Kernels (chapter 2)

- Similarity measures
- Extended example
- Function spaces
- Theory of kernels
  - Positive definite kernels
  - Reproducing kernel map
  - Mercer kernel map

Similarity of Inputs

- symmetric function

$$\begin{aligned} k : \mathcal{X} \times \mathcal{X} &\to \mathbb{R} \\ (x, x') &\mapsto k(x, x') \end{aligned}$$

- for example, if $\mathcal{X} = \mathbb{R}^N$: canonical dot product

$$k(x, x') = \sum_{i=1}^{N} [x]_i [x']_i$$

- if $\mathcal{X}$ is not a vector space: assume that k has a representation as a dot product in a linear space $\mathcal{H}$, i.e., there exists a map $\Phi : \mathcal{X} \to \mathcal{H}$ such that

$$k(x, x') = \langle \Phi(x), \Phi(x') \rangle .$$

- in that case, we can think of the patterns as $\Phi(x), \Phi(x')$, and carry out geometric algorithms in the dot product space ("feature space") $\mathcal{H}$.

The Kernel Trick — Summary

- *any* algorithm that only depends on dot products can benefit from the kernel trick
- this way, we can apply linear methods to vectorial as well as *non-vectorial data*
- think of the kernel as a nonlinear *similarity measure*
- examples of common kernels:

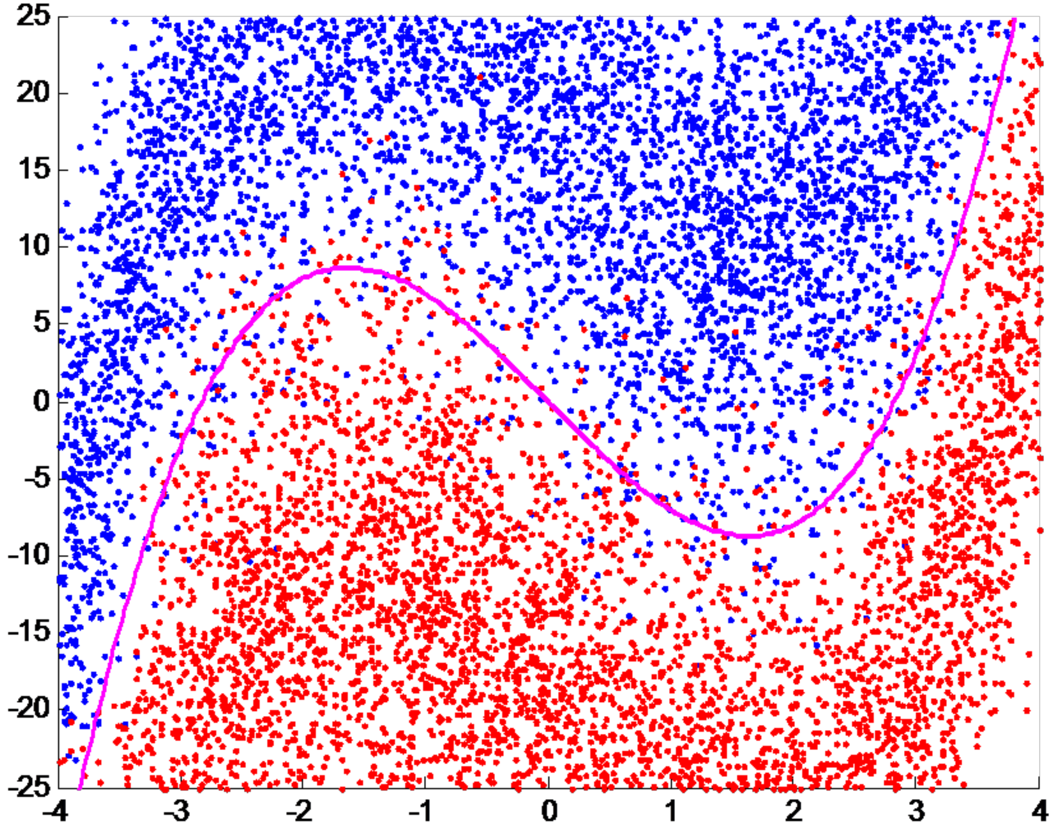
$$\begin{aligned} \text{Polynomial} \quad k(x,x') &= (\langle x,x' \rangle + c)^d \\ \text{Sigmoid} \quad k(x,x') &= \tanh(\kappa \langle x,x' \rangle + \Theta) \\ \text{Gaussian} \quad k(x,x') &= \exp(-\|x-x'\|^2/(2\,\sigma^2)) \end{aligned}$$

- Kernel are studied also in the Gaussian Process prediction community (covariance functions) [71, 68, 72, 40] — cf. Alex Smola's course

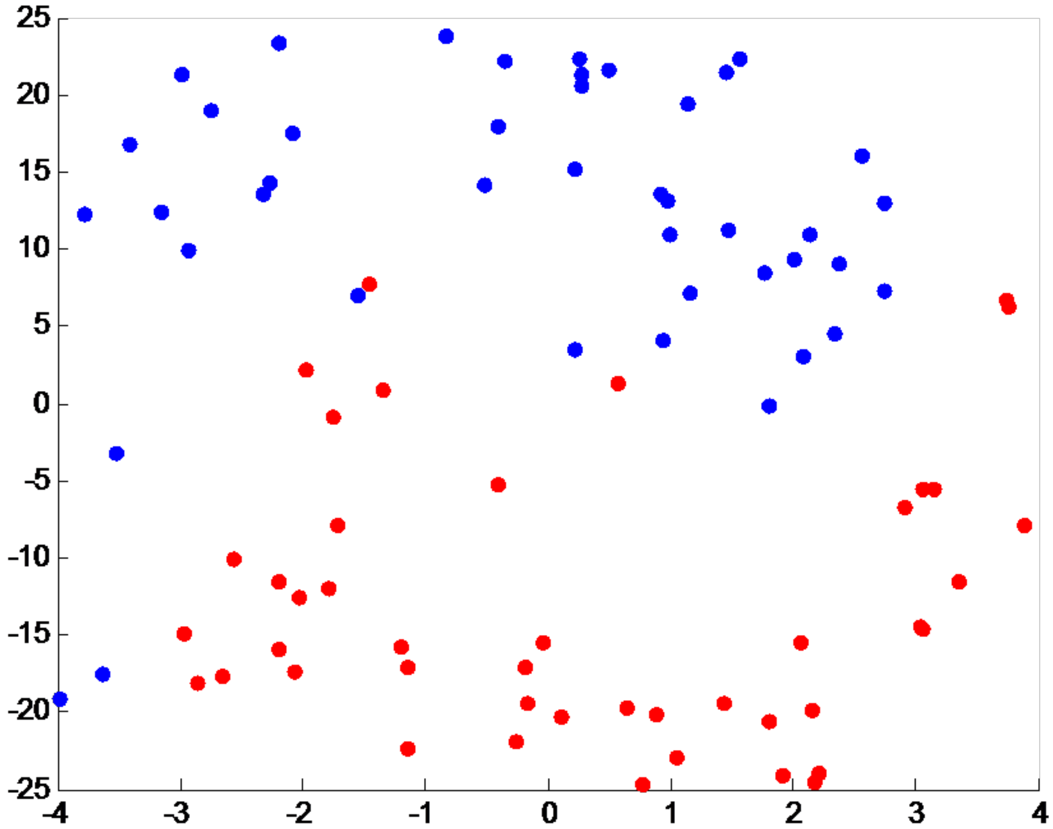
An extended example

- SVMs and other kernel methods do linear classification in (high dimensional) *feature* space.
- This approach is very general in that it works for *any* kernel function.
- We now illustrate how kernel methods work in *input* space.
- The example is based on RBF kernels used with a simple kernel method (described earlier).
- We shall see exactly how the kernel method leads to a non-linear decision boundary.

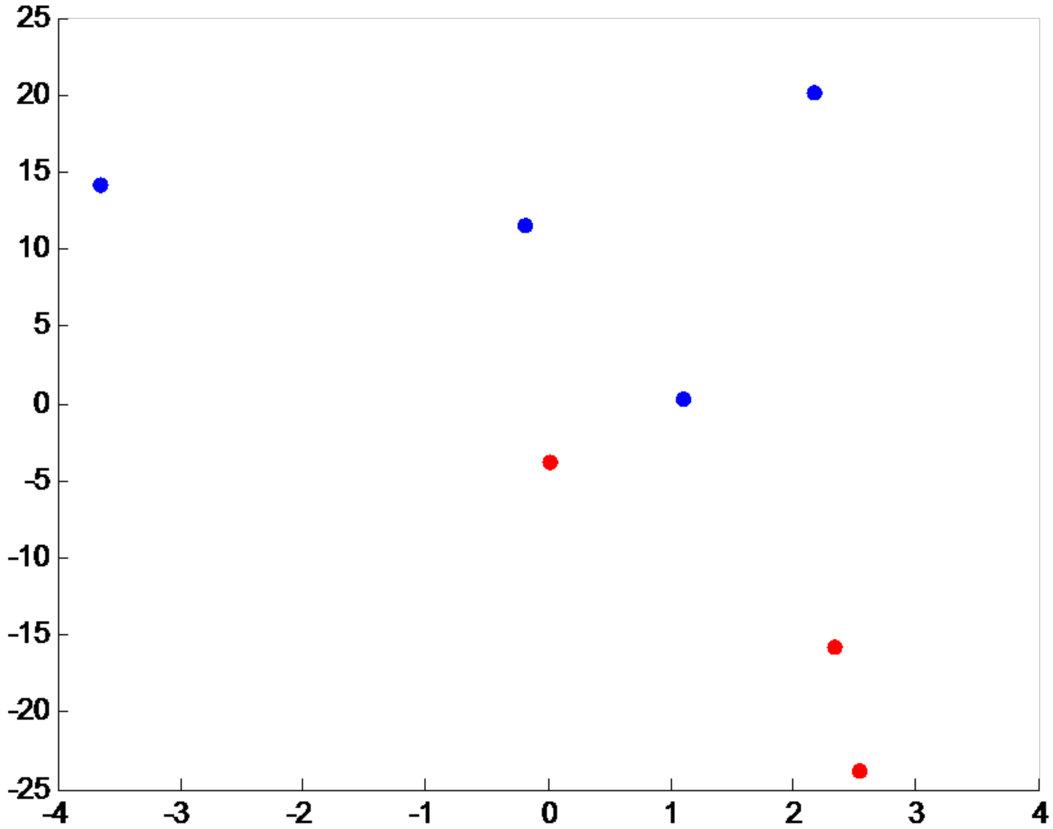
Two distributions and a decision boundary



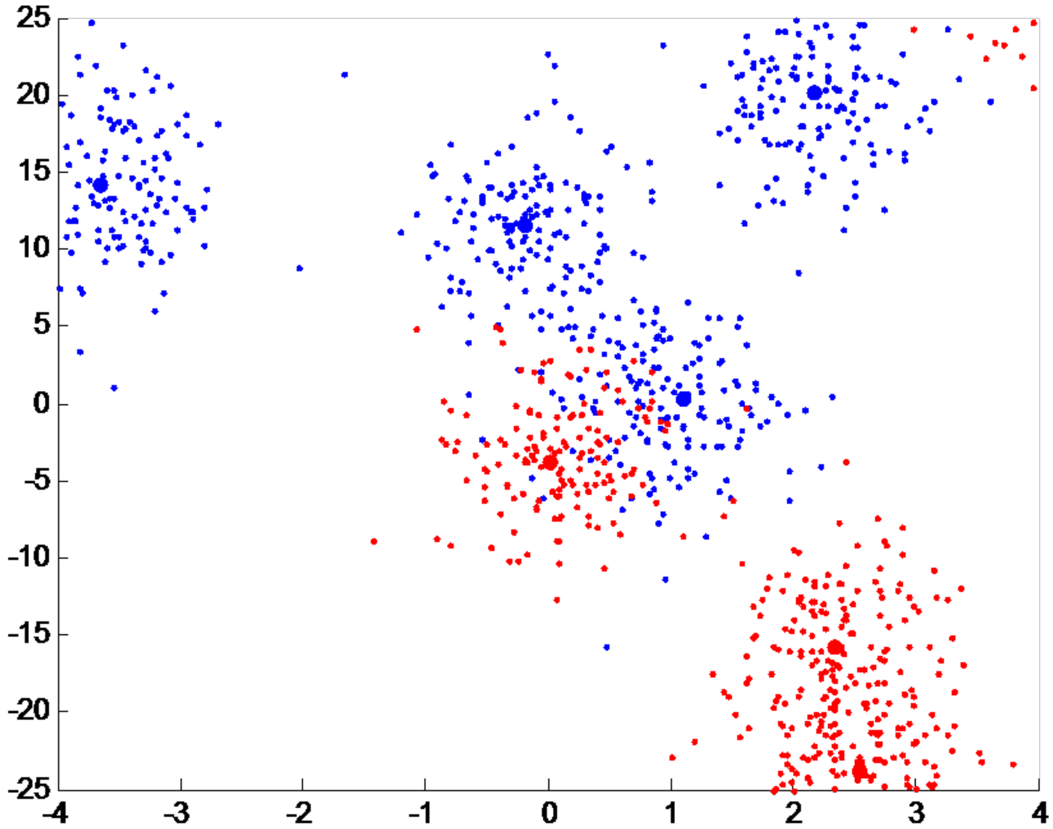
A random sample from the two distributions



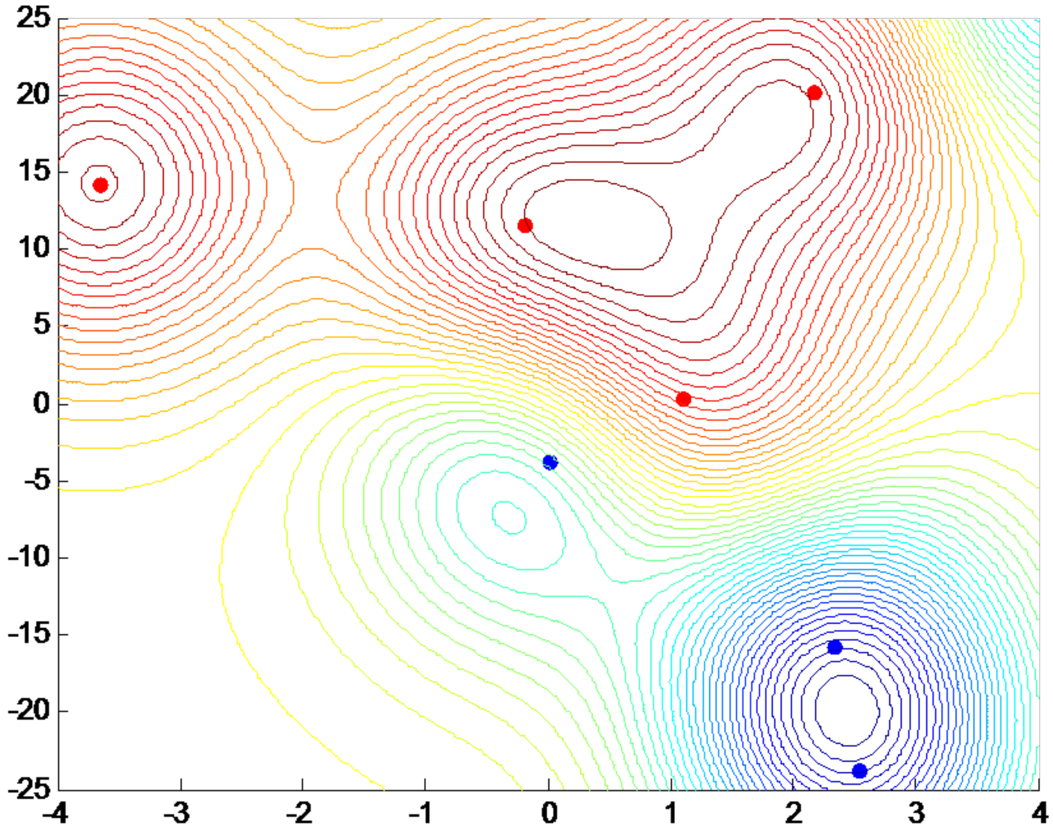
A tiny random sample from the two distributions



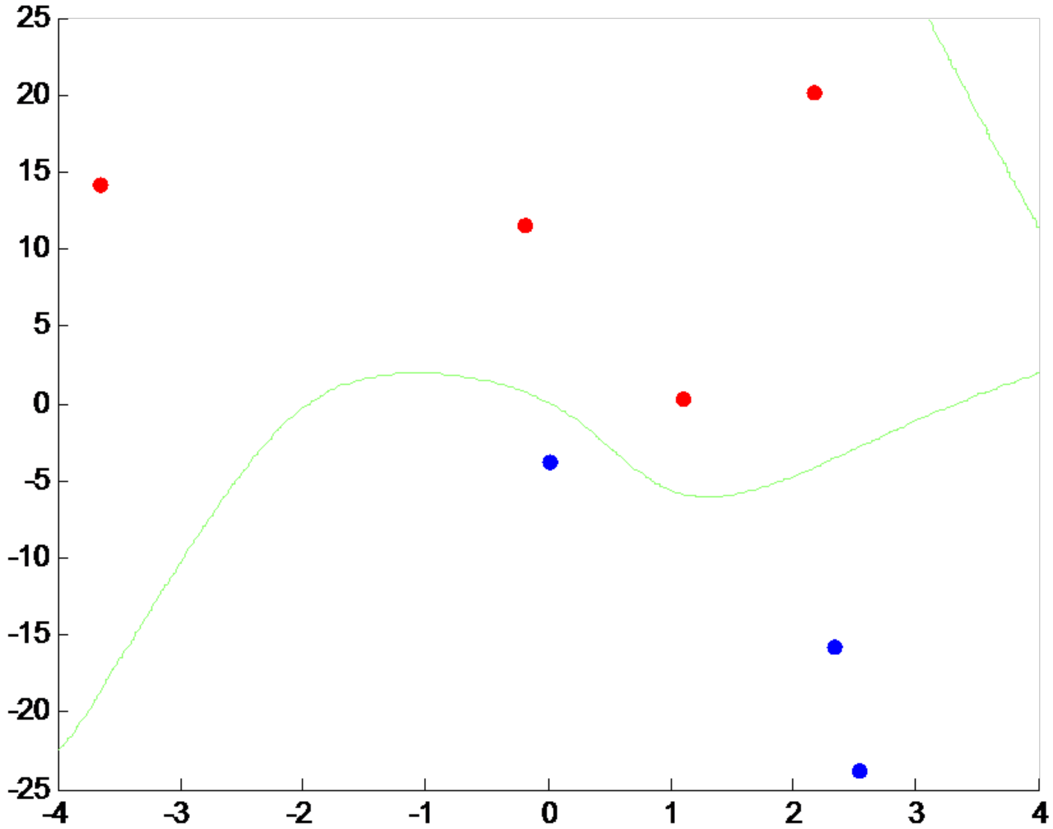
Placing an RBF kernel (a Gaussian distribution) at each sample point



A contour plot of the sum of the kernel values



Estimated decision boundary: the level 0 contour

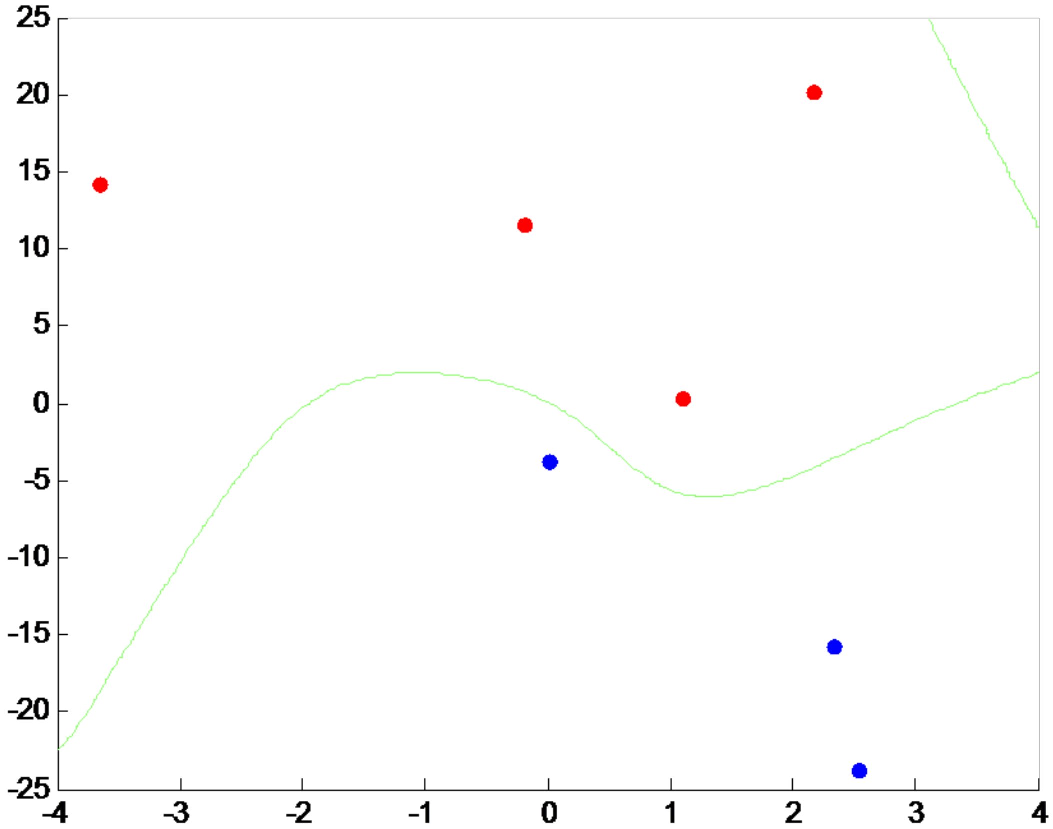


Observation

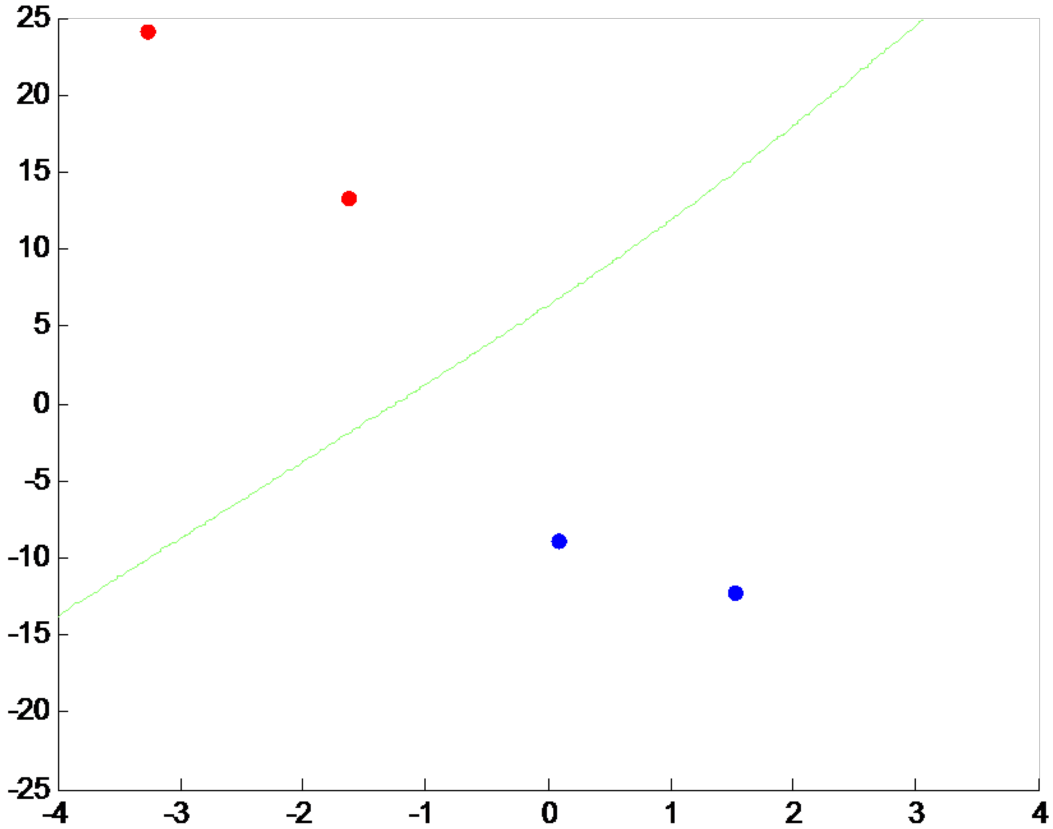
- For very small data samples, the variance in the estimated decision boundary (or of almost anything else) is very high.

- That is, different (very small) data samples can give very different estimates for the decision boundary.

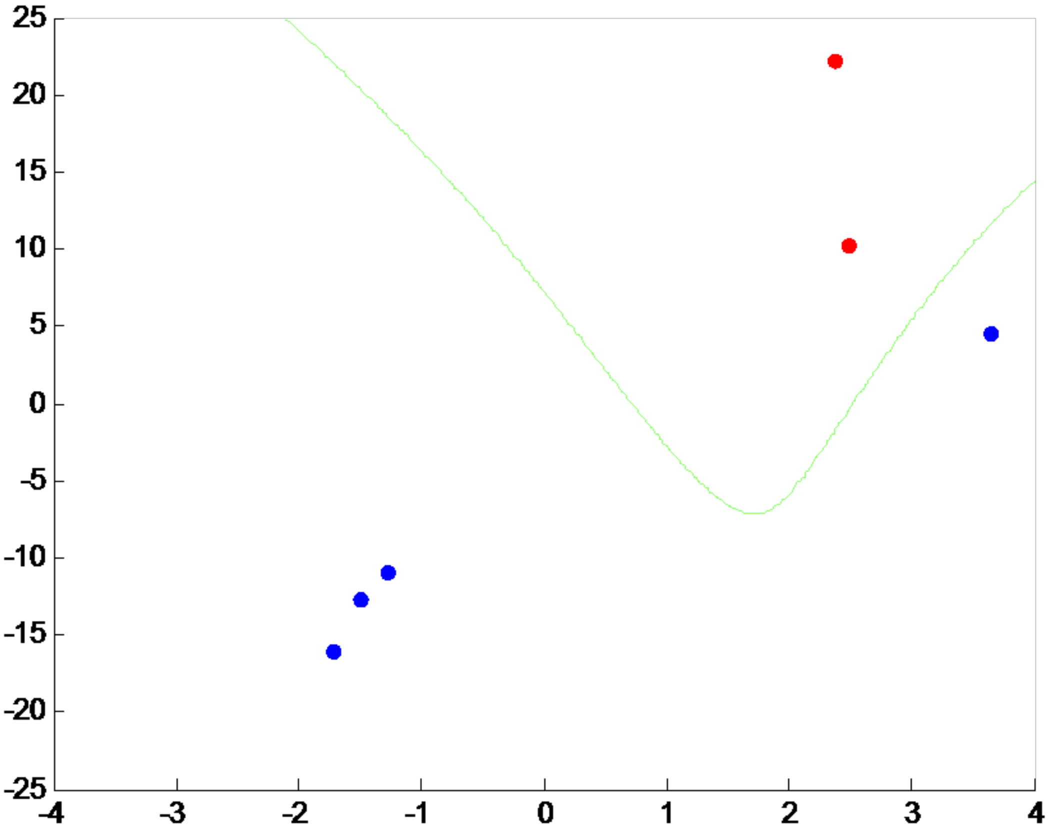
Estimated decision boundary for tiny data sample 1



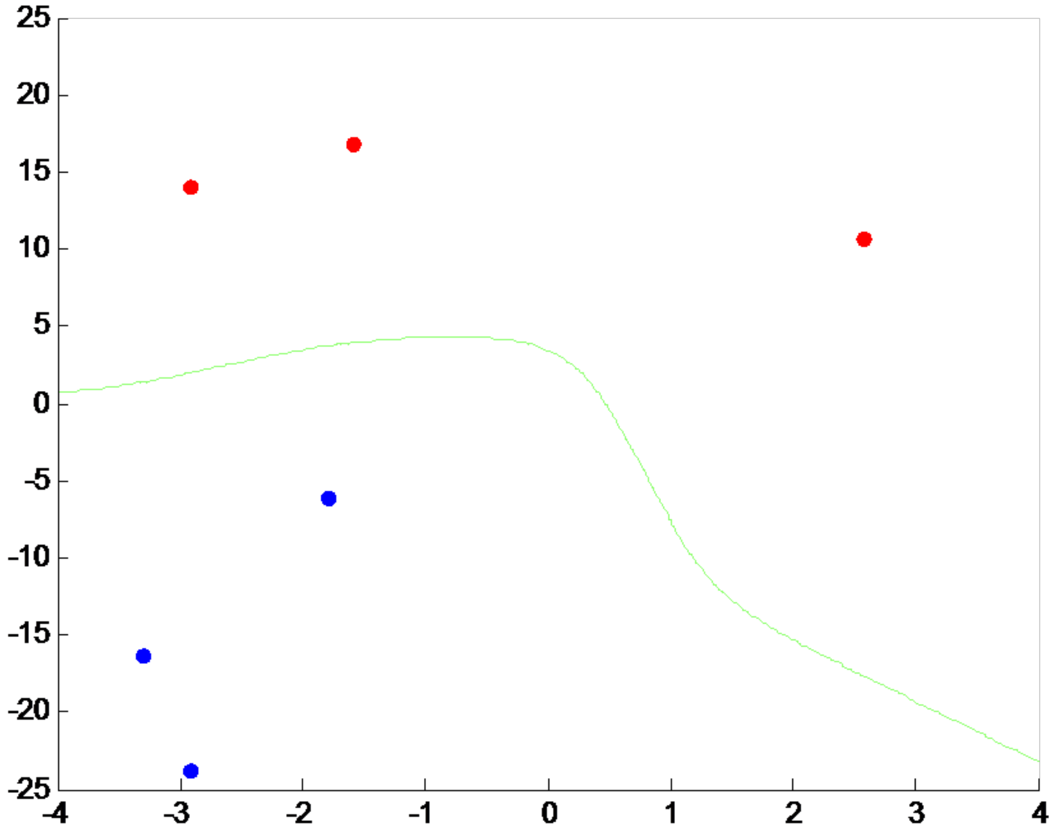
Estimated decision boundary for tiny data sample 2



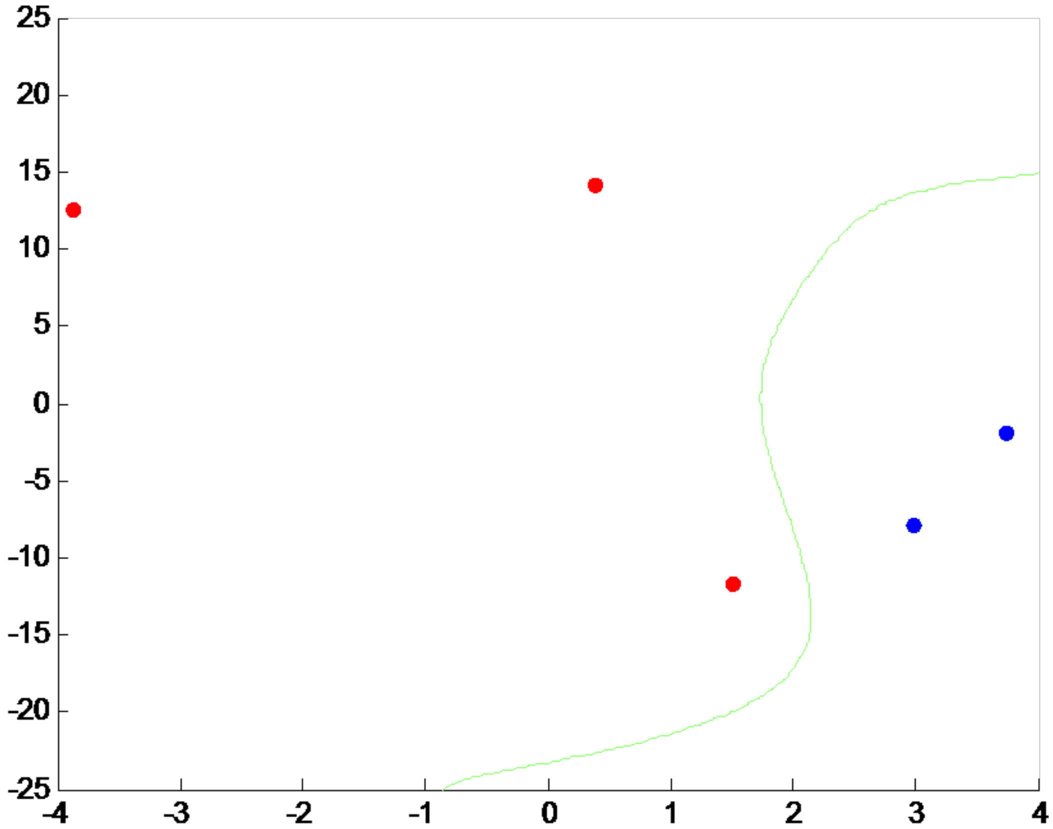
Estimated decision boundary for tiny data sample 3



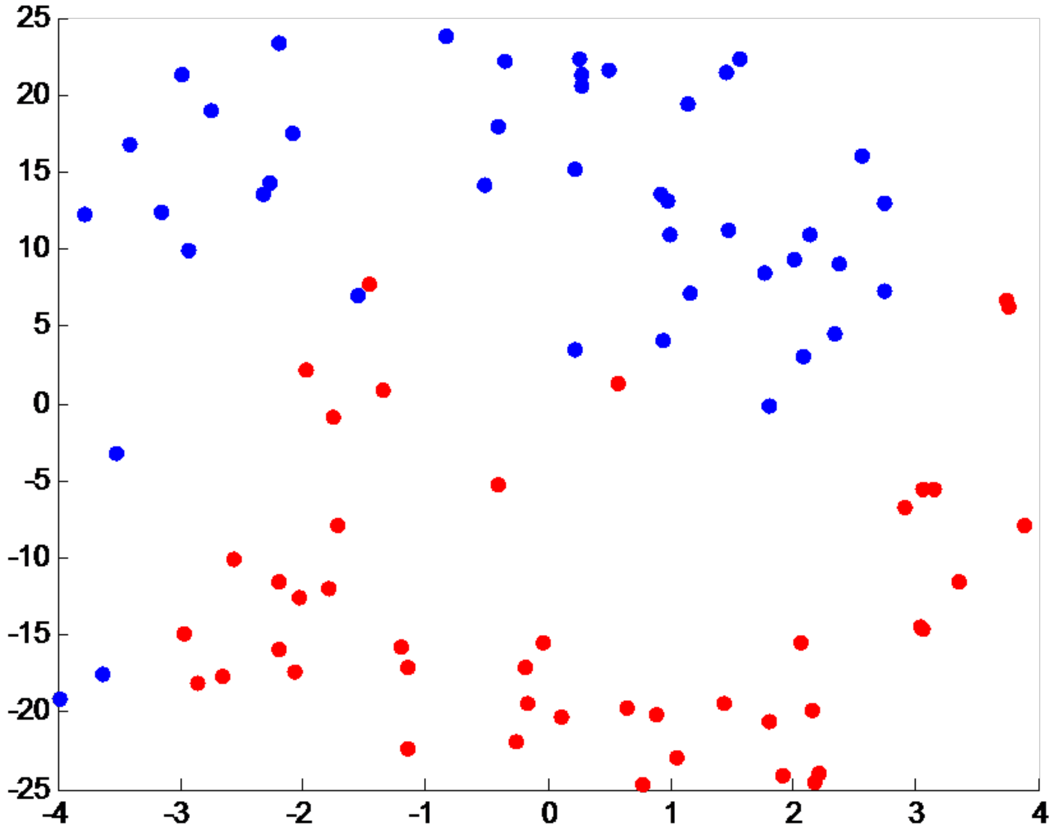
Estimated decision boundary for tiny data sample 4



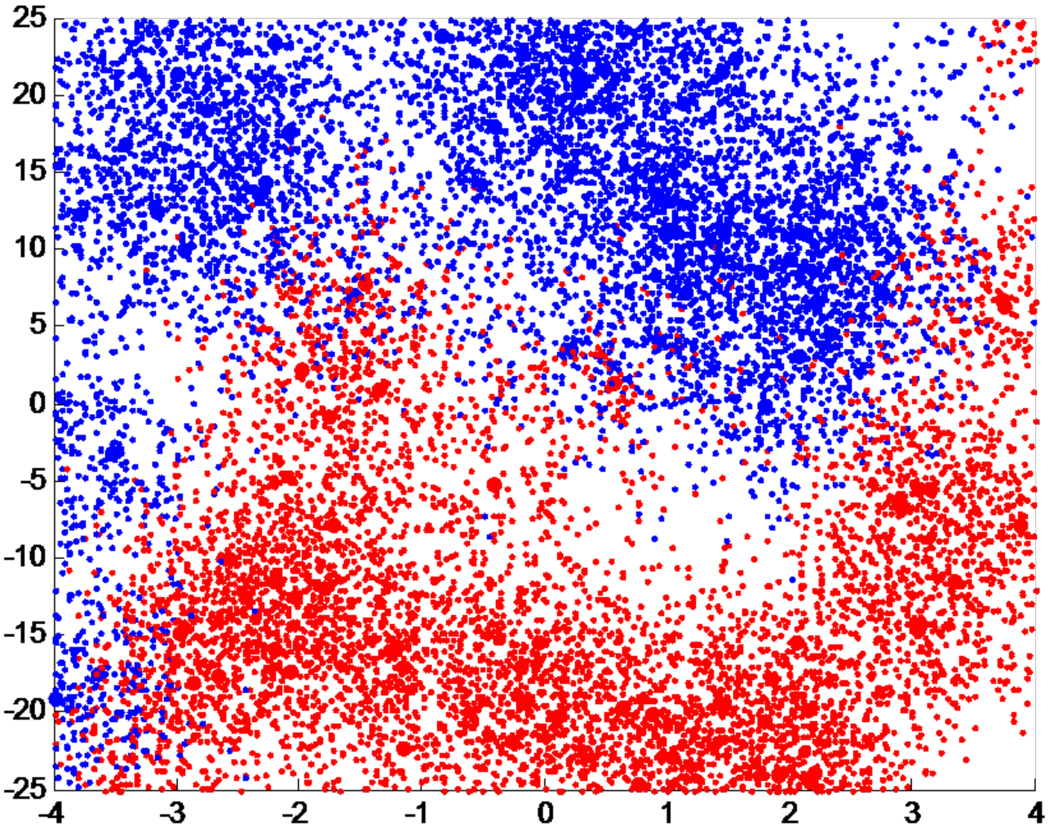
Estimated decision boundary for tiny data sample 5



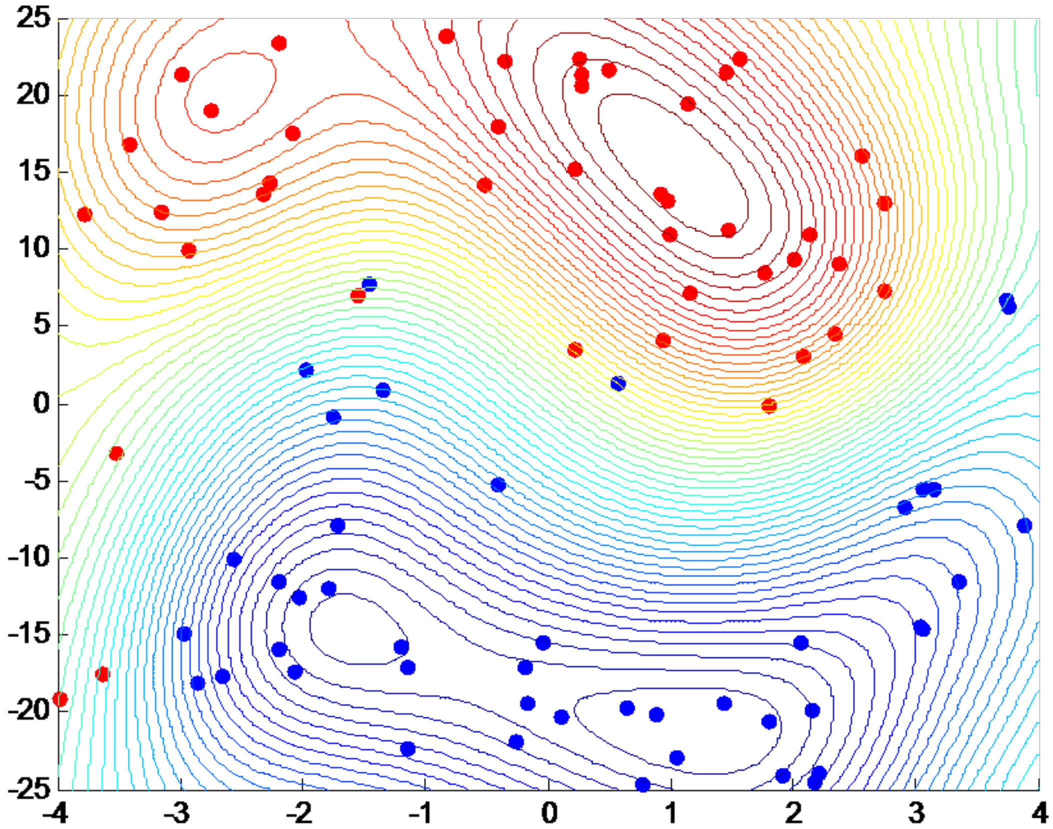
A larger data sample



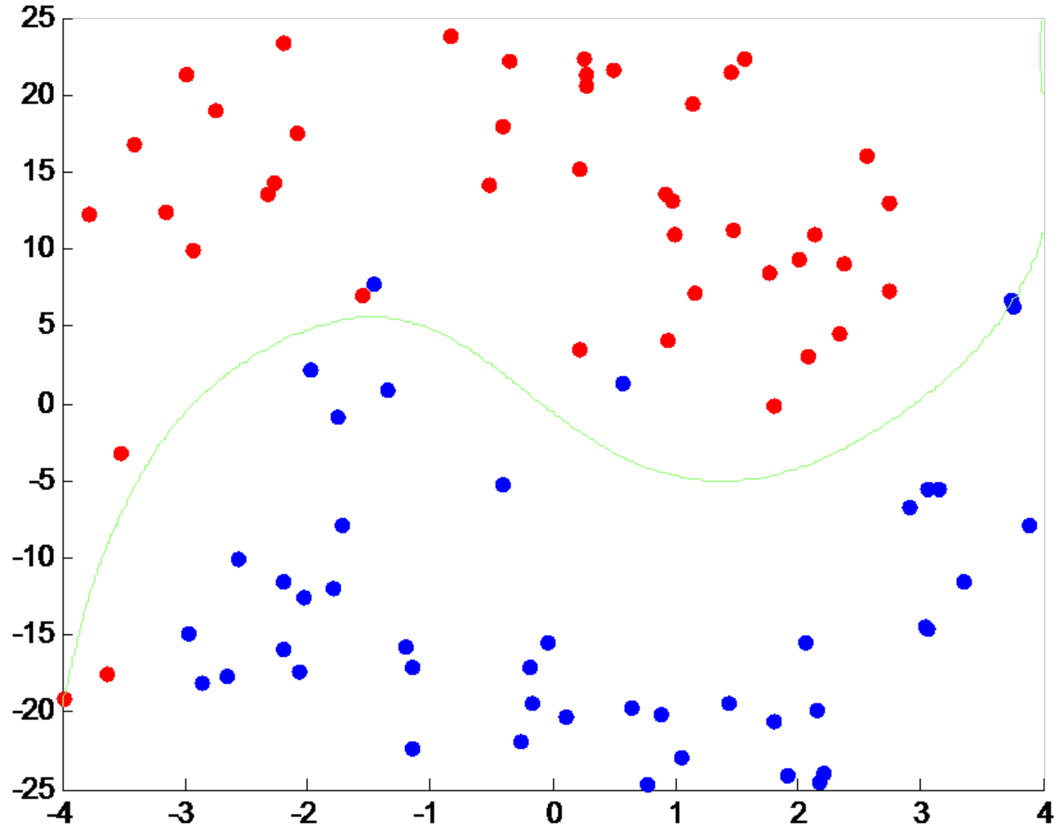
Placing an RBF kernel at each sample point



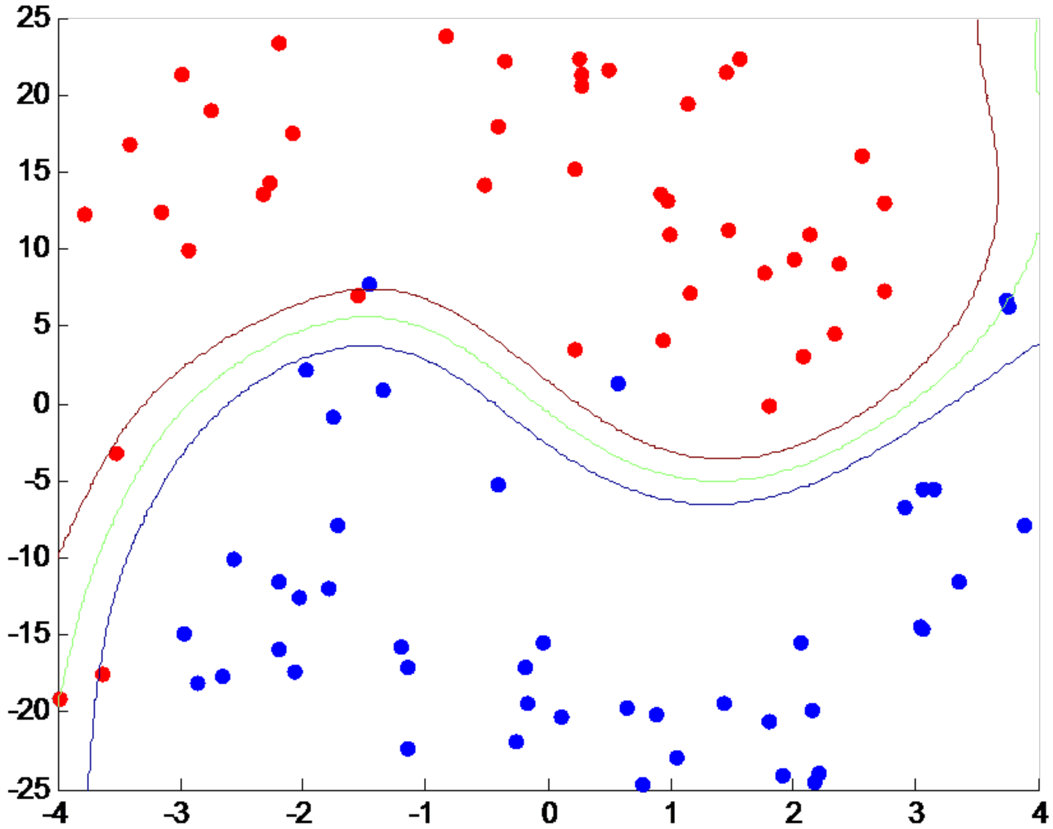
Contour plot of the sum of the kernel values



Estimated decision boundary: level 0 contour



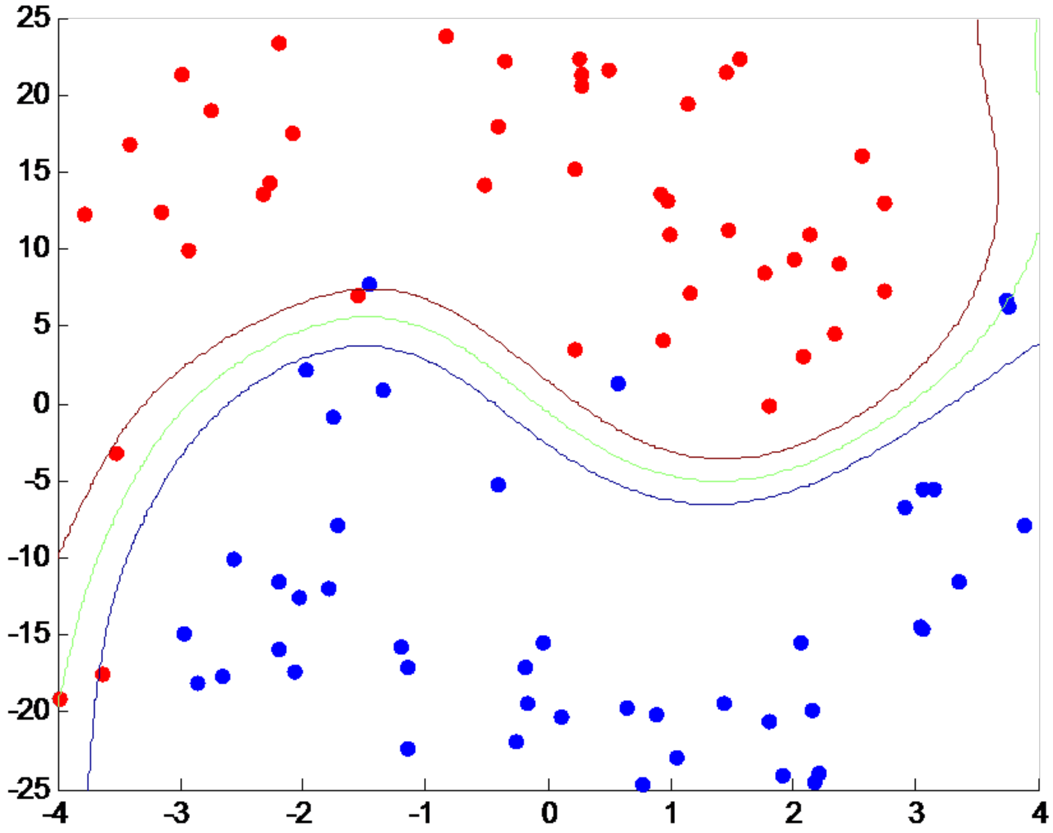
Decision boundary and margins: three contours



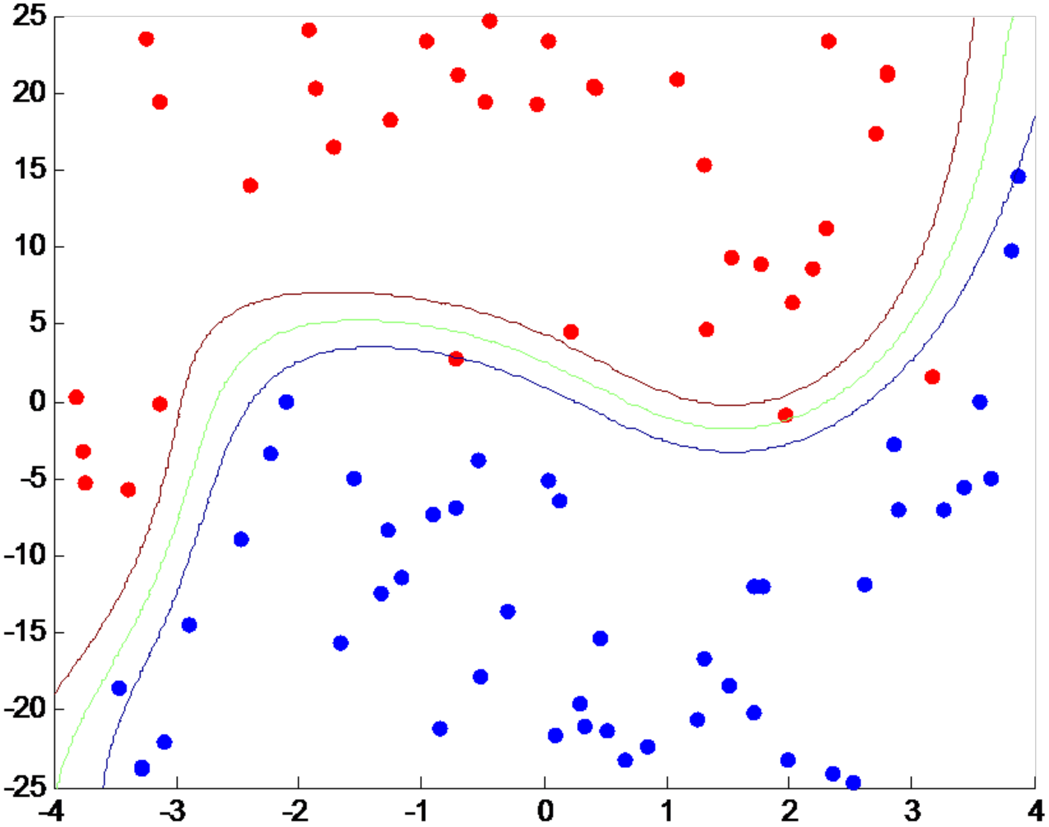
Observation

- The estimated decision boundaries and margins still depend on the data sample.
- But, because of the larger sample size, the estimates are less sensitive to changes in the sample.
- That is, different data samples give roughly similar estimates.
- (Very large data samples would give very similar estimates.)

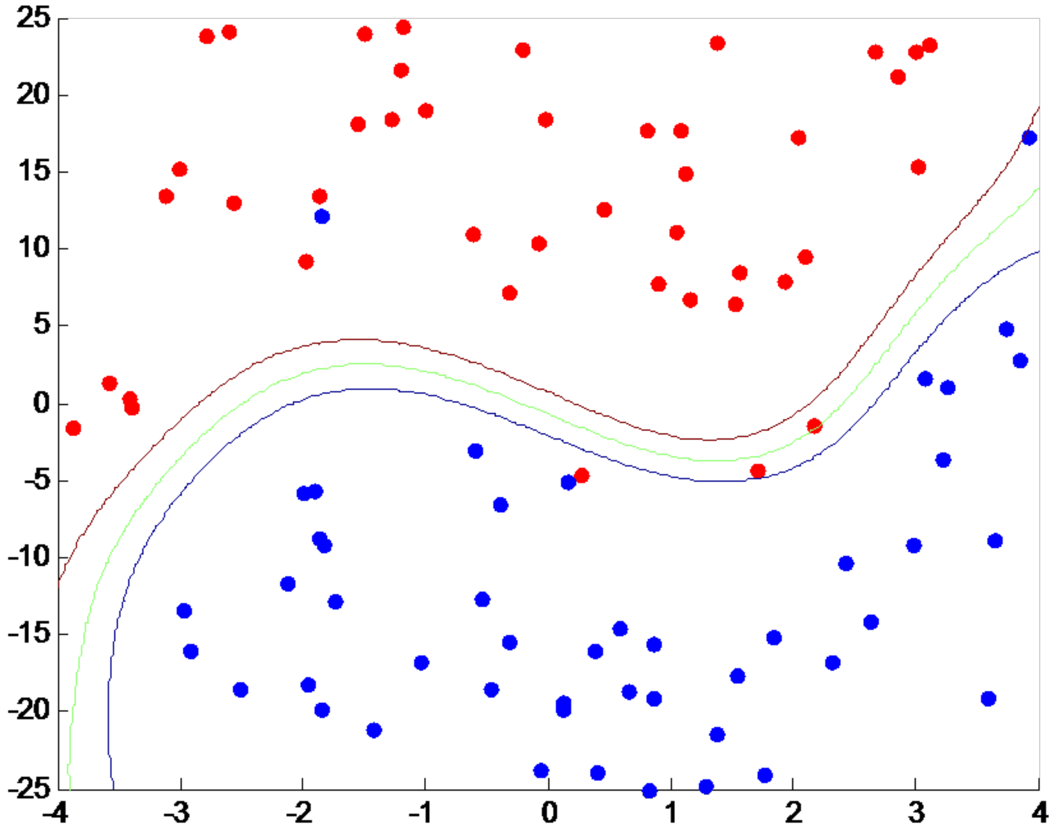
Decision boundary and margins for data sample 1



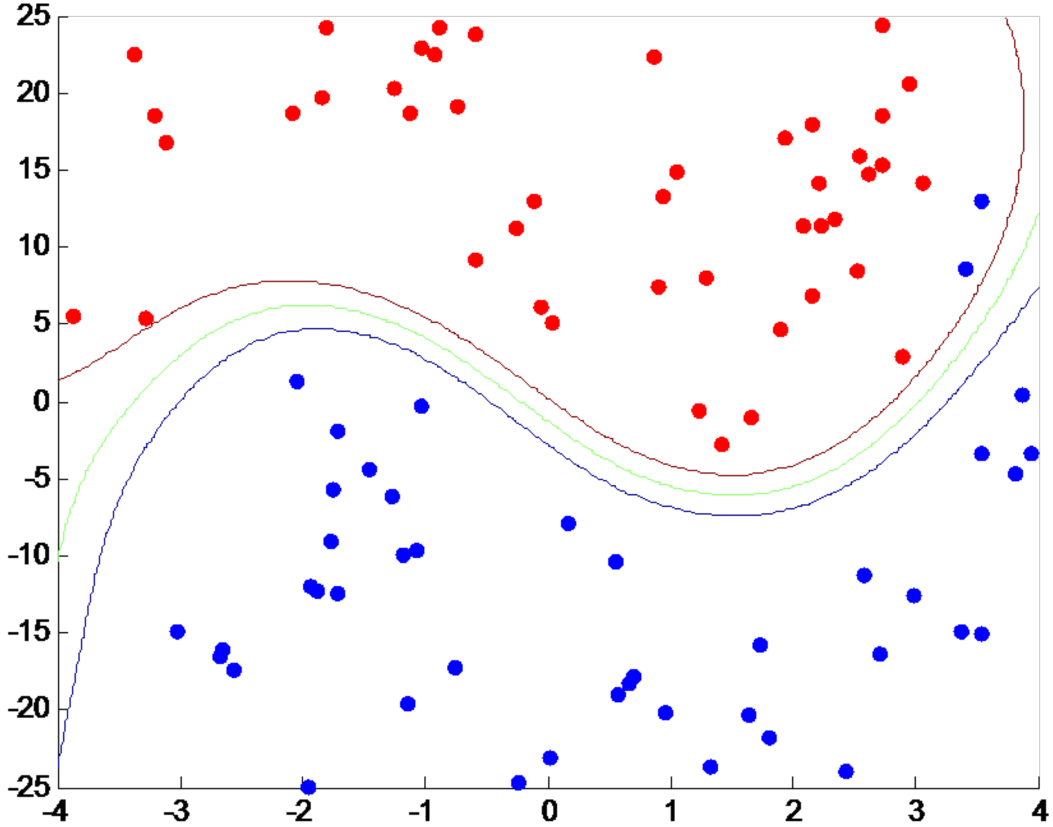
Decision boundary and margins for data sample 2



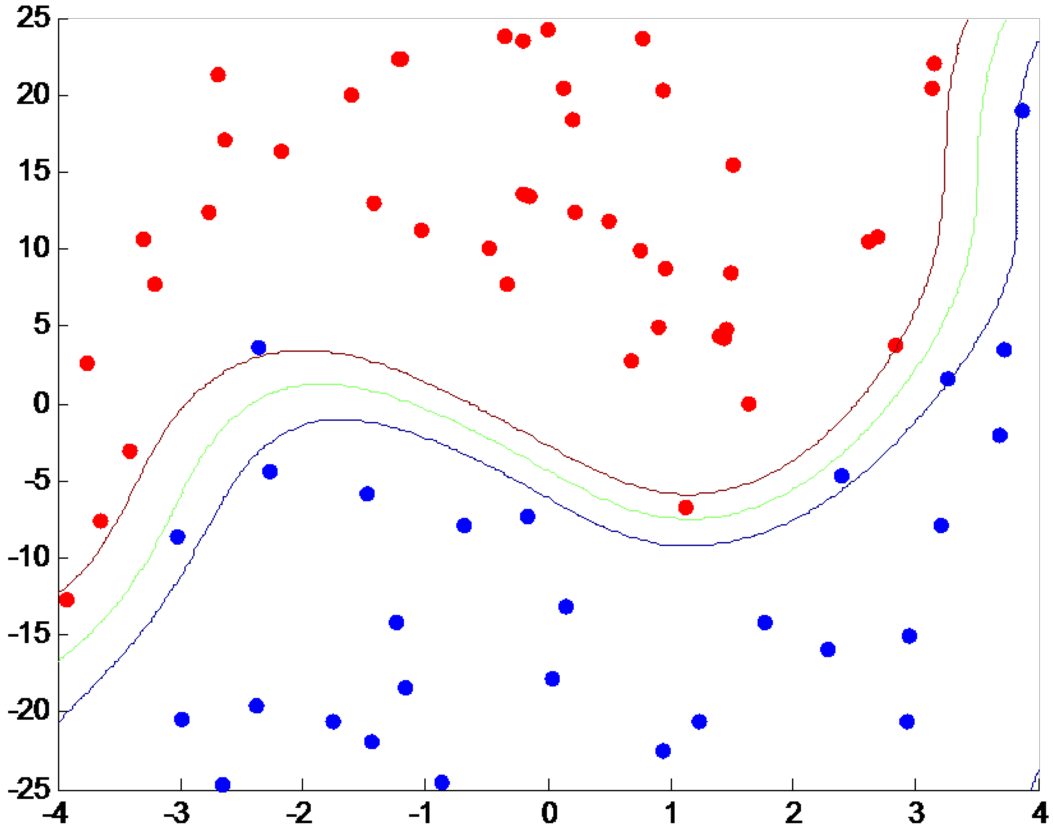
Decision boundary and margins for data sample 3



Decision boundary and margins for data sample 4



Decision boundary and margins for data sample 5



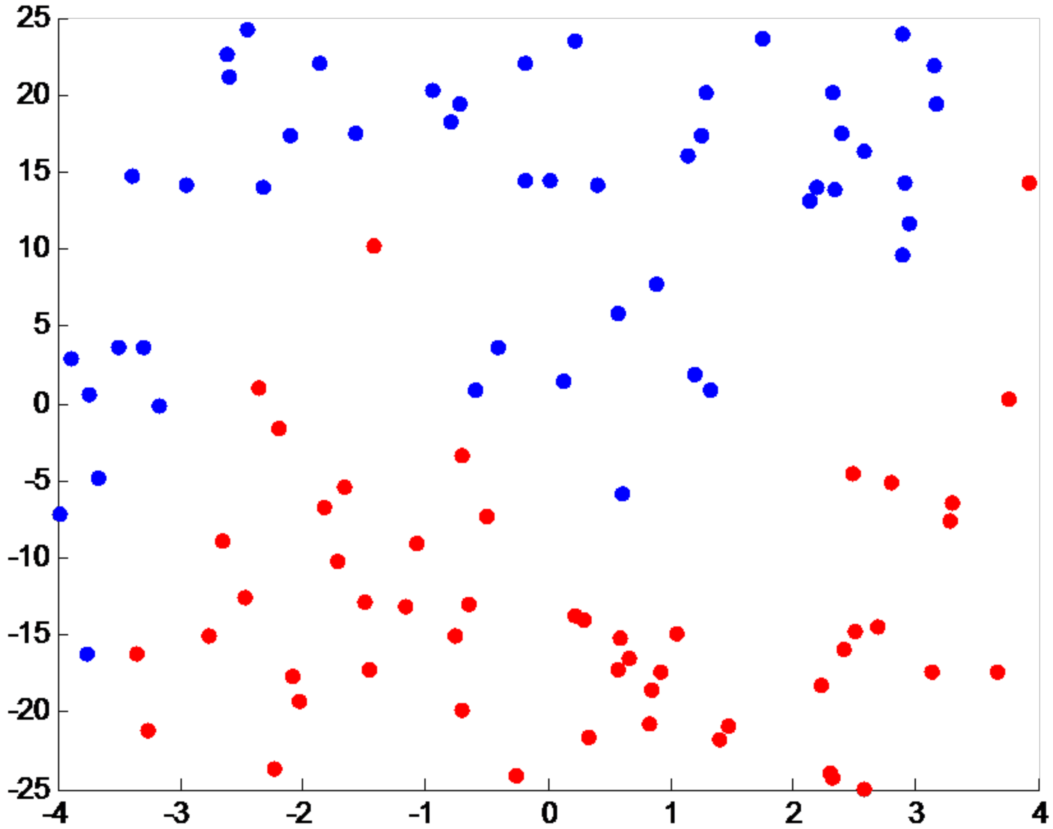
Support vector machines

- They are similar to the simple kernel method just described.
- However, the contour plot comes from a *weighted* sum of kernel values (instead of just a simple sum).
- An SVM determines the *optimal* values of the weights.
- The optimal weights minimize the variance of the decision boundary (i.e., its sensitivity to changes in the data sample).

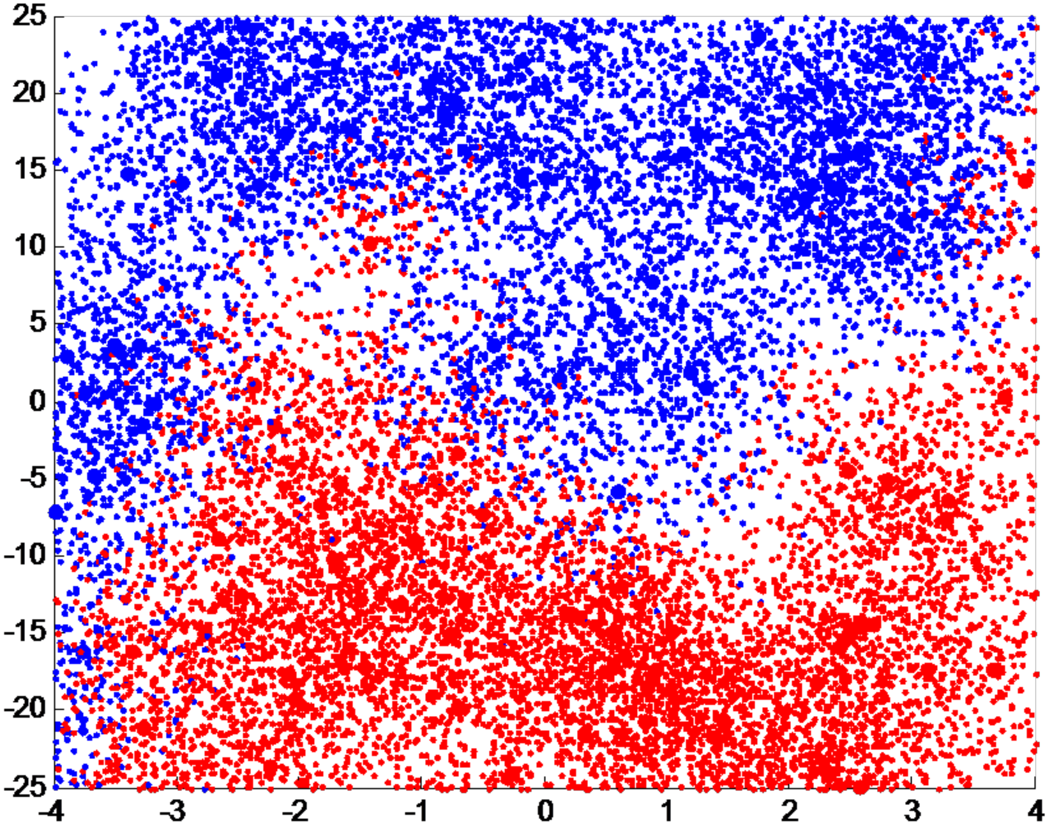
Representer Theorem

- Why place kernels only at the sample points?
- Why not place kernels at other points as well?
- What if we placed kernels at an infinite number of points?
- Couldn't we get a better estimate of the decision boundary this way?
- As we shall see, the answer is NO.
- THEOREM: under a wide range of conditions, placing kernels only at the sample points gives the best estimates (chapter 4).

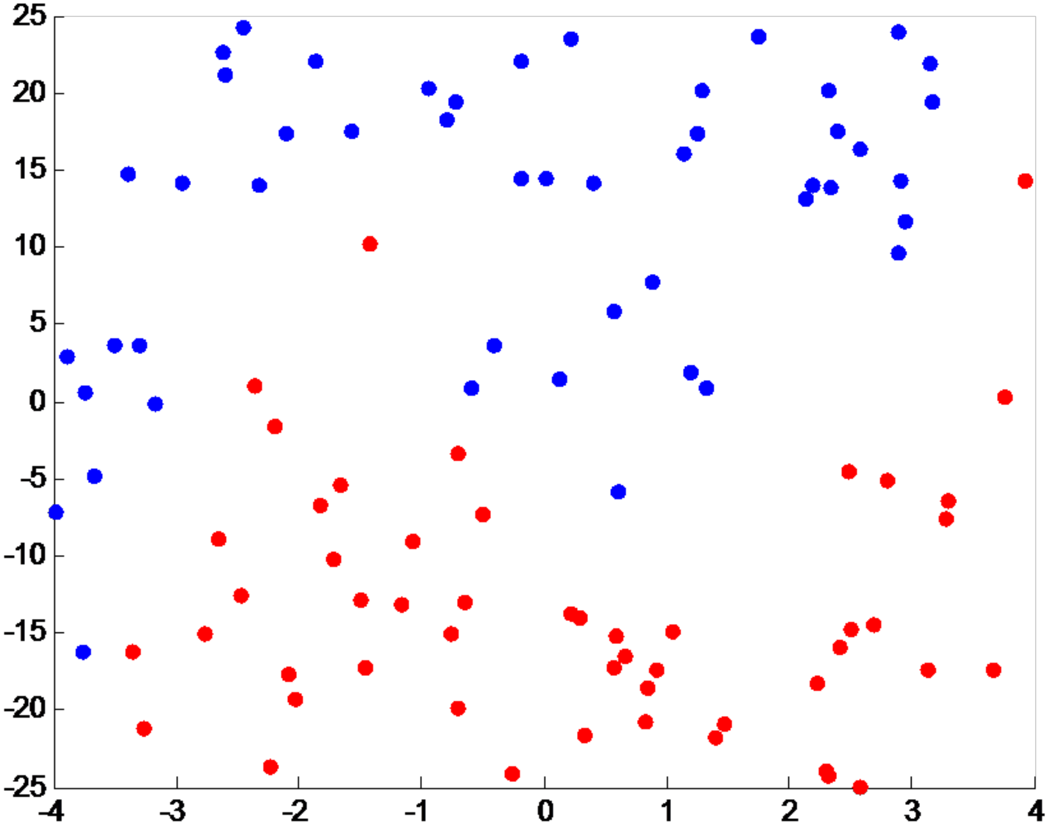
What we have just seen: placing a kernel on each sample point



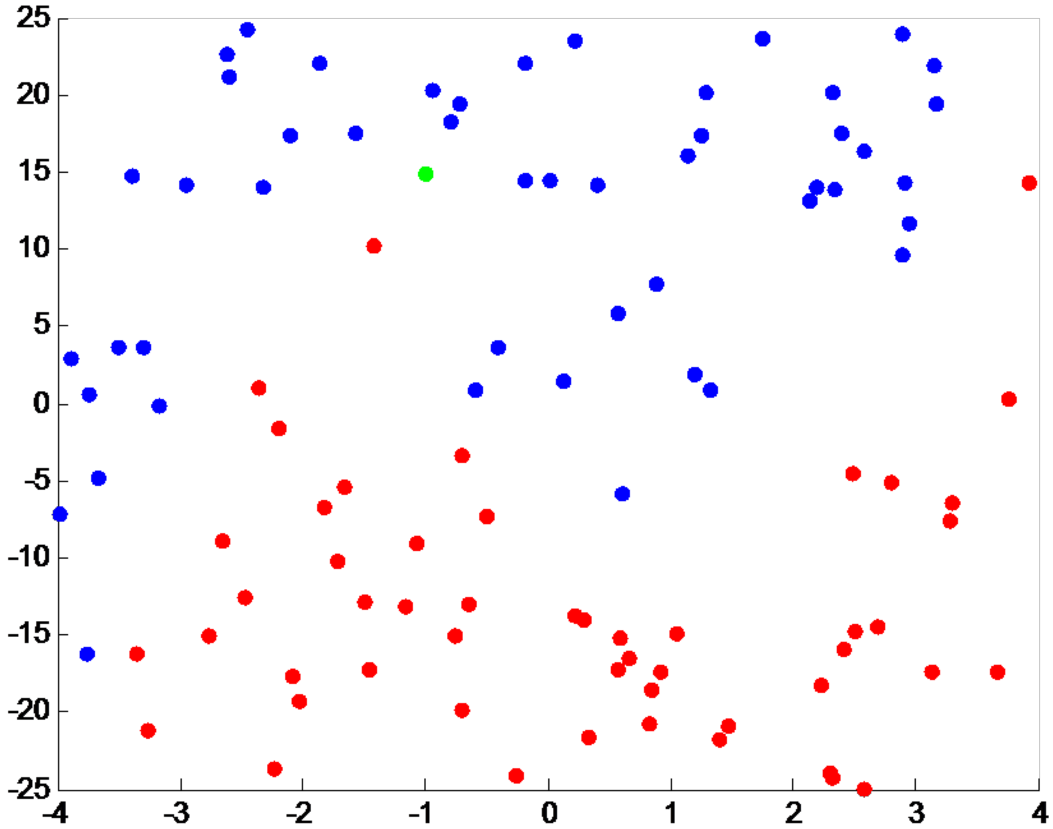
What we have just seen: placing a kernel on each sample point



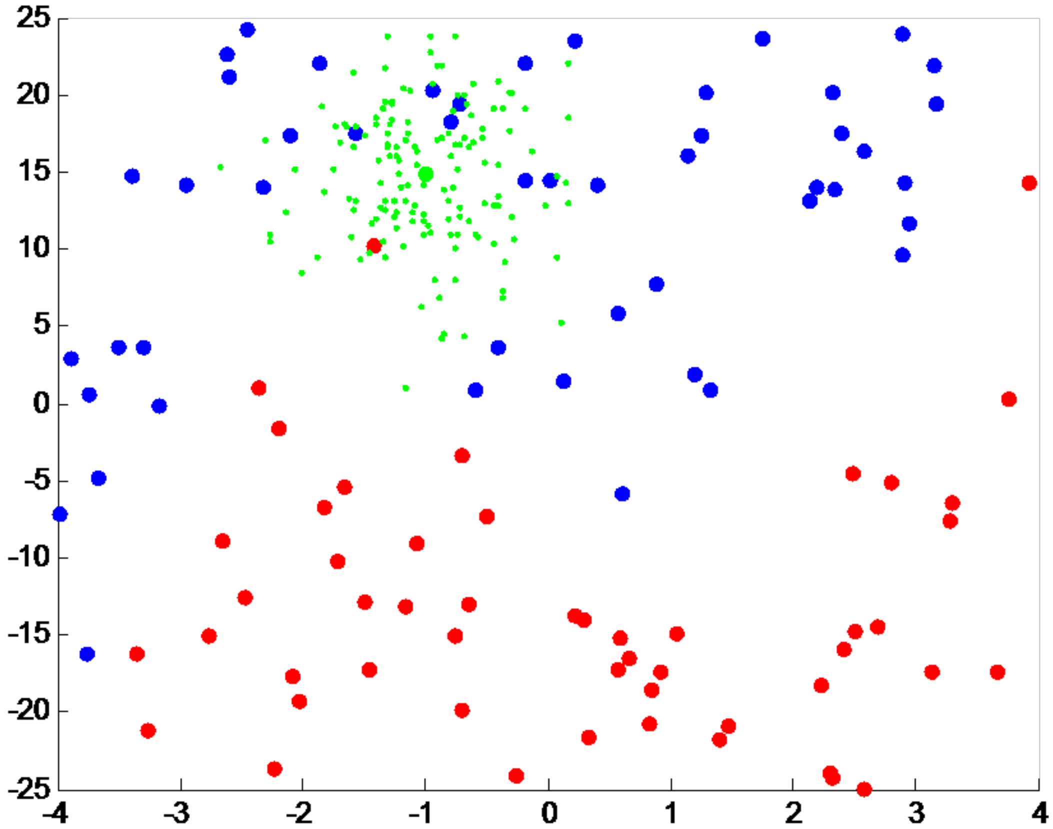
An alternate interpretation: placing a kernel on the test point



An alternate interpretation: placing a kernel on the test point



An alternate interpretation: placing a kernel on the test point



Function spaces

- Vector spaces
  - Functions as vectors
- Inner product spaces
  - Inner products of functions
- Hilbert spaces
  - Infinite-dimensional spaces
- Linear Operators
  - Eigen functions

Vector Spaces (Appendix B.2.1)

- A vector space is a set that is closed under finite linear combinations.
- Basic properties:
  - Linear independence
  - Spanning sets
  - Basis
  - Dimension

Examples of Vector Spaces

- k-tuples
- infinite sequences
- matrices (of given dimension)
- polynomials
- polynomials of degree at most k
- real functions
- continuous functions
- linear combinations of trigonometric functions

Some Important Vector Spaces for this course

- ℓ_2 square-summable sequences
- $L_2[a,b]$ square-integrable functions on [a,b]
- $C[a,b]$ continuous functions on [a,b]

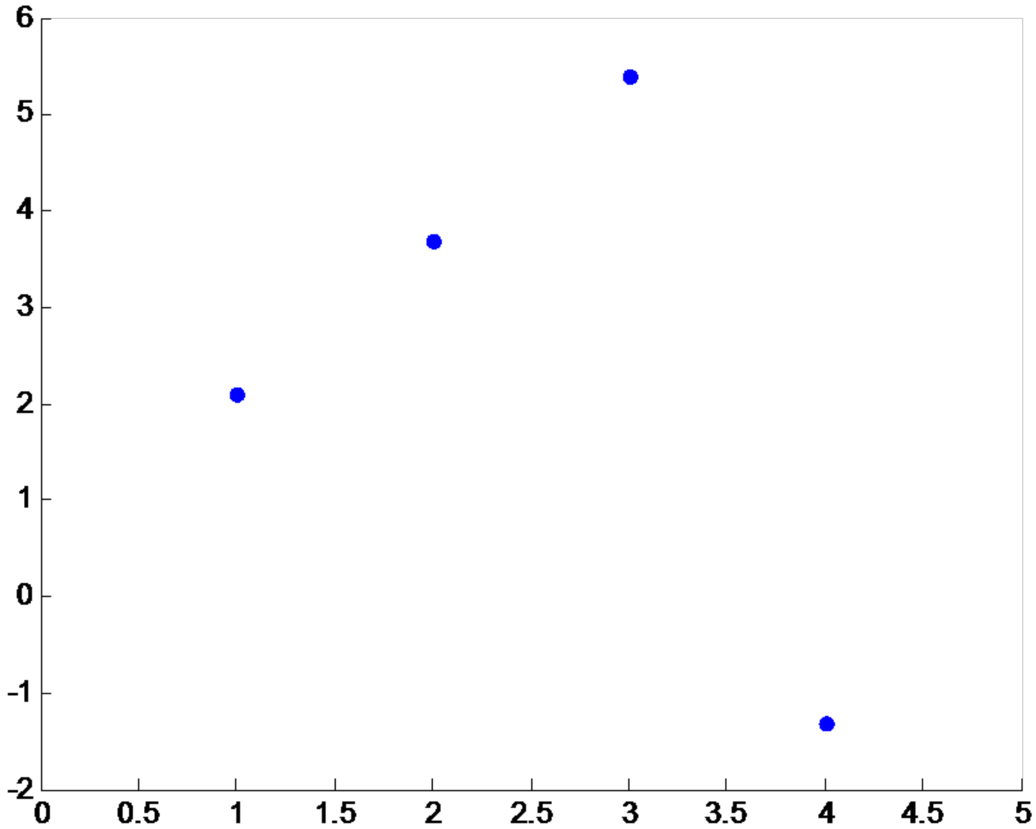
Vectors as functions

- Most common vectors are functions.
- They map an index set to real numbers.
- For example, the tuple v = (2.1, 3.7, 5.4, -1.3) maps the set {1,2,3,4} to real numbers, where
  - v(1) = 2.1
  - v(2) = 3.7
  - v(3) = 5.4
  - v(4) = -1.3

Vectors as functions

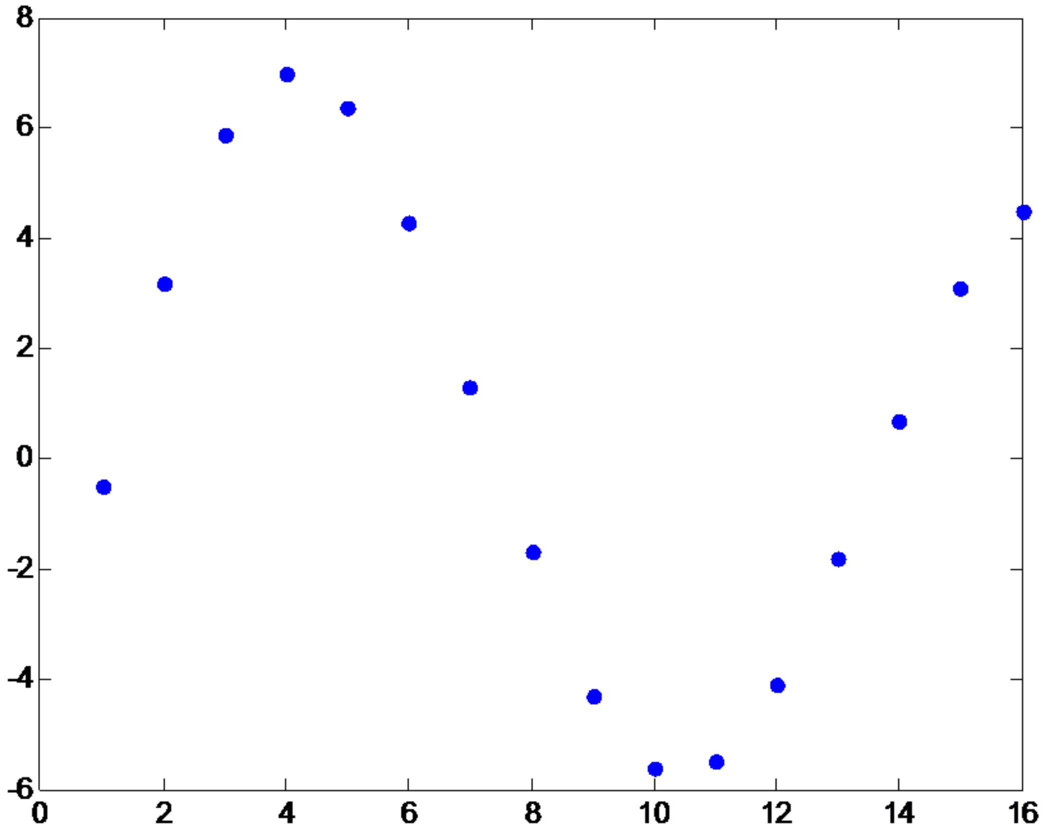
- The infinite sequence v = (1,4,9,16,25,36,...) maps the natural numbers to real numbers, where $v(n) = n^2$.
- Of course, the vector space of polynomials is clearly made up of functions.
- Likewise for other function spaces.
- All such vectors can be plotted as functions.

The vector (2.1, 3.7, 5.4, -1.3)



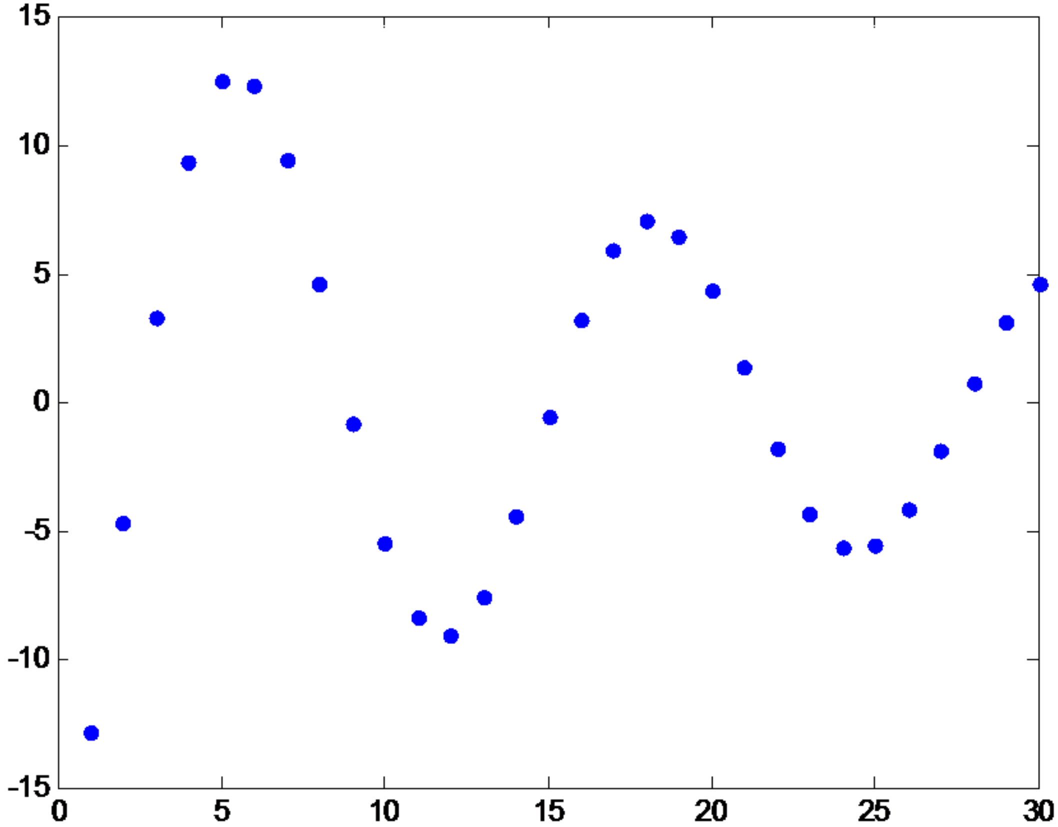
The vector

(-0.5 3.2 5.9 7.0 6.4 4.3 1.3 -1.7 -4.3 -5.6 -5.5 -4.1 -1.8 0.7 3.1 4.5)

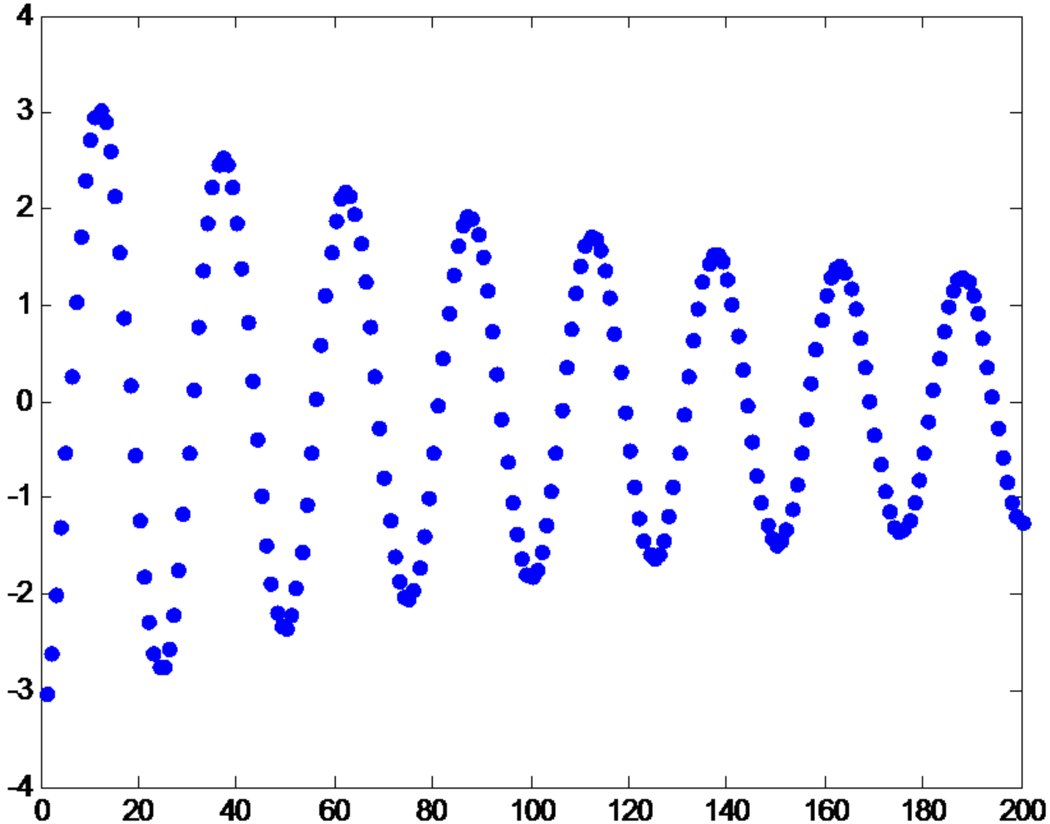


The vector

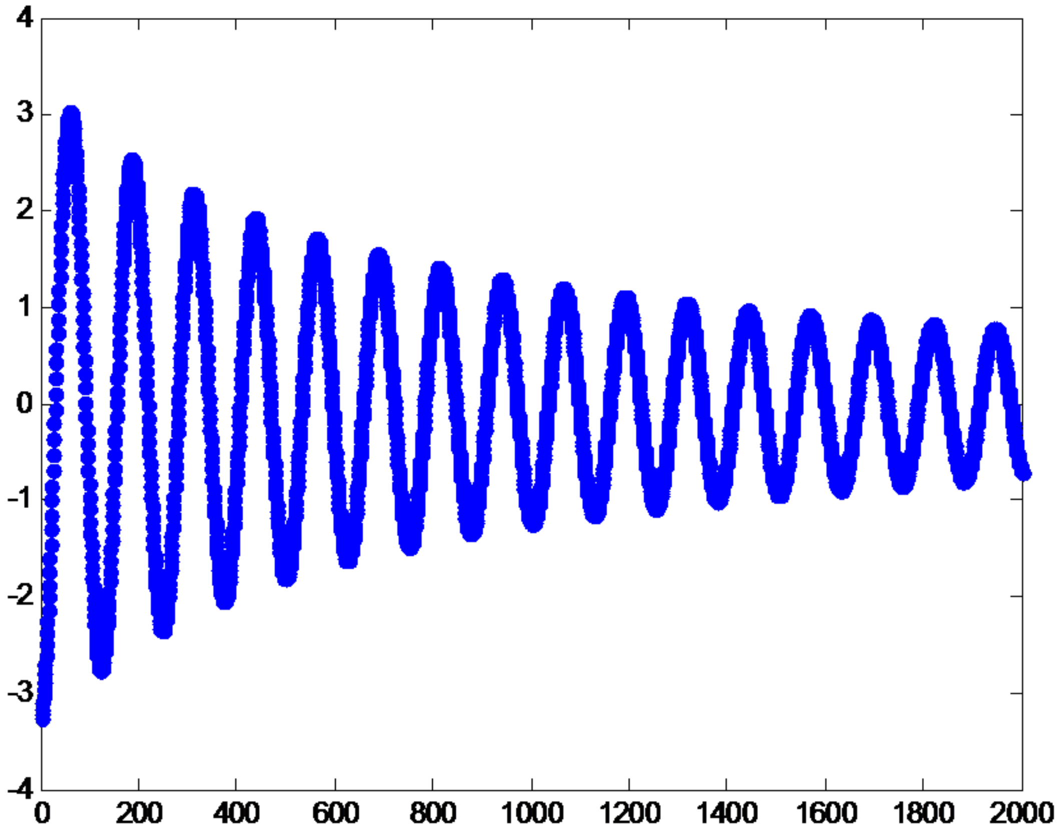
(-12.8 -4.6 3.3 9.3 12.5 12.3 9.3 4.5 -0.7 -5.4 -8.3 -9.0 -7.6 -4.4 -0.5 3.2 5.9 7.0 6.4 4.3 1.3 -1.7 -4.3 -5.6 -5.5 -4.1 -1.8 0.7 3.1 4.5)



A vector of dimension 200



A vector of dimension 2,000



Inner Product Spaces (Appendix B.2.2)

- An inner product space is a vector space on which an inner product is defined.
- An inner product is a function of two arguments that is
  - linear in each argument
  - symmetric
  - positive definite

Geometric Properties of Inner Products

- Cauchy-Schwarz inequality
- Angle
- Orthogonality
- Length
- Triangle inequality
- Pythagorean theorem
- Projection
- Orthonormal bases

Hilbert Spaces (Appendix B.3)

- A Hilbert space is an inner product space that contains all its limit points (cluster points).
- A limit point can be viewed as:
  - a “hole” in a vector space
  - the solution to an optimization problem
  - an infinite linear combination of other points

Examples

- The real numbers are a Hilbert space.
- The rational numbers are *not* a Hilbert space.
- Finite-dimensional real vector spaces are Hilbert spaces.
- C[a,b] is *not* a Hilbert space.
- ℓ_2 and $L_2[a,b]$ are Hilbert spaces.

Optimization

The solution to an optimization problem is a limit point:

– If x is the optimal solution, then there are non-optimal solutions arbitrarily close to x.

– Thus, there is a sequence of non-optimal solutions, x_1, x_2, x_3, ..., that converges to x.

– x is therefore a limit point.

Infinite Linear Combinations

THEOREM:

A point is a limit point iff it is an infinite linear combination of other points.

COROLLARY:

An inner product space is a Hilbert space iff it is closed under infinite linear combinations.

SVM Feature Space

- Making feature space a Hilbert space means
  - it does not have “holes”
  - we can solve optimization problems (e.g., maximizing a margin)
  - we can take limits (as in Euclidean space)
  - SVMs are more powerful than we might have thought, because of infinite linear combinations.
- Also, Hilbert spaces are easy to construct!

Completion

THEOREM:

Any inner product space can be "completed" to form a Hilbert space.

Intuitively, this is done by adding the limit points to the space or by closing it under infinite linear combinations.

Theory of kernels

- Positive definite kernels
- Reproducing kernel map
- Linear operators
- Mercer kernel map

Positive Definite Kernels

It can be shown that (modulo some details) the admissible class of kernels coincides with the one of positive definite (pd) kernels: kernels which are symmetric, and for

- any set of training points $x_1, \ldots, x_m \in \mathcal{X}$ and
- any $a_1, \ldots, a_m \in \mathbb{R}$

satisfy

$$\sum_{i,j} a_i a_j K_{ij} \geq 0, \quad \text{where } K_{ij} := k(x_i, x_j).$$

Elementary Properties of PD Kernels

Kernels from Feature Maps.
If Φ maps $\mathcal{X}$ into a dot product space $\mathcal{H}$, then $\left\langle \Phi(x), \Phi(x') \right\rangle$ is a pd kernel on $\mathcal{X} \times \mathcal{X}$.

Positivity on the Diagonal.
$k(x, x) \geq 0$ for all $x \in \mathcal{X}$

Cauchy-Schwarz Inequality.
$k(x, x')^2 \leq k(x, x)k(x', x')$ (Hint: compute the determinant of the Gram matrix)

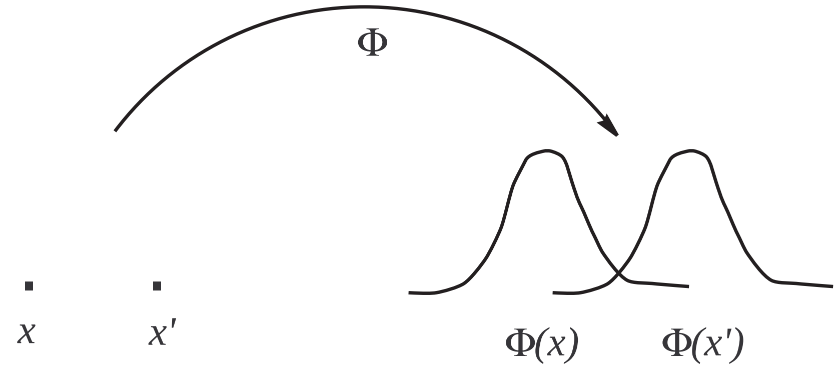
Vanishing Diagonals.
$k(x, x) = 0$ for all $x \in \mathcal{X} \Longrightarrow k(x, x') = 0$ for all $x, x' \in \mathcal{X}$

The Feature Space for PD Kernels [4, 1, 48]

- define a feature map

$$\Phi : \mathcal{X} \to \mathbb{R}^{\mathcal{X}}$$
$$x \mapsto k(.,x).$$

E.g., for the Gaussian kernel:



Next steps:

- turn $\Phi(\mathcal{X})$ into a linear space
- endow it with a dot product satisfying $\left\langle k(.,x_i), k(.,x_j)\right\rangle = k(x_i, x_j)$
- complete the space to get a *reproducing kernel Hilbert space*

Turn it Into a Linear Space

Form linear combinations

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

$$g(.) = \sum_{j=1}^{m'} \beta_j k(., x'_j)$$

$(m, m' \in \mathbb{N},\ \alpha_i, \beta_j \in \mathbb{R},\ x_i, x'_j \in \mathcal{X})$.

Endow it With a Dot Product

$$\langle f, g \rangle := \sum_{i=1}^{m} \sum_{j=1}^{m'} \alpha_i \beta_j k(x_i, x'_j)$$

$$= \sum_{i=1}^{m} \alpha_i g(x_i) = \sum_{j=1}^{m'} \beta_j f(x'_j)$$

- This is well-defined, symmetric, and bilinear.
- It can be shown that it is also strictly positive definite (hence it is a dot product).
- Complete the space in the corresponding norm to get a Hilbert space $\mathcal{H}_k$.

The Reproducing Kernel Property

Two special cases:

- Assume
$$f(.) = k(., x).$$
In this case, we have
$$\langle k(., x), g \rangle = g(x).$$

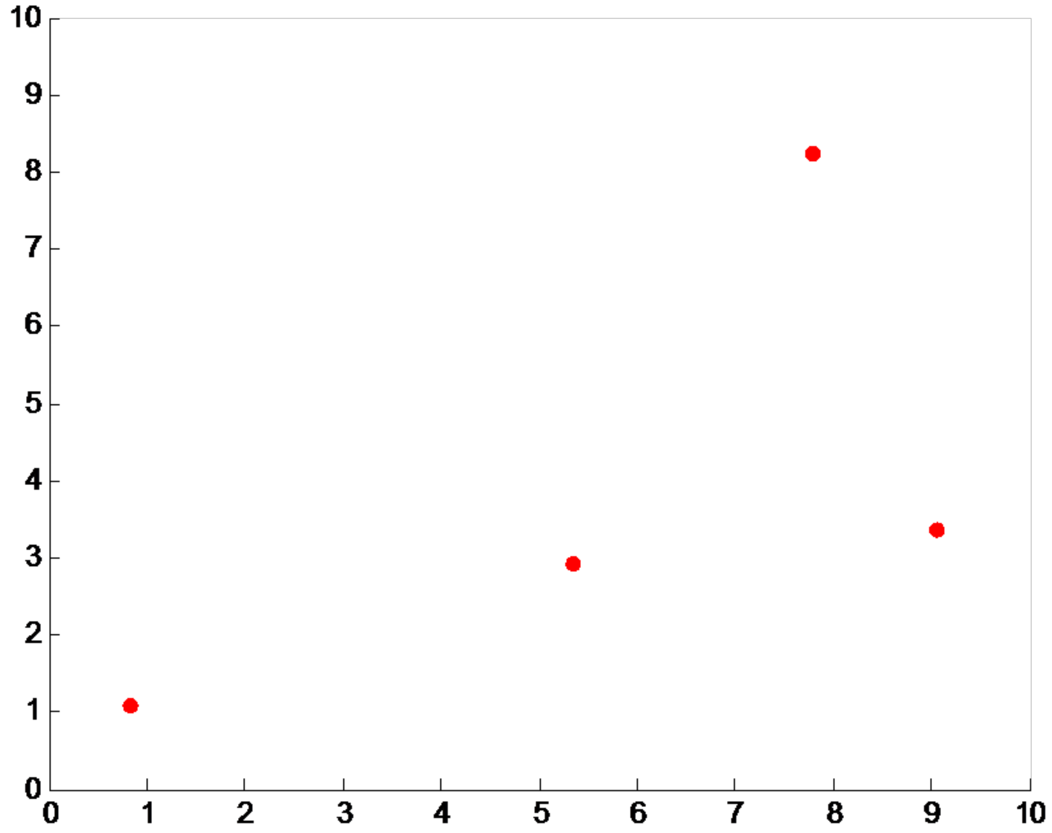
- If moreover
$$g(.) = k(., x'),$$
we have the kernel trick
$$\langle k(., x), k(., x') \rangle = k(x, x').$$

k is called a *reproducing kernel* for $\mathcal{H}_k$.

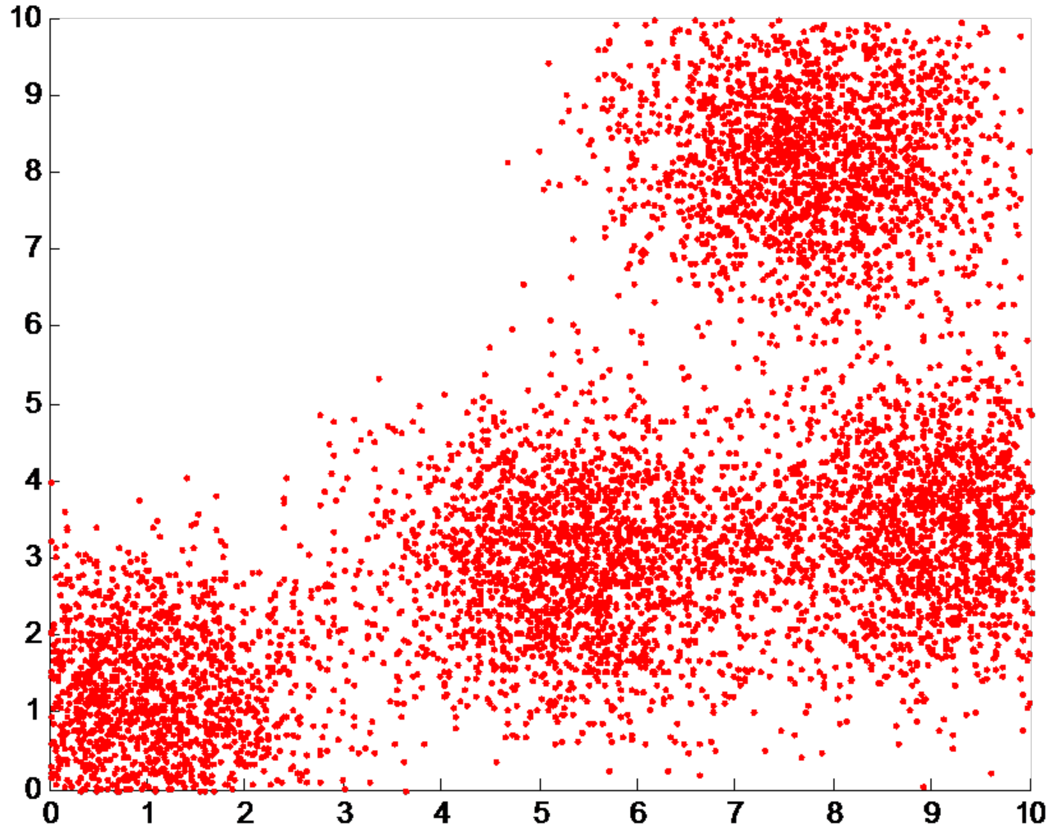
Feature space: linear combinations of kernels

- Each point in feature space is a function constructed as follows:
  - Select some points from input space.
  - Put a kernel on each point.
  - Assign a weight to each kernel.
  - Add up the weighted kernels.
- The contours of this function are potential decision boundaries.

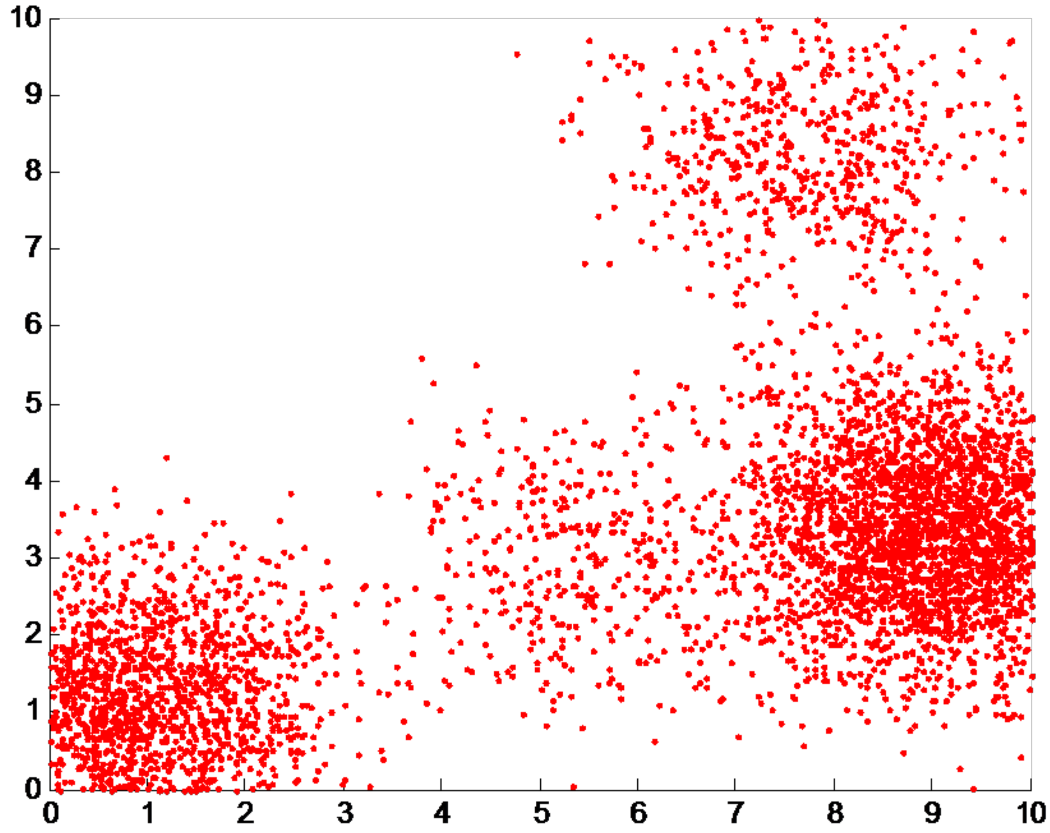
Some points in input space



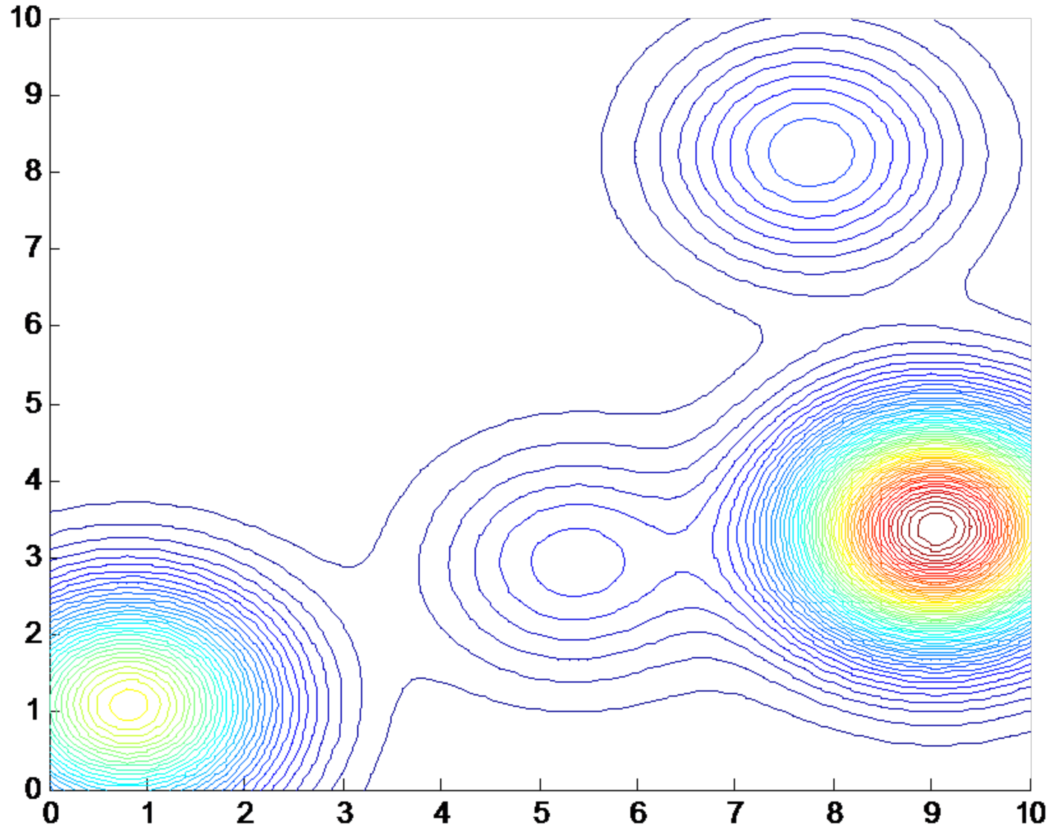
Placing a kernel on each point



Weighting the kernels



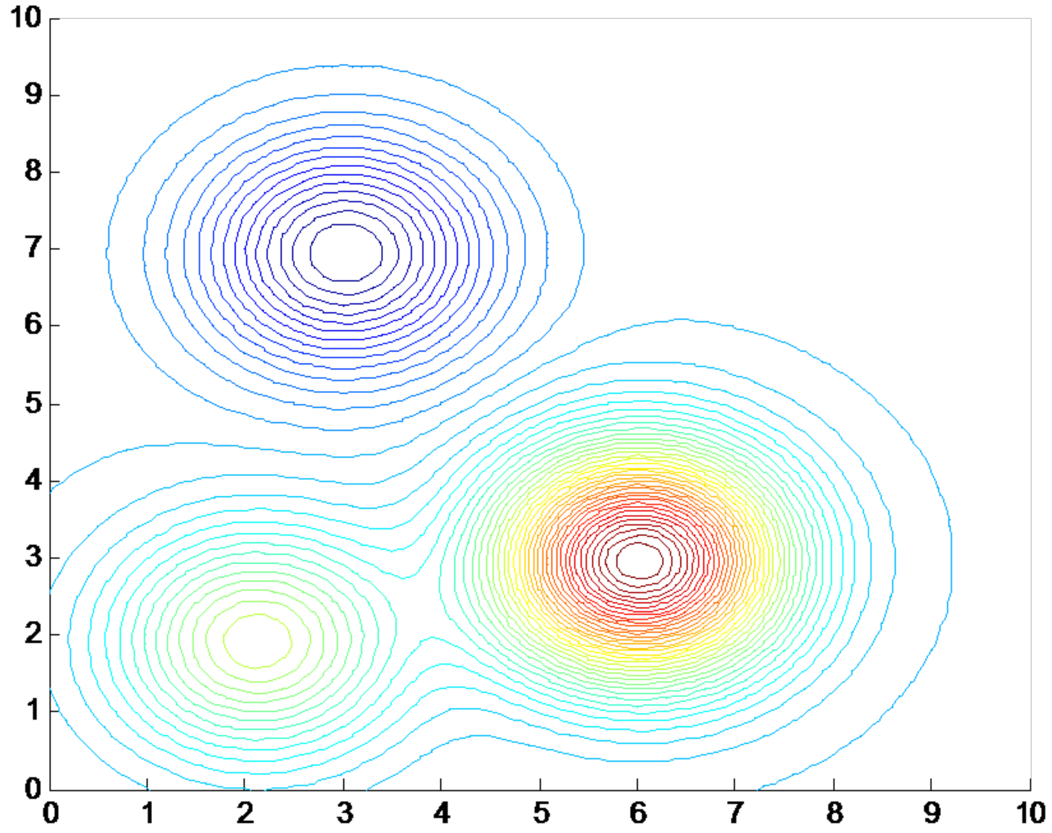
Summing the weighted kernels



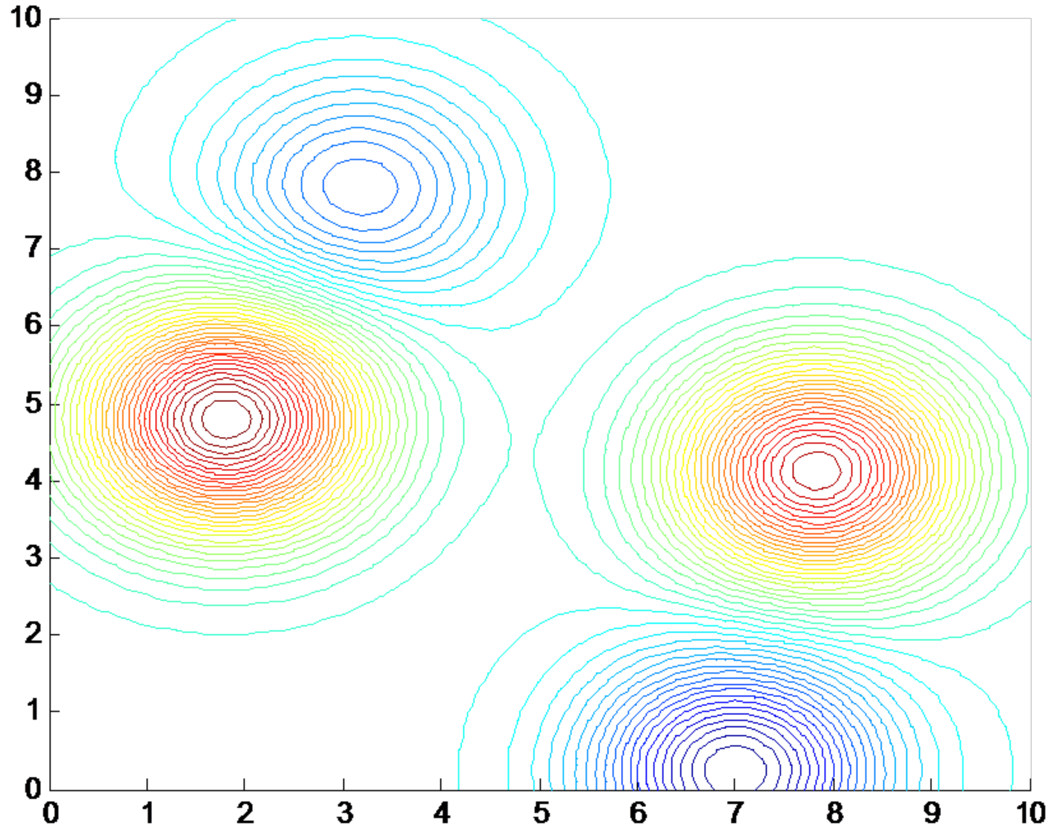
Feature space in pictures

- The previous slide is a contour plot of a function.
- This function is a linear combination of kernels.
- The function is therefore a vector in feature space.
- The next several slides show different contour plots, each representing a different vector in feature space.

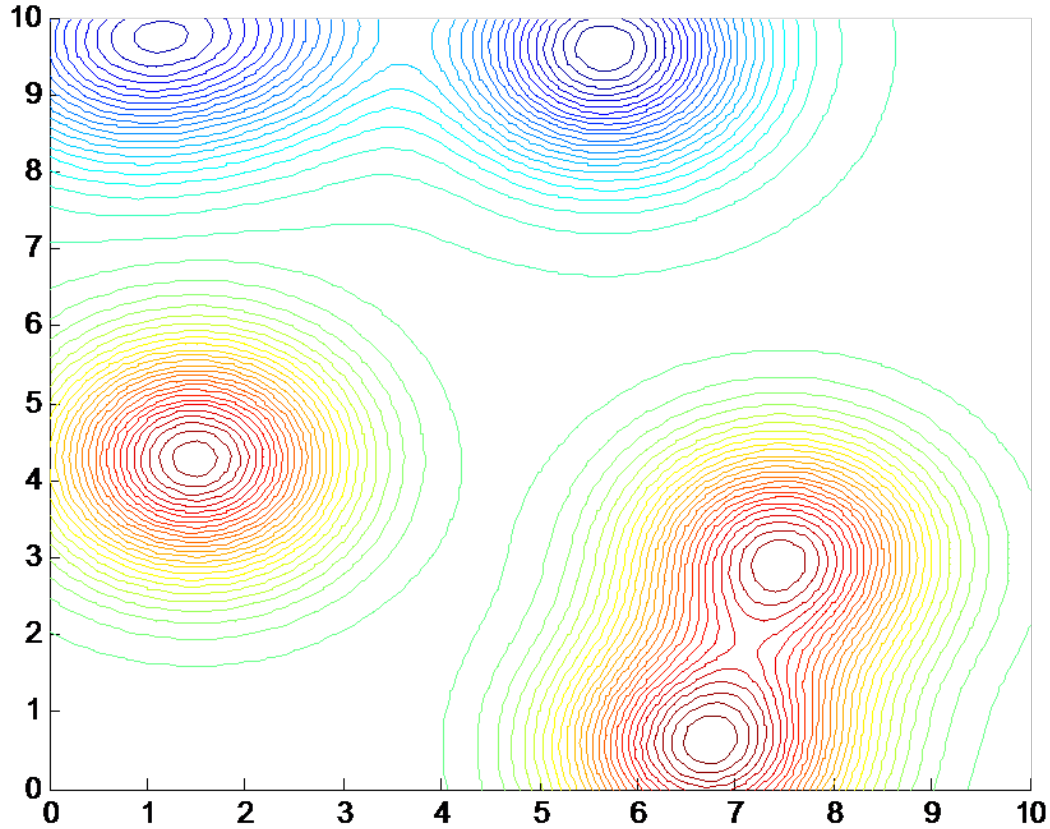
A linear combination of 3 kernels



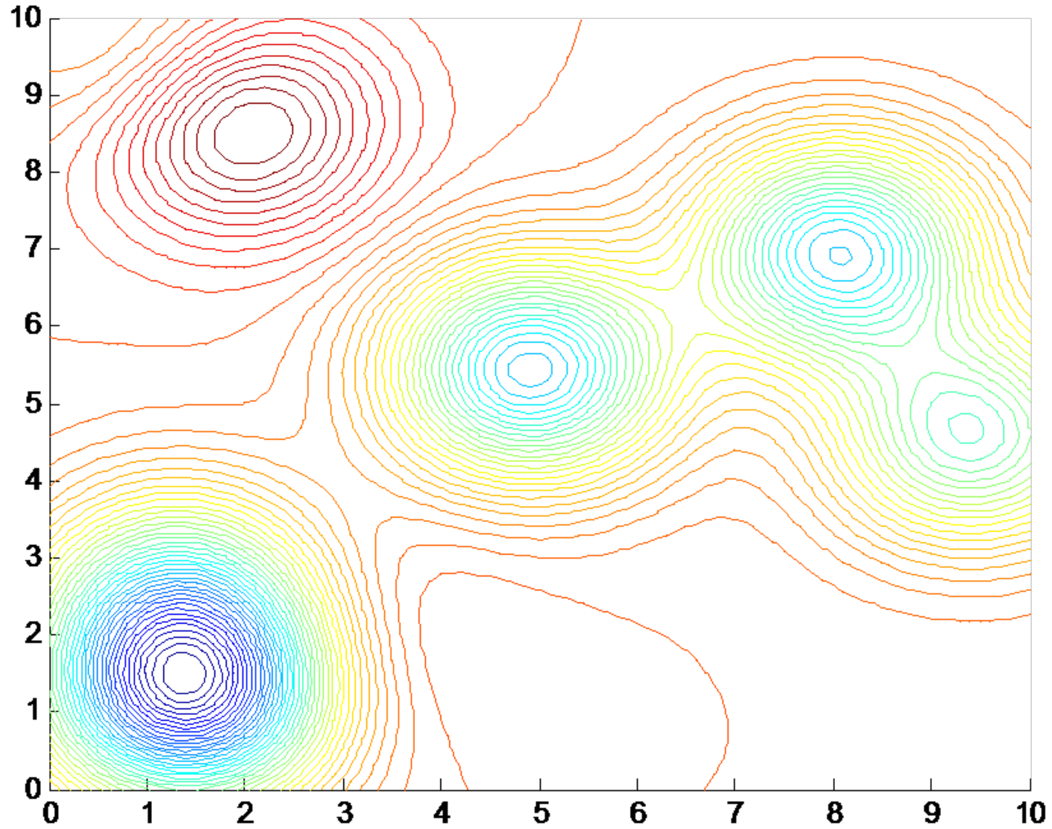
A linear combination of 4 kernels



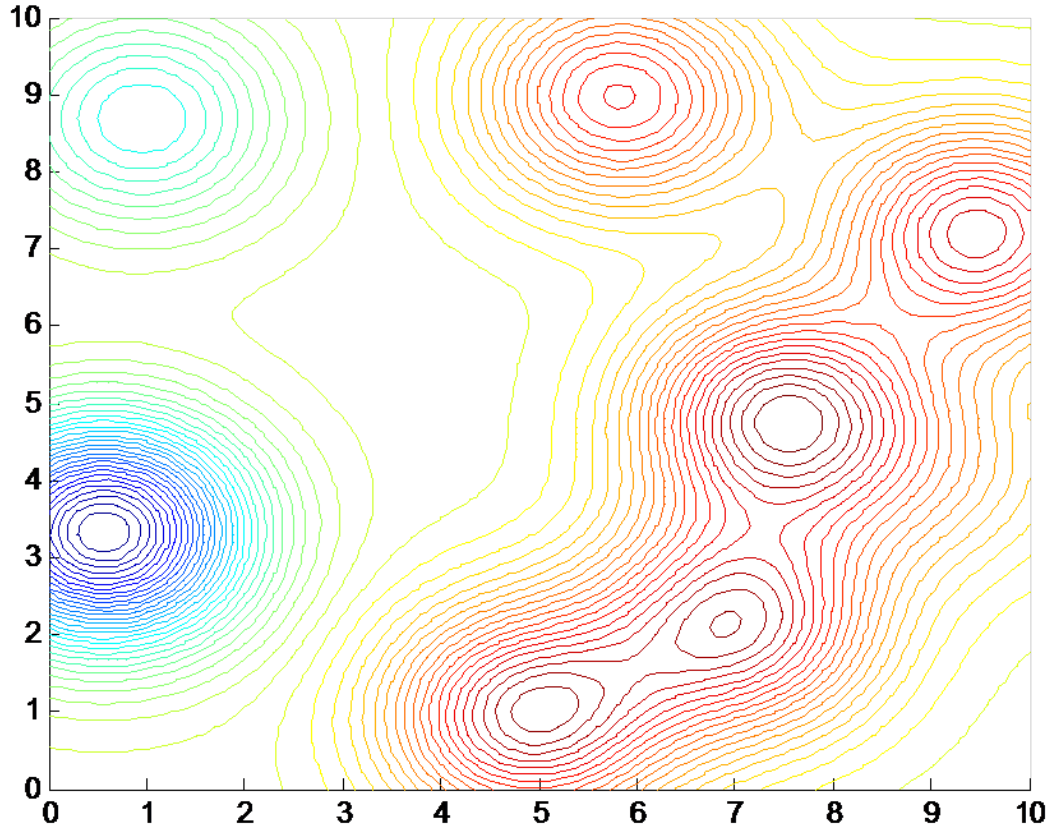
5 kernels



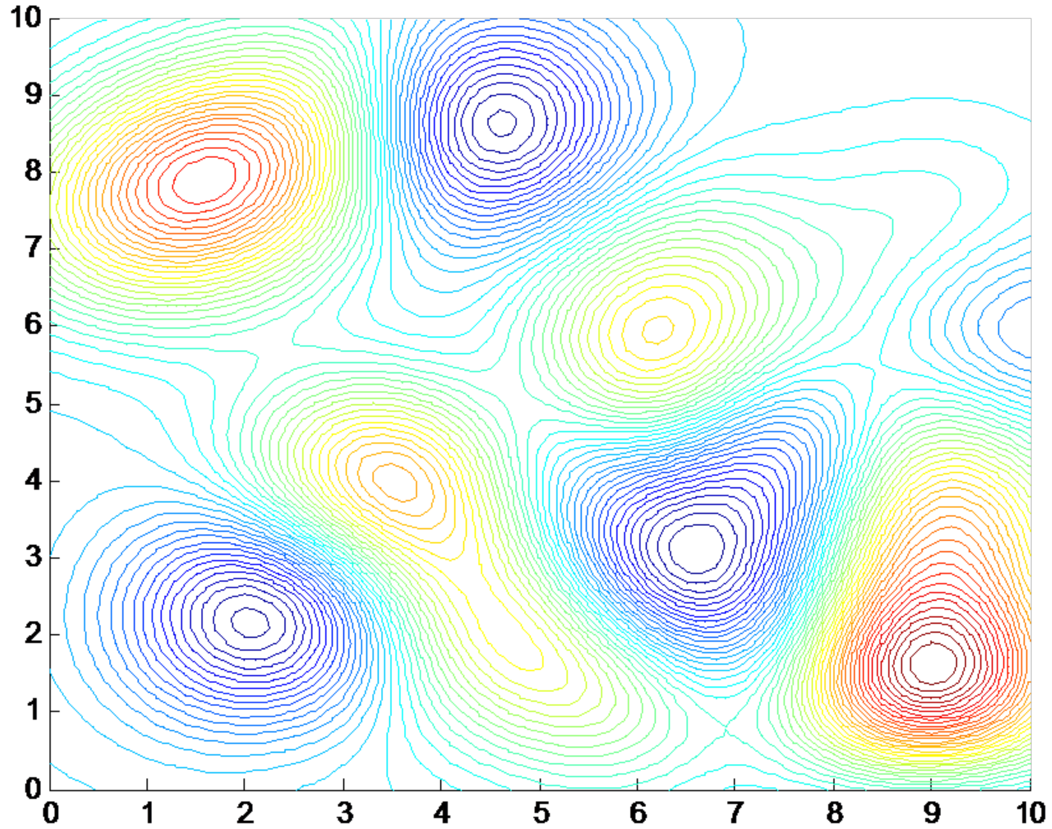
6 kernels



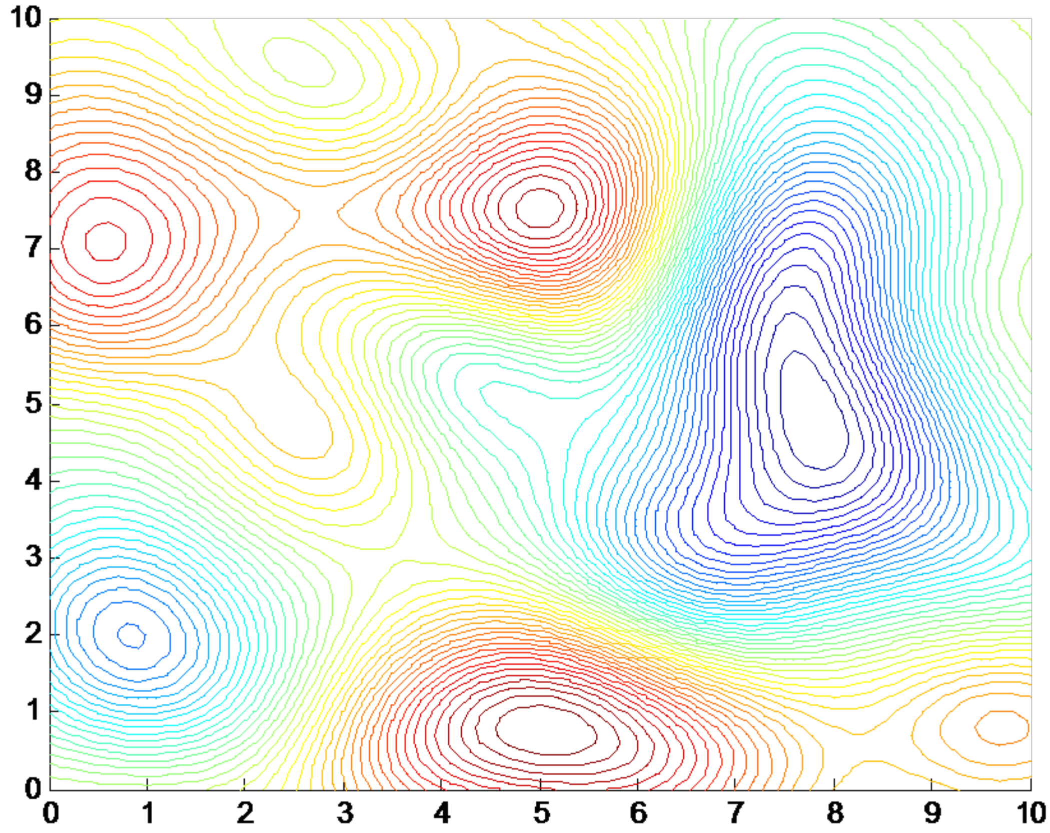
8 kernels



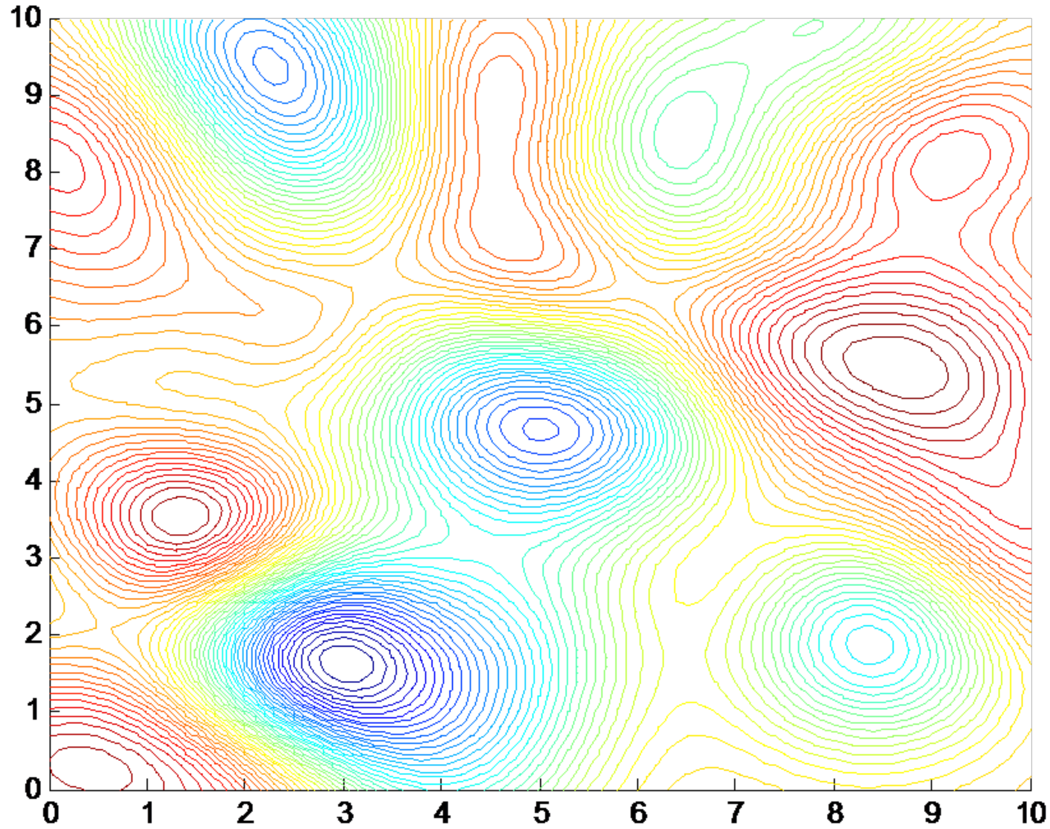
15 kernels



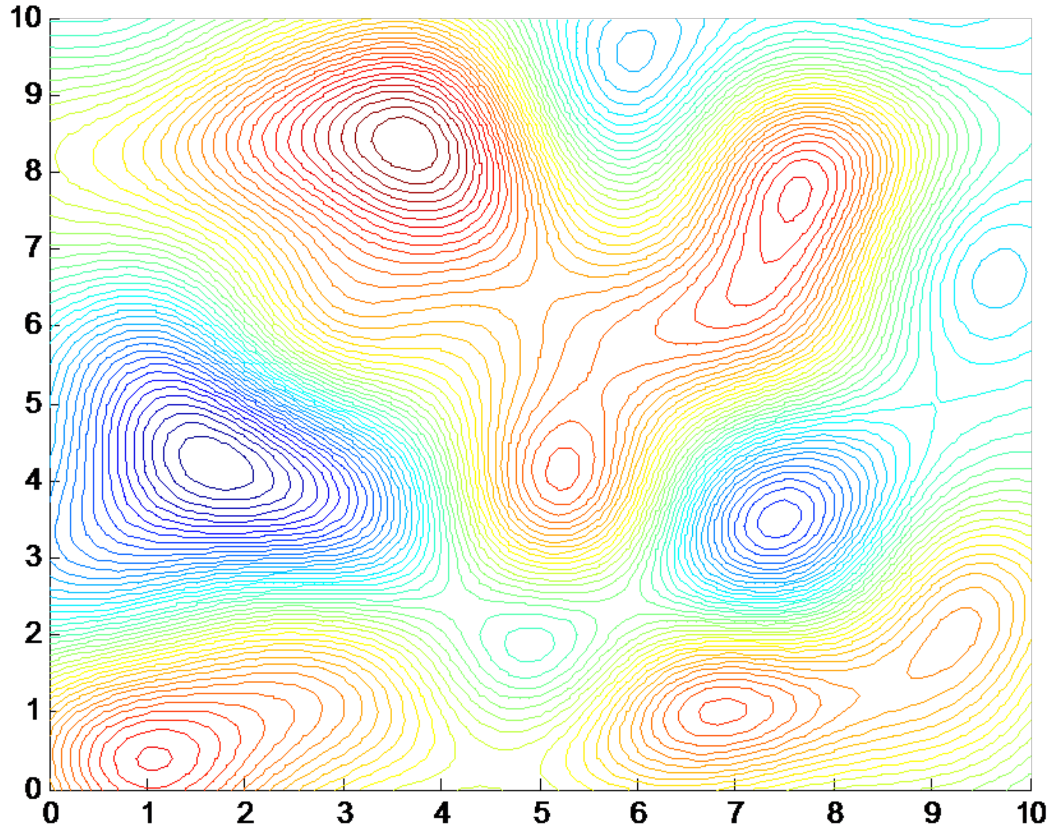
25 kernels



50 kernels



100 kernels



Contours

- Each (highly non-linear) contour corresponds to a straight line in feature space.
- Each contour is a potential decision boundary in input space.
- Given training data, an SVM chooses the best function and contour.
- The SVM margins correspond to contours on either side of the decision boundary.

Symmetric, positive-definite matrices

- Symmetric matrices:
  - Eigenvectors for distinct eigenvalues are orthogonal.
  - An nxn matrix has n orthogonal eigenvectors.
  - $A = EDE^T$, where E is orthogonal and D is diagonal.
- Positive-definite matrices:
  - All eigenvalues are non-negative.
  - The determinant is non-negative.

Mercer's Theorem

If k is a continuous kernel of a positive definite integral operator on $L_2(\mathcal{X})$ (where $\mathcal{X}$ is some compact space),

$$\int_{\mathcal{X}} k(x, x') f(x) f(x') \, dx \, dx' \geq 0,$$

it can be expanded as

$$k(x, x') = \sum_{i=1}^{\infty} \lambda_i \psi_i(x) \psi_i(x')$$

using eigenfunctions ψ_i and eigenvalues $\lambda_i \geq 0$ [41].

The Mercer Feature Map

In that case

$$\Phi(x) := \begin{pmatrix} \sqrt{\lambda_1}\psi_1(x) \\ \sqrt{\lambda_2}\psi_2(x) \\ \vdots \end{pmatrix}$$

satisfies $\langle \Phi(x), \Phi(x') \rangle = k(x, x')$.

Proof:

$$\langle \Phi(x), \Phi(x') \rangle = \left\langle \begin{pmatrix} \sqrt{\lambda_1}\psi_1(x) \\ \sqrt{\lambda_2}\psi_2(x) \\ \vdots \end{pmatrix}, \begin{pmatrix} \sqrt{\lambda_1}\psi_1(x') \\ \sqrt{\lambda_2}\psi_2(x') \\ \vdots \end{pmatrix} \right\rangle$$

$$= \sum_{i=1}^{\infty} \lambda_i \psi_i(x) \psi_i(x') = k(x, x')$$

Data-dependent feature spaces

- In support-vector classification and regression, the optimal hyperplane can be found by solving a dual problem.
- The dual problem depends only on the training data, not the entire input space.
- We can therefore pretend that the training data *is* the entire input space.
- This leads to data-dependent feature spaces.

Kernels

Nonlinearity via Feature Maps

Replace x_i by $\Phi(x_i)$ in the optimization problem.

Equivalent optimization problem

$$\text{minimize } \frac{1}{2}\sum_{i,j=1}^{m} \alpha_i\alpha_j y_i y_j k(x_i, x_j) - \sum_{i=1}^{m} \alpha_i$$

$$\text{subject to } \sum_{i=1}^{m} \alpha_i y_i = 0 \text{ and } \alpha_i \geq 0$$

Decision Function

$$w = \sum_{i=1}^{m} \alpha_i y_i \Phi(x_i) \text{ implies}$$

$$f(x) = \langle w, \Phi(x)\rangle + b = \sum_{i=1}^{m} \alpha_i y_i k(x_i, x) + b.$$

The Empirical Kernel Map

Recall the feature map

$$\begin{aligned} \Phi : \mathcal{X} &\to \mathbb{R}^{\mathcal{X}} \\ x &\mapsto k(.,x). \end{aligned}$$

- each point is represented by its similarity to *all* other points
- how about representing it by its similarity to a *sample* of other points?

Consider

$$\begin{aligned} \Phi_m : \mathbb{R}^N &\to \mathbb{R}^m \\ x &\mapsto k(.,x)|_{\{x_1,\dots,x_m\}} = (k(x_1,x),\dots,k(x_m,x))^\top \end{aligned}$$

(cf. Tsuda, 1999)

ctd.

- $\Phi_m(x)$ contains *all* available information about x
- the Gram matrix $G_{ij} := \left\langle \Phi_m(x_i), \Phi_m(x_j) \right\rangle$ satisfies $G = K^2$ where $K_{ij} = k(x_i, x_j)$
- modify Φ_m to

$$\begin{aligned} \Phi_m^w : \mathbb{R}^N &\to \mathbb{R}^m \\ x &\mapsto K^{-\frac{1}{2}}(k(x_1, x), \ldots, k(x_m, x))^\top \end{aligned}$$

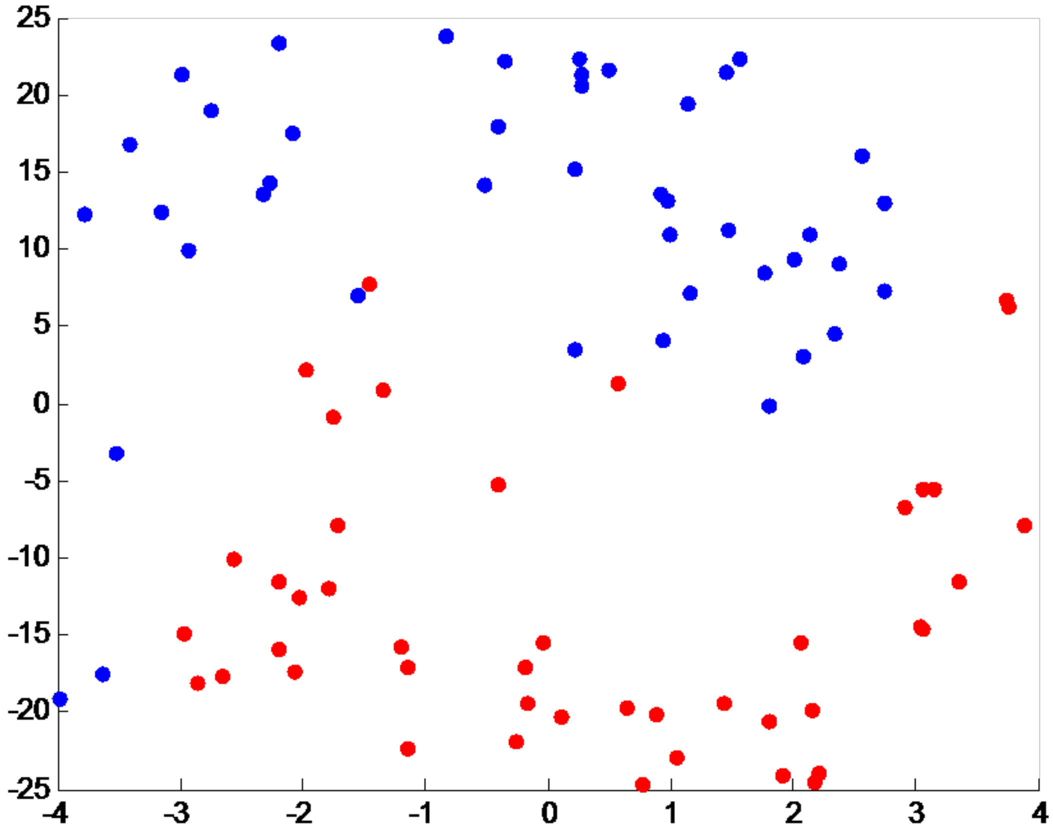
- this "whitened" map ("kernel PCA map") satifies

$$\left\langle \Phi_m^w(x_i), \Phi_m^w(x_j) \right\rangle = k(x_i, x_j)$$

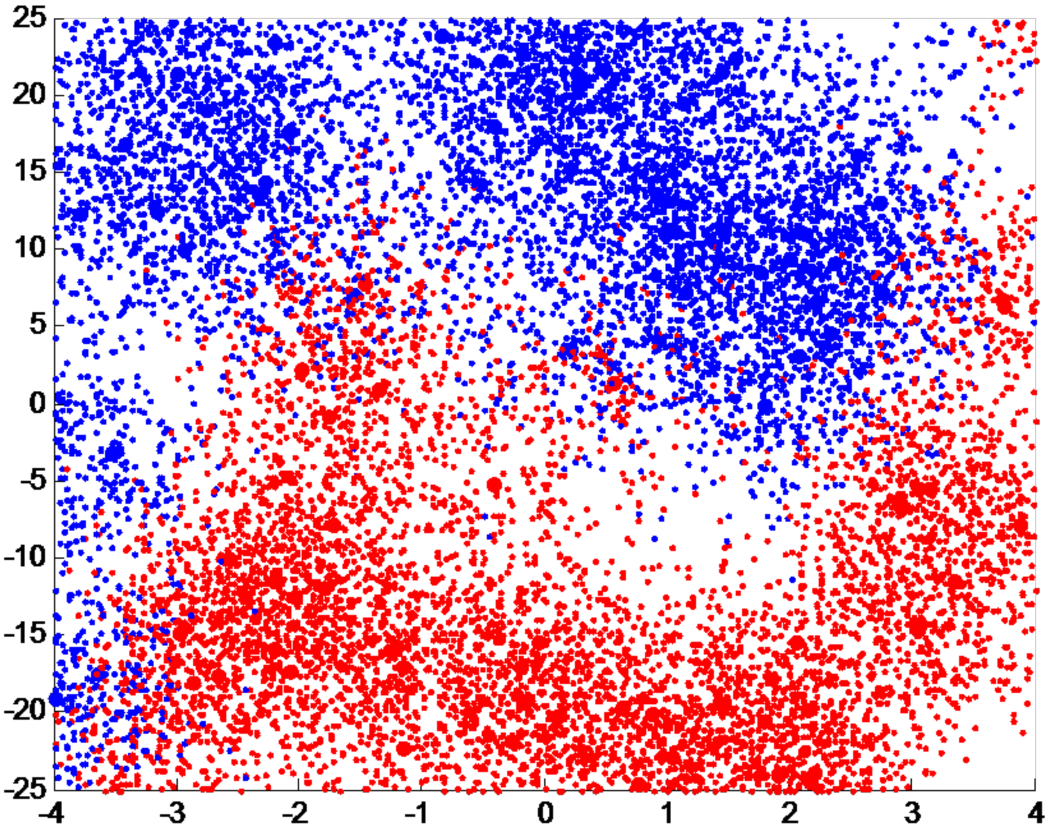
The Representer Theorem

- In support-vector classification, the SVM places a kernel on each training point.
- It then estimates an optimal weight for each kernel, and adds them up.
- The decision boundary is a contour of the sum.
- Placing kernels on other input points would *not* lead to a better decision boundary.
- Many other kernel problems have this extremely-useful property.

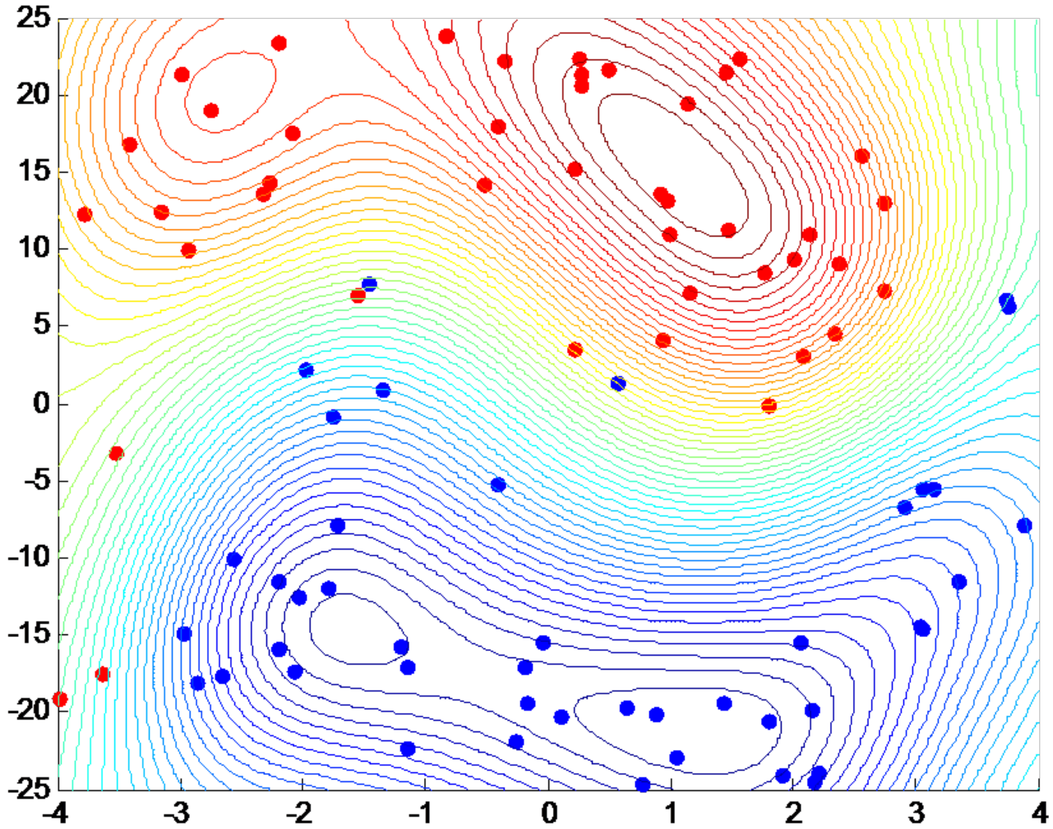
A data sample



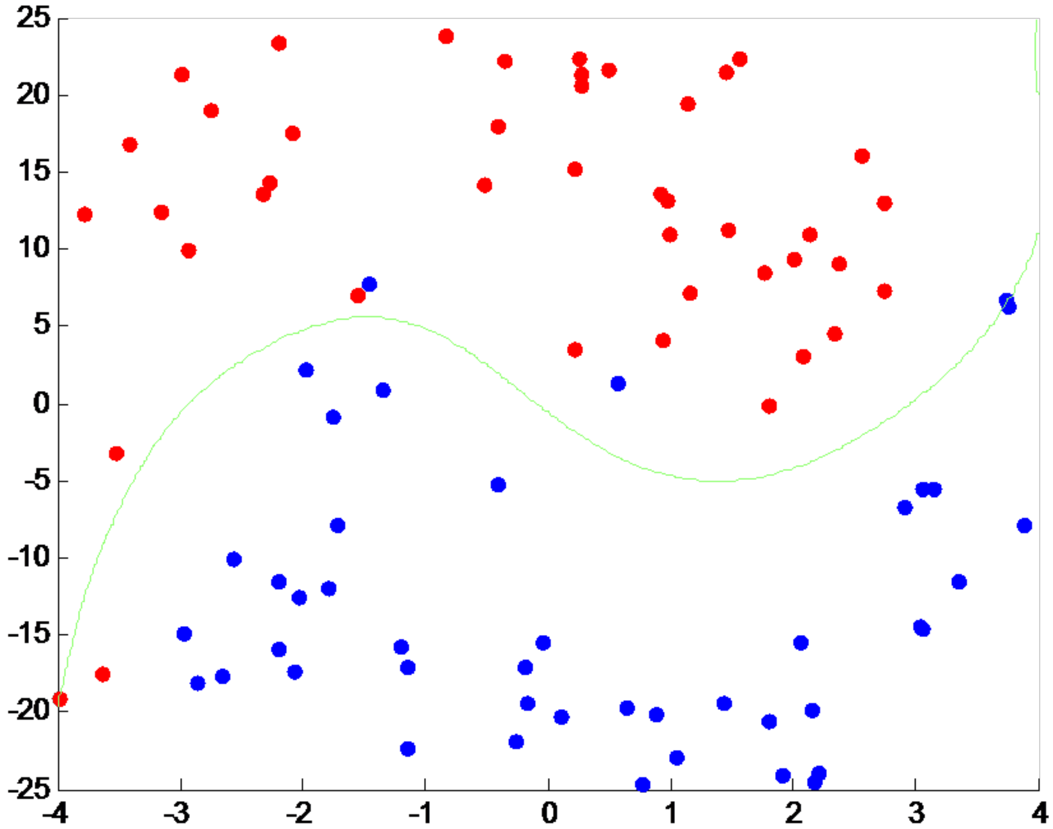
Placing an RBF kernel at each sample point



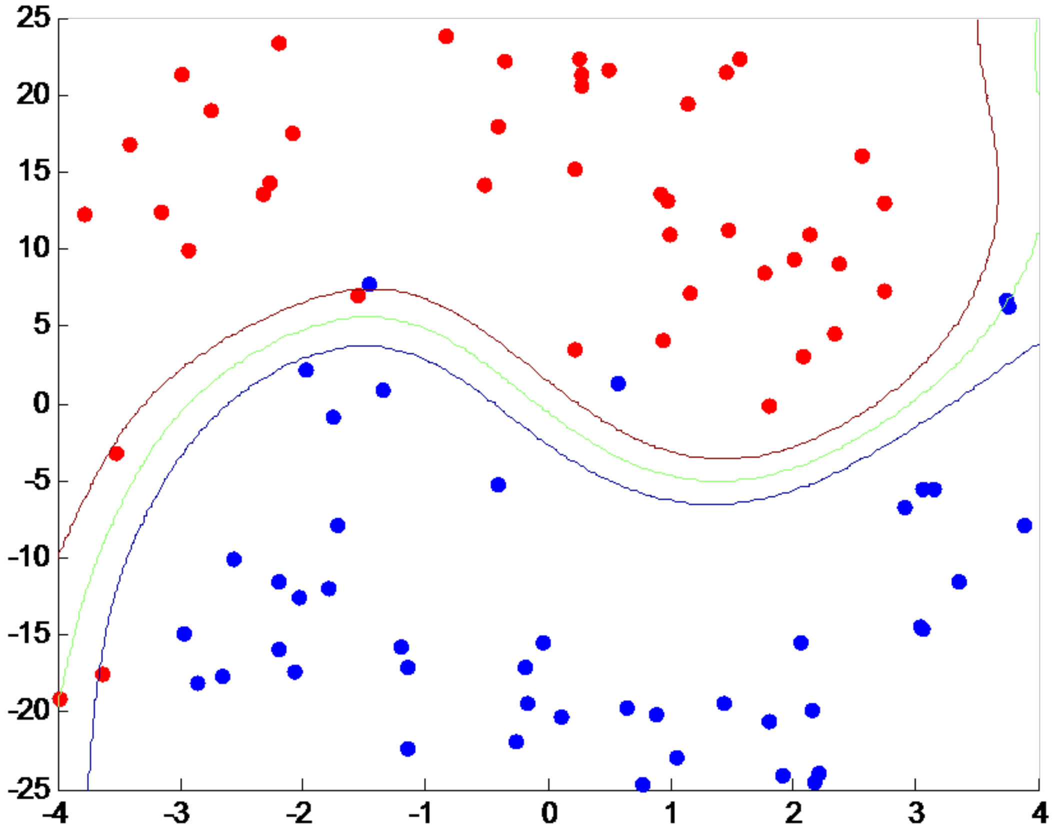
Contour plot of the sum of the kernel values



Estimated decision boundary: level 0 contour



Decision boundary and margins: three contours



The Representer Theorem

Theorem 4 *Given: a p.d. kernel k on $\mathcal{X} \times \mathcal{X}$, a training set $(x_1, y_1), \ldots, (x_m, y_m) \in \mathcal{X} \times \mathbb{R}$, a strictly monotonic increasing real-valued function Ω on $[0, \infty[$, and an arbitrary cost function $c : (\mathcal{X} \times \mathbb{R}^2)^m \to \mathbb{R} \cup \{\infty\}$*

Any $f \in \mathcal{F}$ minimizing the regularized risk functional

$$c\left((x_1, y_1, f(x_1)), \ldots, (x_m, y_m, f(x_m))\right) + \Omega\left(\|f\|\right) \qquad (3)$$

admits a representation of the form

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

Remarks

- significance: many learning algorithms have optimal solutions that can be expressed as expansions in terms of the training examples
- original form, with mean squared loss

$$c((x_1, y_1, f(x_1)), \ldots, (x_m, y_m, f(x_m))) = \frac{1}{m} \sum_{i=1}^{m} (y_i - f(x_i))^2,$$

  and $\Omega(\|f\|) = \lambda \|f\|^2$ ($\lambda > 0$): [37]
- generalization to non-quadratic cost functions: [16]
- present form: non-quadratic regularizers [53]

Proof

Decompose $f \in \mathcal{F}$ into a part in the span of the $k(x_i, .)$ and an orthogonal one:

$$f = \sum_i \alpha_i k(x_i, .) + f_\perp,$$

where for all j

$$\langle f_\perp, k(x_j, .)\rangle = 0.$$

Application of f to an arbitrary training point x_j yields

$$\begin{aligned} f(x_j) &= \langle f, k(x_j, .)\rangle \\ &= \left\langle \sum_i \alpha_i k(x_i, .) + f_\perp, k(x_j, .) \right\rangle \\ &= \sum_i \alpha_i \langle k(x_i, .), k(x_j, .)\rangle, \end{aligned}$$

independent of $f_\perp$.

Proof: second part of (3)

Since $f_\perp$ is orthogonal to $\sum_i \alpha_i k(x_i, .)$, and Ω is strictly monotonic, we get

$$\begin{aligned}
\Omega(\|f\|) &= \Omega\left(\|\sum_i \alpha_i k(x_i, .) + f_\perp\|\right) \\
&= \Omega\left(\sqrt{\|\sum_i \alpha_i k(x_i, .)\|^2 + \|f_\perp\|^2}\right) \\
&\geq \Omega\left(\|\sum_i \alpha_i k(x_i, .)\|\right), \qquad (4)
\end{aligned}$$

with equality occuring if and only if $f_\perp = 0$.
Hence, any minimizer must have $f_\perp = 0$. Consequently, any solution takes the form $f = \sum_i \alpha_i k(x_i, .)$, i.e.

$$f(.) = \sum_i \alpha_i k(x_i, .).$$

Application 1: Support Vector Classification

Here, $y_i \in \{\pm 1\}$. Use

$$c\left(\left(x_i, y_i, f\left(x_i\right)\right)_i\right) = \frac{1}{\lambda} \sum_i \max\left(0, 1 - y_i f\left(x_i\right)\right),$$

and the regularizer $\Omega\left(\|f\|\right) = \|f\|^2$.
$\lambda \to 0$ leads to the hard margin SVM

Some Properties of Kernels [53]

If $k_1, k_2, \ldots$ are pd kernels, then so are

- αk_1, provided $\alpha \geq 0$
- $k_1 + k_2$
- $k_1 \cdot k_2$
- $k(x, x') := \lim_{n\to\infty} k_n(x, x')$, provided it exists
- $k(A, B) := \sum_{x\in A, x'\in B} k_1(x, x')$, where A, B are finite subsets of $\mathcal{X}$
  (using the feature map $\tilde{\Phi}(A) := \sum_{x\in A} \Phi(x)$)

Further operations to construct kernels from kernels: tensor products, direct sums, convolutions [28].