Kernel trick. Deep learning.

© Victor Kitov

v.v.kitov@yandex.ru

Summer school on Machine Learning in High Energy Physics

in partnership with

August 2015

Table of Contents

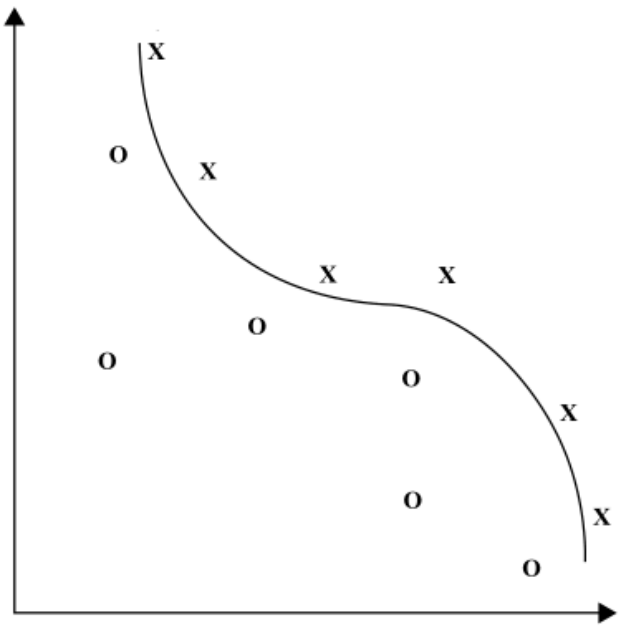
Kernel trick

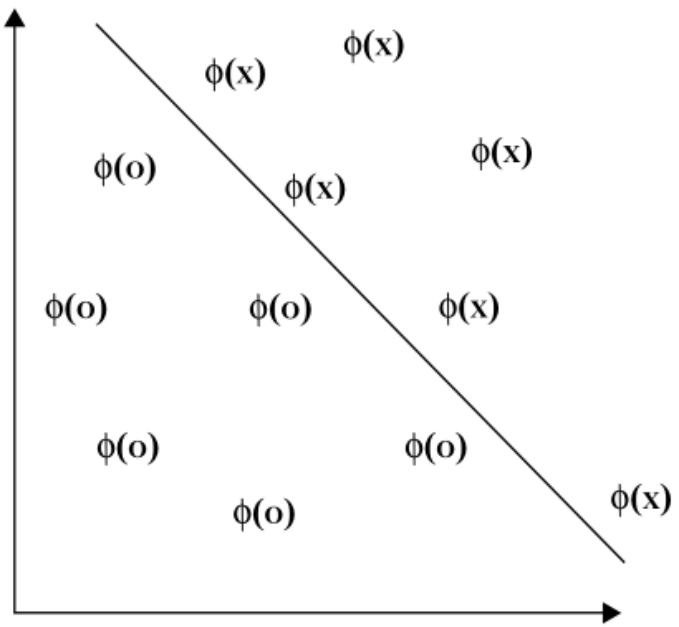
- Kernel trick:
  - replaces x with $\phi(x)$
  - $\langle \phi(x), \phi(x') \rangle$ are calculated, using Kernel function $K(x, x')$
- Algorithm:
  1. Obtain "dual solution" of the problem in terms of $\langle x, x' \rangle$
  2. Replace $\langle x, x' \rangle$ with $K(x, x')$
  3. Solve the task.
- Linear solution in transformed space may be non-linear in original space.

Benefits of kernel trick

- $O(D)$ dot product complexity now takes $O(1)$.
- can apply standard well-developed linear tools for non-linear case
- can apply methods for non-vector objects:
  - strings of variable length
  - graphs
  - structured objects
  - etc.

Non-linear features transformation





Kernel definition

- x is replaced with $\phi(x)$
- $[x] \to [x, x^2, x^3]$

Kernel

Function $K(x, y) : X \times X \to \mathbb{R}$ is a kernel function if it may be represented as $K(x, y) = \langle \psi(x), \psi(y) \rangle$ for some mapping $\psi : X \to H$, with scalar product defined on H.

- $< x, y >$ is replaced by $< \phi(x), \phi(y) >= K(x, y)$

Kernel properties

Theorem (Mercer): Function $K(x, x')$ is a kernel is and only if

- it is symmetric: $K(x, x') = K(x', x)$
- it is non-negative definite: for every function $g : X \to \mathbb{R}$

$$\int_X \int_X K(x, x')g(x)g(x')dxdx' \geq 0$$

- Example: $K(x, z) = (1 + x^T z) = (1 + x_1 z_1 + x_2 z_2)^2 = 1 + 2x_1 z_1 + 2x_2 z_2 + 2x_1 z_1 x_2 z_2 + x_1^2 z_1^2 + x_2^2 z_2^2 = \phi^T(x)\phi(z)$
- $\phi(x) = (1, \sqrt{2}x_1, \sqrt{2}x_2, \sqrt{2}x_1 x_2, x_1^2, x_2^2)$

Kernel properties

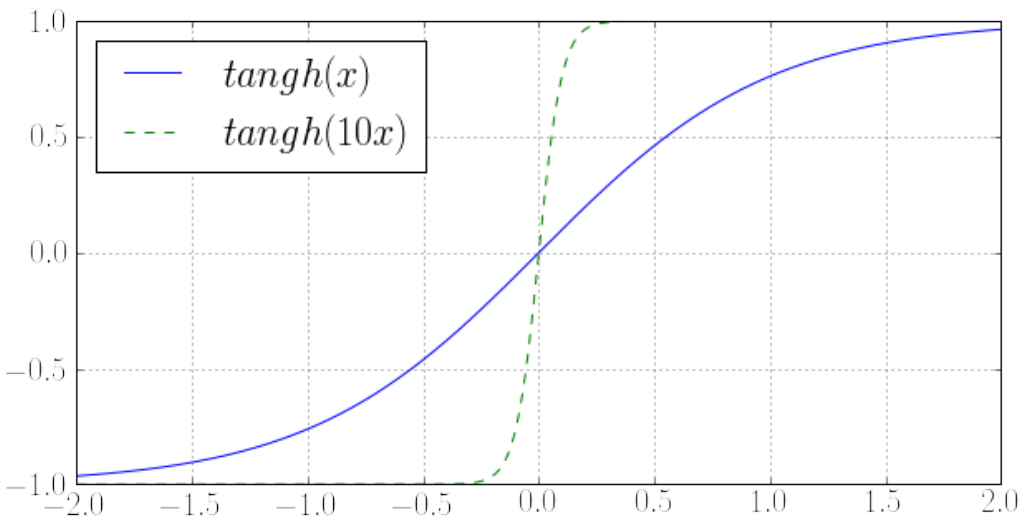
Kernels can be constructed manually:

- Scalar product $\langle x, x' \rangle$ is a kernel
- Constant $K(x, x') \equiv 1$ is a kernel
- Product of kernels $K(x, x') = K_1(x, x')K_2(x, x')$ is a kernel.
- For every function $\psi : X \to \mathbb{R}$ the product $K(x, x') = \psi(x)\psi(x')$ is a kernel
- Linear combination of kernels $K(x, x') = \alpha_1 K_1(x, x') + \alpha_2 K(x, x')$ with positive coefficients is a kernel
- Composition of function $\varphi : X \to X$ and kernel K_0 is a kernel: $K(x, x') = K_0(\varphi(x), \varphi(x'))$
- etc.

Commonly used kernels

Let x and y be two objects.

Kernel	Mathematical form
linear	$\langle x, y \rangle$
polynomial	$(\gamma \langle x, y \rangle + r)^d$
RBF	$\exp(-\gamma \lvert x - y \rvert^2)$
sigmoid	$\text{tangh}(\gamma \langle x, y \rangle + r)$



Kernelized ridge regression

Task: $(y - Xw)^T(y - Xw) + \lambda w^T w \to \min_w$

Solution

$$w = (X^T X + \lambda I_n)^{-1} X^T y$$

Then

$$X^T Xw + \lambda w = X^T y$$

$$w = \frac{1}{\lambda}(X^T y - X^T Xw) = \frac{1}{\lambda} X^T (y - Xw) = X^T \alpha$$

$\alpha = \frac{1}{\lambda} X^T (y - Xw) => \lambda\alpha = (y - XX^T\alpha) => (XX^T + \lambda I)\alpha = y$
$=> \alpha = (XX^T + \lambda I)^{-1} y = (G + \lambda I)^{-1} y$ where $\{G\}_{ij} = K(x_i, x_j)$.
Prediction:

$$\widehat{y} = \langle w, x \rangle = \sum_{n=1}^{N} \alpha_n \langle x_i, x \rangle = \sum_{n=1}^{N} \alpha_n K(x_i, x)$$

Other kernelized algorithms

- K-NN
- K-means, K-medoinds
- nearest medoid
- PCA
- SVM

Table of Contents

Linear SVM reminder

- Solution for weights:

$$w = \sum_{i \in \mathcal{SV}} \alpha_i y_i x_i$$

Discriminant function

$$g(x) = \sum_{i \in \mathcal{SV}} \alpha_i y_i < x_i, x > + w_0$$

$$w_0 = \frac{1}{n_{\widetilde{\mathcal{SV}}}} \left(\sum_{i \in \widetilde{\mathcal{SV}}} y_i - \sum_{i \in \widetilde{\mathcal{SV}}} \sum_{j \in \mathcal{SV}} \alpha_i y_i \langle x_i, x_j \rangle \right)$$

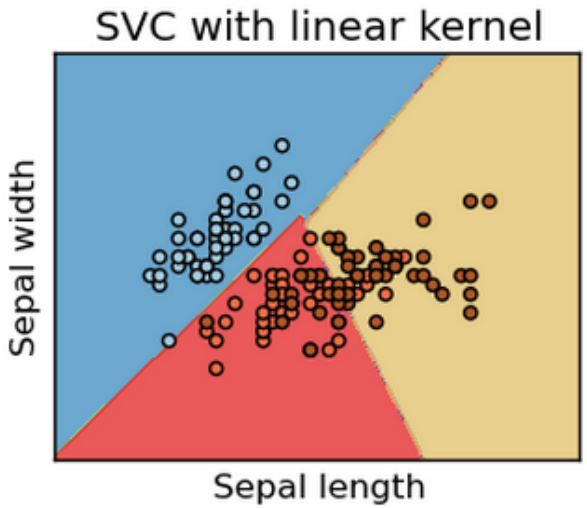
Kernel SVM

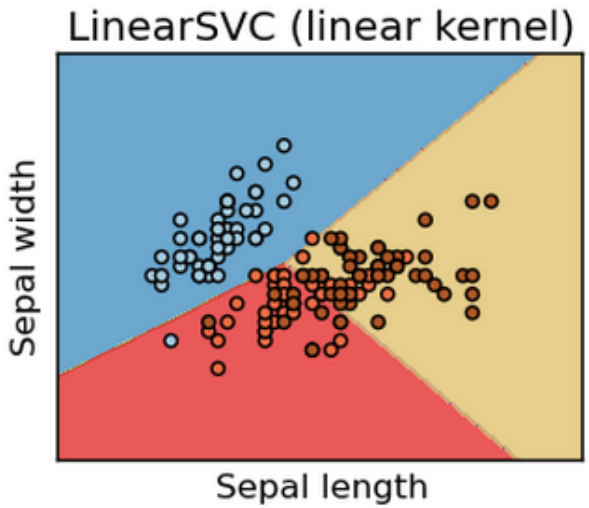
Discriminant function

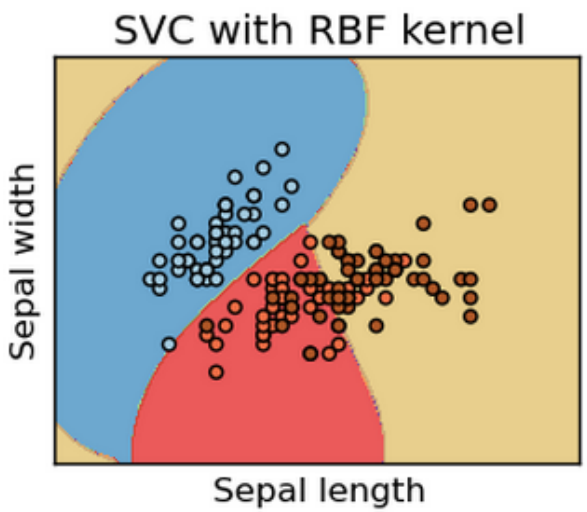
$$g(x) = \sum_{i \in \mathcal{SV}} \alpha_i y_i K(x_i, x) + w_0$$

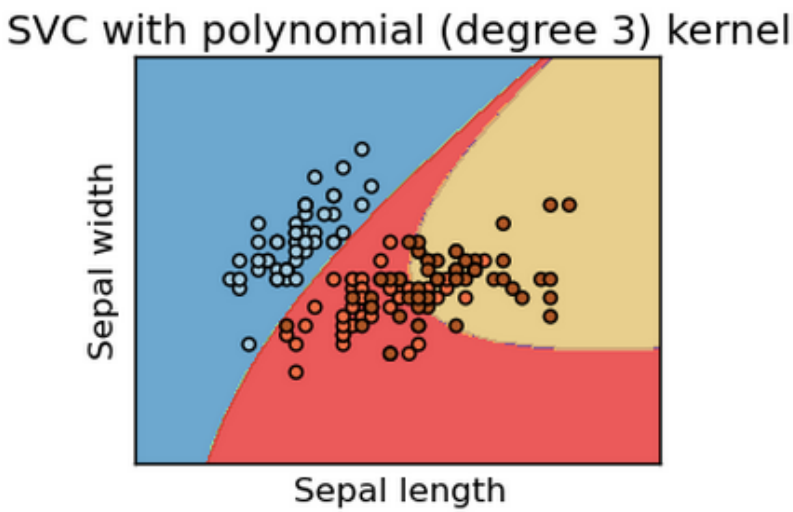
$$w_0 = \frac{1}{n_{\widetilde{\mathcal{SV}}}} \left(\sum_{i \in \widetilde{\mathcal{SV}}} y_i - \sum_{i \in \widetilde{\mathcal{SV}}} \sum_{j \in \mathcal{SV}} \alpha_i y_i K(x_i, x_j) \right)$$

Kernel results

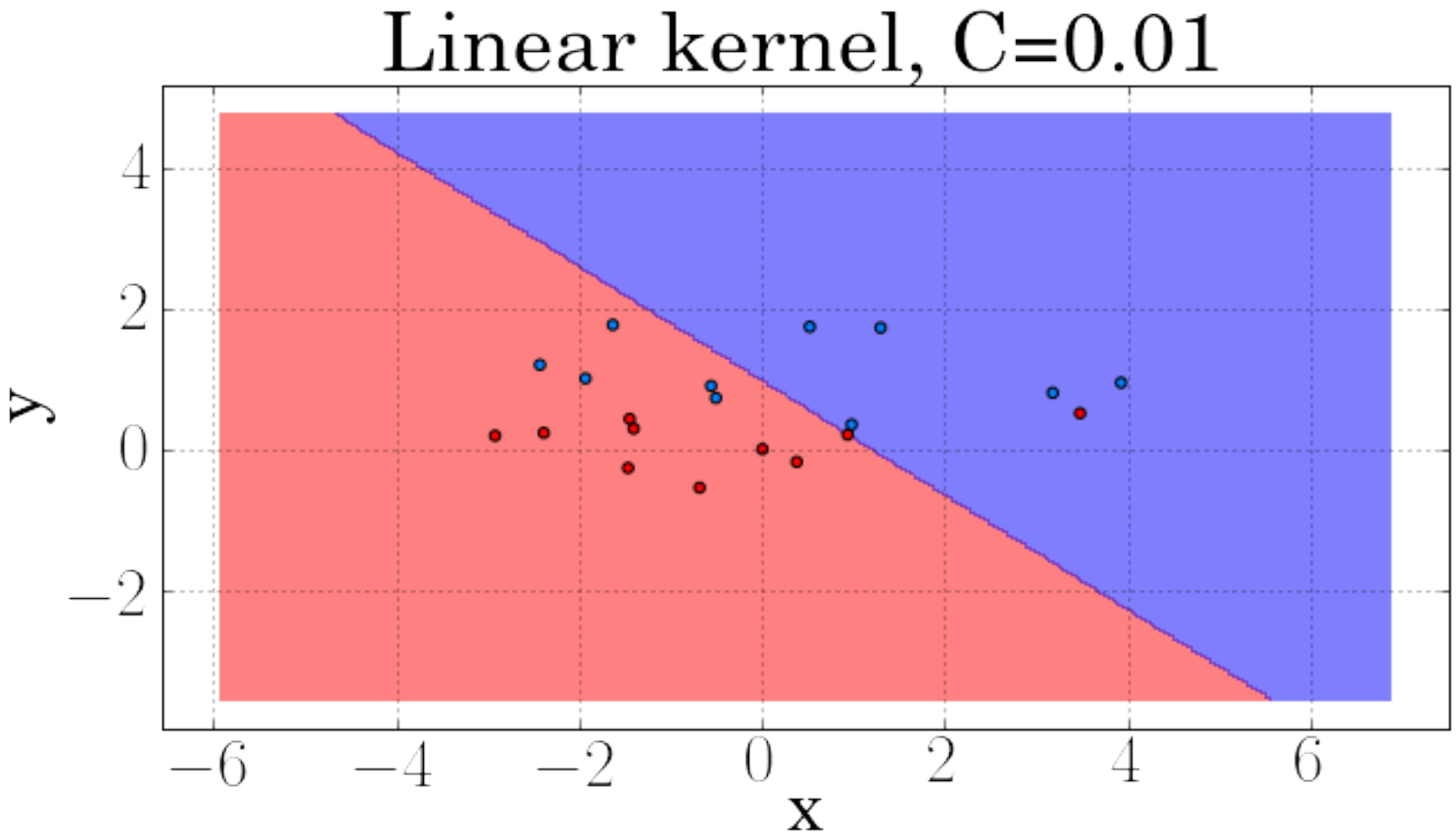




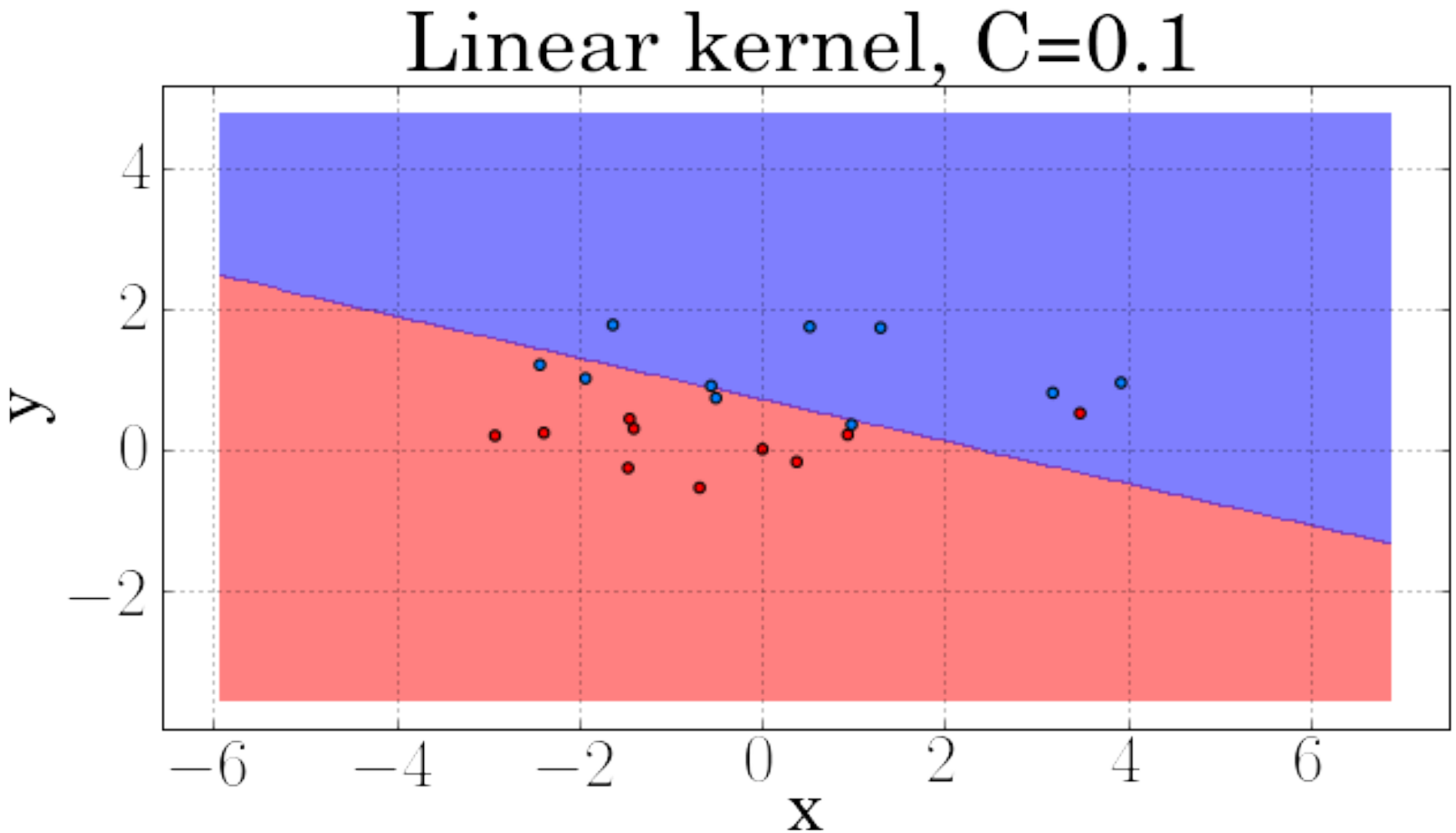




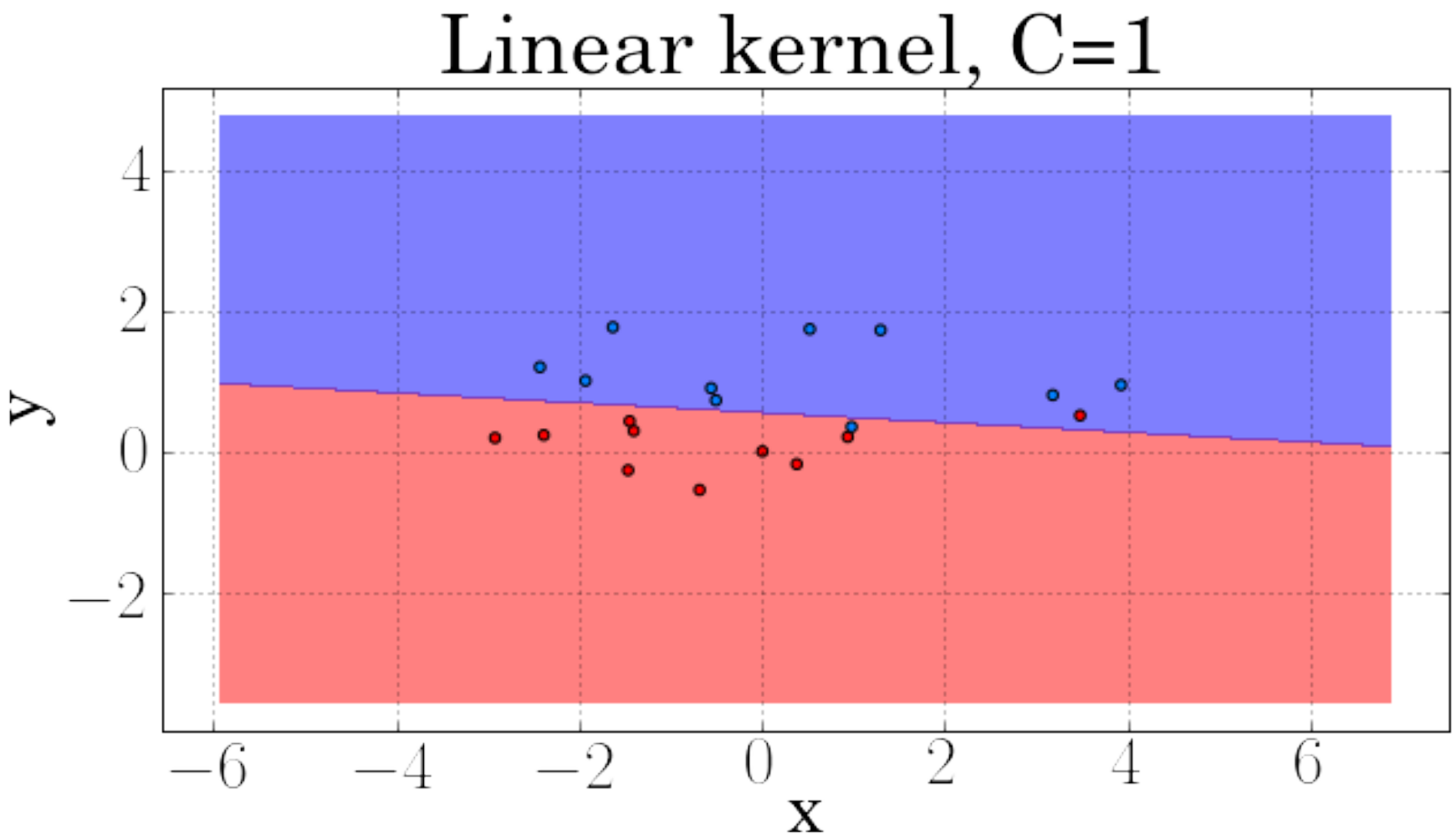
Linear kernel - variable C



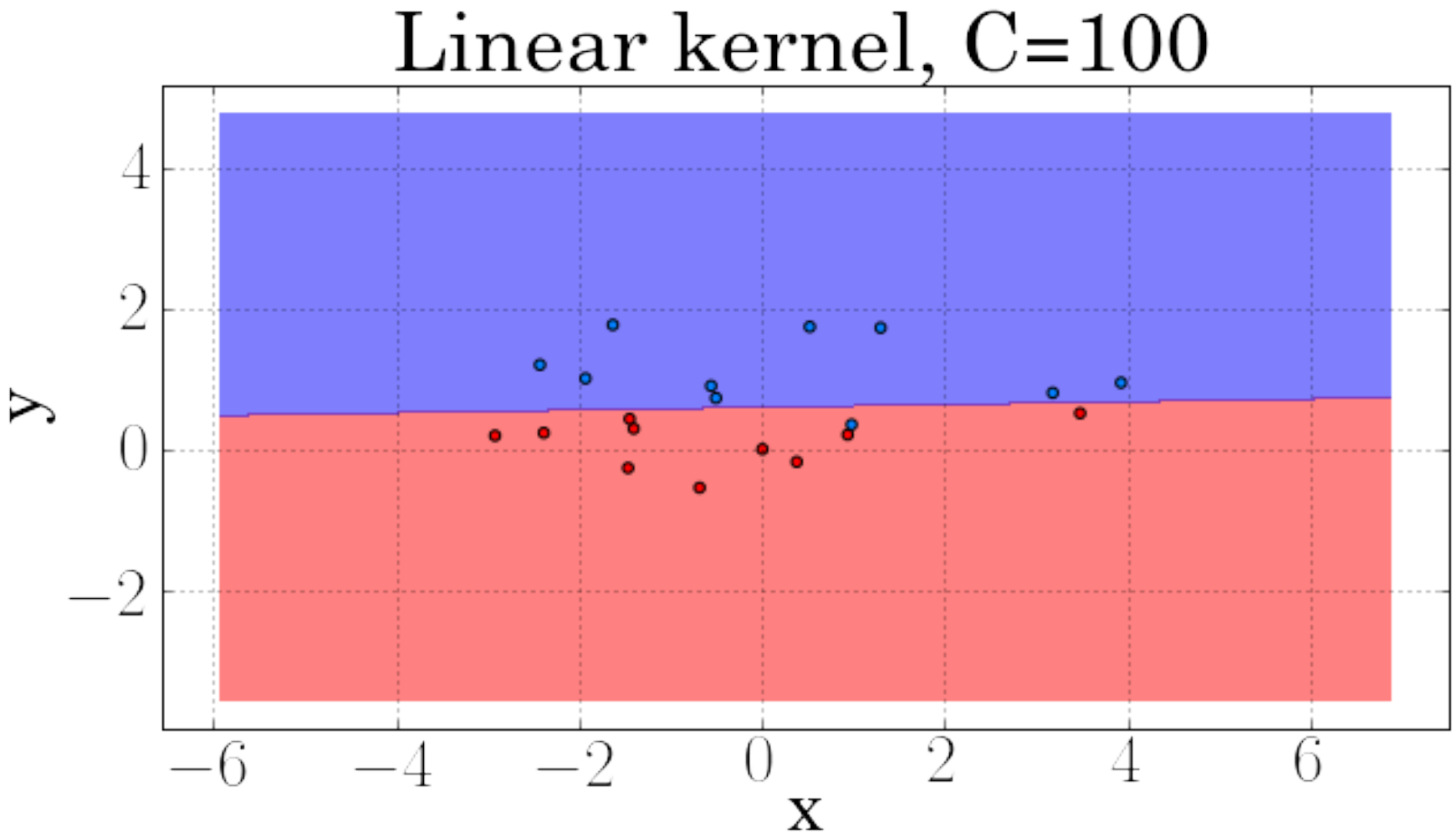
Linear kernel - variable C



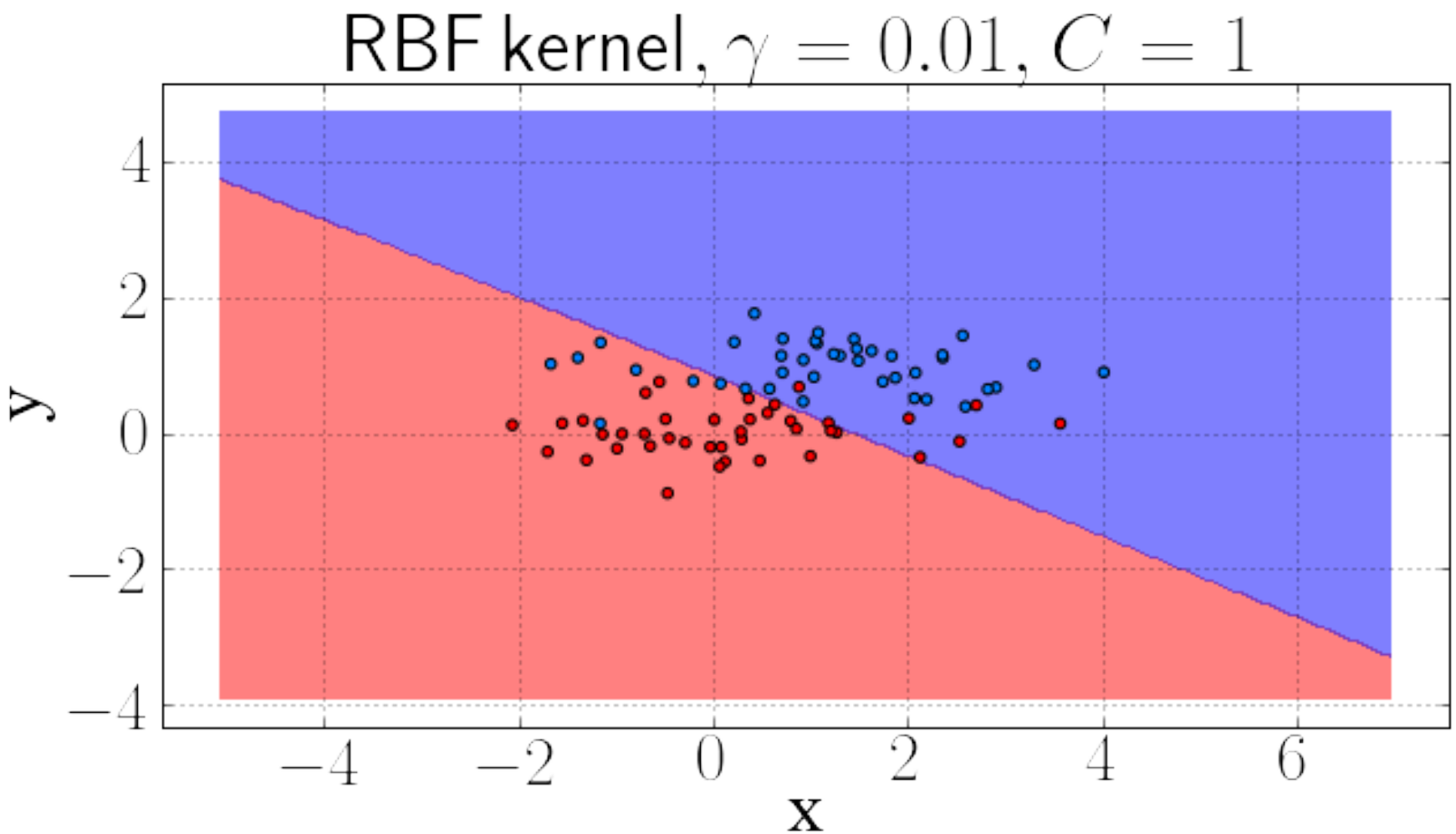
Linear kernel - variable C



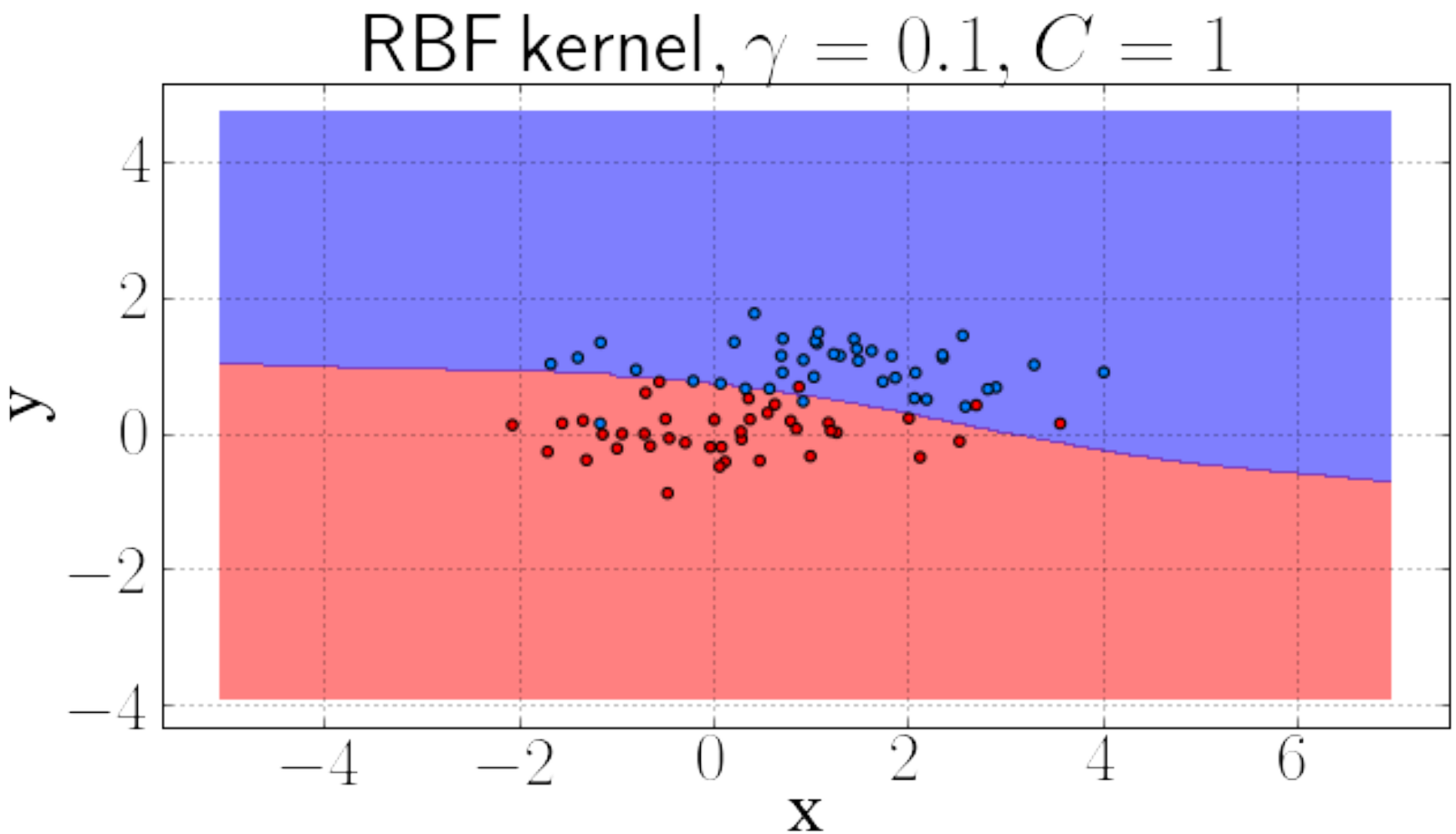
Linear kernel - variable C



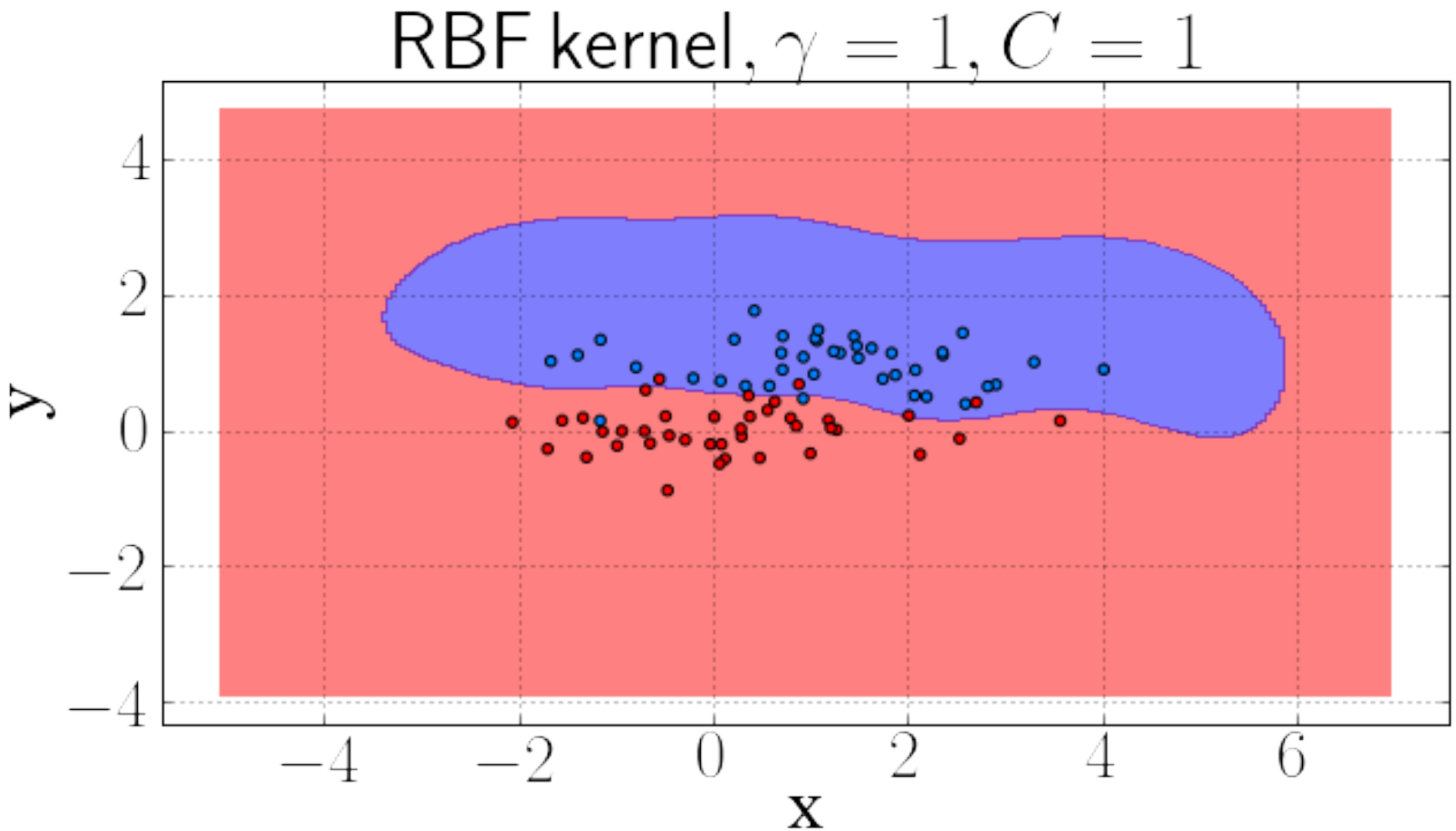
RBF kernel - variable γ



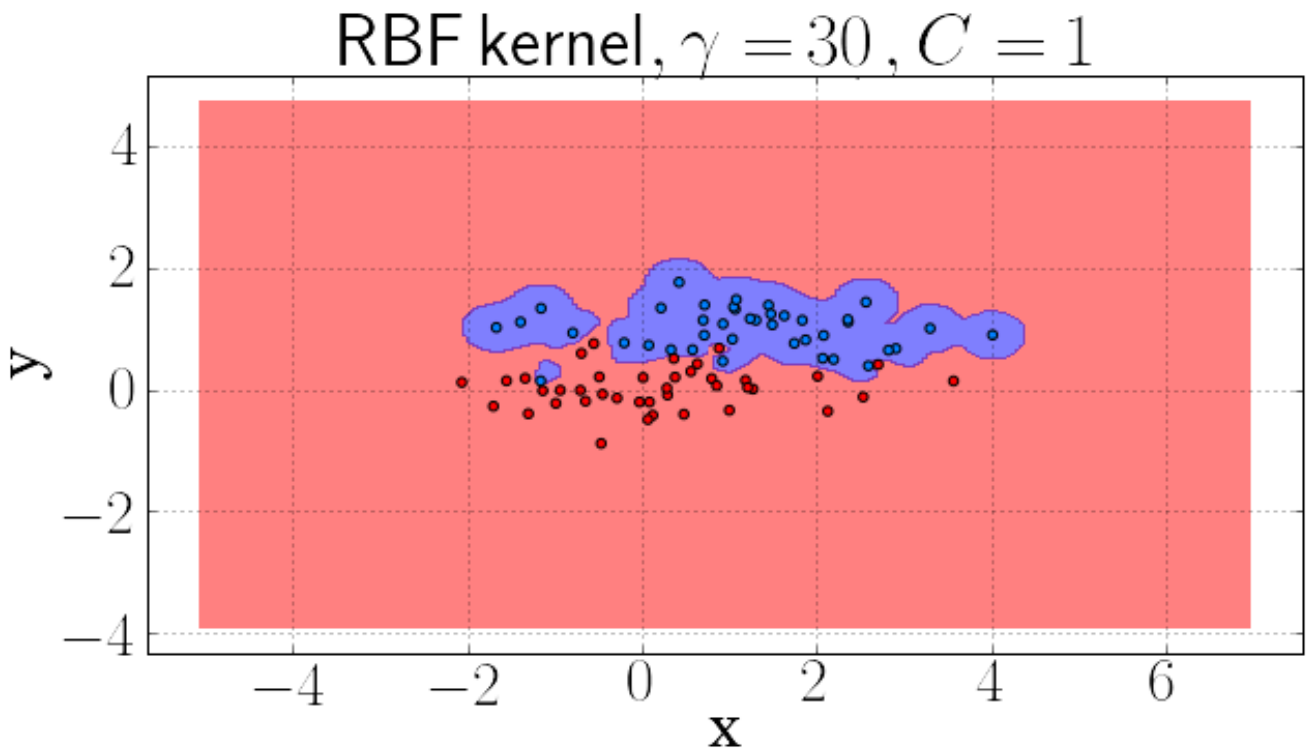
RBF kernel - variable γ



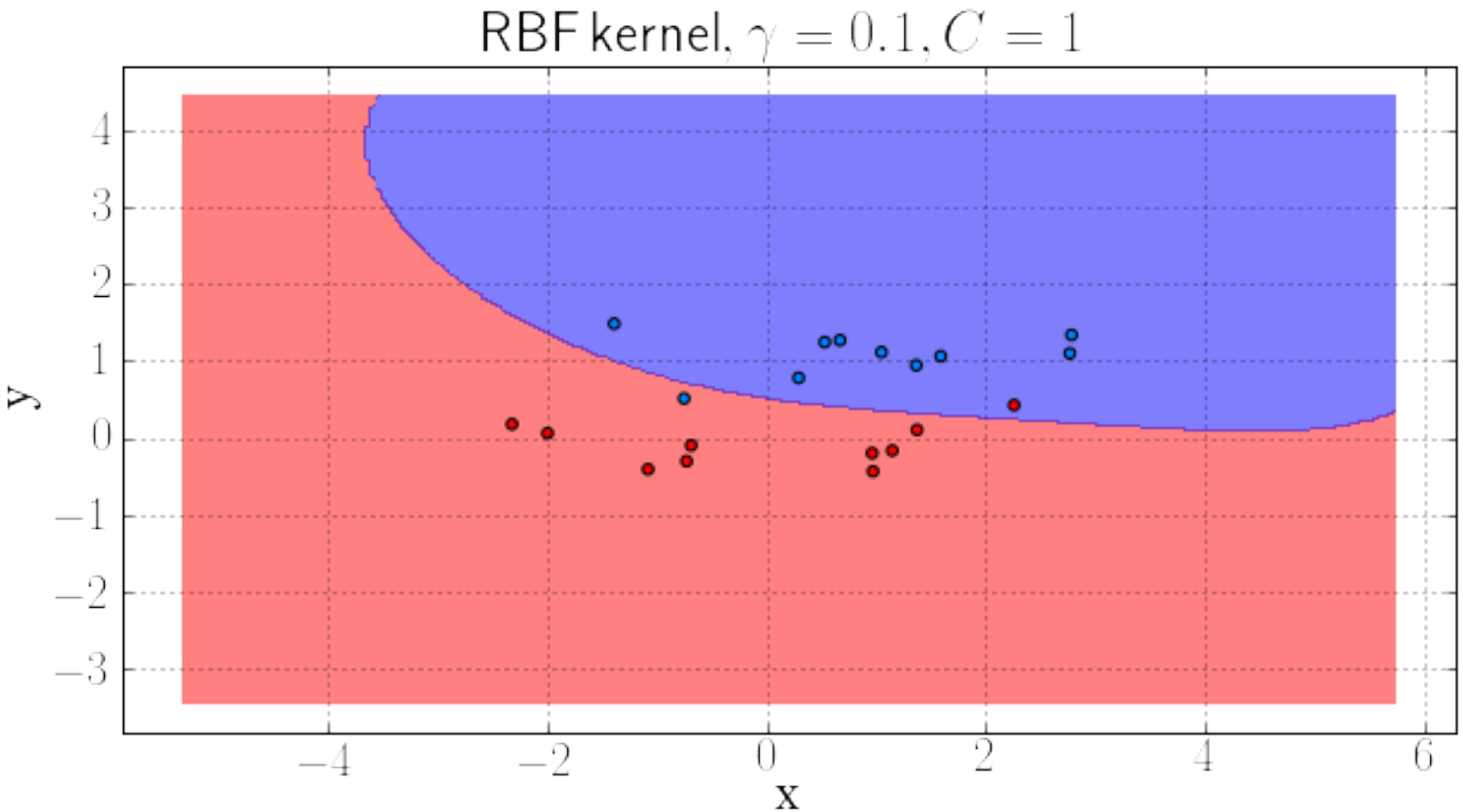
RBF kernel - variable γ



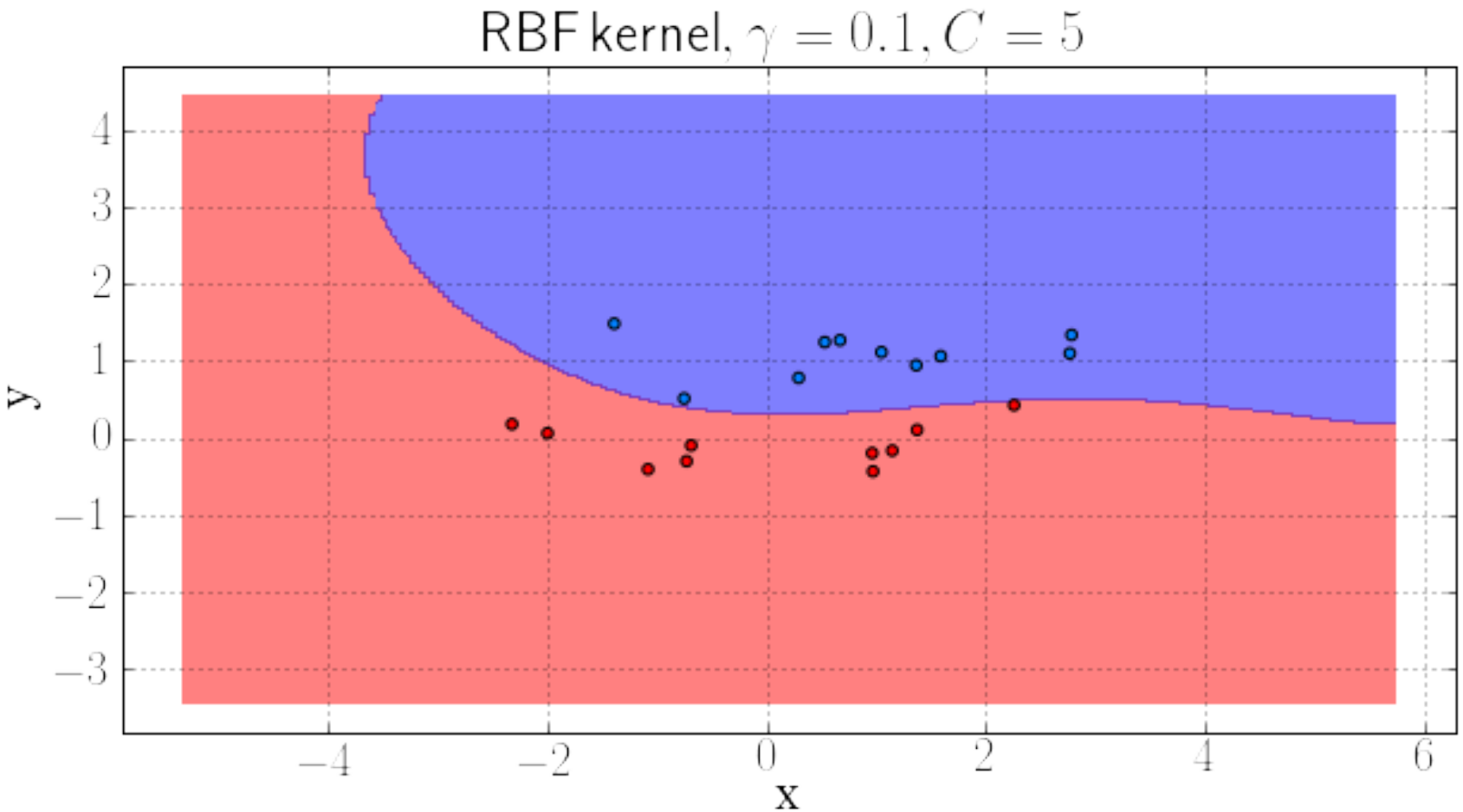
RBF kernel - variable γ



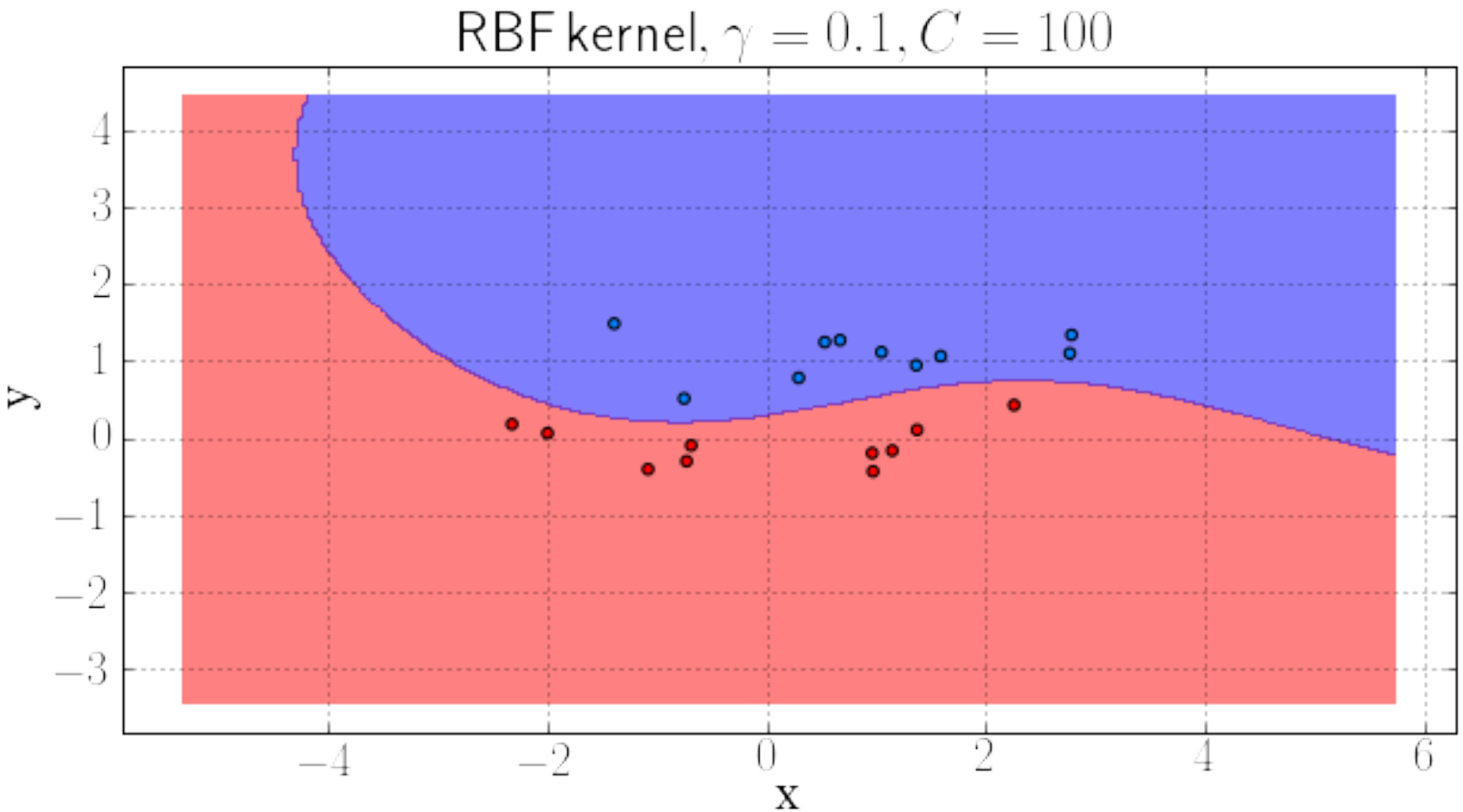
RBF kernel - variable C



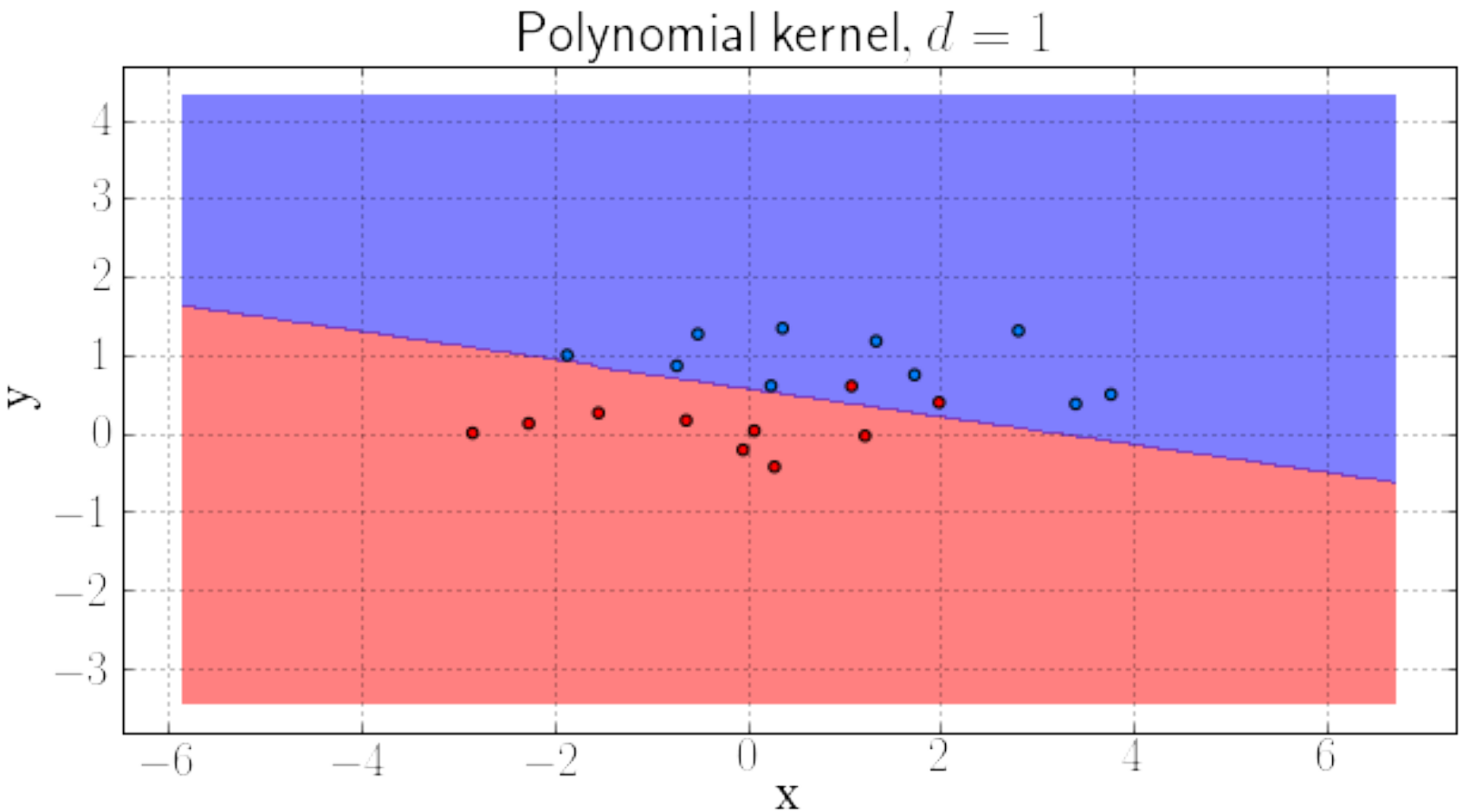
RBF kernel - variable C



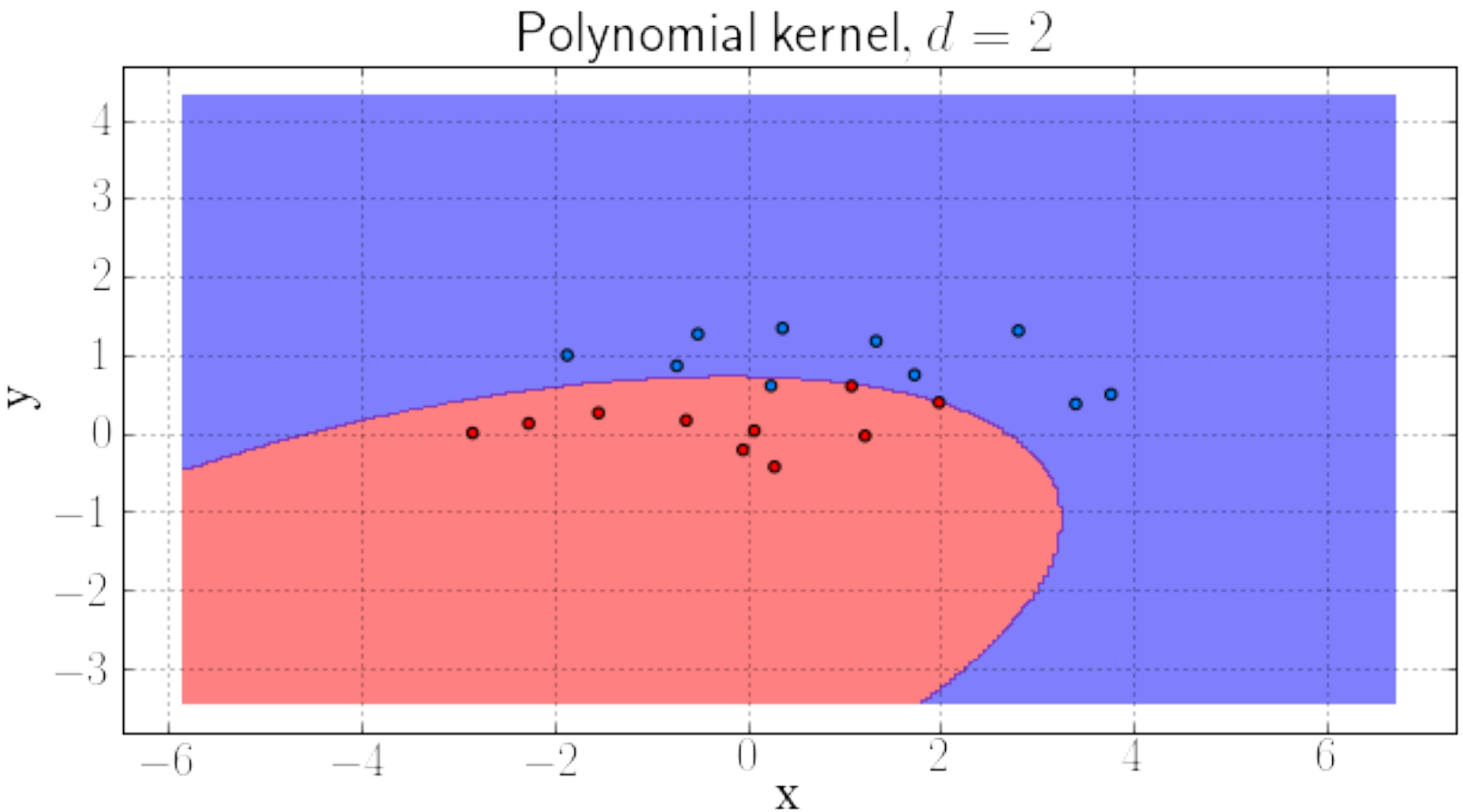
RBF kernel - variable C



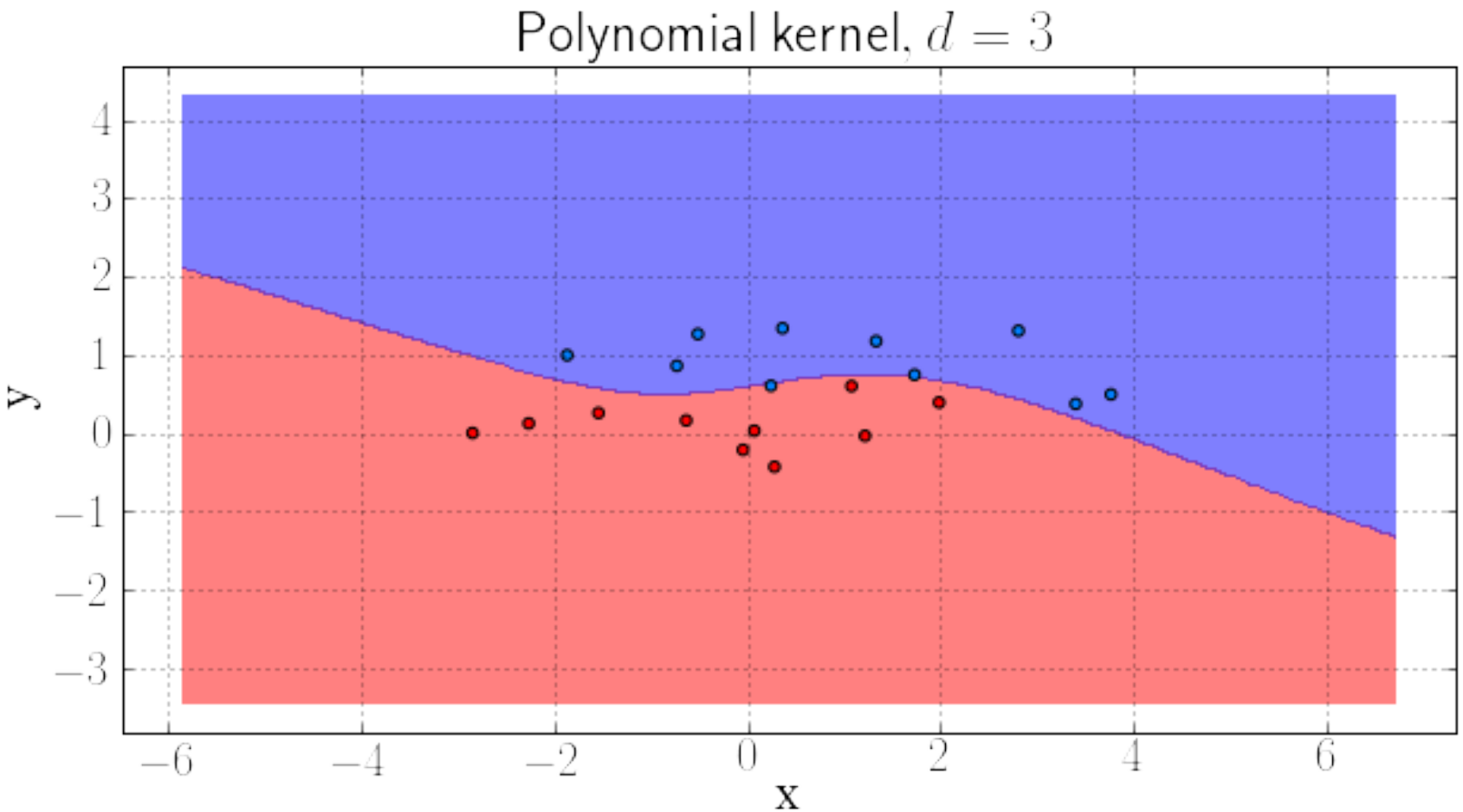
Polynomial kernel - variable d



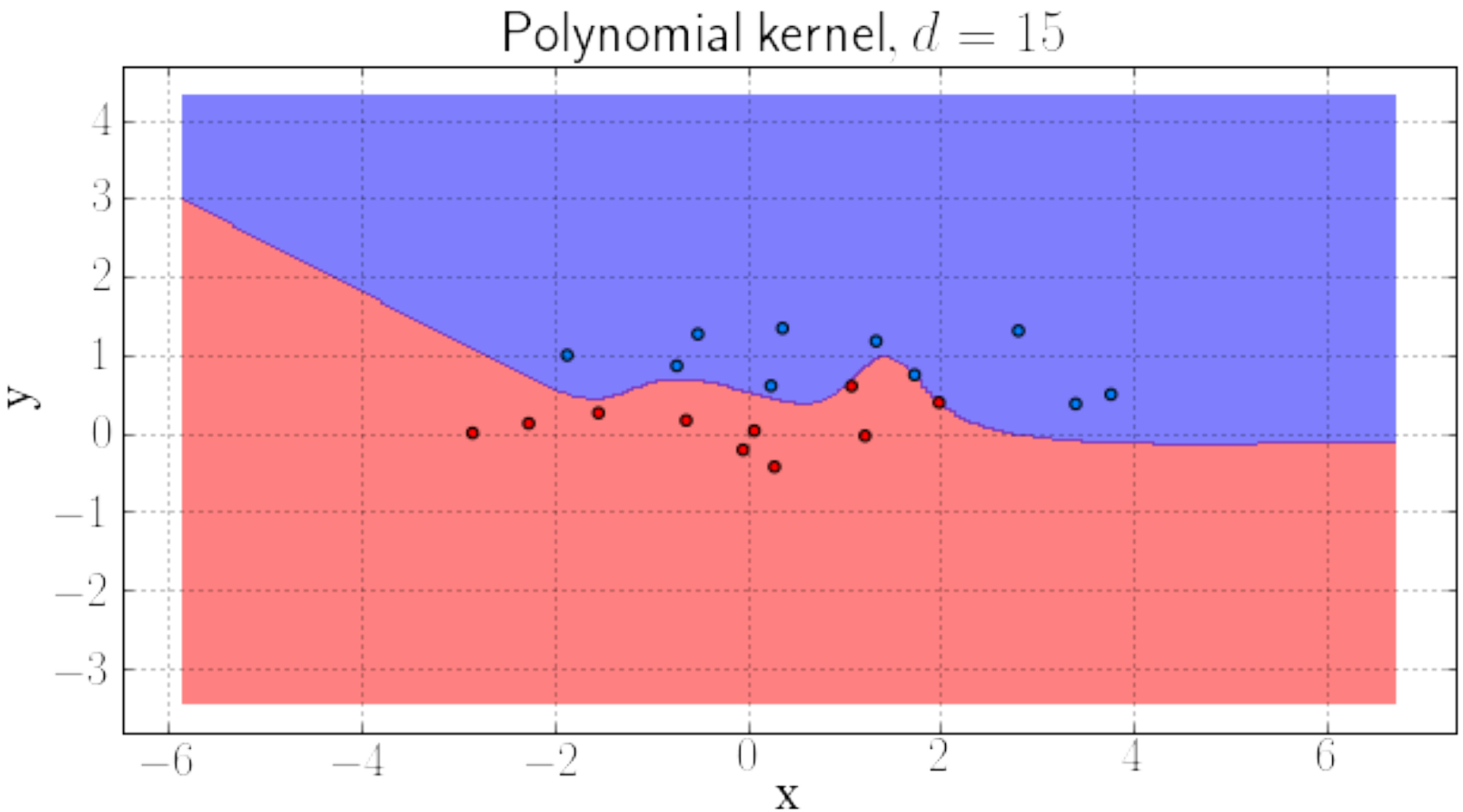
Polynomial kernel - variable d



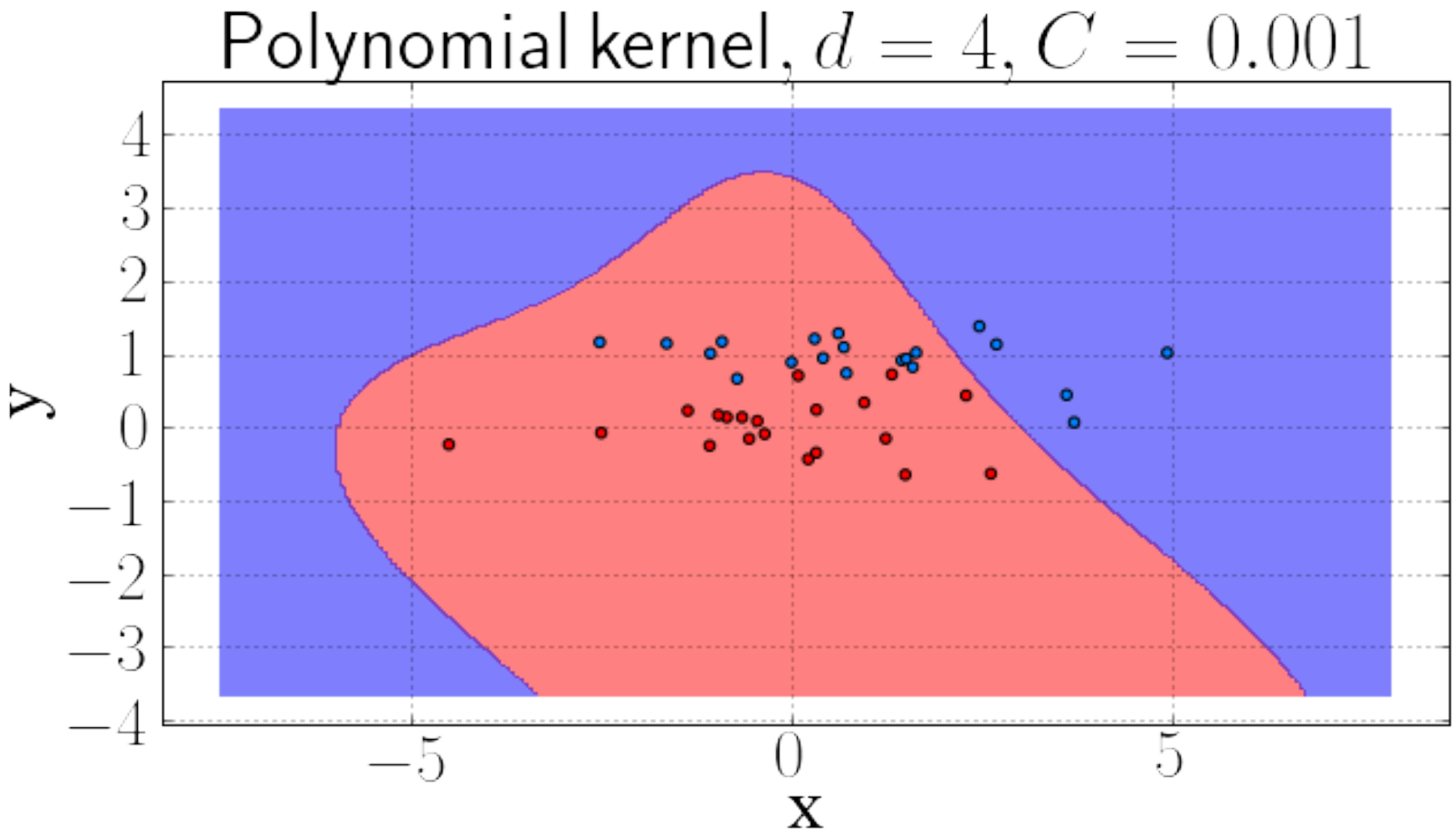
Polynomial kernel - variable d



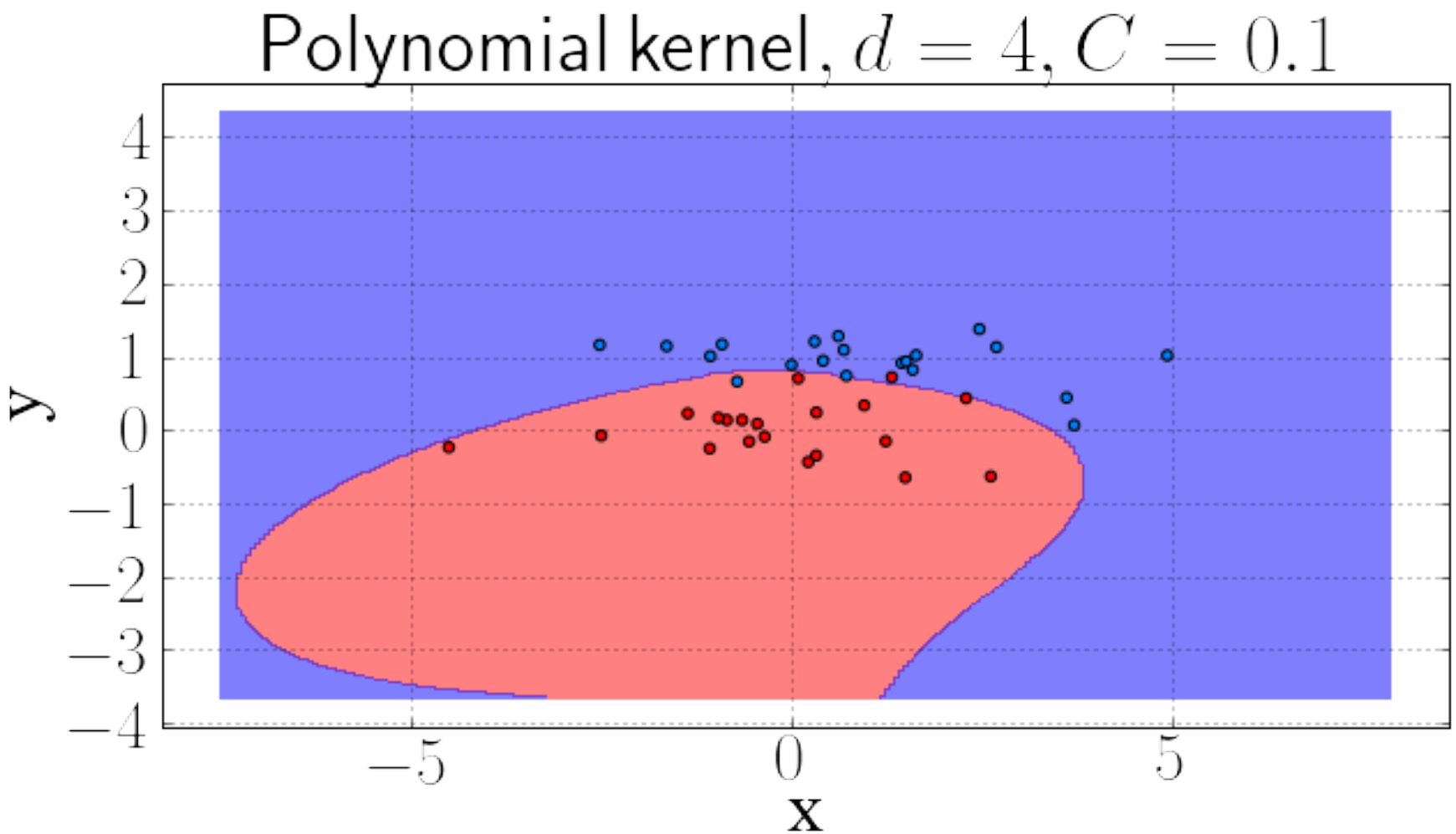
Polynomial kernel - variable d



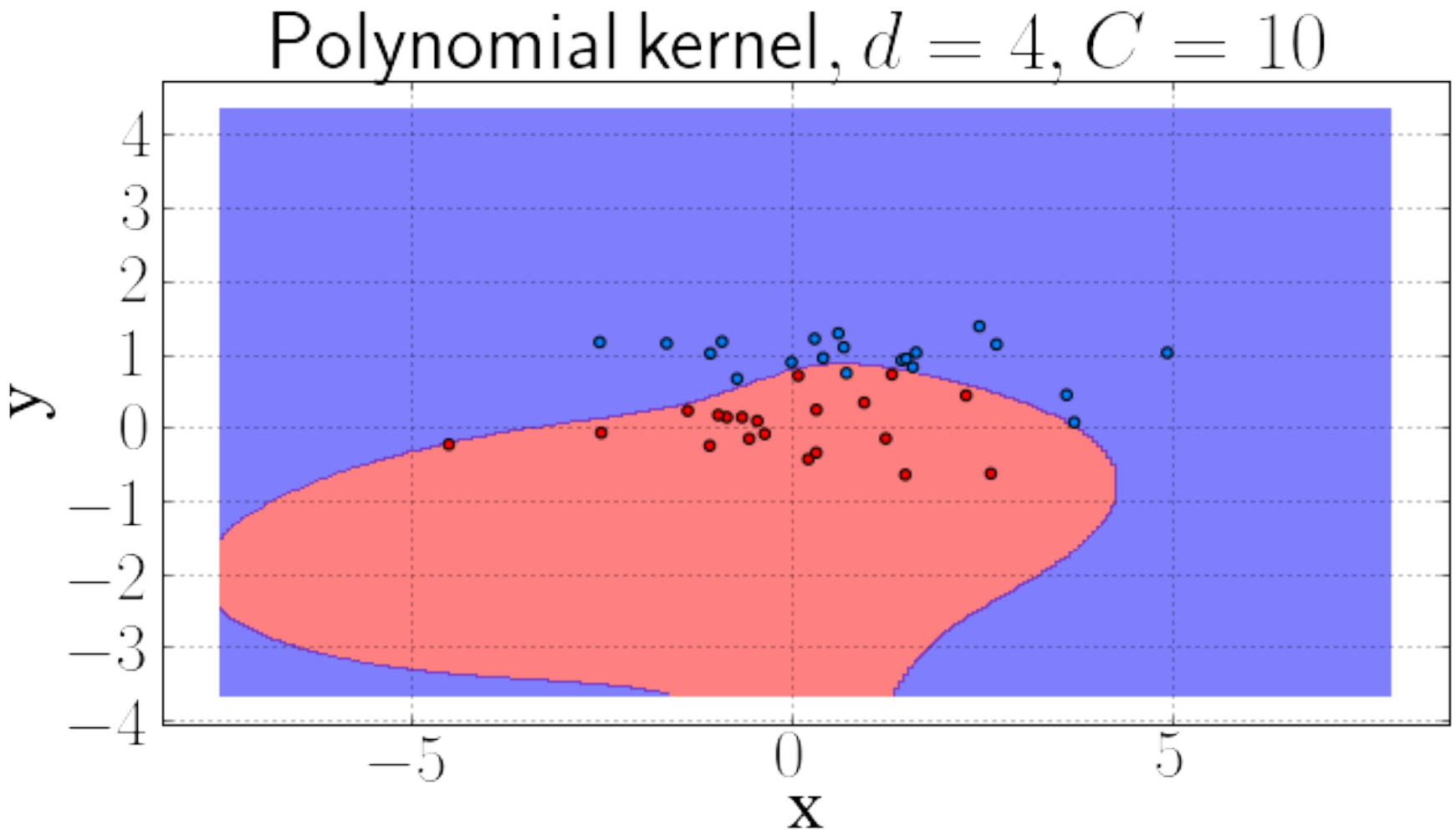
Polynomial kernel - variable C



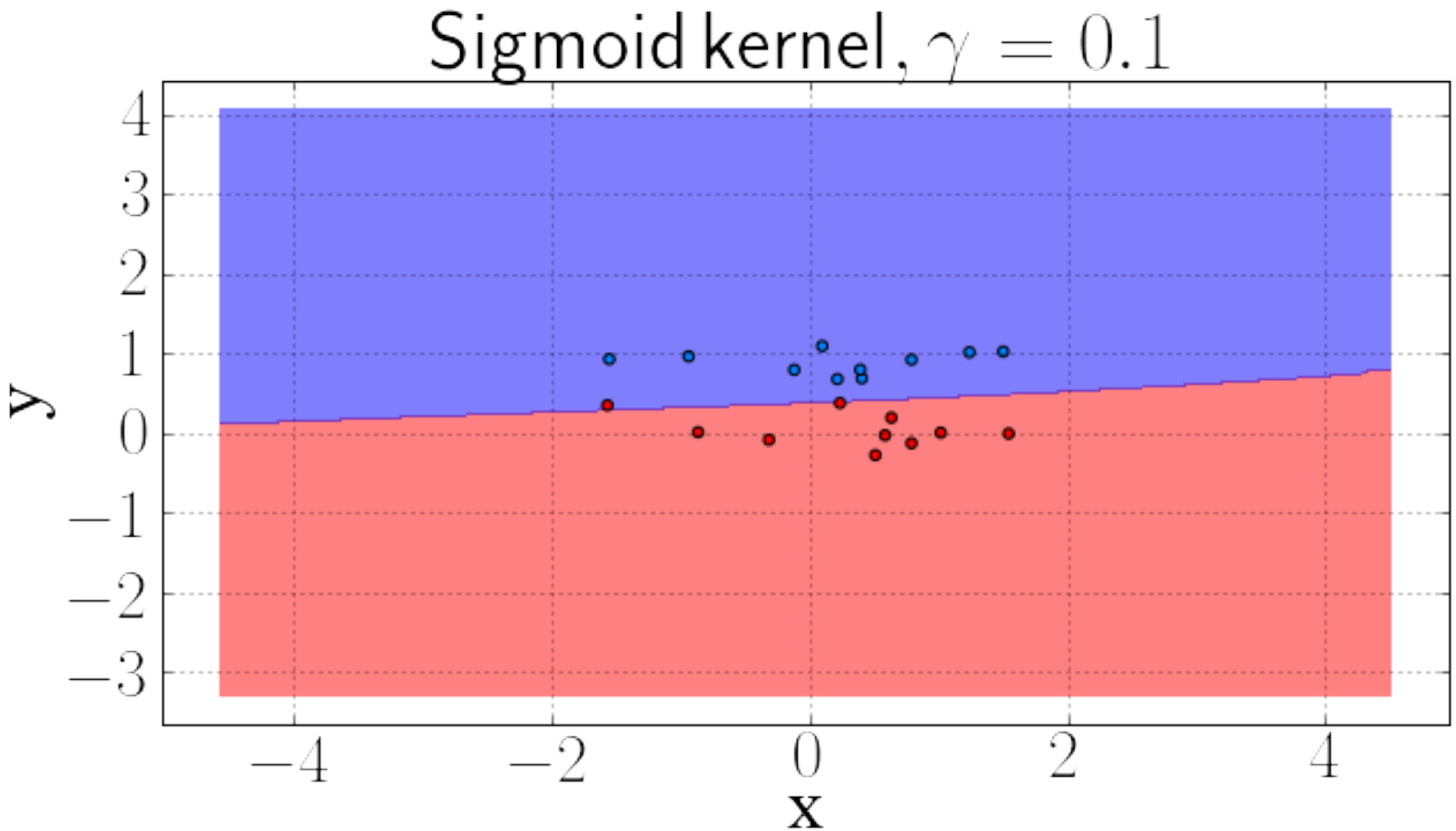
Polynomial kernel - variable C



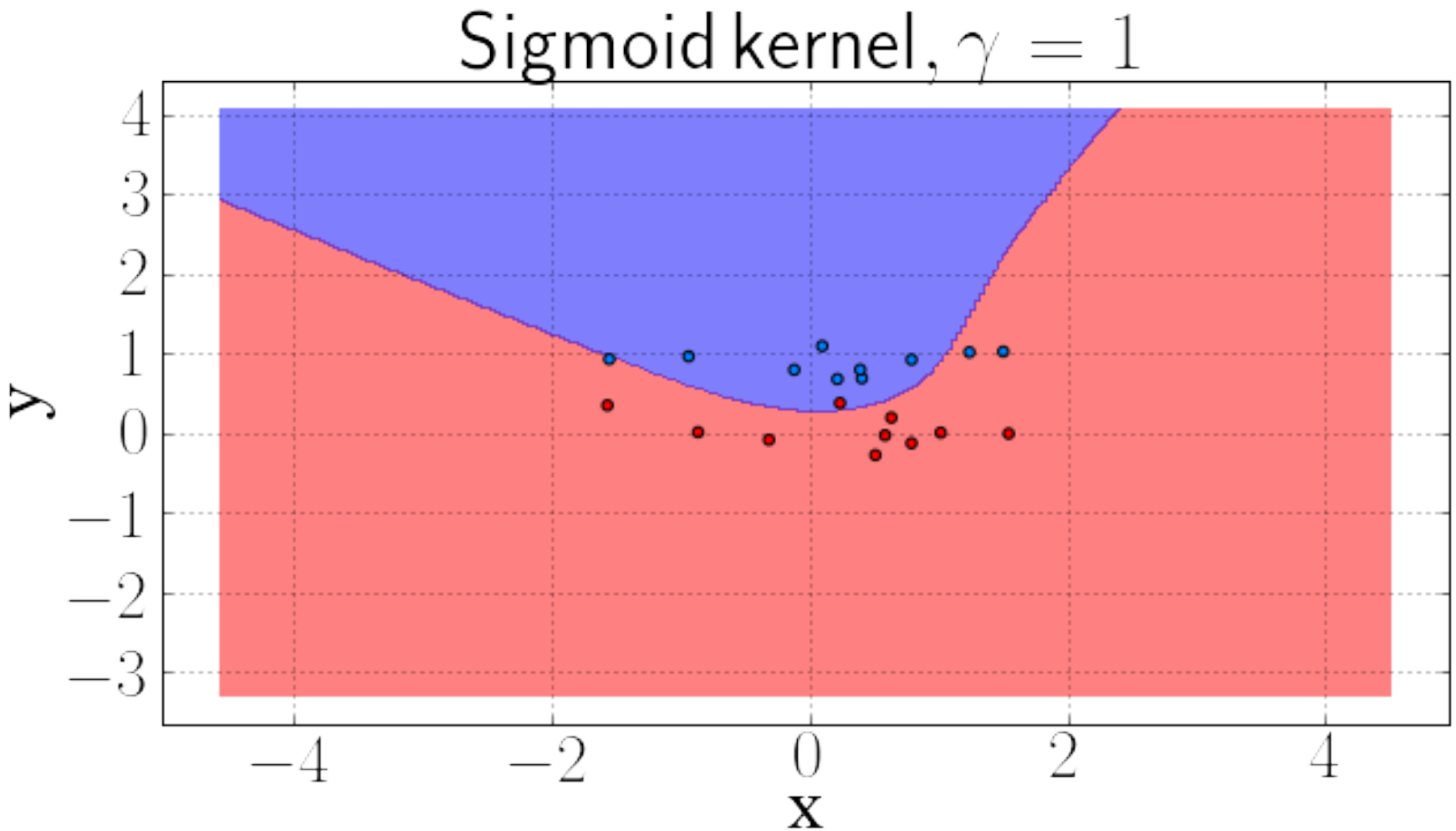
Polynomial kernel - variable C



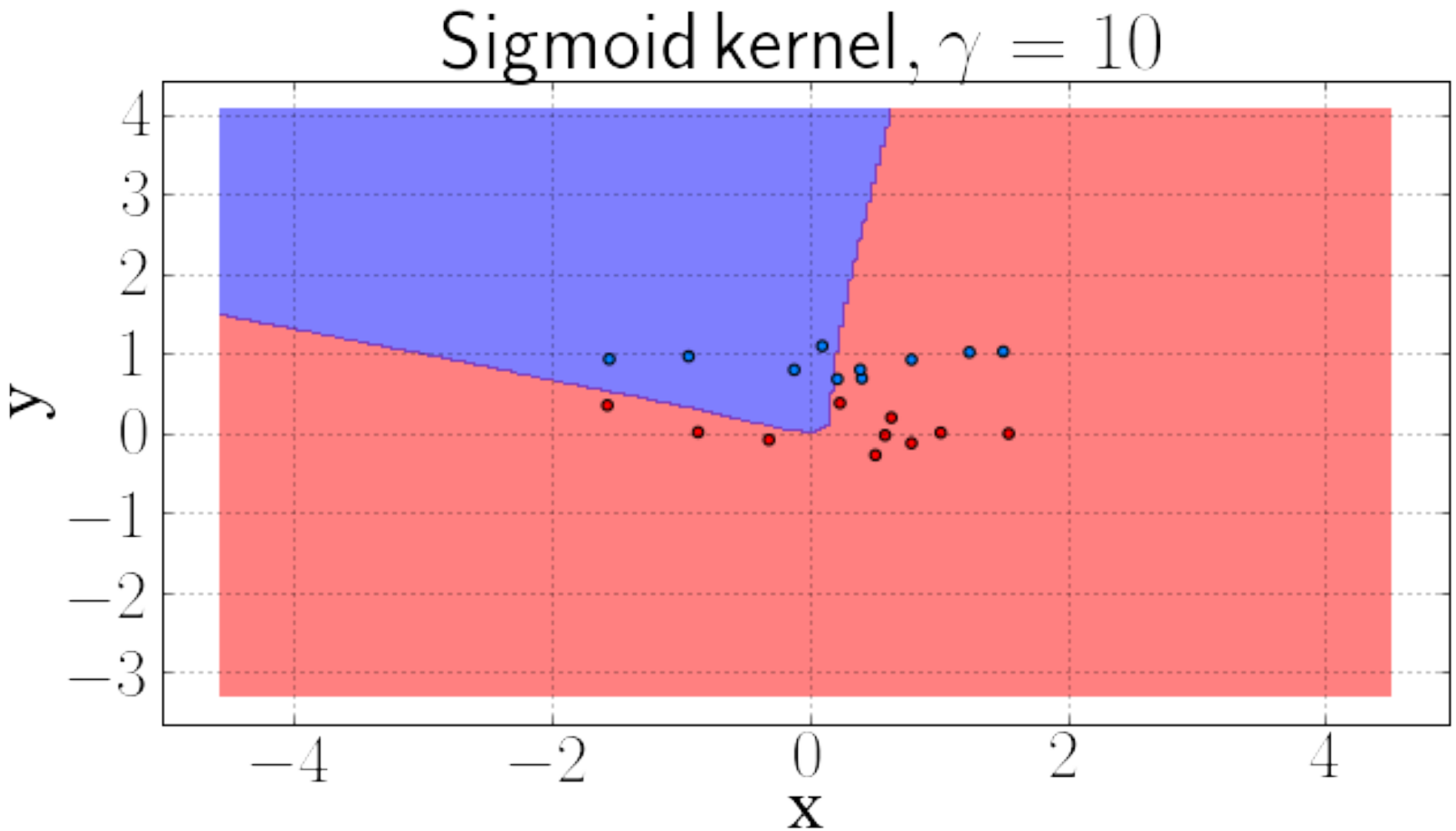
Sigmoid kernel - variable γ



Sigmoid kernel - variable γ



Sigmoid kernel - variable γ



Sigmoid kernel - variable C

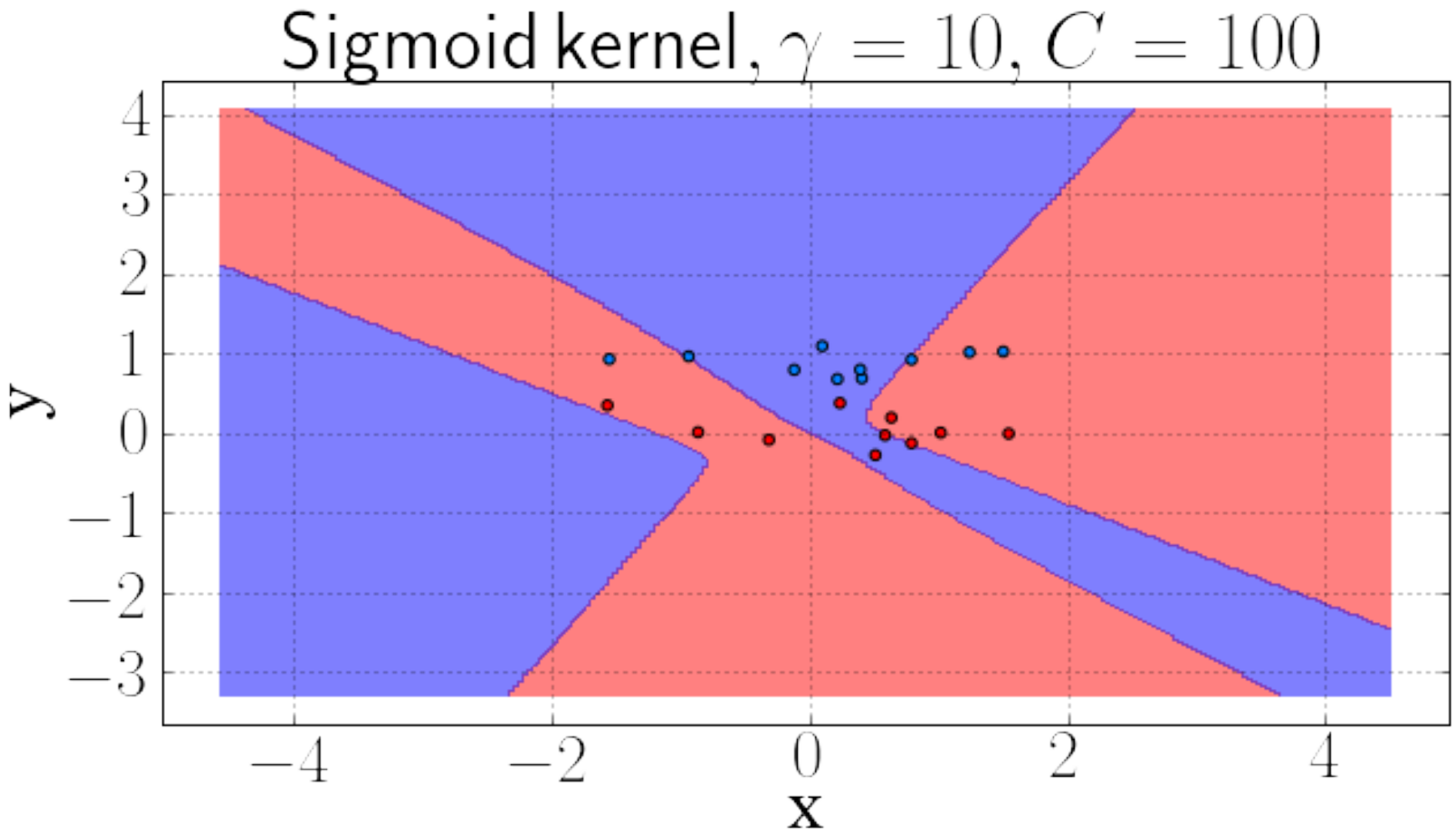


Table of Contents

Deep learning

- Multiple layers
- Layers trained one by one:
  - using restricted Bolzman machines
  - using autoencoders
- Autoencoders
  - bottleneck layer
  - sparse regularization
  - denoising autoencoders

Other

- Recommended materials in deep learning:
  - deeplearning.net
  - articles/lectures of Hinton, Bengio, LeCun
- Other types of learning:
  - transductive, semi-supervised, active, reinforcement, multi-task, transfer, representation.