Kernel trick

Victor Kitov

v.v.kitov@yandex.ru

Kernel trick

Perform feature transformation: $x \to \phi(x)$. Scalar product becomes $\langle x, x' \rangle \to \langle \phi(x), \phi(x') \rangle = K(x, x')$

Kernel trick

Define not the feature representation x but only scalar product function $K(x, x')$

- Comments:
  - required that the solution depends only on scalar products.Kernels can be constructed from other kernels, for example from:
    1. scalar product $\langle x, x' \rangle$
    2. constant $K(x, x') \equiv 1$
    3. $x^T A x$ for any $A \succcurlyeq 0$[1]
  - feature representation $\phi(x)$ not needed
  - $\langle x, x' \rangle$ has complexity $O(D)$. Complexity of $K(x, x')$ may be $O(1)$.
  - we can define natural linear kernel:

Kernelizable algorithms

- ridge regression:
- K-NN
- K-means
- PCA
- SVM
- many more...

Kernel trick use cases

- high-dimensional data
  - polynomial of order up to M
  - Gaussian kernel $K(x,x') = e^{-\frac{1}{2\sigma^2}\left\|x-x'\right\|^2}$ corresponds to infinite-dimensional feature space.
- hard to vectorize data
  - strings, sets, images, texts, graphs, 3D-structures, sequences, etc.
- natural scalar product exist
  - strings: number of co-occuring substrings
  - sets: size of intersection of sets
    - example: for sets S_1 and S_2: $K(S_1,S_2) = 2^{|S_1\cap S_2|}$ is a possible kernel.
  - etc.
- scalar product can be computed efficiently

General motivation for kernel trick

- perform generalization of linear methods to non-linear case
  - as efficient as linear methods
  - local minimum is global minimum
  - no local optima=>less overfitting
- non-vectorial objects
  - hard to obtain vector representation

Kernel definition

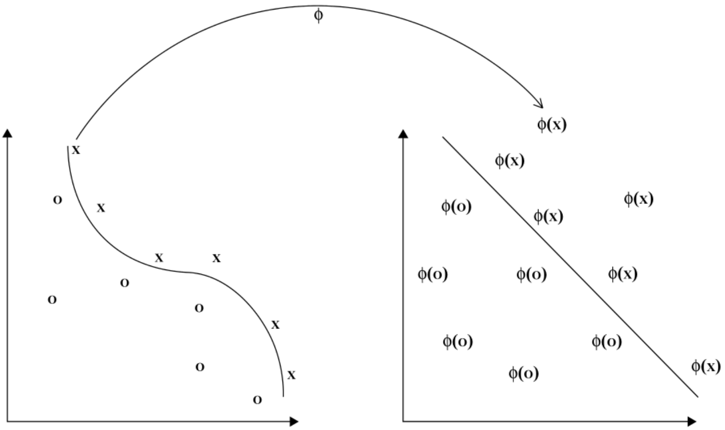
- x is replaced with $\phi(x)$
  - Example: $[x] \to [x,\ x^2,\ x^3]$

Kernel

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

- $\langle x, x' \rangle$ is replaced by $\langle \phi(x), \phi(x') \rangle = K(x, x')$

Illustration



Specific types of kernels

- $K(x, x') = K(x - x')$ - stationary kernels (invariant to translations)
- $K(x, x') = K(\|x - x'\|)$ - radial basis functions

Polynomial kernel[2]

- Example 1: let $D = 2$.

$$\begin{aligned} K(x,z) &= (x^T z)^2 = (x_1 z_1 + x_2 z_2)^2 = \\ &= x_1^2 z_1^2 + x_2^2 z_2^2 + 2 x_1 z_1 x_2 z_2 \\ &= \phi^T(x)\phi(z) \end{aligned}$$

for $\phi(x) = (x_1^2, x_2^2, \sqrt{2} x_1 x_2)$

[2]What kind of feature transformation will correspond to $K(x,z) = (x^T z)^M$ for arbitrary M and D?

Polynomial kernel[3]

- Example 2: let $D=2$.

$$\begin{aligned} K(x,z) &= (1+x^T z)^2 = (1+x_1 z_1 + x_2 z_2)^2 = \\ &= 1 + x_1^2 z_1^2 + x_2^2 z_2^2 + 2x_1 z_1 + 2x_2 z_2 + 2x_1 z_1 x_2 z_2 \\ &= \phi^T(x)\phi(z) \end{aligned}$$

for $\phi(x) = (1,\, x_1^2,\, x_2^2,\, \sqrt{2}x_1,\, \sqrt{2}x_2,\, \sqrt{2}x_1 x_2)$

[3] What kind of feature transformation will correspond to $K(x,z) = (1+x^T z)^M$ kernels for arbitrary M and D?

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:
  - definition 1: for every function $g : X \to \mathbb{R}$

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

  - definition 2 (equivalent): for every finite set $x_1, x_2, ...x_M$ Gramm matrix $\{K(x_i, x_j)\}_{i,j=1}^{M} \succeq 0$ (p.s.d.)

Kernel construction

- Kernel learning - separate field of study.
- Hard to prove non-negative definitness of kernel in general.
- Kernels can be constructed from other kernels, for example from:
  1. scalar product $\langle x, x' \rangle$
  2. constant $K(x, x') \equiv 1$
  3. $x^T A x$ for any $A \succcurlyeq 0$[4]

[4] Under what feature transformation will case 1 transform to cases 2 and 3? You may use Choletsky decomposition.

Constructing kernels from other kernels

If $K_1(x, x')$, $K_2(x, x')$ are arbitrary kernels, $c > 0$ is a constant, $q(\cdot)$ is a polynomial with non-negative coefficients, $h(x)$ and $\varphi(x)$ are arbitrary functions $\mathcal{X} \to \mathbb{R}$ and $\mathcal{X} \to \mathbb{R}^M$ respectively, then these are valid kernels[5]:

1. $K(x, x') = cK_1(x, x')$
2. $K(x, x') = K_1(x, x')K_2(x, x')$
3. $K(x, x') = K_1(x, x') + K_2(x, x')$
4. $K(x, x') = K_1(\varphi(x), \varphi(x'))$
5. $K(x, x') = h(x)K_1(x, x')h(x')$
6. $K(x, x') = e^{K_1(x, x')}$

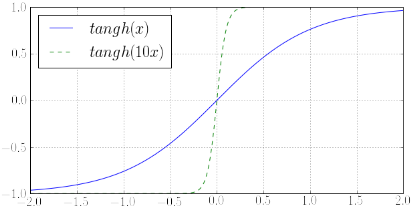
[5] prove some of these statements

Commonly used kernels

Let x and x' be two objects.

Kernel	Mathematical form
linear	$\langle x, x' \rangle$
polynomial	$(\gamma \langle x, x' \rangle + r)^d$
RBF	$\exp(-\gamma \left\Vert x - x' \right\Vert^2)$

- Standard transformation is also sigmoid=tangh$(\gamma \langle x, y \rangle + r)$ but its not a Mercer kernel.



Addition[6]

- Other kernelized algorithms: K-NN, K-means, K-medoids, nearest medoid, PCA, SVM, etc.
- Kernelization of distance:

[6] How can we calculate scalar product between normalized (unit norm) vectors $\phi(x)$ and $\phi(x')$?

Addition[6]

- Other kernelized algorithms: K-NN, K-means, K-medoids, nearest medoid, PCA, SVM, etc.
- Kernelization of distance:

$$\begin{aligned}\rho(x,x')^2 &= \langle\phi(x)-\phi(x'),\phi(x)-\phi(x')\rangle \\ &= \langle\phi(x),\phi(x)\rangle+\langle\phi(x'),\phi(x')\rangle-2\langle\phi(x),\phi(x')\rangle \\ &= K(x,x)+K(x',x')-2K(x,x')\end{aligned}$$

[6] How can we calculate scalar product between normalized (unit norm) vectors $\phi(x)$ and $\phi(x')$?

Table of Contents

Making predictions

1. Solve dual task to find α_i^*, $i = 1, 2, ...N$

$$\begin{cases} \sum_{i=1}^N \alpha_i - \frac{1}{2}\sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y_i y_j \langle x_i, x_j \rangle \to \max_\alpha \\ \sum_{i=1}^N \alpha_i y_i = 0 \\ 0 \le \alpha_i \le C \quad \text{(using (\textbf{??}) and that } \alpha_i \ge 0,\ r_i \ge 0) \end{cases}$$

2. Find optimal w_0:

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

3. Make prediction for new x:

$$\widehat{y} = \text{sign}[w^T x + w_0] = \text{sign}[\sum_{i \in \mathcal{SV}} \alpha_i^* y_i \langle x_i, x \rangle + w_0]$$

Making predictions

1. Solve dual task to find α_i^*, $i = 1, 2, ...N$

$$\begin{cases} L_D = \sum_{i=1}^N \alpha_i - \frac{1}{2}\sum_{i=1}^N \sum_{j=1}^N \alpha_i\alpha_j y_i y_j \langle x_i, x_j \rangle \to \max_\alpha \\ \sum_{i=1}^N \alpha_i y_i = 0 \\ 0 \le \alpha_i \le C \quad \text{(using (??) and that } \alpha_i \ge 0,\ r_i \ge 0) \end{cases}$$

2. Find optimal w_0:

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

3. Make prediction for new x:

$$\hat{y} = \text{sign}[w^T x + w_0] = \text{sign}[\sum_{i\in\mathcal{SV}} \alpha_i^* y_i \langle x_i, x \rangle + w_0]$$

- On all steps we don't need exact feature representations, only scalar products $\langle x, x' \rangle$!

Kernel trick generalization

1. Solve dual task to find α_i^*, $i = 1, 2, ...N$

$$\begin{cases} L_D = \sum_{i=1}^N \alpha_i - \frac{1}{2}\sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y_i y_j K(x_i, x_j) \to \max_\alpha \\ \sum_{i=1}^N \alpha_i y_i = 0 \\ 0 \le \alpha_i \le C \quad \text{(using (??) and that } \alpha_i \ge 0,\ r_i \ge 0) \end{cases}$$

2. Find optimal w_0:

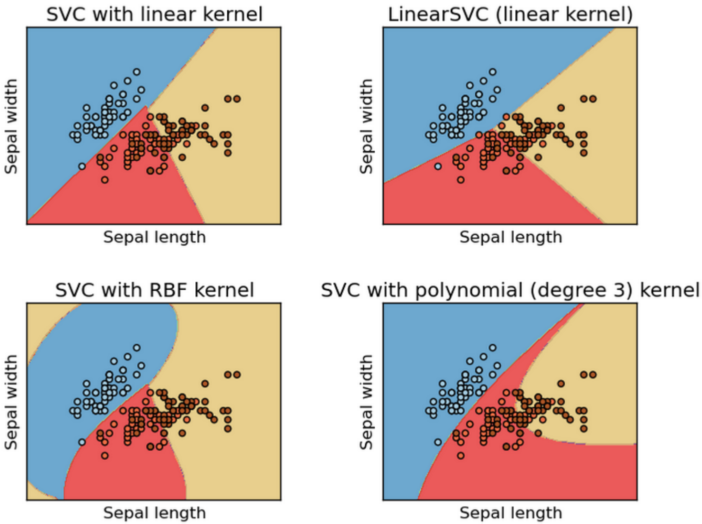
$$w_0 = \frac{1}{n_{\tilde{SV}}}\left(\sum_{j \in \tilde{SV}} y_j - \sum_{j \in \tilde{SV}} \sum_{i \in \mathcal{SV}} \alpha_i^* y_i K(x_i, x_j)\right)$$

3. Make prediction for new x:

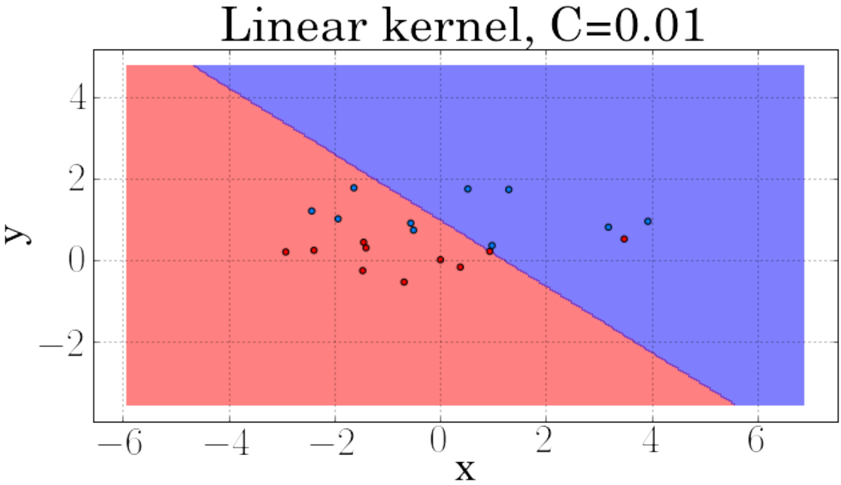
$$\hat{y} = \text{sign}[w^T x + w_0] = \text{sign}[\sum_{i \in \mathcal{SV}} \alpha_i^* y_i K(x_i, x_j) + w_0]$$

- We replaced $\langle x, x' \rangle \to K(x, x')$ for $K(x, x') = \langle \phi(x), \phi(x') \rangle$ for some feature transformation $\phi(\cdot)$.

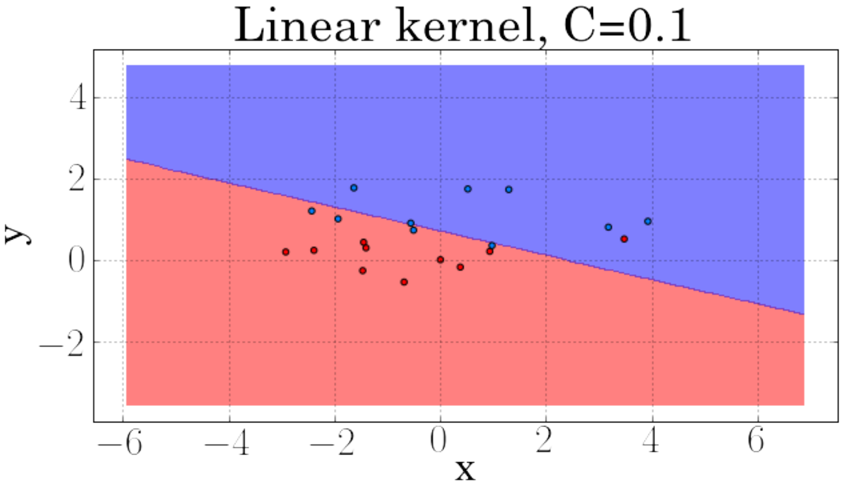
Kernel results



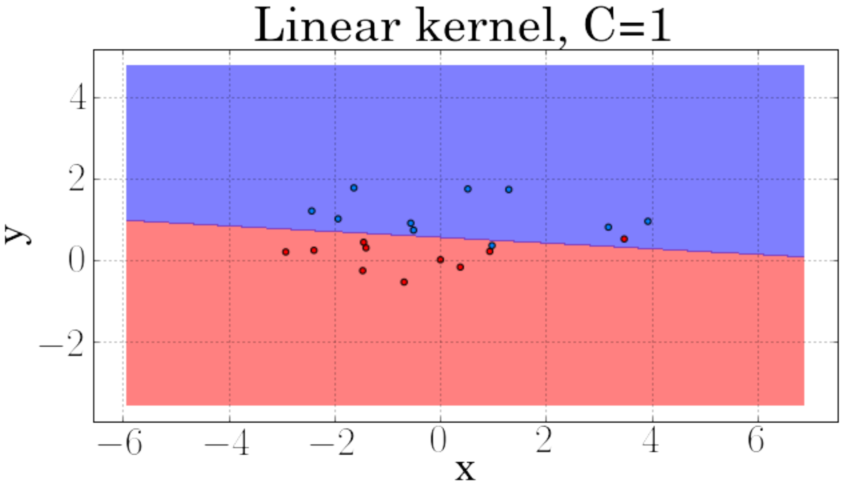
Linear kernel - variable C



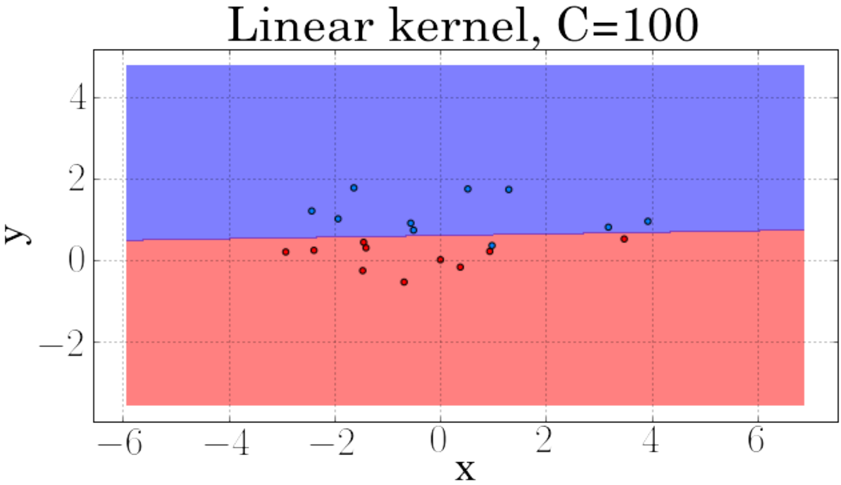
Linear kernel - variable C



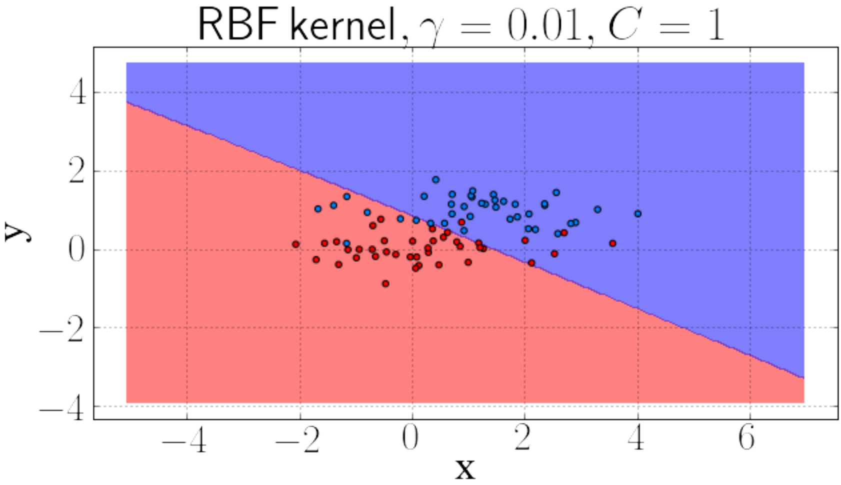
Linear kernel - variable C



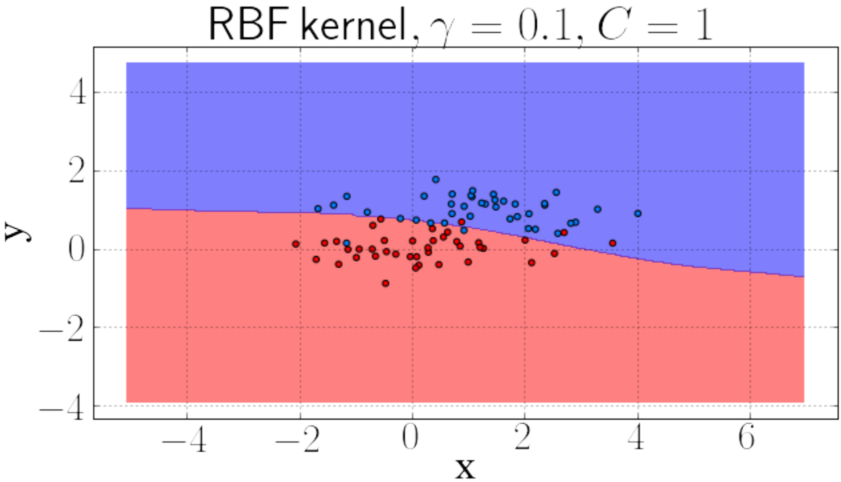
Linear kernel - variable C



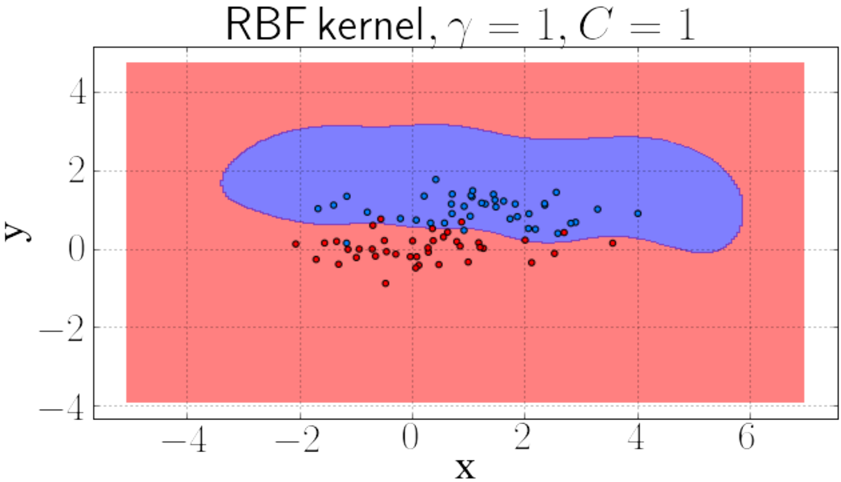
RBF kernel - variable γ



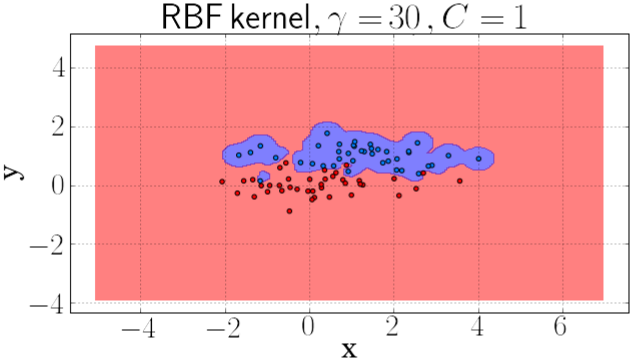
RBF kernel - variable γ



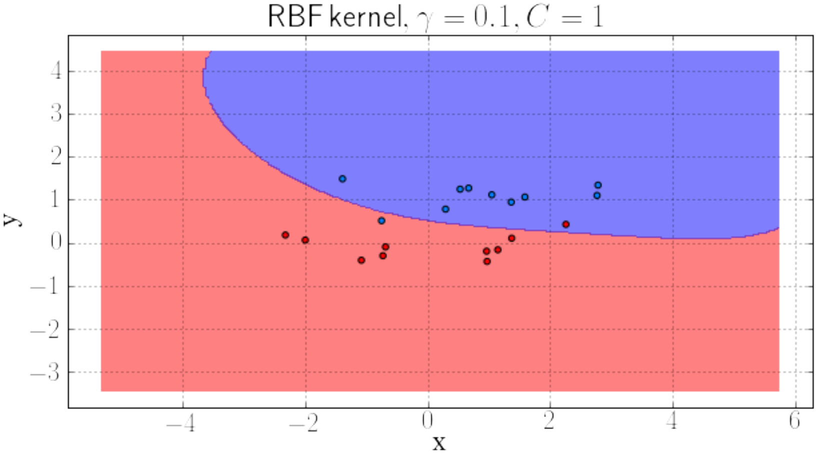
RBF kernel - variable γ



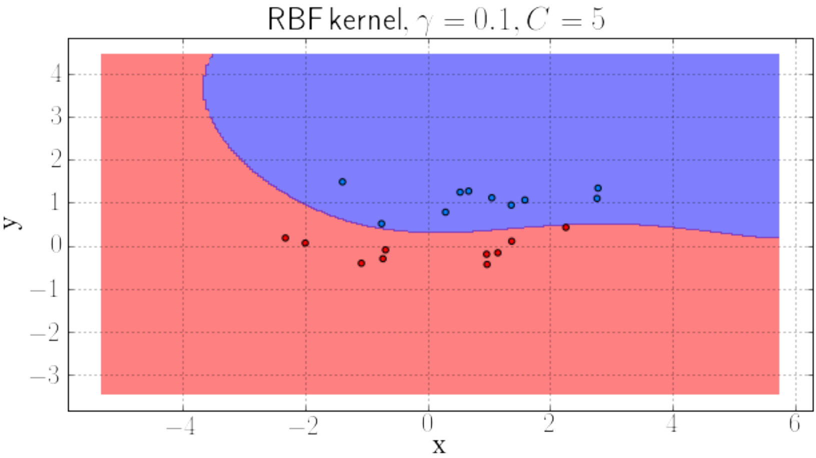
RBF kernel - variable γ



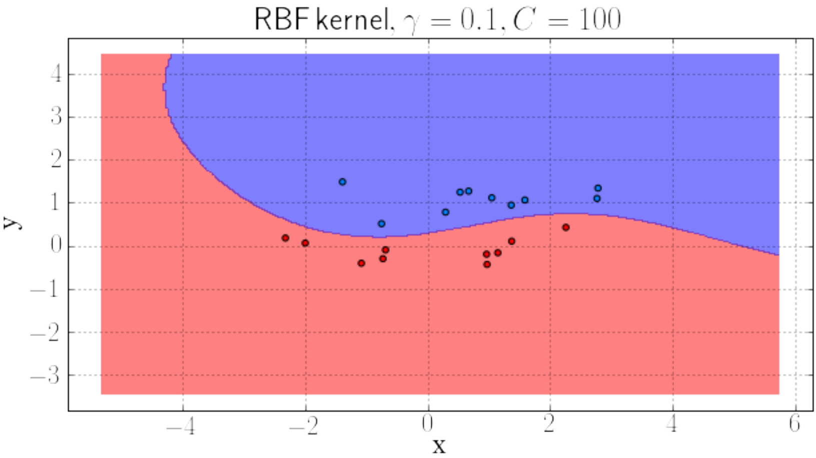
RBF kernel - variable C



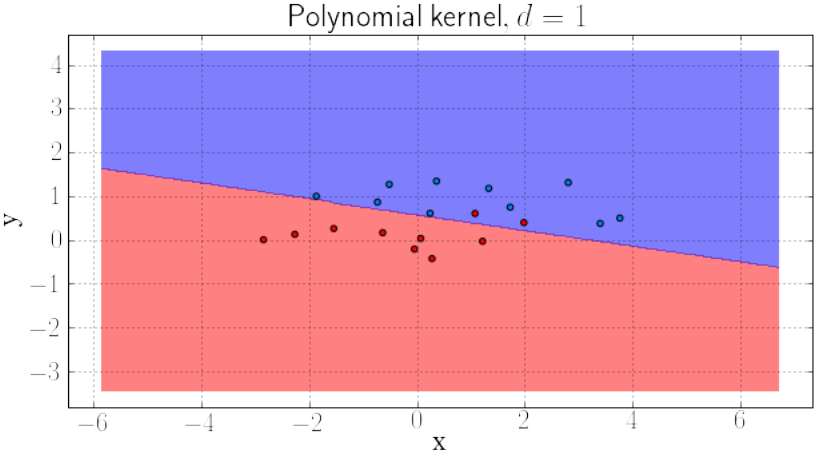
RBF kernel - variable C



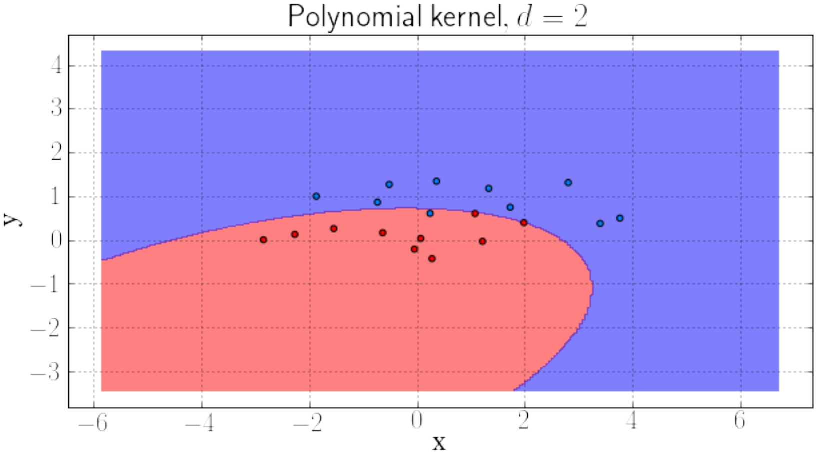
RBF kernel - variable C



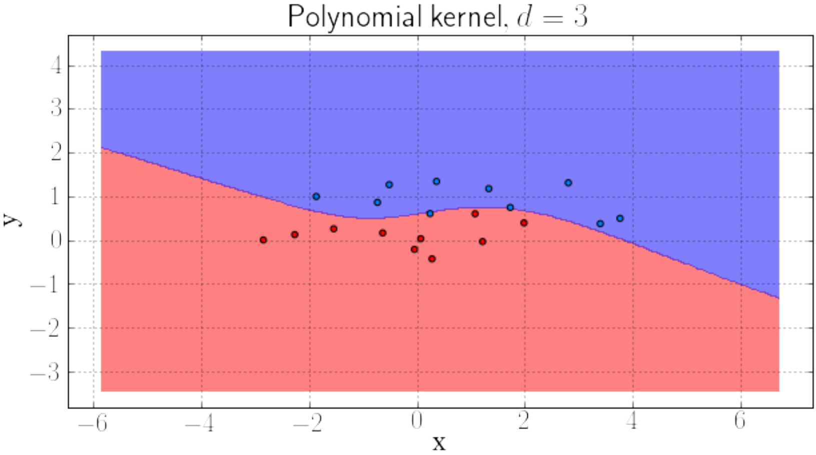
Polynomial kernel - variable d



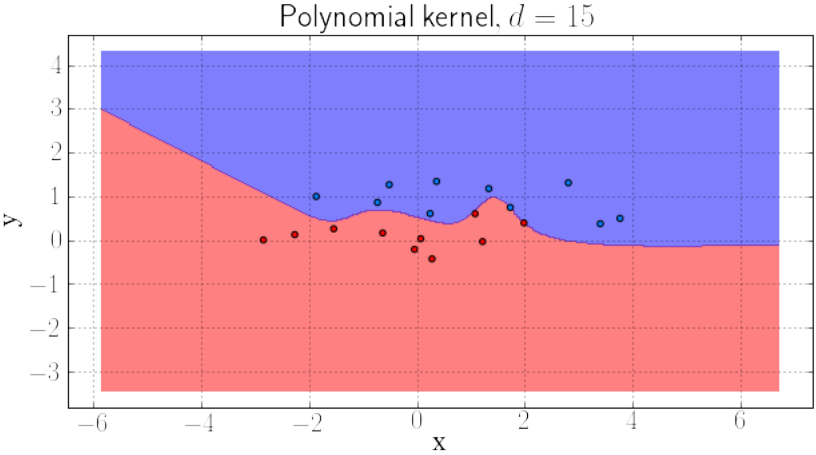
Polynomial kernel - variable d



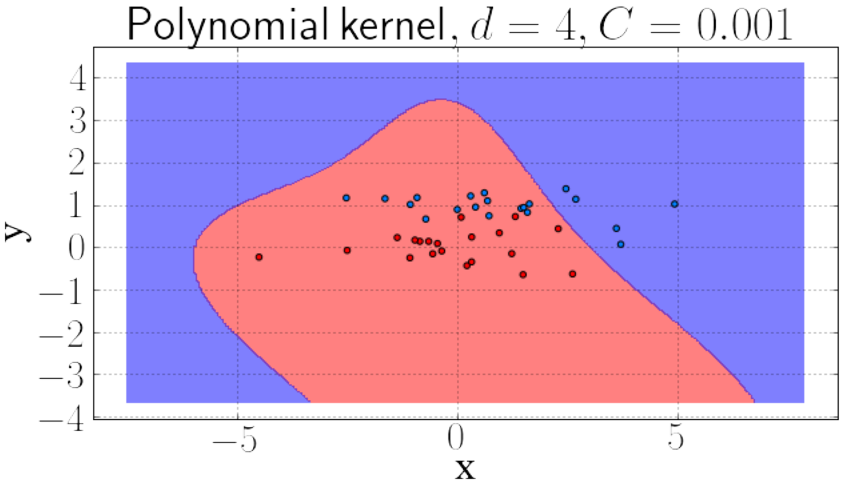
Polynomial kernel - variable d



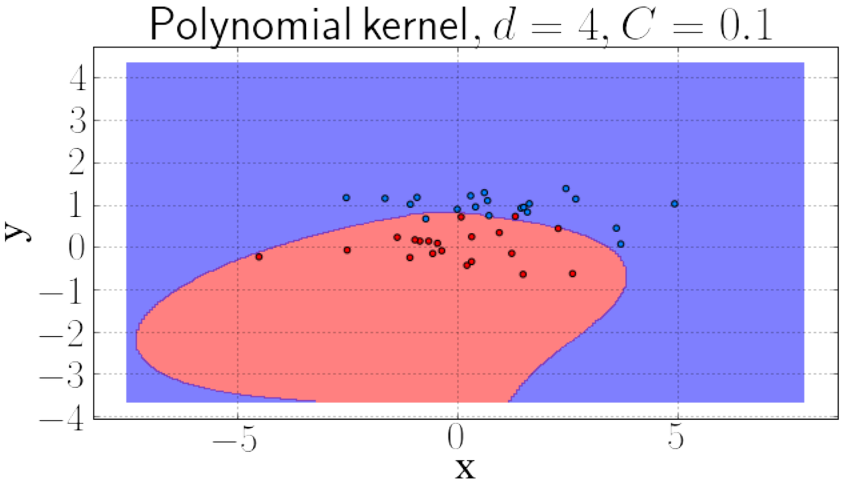
Polynomial kernel - variable d



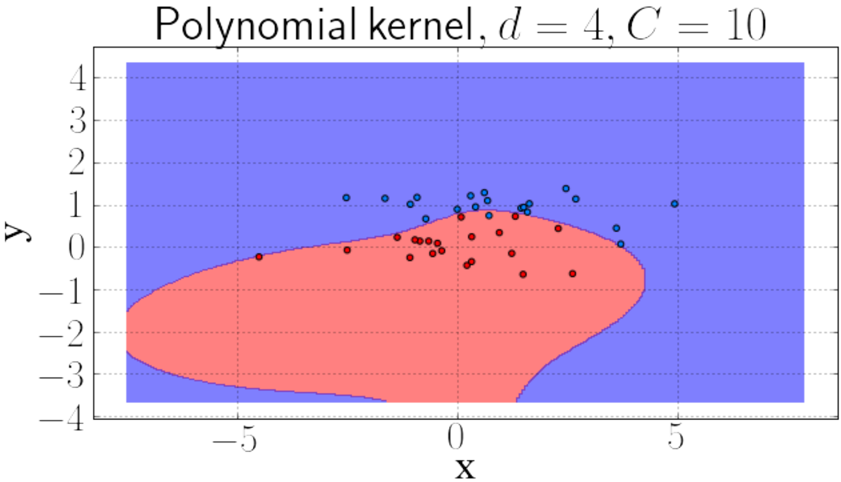
Polynomial kernel - variable C



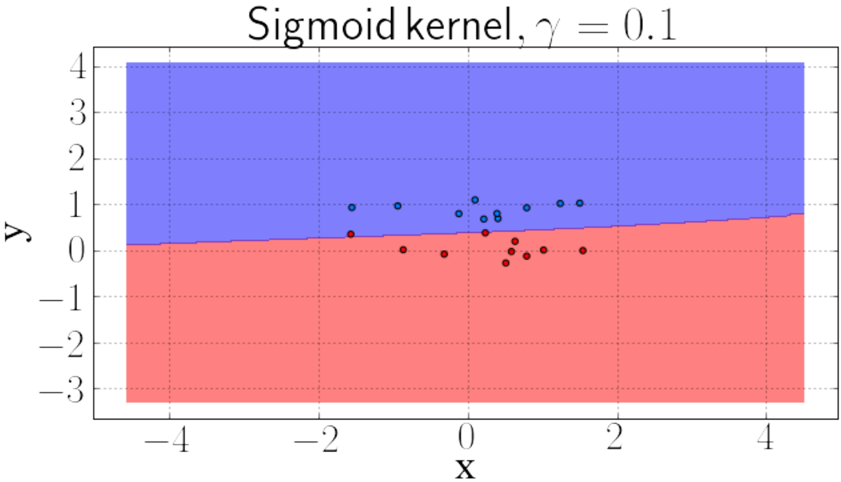
Polynomial kernel - variable C



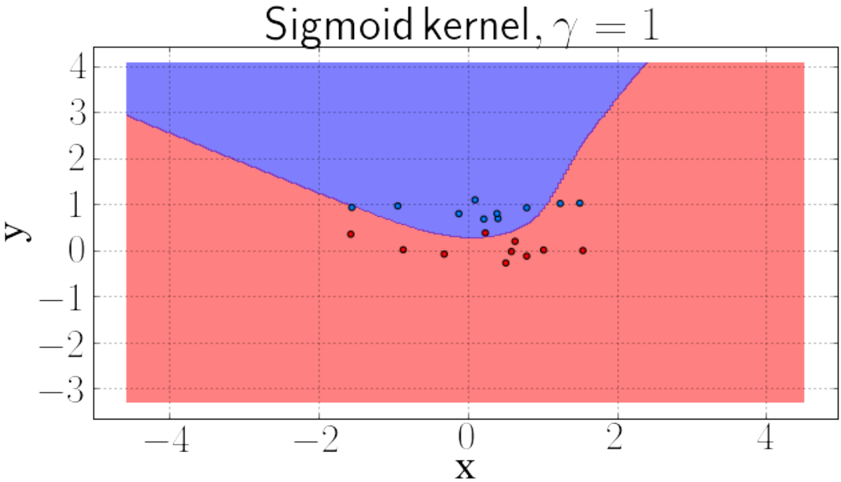
Polynomial kernel - variable C



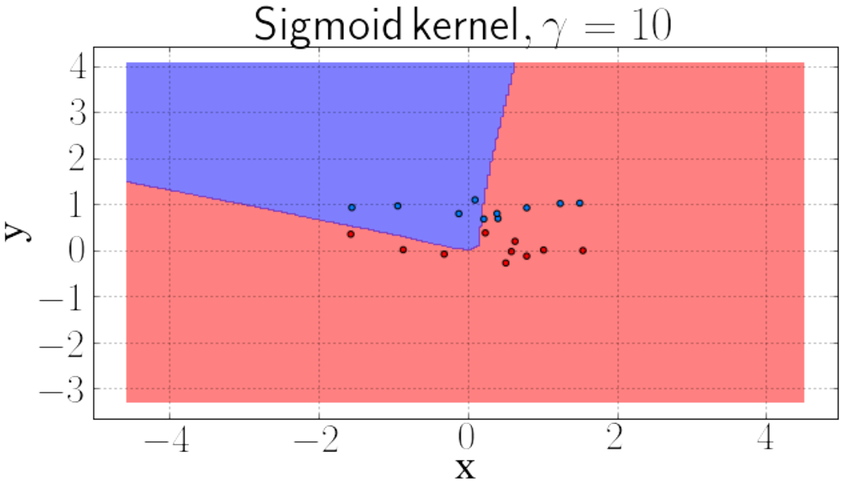
Sigmoid kernel - variable γ



Sigmoid kernel - variable γ



Sigmoid kernel - variable γ



Sigmoid kernel - variable C

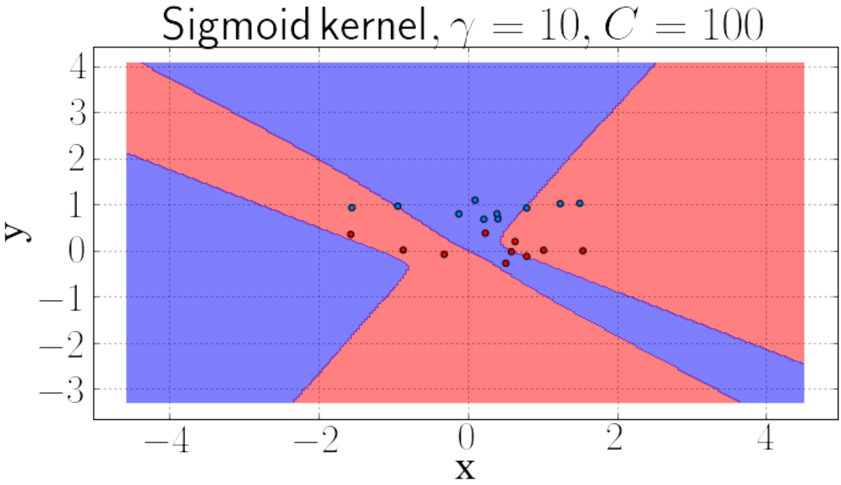


Table of Contents

Ridge regression

- Ridge regression criterion:

$$Q(\beta)=\sum_{n=1}^{N}\left(x_n^T\beta-y_n\right)^2+\lambda\sum_{d=1}^{D}\beta_d^2\to\min_{\beta}$$

- Stationarity condition:

$$\frac{dQ(\beta)}{d\beta}=2\sum_{n=1}^{N}\left(x_n^T\beta-y_n\right)x_n+2\lambda\beta=0$$

- In vector form:

$$X^T\left(X\beta-Y\right)+\lambda\beta=0$$

Ridge regression

- Primal solution:

$$X^TX + \lambda I\beta = X^TY$$
$$\beta = (X^TX + \lambda I)^{-1}X^TY$$

- Comment: $X^TX \succcurlyeq 0$ (positive semi-definite) and $X^TX + \lambda I \succ 0$ (positive definite), so ridge regression is always identifiable.
- Cost of estimation:
    - $X^TX + \lambda I$: $ND^2 + D$
    - X^TY: DN
    - $(X^TX + \lambda I)^{-1}$: D^3
    - $(X^TX + \lambda I)^{-1}X^TY$: D^2
    - Total training cost is $O(ND^2 + D^3) = O(D^2(N + D))$.
- Cost of prediction $\widehat{y}(x) = \langle x, \beta \rangle$ is D.

Dual solution

From vector stationarity condition:

$$X^T(X\beta - Y) + \lambda\beta = 0$$

follows the dual solution (a linear combination of training vectors):

$$\beta = \frac{1}{\lambda}X^T(Y - X\beta) = X^T\alpha \tag{1}$$

where

$$\alpha = \frac{1}{\lambda}(Y - X\beta) \tag{2}$$

is called a vector of *dual variables*.

Prediction:

$$\widehat{y}(x) = x^T\beta = x^TX^T\alpha = \sum_{i=1}^{N}\alpha_i\langle x, x_i\rangle$$

Dual solution

To find α we plug (1) into (2):

$$\alpha = \frac{1}{\lambda}(Y - X\beta) = \frac{1}{\lambda}(Y - XX^T\alpha)$$
$$\left(XX^T + \lambda I\right)\alpha = Y$$
$$\alpha = \left(XX^T + \lambda I\right)^{-1} Y$$

Cost of estimation:
$XX^T + \lambda I$: $N^2D + N$
$\left(XX^T + \lambda I\right)^{-1}$: N^3
$\left(XX^T + \lambda I\right)^{-1} Y$: N^2
Total training cost is $O(N^2D + N^3) = O(N^2(D + N))$.
Cost of prediction $\widehat{y}(x) = \langle x, \beta \rangle$ is ND.

Dual solution motivation

- Optimal α depends not on exact features but only on scalar products:

$$\alpha = \left(XX^T + \lambda I\right)^{-1} Y = (G + \lambda I)^{-1} Y$$

where $G \in \mathbb{R}^{NxN}$ and $\{G\}_{ij} = \langle x_i, x_j \rangle$ - G is called *Gram matrix*.

- Prediction also depends only on scalar products:

$$\widehat{y}(x) = \sum_{i=1}^{N} \alpha_i \langle x, x_i \rangle$$

- Cost of prediction $\widehat{y}(x)$ is *ND*.

Kernel ridge regression

Ridge regression for general $K(x, x')$:

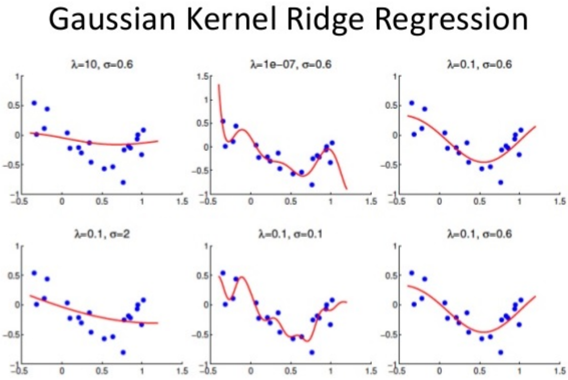
1. Estimate Gramm matrix $\{G\}_{i,j} = K(x_i, x_j)$
2. Estimate *dual variables*

$$\alpha = (G + \lambda I)^{-1} Y$$

3. For every x make prediction with

$$\widehat{y}(x) = \sum_{i=1}^{N} \alpha_i K(x, x_i)$$

Visualization



Introduction to RKHS, and some simple kernel Algorithms, Arthur Gretton, January 27, 2015

Decresing λ or decreasing σ leads to more complex model in ridge regression with Gaussian (RBF) kernel.