2)$ is a valid kernel function.

Data61 | CSIRO
The Australian National University

Kernels from Probabilistic Generative Models

- Given $p(\mathbf{x})$, we can define a kernel

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

  which means two inputs $\mathbf{x}$ and $\mathbf{x}'$ are similar if they both have high probabilities.
- Include a weighting function $p(i)$ and extend the kernel to

$$k(\mathbf{x}, \mathbf{x}') = \sum_i p(\mathbf{x} \mid i)\, p(\mathbf{x}' \mid i) p(i).$$

- For a continous variable $\mathbf{z}$

$$k(\mathbf{x}, \mathbf{x}') = \int p(\mathbf{x} \mid \mathbf{z})\, p(\mathbf{x}' \mid \mathbf{z}) p(\mathbf{z}) d\mathbf{z}.$$

- Hidden Markov Model with sequences of length L.

Kernels for Classification: Summary

- Pick a suitable kernel function $k(\mathbf{x}, \mathbf{x}')$
  - e.g. by computing inner product of some basis functions
- Make predictions by suitably combining $k(\mathbf{x}, \mathbf{x}_n)$ for each training example $\mathbf{x}_n$
  - implicitly, a linear model in some high-dimensional space
- For linear regression, we go from

$$y(\mathbf{x}) = \boldsymbol{\phi}(\mathbf{x})^\top (\lambda \mathbf{I} + \boldsymbol{\Phi}^\top \boldsymbol{\Phi})^{-1} \boldsymbol{\Phi}^\top \mathbf{t}$$

to

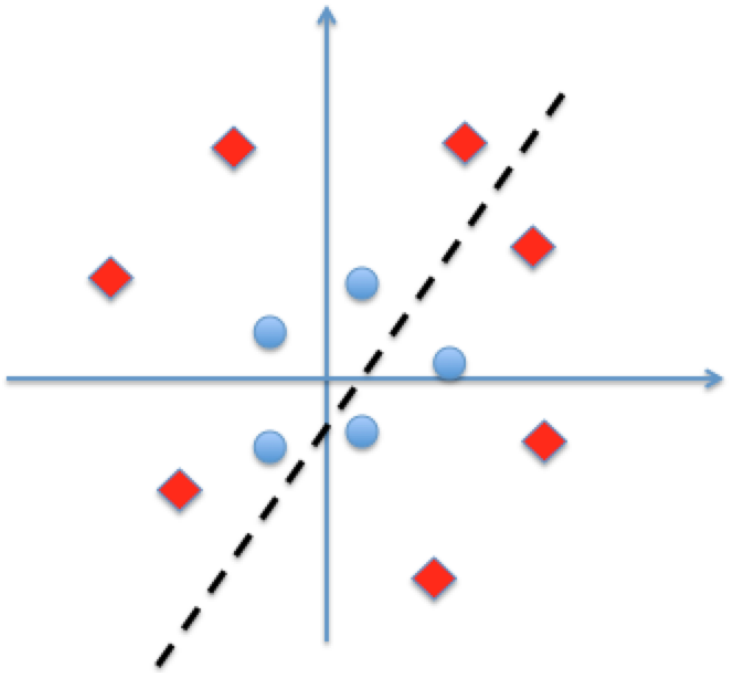
$$y(\mathbf{x}) = \mathbf{k}(\mathbf{x})^\top (\mathbf{K} + \lambda \mathbf{I}_N)^{-1} \mathbf{t}$$

  - can plug in suitable kernel function to implicitly perform nonlinear transformation

Ong & Walder & Webers
Data61 | CSIRO
The Australian National University

Kernels for Classification: Summary

- Working with a nonlinear kernel, we are implicitly performing a nonlinear transformation of our data

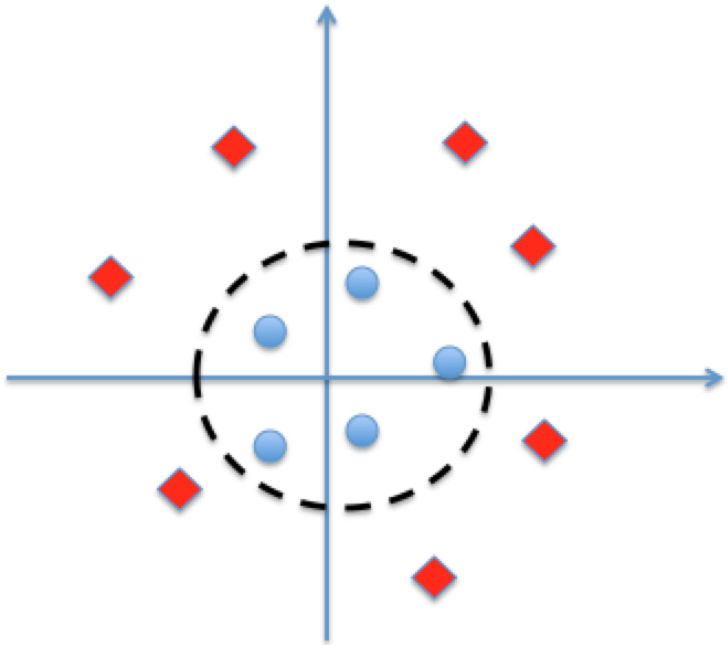


Classification with linear kernel $k(\mathbf{x}, \mathbf{x}') = \mathbf{x}^\top \mathbf{x}'$

Ong & Walder & Webers
Data61 | CSIRO
The Australian National University

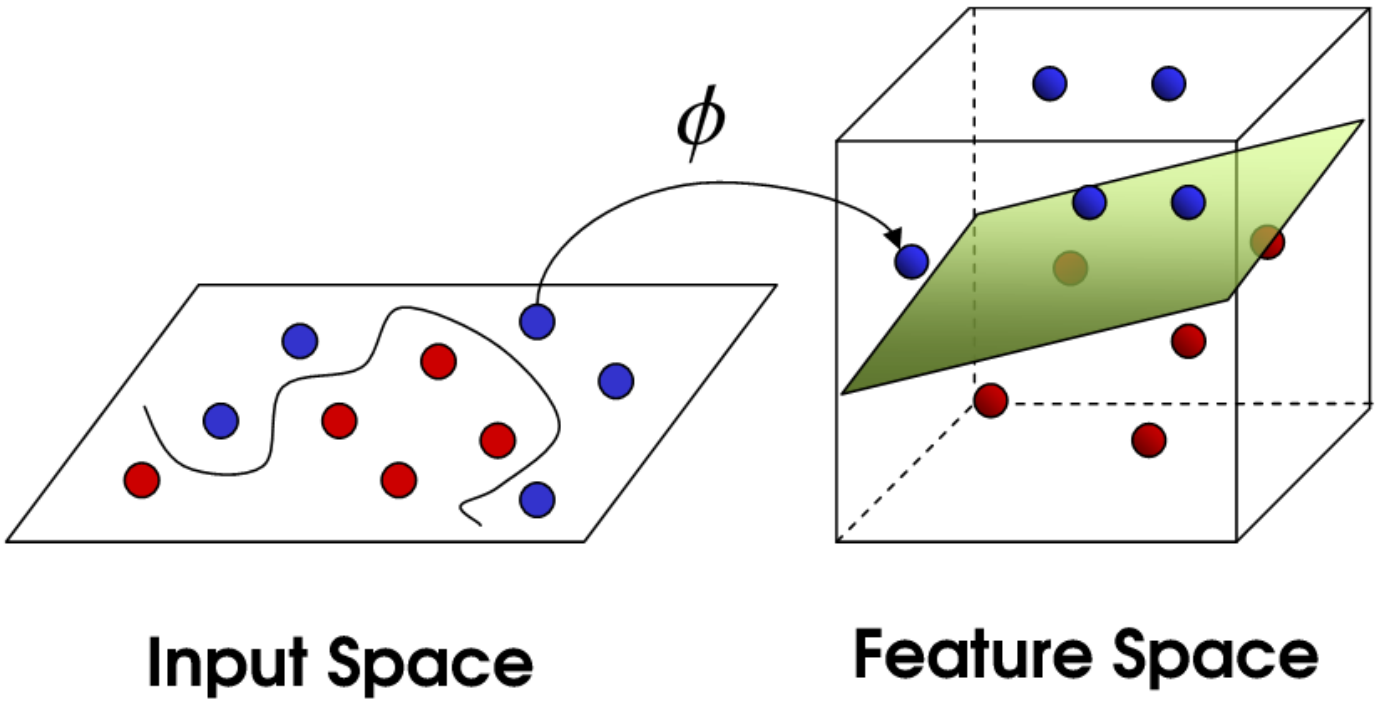
Kernels for Classification: Summary

- Working with a nonlinear kernel, we are implicitly performing a nonlinear transformation of our data



Classification with nonlinear kernel $k(\mathbf{x}, \mathbf{x}') = (\mathbf{x}^\top \mathbf{x}')^2$

Kernels for Classification: Summary



We need not explictly work with ϕ!