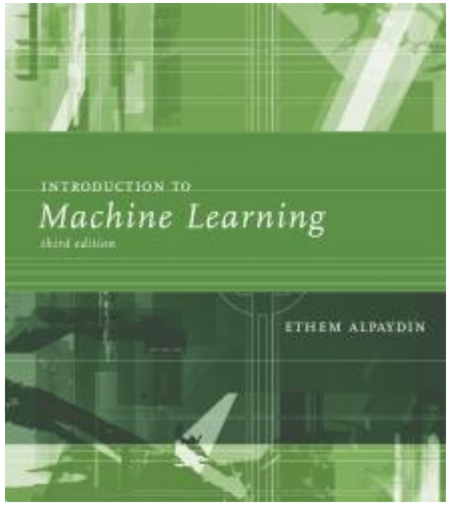


Support Vector Machines (SVM)

Lecture notes by Ethem Alpaydın
Introduction to Machine Learning (Boğaziçi Üniversitesi)

Lecture notes by Kevyn Collins-Thompson
Applied Machine Learning (Coursera)

Kernel Machines

- Discriminant-based: No need to estimate densities first
  - It is not necessary to estimate where the class densities $P(x|C_i)$ or the exact posterior probability values $P(C_i|x)$
  - We only need to estimate where the class boundaries lie.
- Define the discriminant in terms of support vectors
  - After training, the parameter of the linear model, the weight vector can be written down in terms of a subset of the training set, which are so-called support vectors.

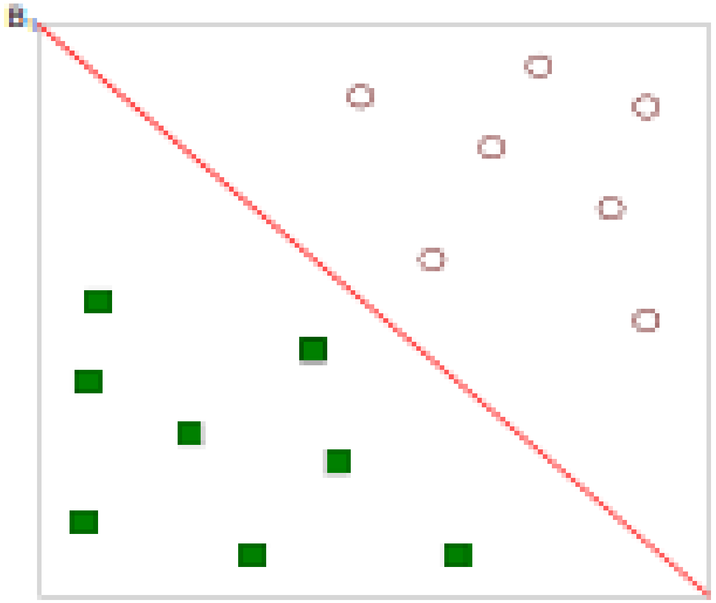
Kernel Machines

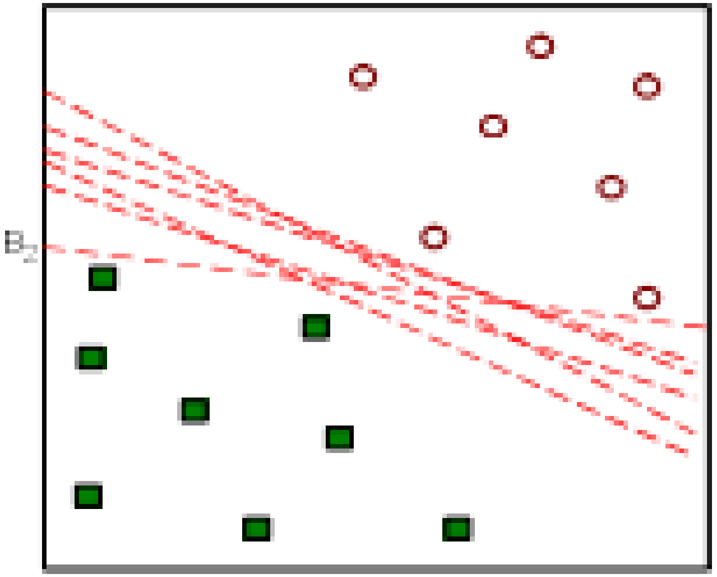
- The output is written as a sum of the influence of support vectors and these are given by **kernel functions** that are application-specific measures of similarity between data instances
- No need to represent instances as vectors

Kernel Machines

- Kernel-based algorithms are formulated as convex optimization problems and there is a single optimum that we can solve for analytically.

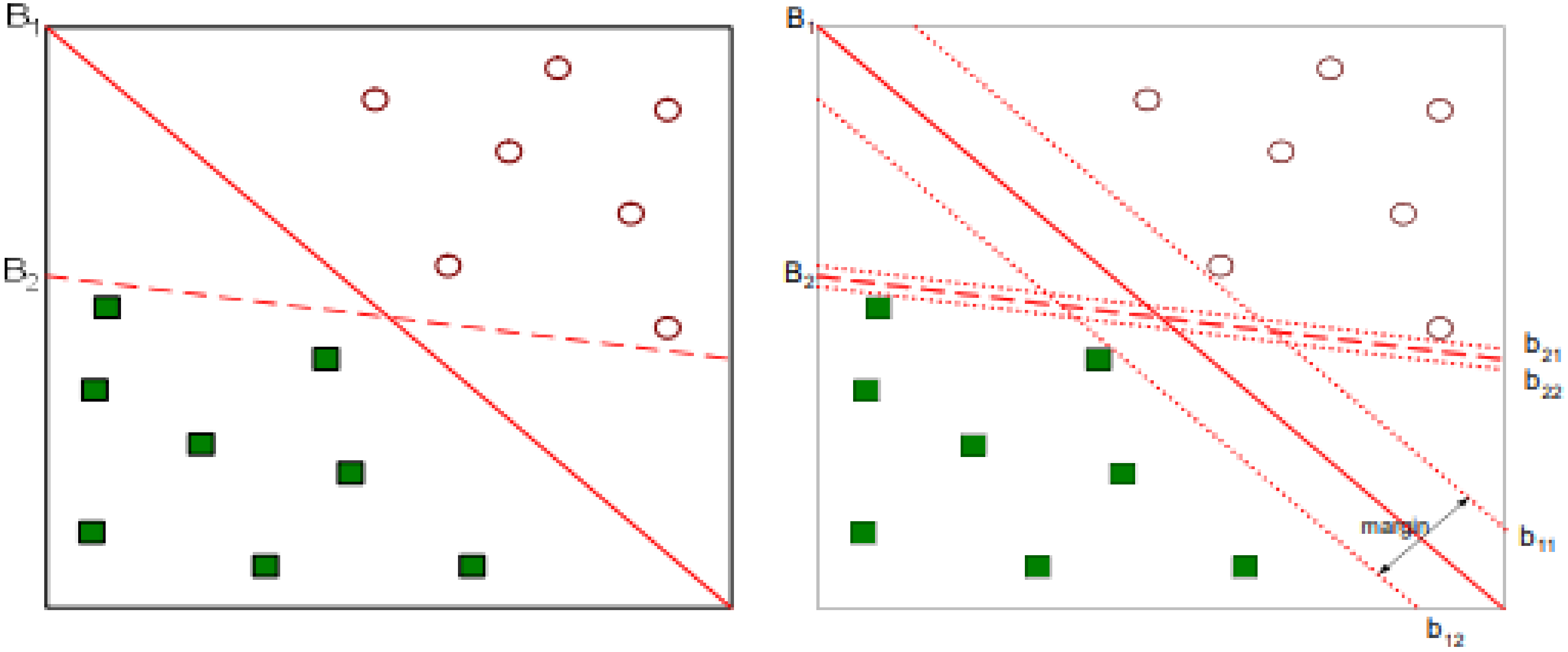
Decision Boundaries





https://ninova.itu.edu.tr/tr/dersler/fen-bilimleri-enstitusu/998/blg-607/

Decision Boundaries



Which one is better? B_1 or B_2

https://ninova.itu.edu.tr/tr/dersler/fen-bilimleri-enstitusu/998/blg-607/

Optimal Separating Hyperplane

$$\mathcal{X} = \left\{\mathbf{x}^t, r^t\right\}_t \text{ where } r^t = \begin{cases} +1 & \text{if } \mathbf{x}^t \in C_1 \\ -1 & \text{if } \mathbf{x}^t \in C_2 \end{cases}$$

find $\mathbf{w}$ and w_0 such that

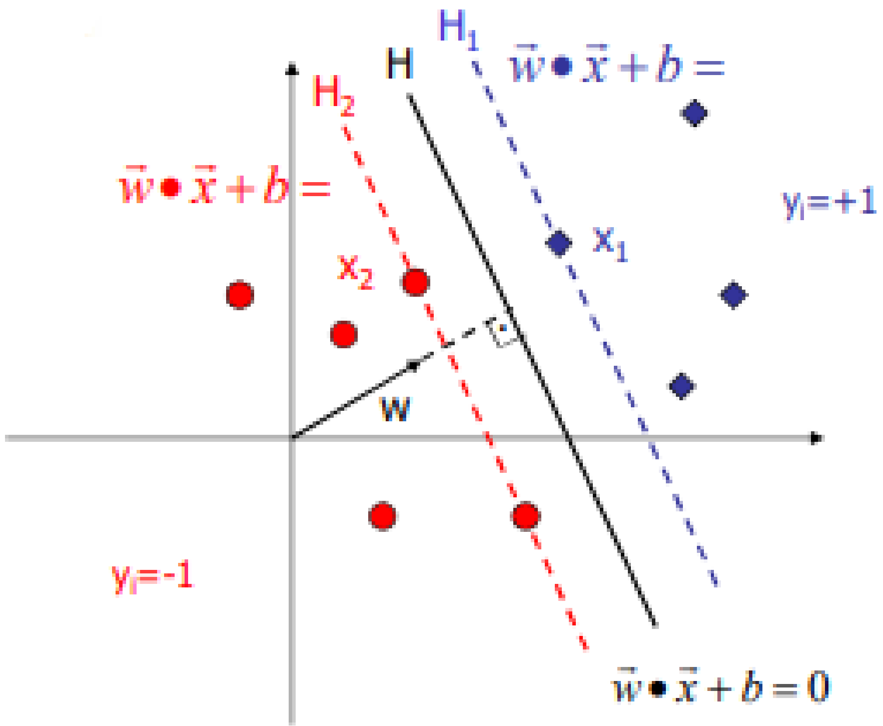
$$\mathbf{w}^T \mathbf{x}^t + w_0 \geq +1 \text{ for } r^t = +1$$

$$\mathbf{w}^T \mathbf{x}^t + w_0 \leq -1 \text{ for } r^t = -1$$

which can be rewritten as

$$r^t \left(\mathbf{w}^T \mathbf{x}^t + w_0\right) \geq +1$$

Margin



H_1
H
$\vec{w}\bullet\vec{x}+b=$
H_2
$\vec{w}\bullet\vec{x}+b=$
$y_i=+1$
x_1
x_2
w
$y_i=-1$
$\vec{w}\bullet\vec{x}+b=0$

Distance from the discriminant to the closest instances on either side is called *margin* which we want to maximize for best generalization.

https://ninova.itu.edu.tr/tr/dersler/fen-bilimleri-enstitusu/998/blg-607/

Margin

- Distance of x^t to the hyperplane $\frac{|w^T x^t + w_0|}{||w||}$
- When $r^t \in \{-1, +1\} \rightarrow \frac{r^t(w^T x^t + w_0)}{||w||}$
- We require $\frac{r^t(w^T x^t + w_0)}{||w||} \geq \rho$
- For a unique sol'n, fix $\rho||\boldsymbol{w}||$=1, and to max margin

$$\min \frac{1}{2}\|\mathbf{w}\|^2 \text{ subject to } r^t\left(\mathbf{w}^T\mathbf{x}^t + w_0\right) \geq +1, \forall t$$

$$\text{min } \frac{1}{2}\|\mathbf{w}\|^2 \text{ subject to } r^t\left(\mathbf{w}^T\mathbf{x}^t + w_0\right) \geq +1, \forall t$$

$$L_p = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{t=1}^{N}\alpha^t\left[r^t\left(\mathbf{w}^T\mathbf{x}^t + w_0\right) - 1\right]$$

$$= \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{t=1}^{N}\alpha^t r^t\left(\mathbf{w}^T\mathbf{x}^t + w_0\right) + \sum_{t=1}^{N}\alpha^t$$

$$\frac{\partial L_p}{\partial \mathbf{w}} = 0 \Rightarrow \mathbf{w} = \sum_{t=1}^{N}\alpha^t r^t \mathbf{x}^t$$

$$\frac{\partial L_p}{\partial w_0} = 0 \Rightarrow \sum_{t=1}^{N}\alpha^t r^t = 0$$

solve the dual problem

$$L_d = \frac{1}{2}\left(\mathbf{w}^T\mathbf{w}\right) - \mathbf{w}^T\sum_t \alpha^t r^t \mathbf{x}^t - w_0\sum_t \alpha^t r^t + \sum_t \alpha^t$$

$$= -\frac{1}{2}\left(\mathbf{w}^T\mathbf{w}\right) + \sum_t \alpha^t$$

$$= -\frac{1}{2}\sum_t\sum_s \alpha^t\alpha^s r^t r^s \left(\mathbf{x}^t\right)^T \mathbf{x}^s + \sum_t \alpha^t$$

subject to $\sum_{\mathrm{t}} \alpha^t r^t = 0$ and $\alpha^t \geq 0, \forall t$

Most α^t are 0 and only a small number have α^t >0; they are the support vectors

Solution

w is written as the weighted sum of these training instances that are selected as the support vectors

$$w = \sum_t \alpha^t r^t x^t$$

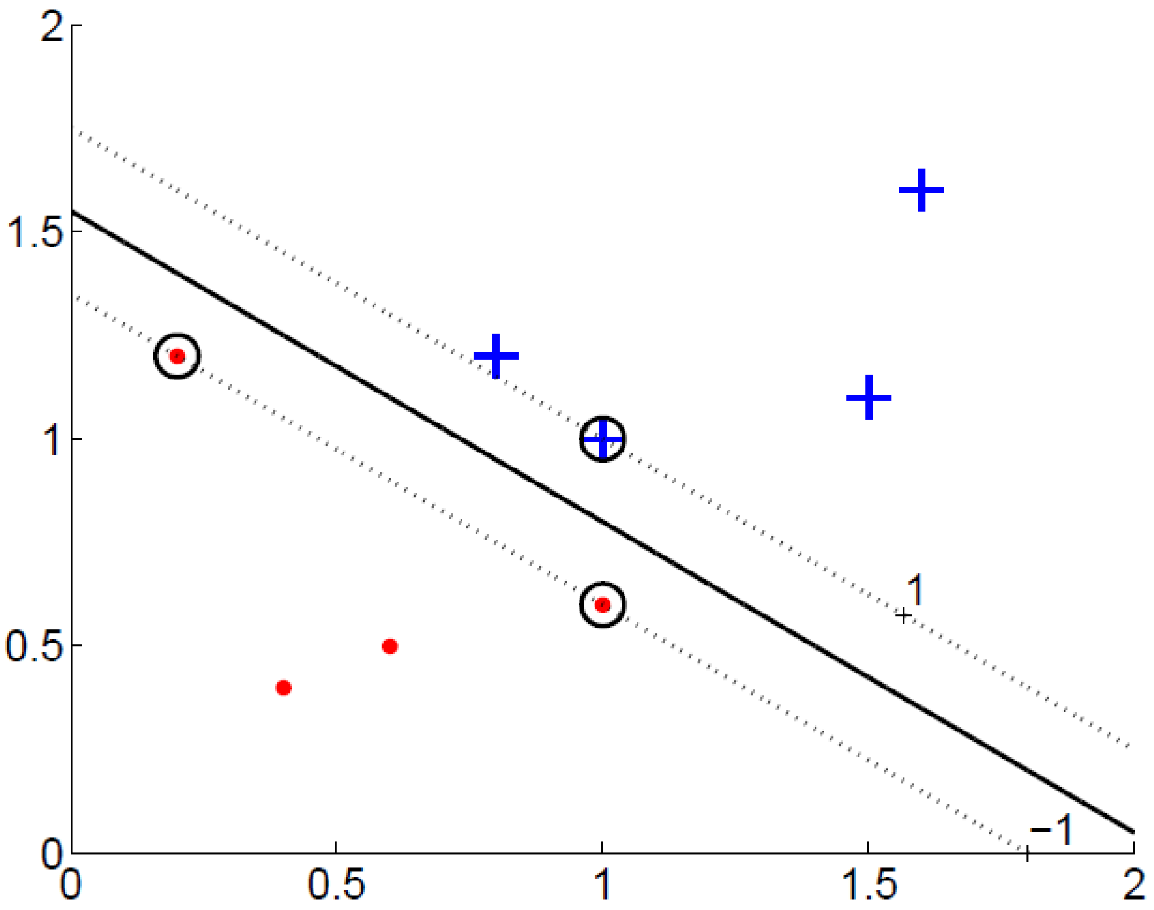
- These are the x^t that satisfy

$$r^t(w^T x^t + w_0) = 1$$

and lie on the margin. We can use this fact to calculate w_0 from any support vector as

$$w_0 = r^t - w^T x^t$$

Margin



For a two-class problem where the instances of the classes are shown by plus signs and dots, the thick line is the boundary and the dashed lines define the margins on either side.

Circled instances are the support vectors.

Testing

- During testing, we do not enforce a margin.
- We calculate

$$g(x) = w^T x + w_0$$

and choose according to the sign of $g(x)$:

Choose C1 if $g(x) > 0$ and C2 otherwise

The non-separable case: Soft Margin Hyperplane

- The data is not linearly separable
- In this case, if the two classes are not linearly separable such that there is no hyperplane to separate, then, we look for the one that incurs the least error.
- Define slack variables $\xi^t \geq 0$, which store the deviation from the margin

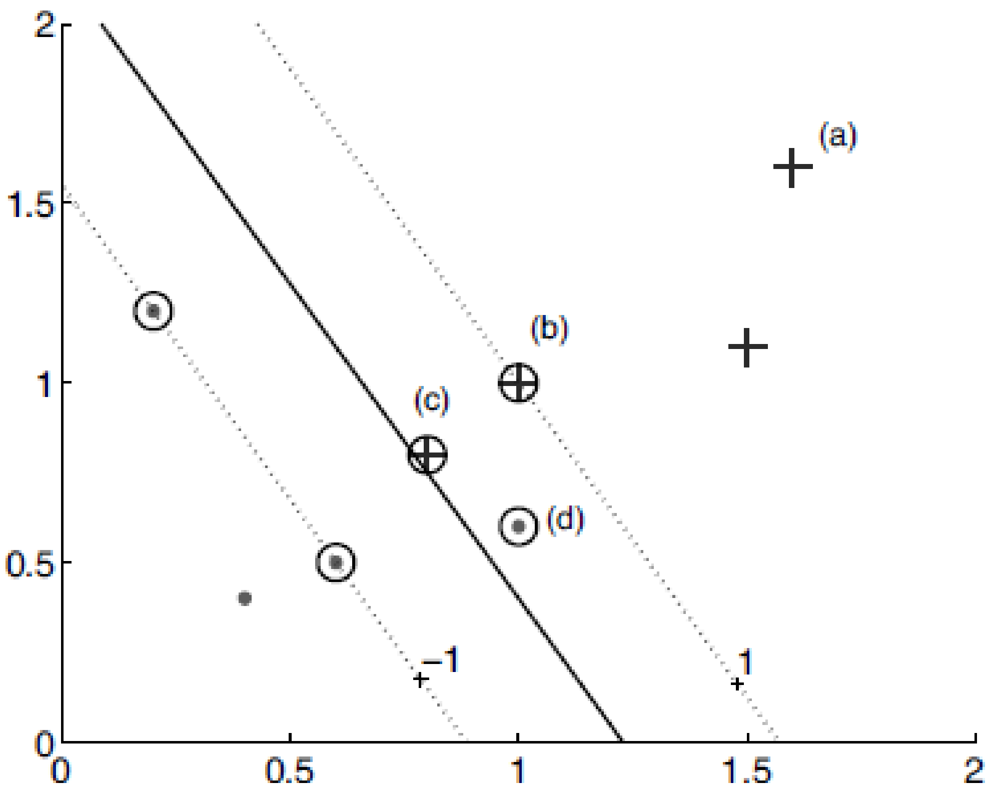


Figure 13.2 In classifying an instance, there are four possible cases: In (a), the instance is on the correct side and far away from the margin; $r^t g(\boldsymbol{x}^t) > 1$, $\xi^t = 0$. In (b), $\xi^t = 0$; it is on the right side and on the margin. In (c), $\xi^t = 1 - g(\boldsymbol{x}^t), 0 < \xi < 1$; it is on the right side but is in the margin and not sufficiently away. In (d), $\xi^t = 1 + g(\boldsymbol{x}^t) > 1$; it is on the wrong side—this is a misclassification. All cases except (a) are support vectors. In terms of the dual variable, in (a), $\alpha^t = 0$; in (b), $\alpha^t < C$; in (c) and (d), $\alpha^t = C$.

Soft Margin Hyperplane

- Not linearly separable

$$r^t\left(\mathbf{w}^T x^t + w_0\right) \geq 1 - \xi^t$$

- The number of misclassification is $\#\{\xi^t > 1\}$
- The number of nonseparable points is $\#\{\xi^t > 0\}$

Soft Margin Hyperplane

- Soft error

$$\sum_t \xi^t$$

- $L_p = \frac{1}{2}\|w\|^2 + C\sum_t \xi^t$

subject to $r^t(w^T x^t + w_0) \geq 1 - \xi^t$

C is penalty factor.

Soft Margin Hyperplane

New primal is

$$L_p = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_t \xi^t - \sum_t \alpha^t \left[r^t\left(\mathbf{w}^T x^t + w_0\right) - 1 + \xi^t\right] - \sum_t \mu^t \xi^t$$

Solution

$w = \sum_t \alpha^t r^t x^t$

$w_0 = r^t(1 - \xi^t) - w^T x^t$

ν-SVM

- There is another, equivalent formulation of the soft margin hyperplane that uses a parameter $\nu \in [0,1]$ instead of C.
- ν controls the fraction of support vectors

$$\min \frac{1}{2}\|\mathbf{w}\|^2 - \nu\rho + \frac{1}{N}\sum_t \xi^t$$

subject to

$$r^t\left(\mathbf{w}^T\mathbf{x}^t + w_0\right) \geq \rho - \xi^t, \xi^t \geq 0, \rho \geq 0$$

$$L_d = -\frac{1}{2}\sum_{t=1}^{N}\sum_s \alpha^t \alpha^s r^t r^s \left(x^t\right)^T x^s$$

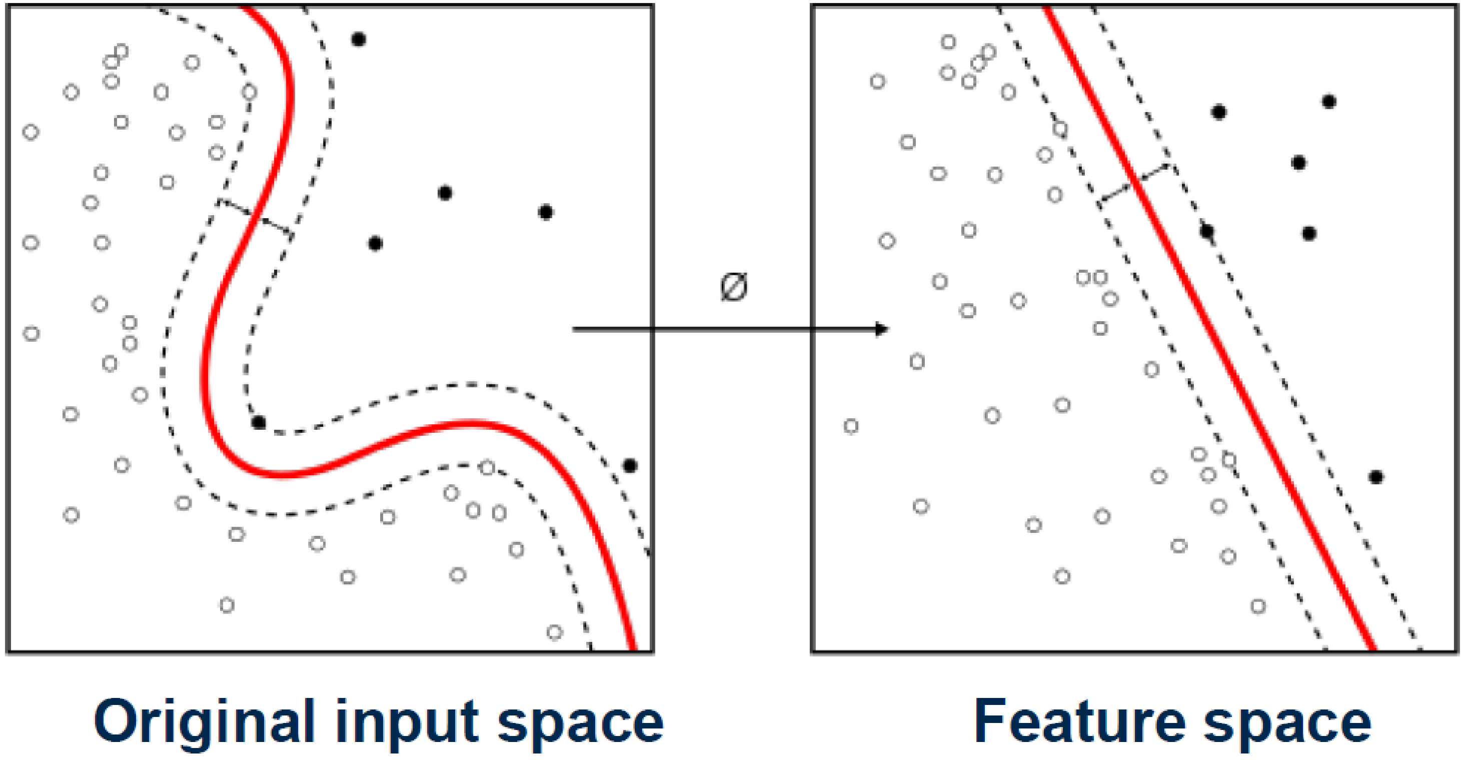
subject to

$$\sum_{\mathrm{t}} \alpha^t r^t = 0, 0 \leq \alpha^t \leq \frac{1}{N}, \sum_t \alpha^t \leq \nu$$

Kernel Trick

- Problem is non-linear
- We can map the problem to a new space by doing a nonlinear transformation using suitably chosen basis functions and then use a linear model in this new space.

Kernel Trick



Source: Wikipedia "Kernel Machine" article.
https://commons.wikimedia.org/w/index.php?curid=47868867

Kernel Trick

- We have new dimensions calculated through the basis functions:

  $\boldsymbol{z} = \boldsymbol{\varphi}(\boldsymbol{x})$

- Mapping from d-dimensional **x** space to k-dimensional **z** space
- The discriminant is

$$g(\boldsymbol{z}) = \boldsymbol{w}^T\boldsymbol{z} = \boldsymbol{w}^T\boldsymbol{\varphi}(\boldsymbol{x}) = \sum_{j=1}^{k} w_j \varphi_j(x)$$

assume that $z_1 = \varphi_1(x) = 1$

- k is much larger than d. k may also be larger than N

Kernel Trick

- The idea in kernel machines is to replace the inner product of basis function $\varphi(x^t)^T\varphi(x^s)$ by a kernel function, $K(x^t, x^s)$, between instances in the original space.
- Instead of mapping two instances x^t and x^s to the z-space and doing a dot product there, we directly apply the kernel function in the originl space.

Kernel Trick

- The SVM solution

$$\mathbf{w} = \sum_t \alpha^t r^t \mathbf{z}^t = \sum_t \alpha^t r^t \boldsymbol{\varphi}\left(\mathbf{x}^t\right)$$

$$g(\mathbf{x}) = \mathbf{w}^T \boldsymbol{\varphi}(\mathbf{x}) = \sum_t \alpha^t r^t \boldsymbol{\varphi}\left(\mathbf{x}^t\right)^T \boldsymbol{\varphi}(\mathbf{x})$$

$$g(\mathbf{x}) = \sum_t \alpha^t r^t K\left(\mathbf{x}^t, \mathbf{x}\right)$$

Kernel Trick

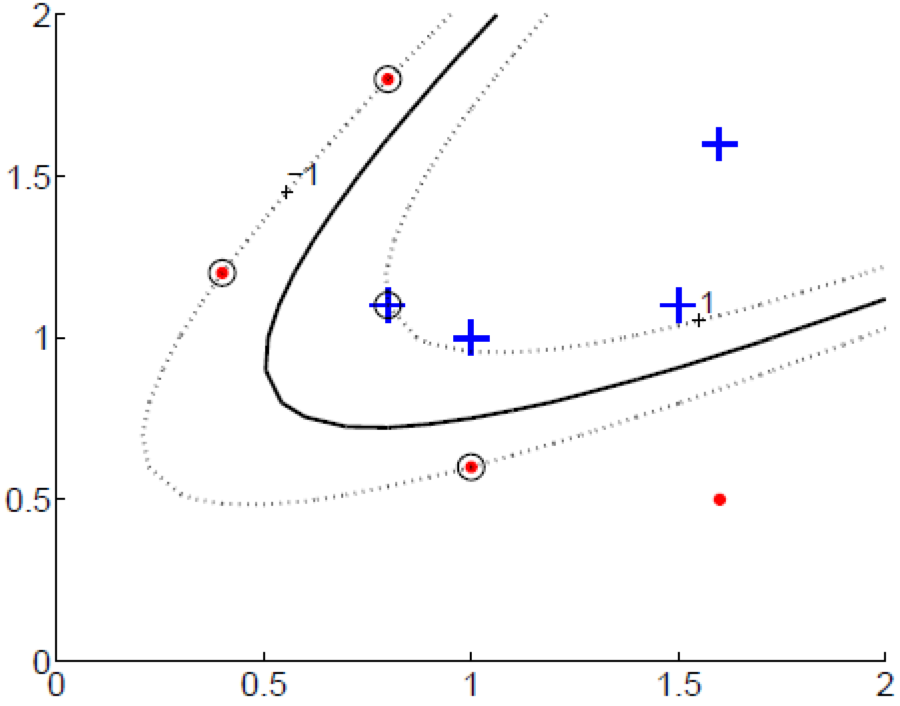
- The matrix of kernal values, K, where $K_{ts} = K(x^t, x^s)$ is called Gram matrix, which should be symmetric and positive semidefinite.

Vectorial Kernels

- Polynomials of degree q

$$K\left(\mathbf{x}^t,\mathbf{x}\right)=\left(\mathbf{x}^T\mathbf{x}^t+1\right)^q$$

- For example, when *q* = 2 and *d* = 2,



$$\begin{aligned}K(\mathbf{x},\mathbf{y})&=\left(\mathbf{x}^T\mathbf{y}+1\right)^2\\&=\left(x_1y_1+x_2y_2+1\right)^2\\&=1+2x_1y_1+2x_2y_2+2x_1x_2y_1y_2+x_1^2y_1^2+x_2^2y_2^2\end{aligned}$$

corresponds to the inner product of the basis function

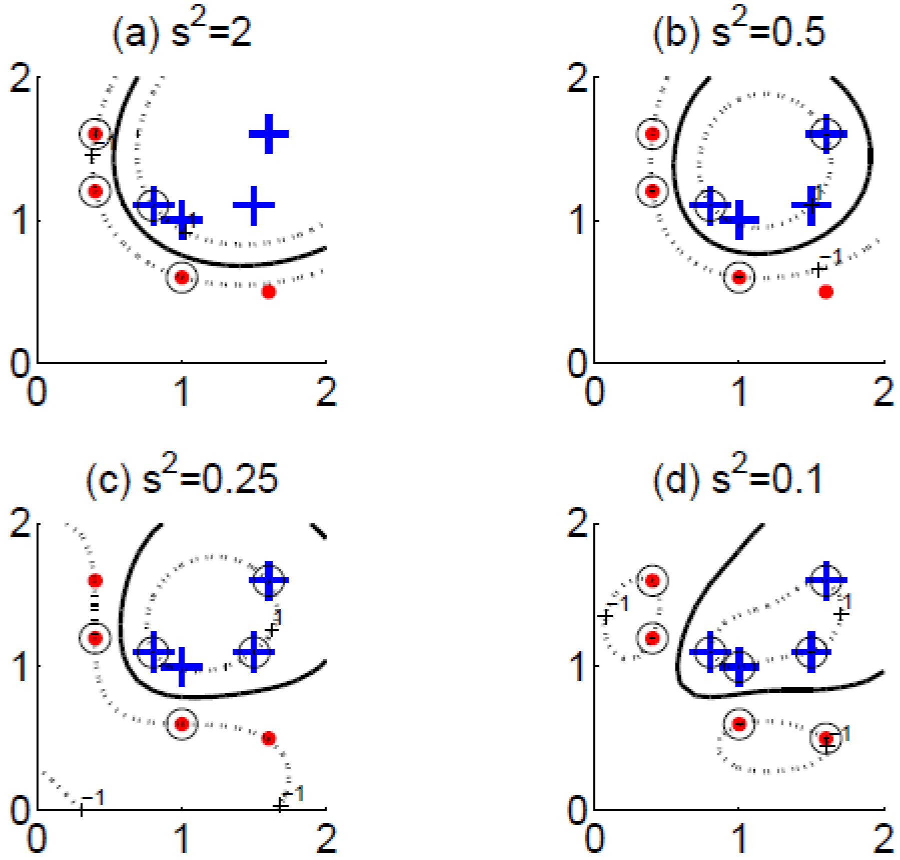
$$\phi(\mathbf{x})=\left[1,\sqrt{2}x_1,\sqrt{2}x_2,\sqrt{2}x_1x_2,x_1^2,x_2^2\right]^T$$

Vectorial Kernels

- Radial-basis functions (RBF)

$$K\left(\mathbf{x}^t,\mathbf{x}\right)=\exp\left[-\frac{\left\|\mathbf{x}^t-\mathbf{x}\right\|^2}{2s^2}\right]$$

RBF defines spherical kernel where x^t is the center and s defines radius



(a) $s^2=2$

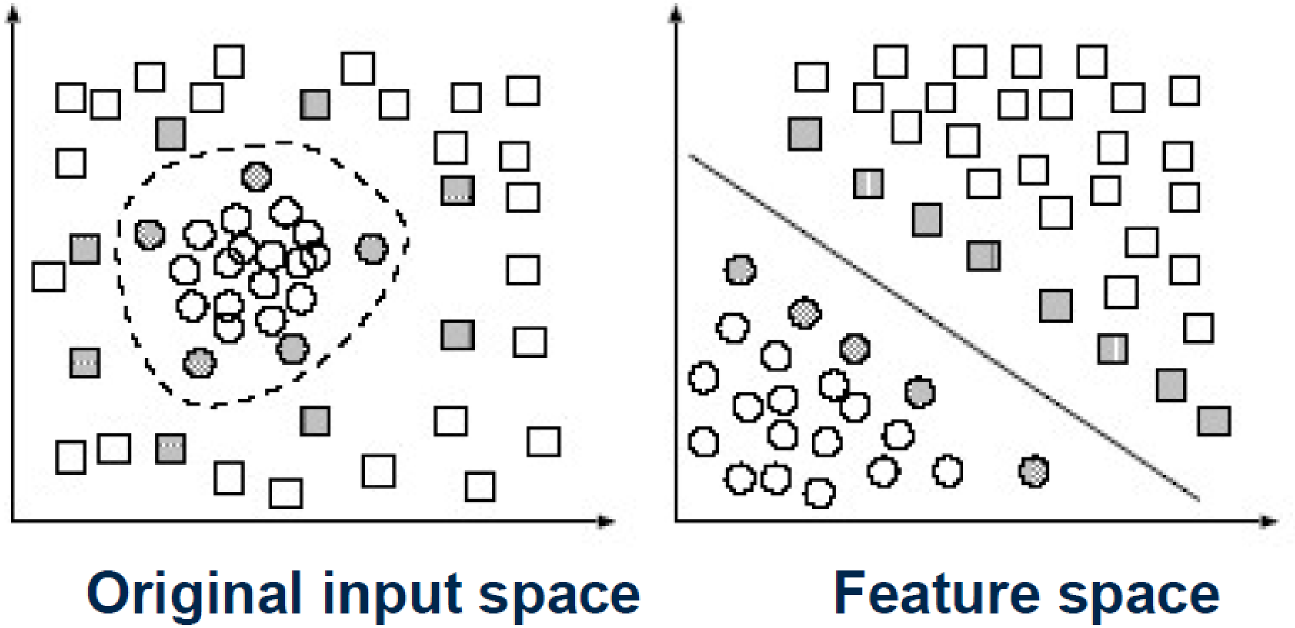
(b) $s^2=0.5$

(c) $s^2=0.25$

(d) $s^2=0.1$

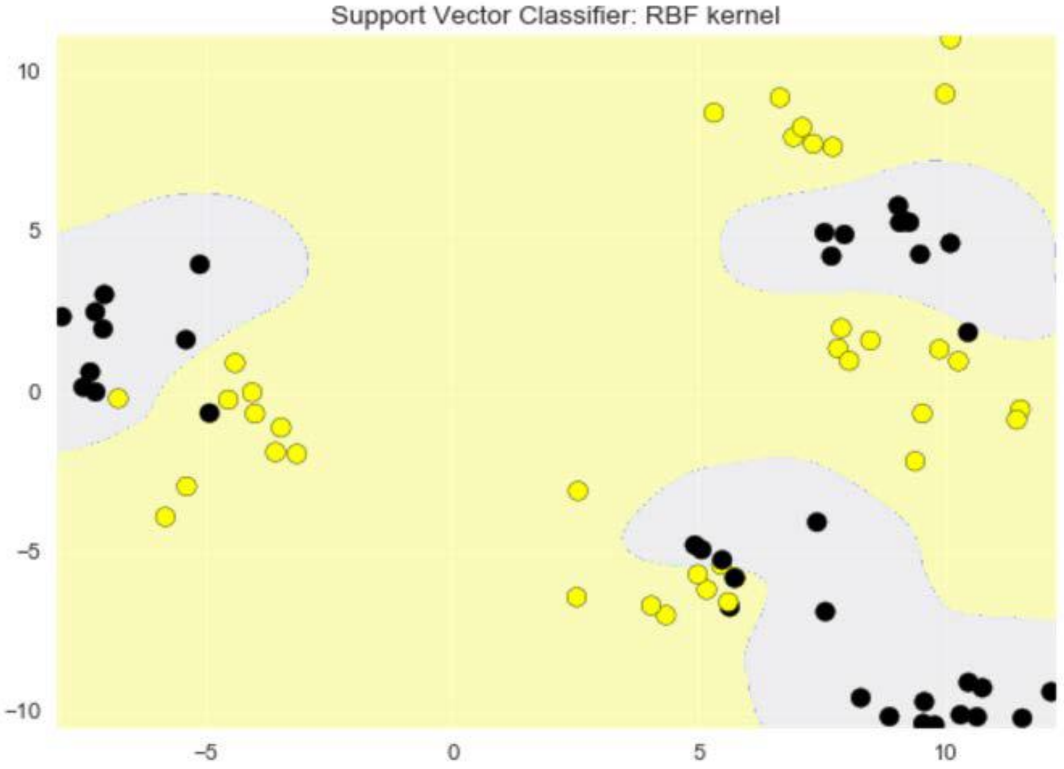
Radial-basis functions

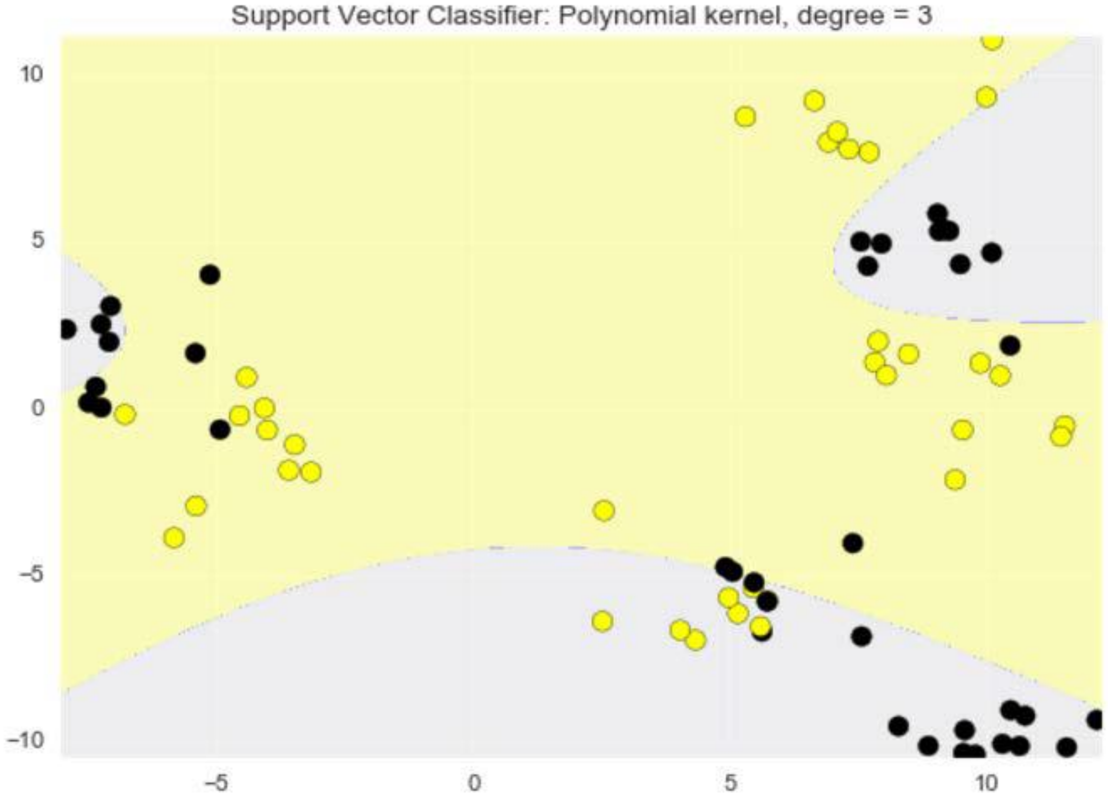
$$K(\boldsymbol{x}, \boldsymbol{x}') = \exp\ [-\gamma \cdot \|\boldsymbol{x} - \boldsymbol{x}'\|^2]$$



A kernel is a similarity measure (modified dot product) between data points

Radial Basis Kernel vs Polynomial Kernel

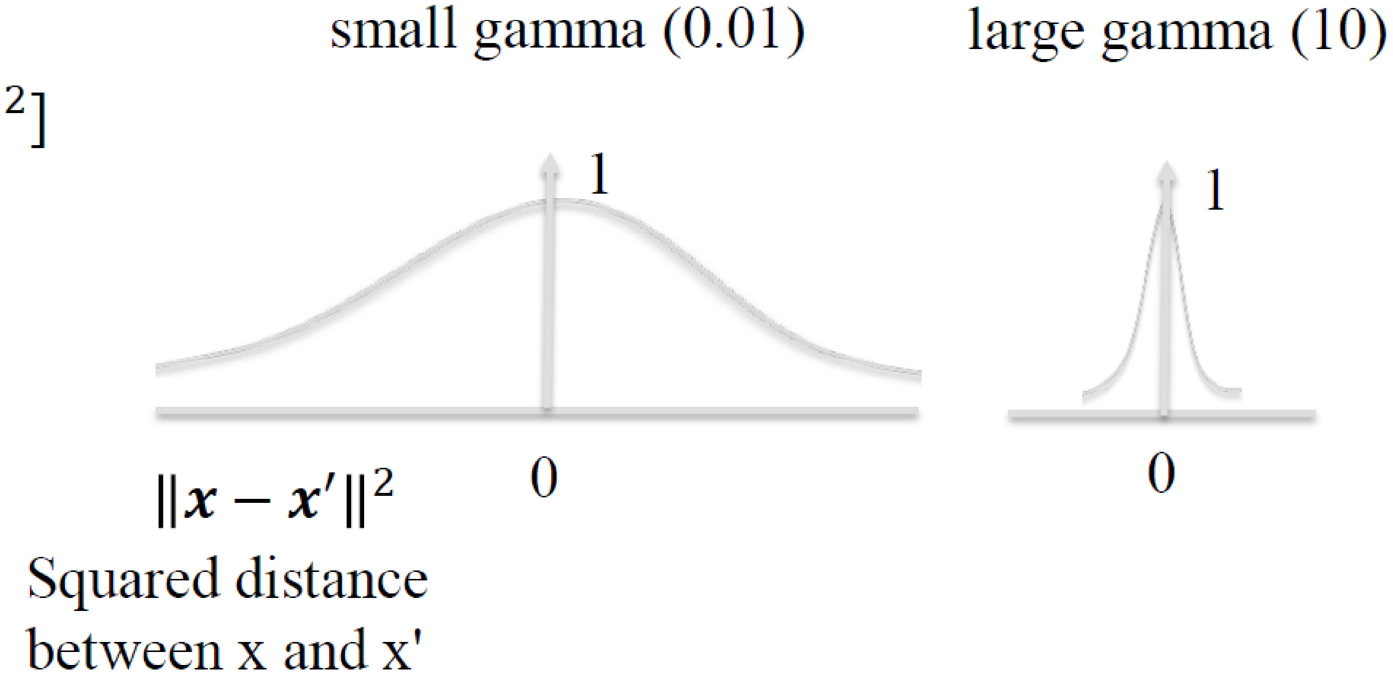




Radial Basis Function kernel: Gamma Parameter

$$K(\boldsymbol{x}, \boldsymbol{x}') = \exp\left[-\gamma \cdot \|\boldsymbol{x} - \boldsymbol{x}'\|^2\right]$$

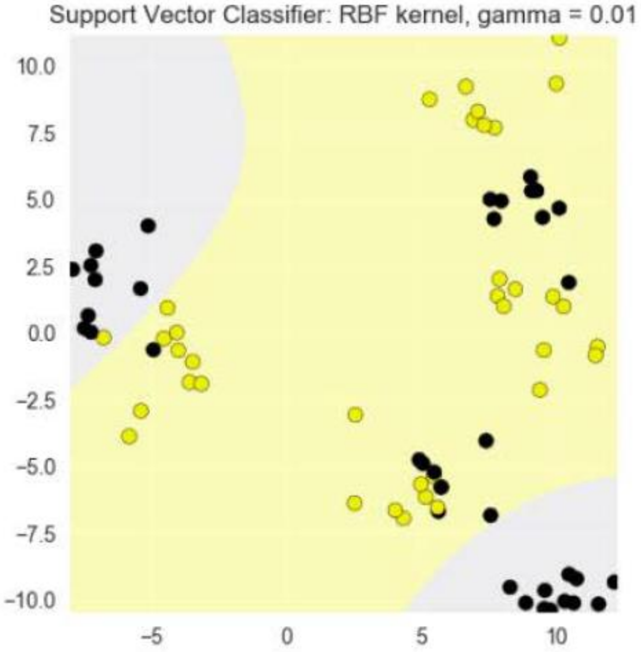
gamma (γ): kernel width parameter



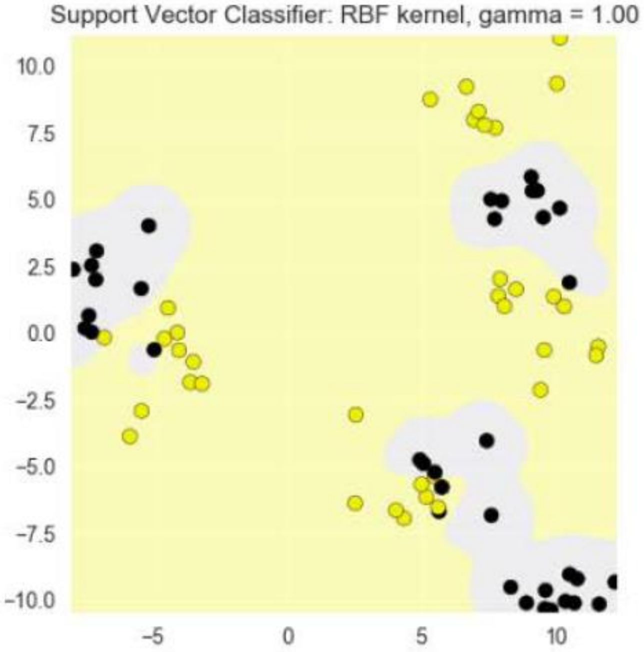
small gamma (0.01)

large gamma (10)

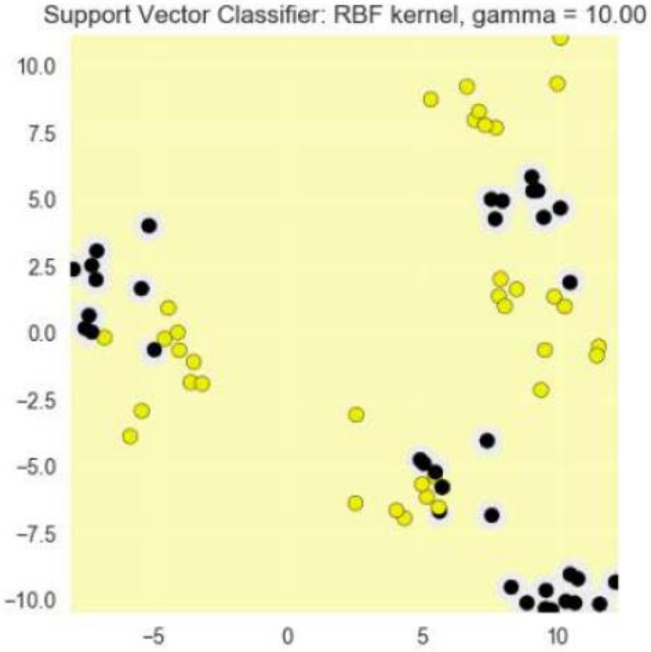
The effect of the RBF gamma parameter on decision boundaries



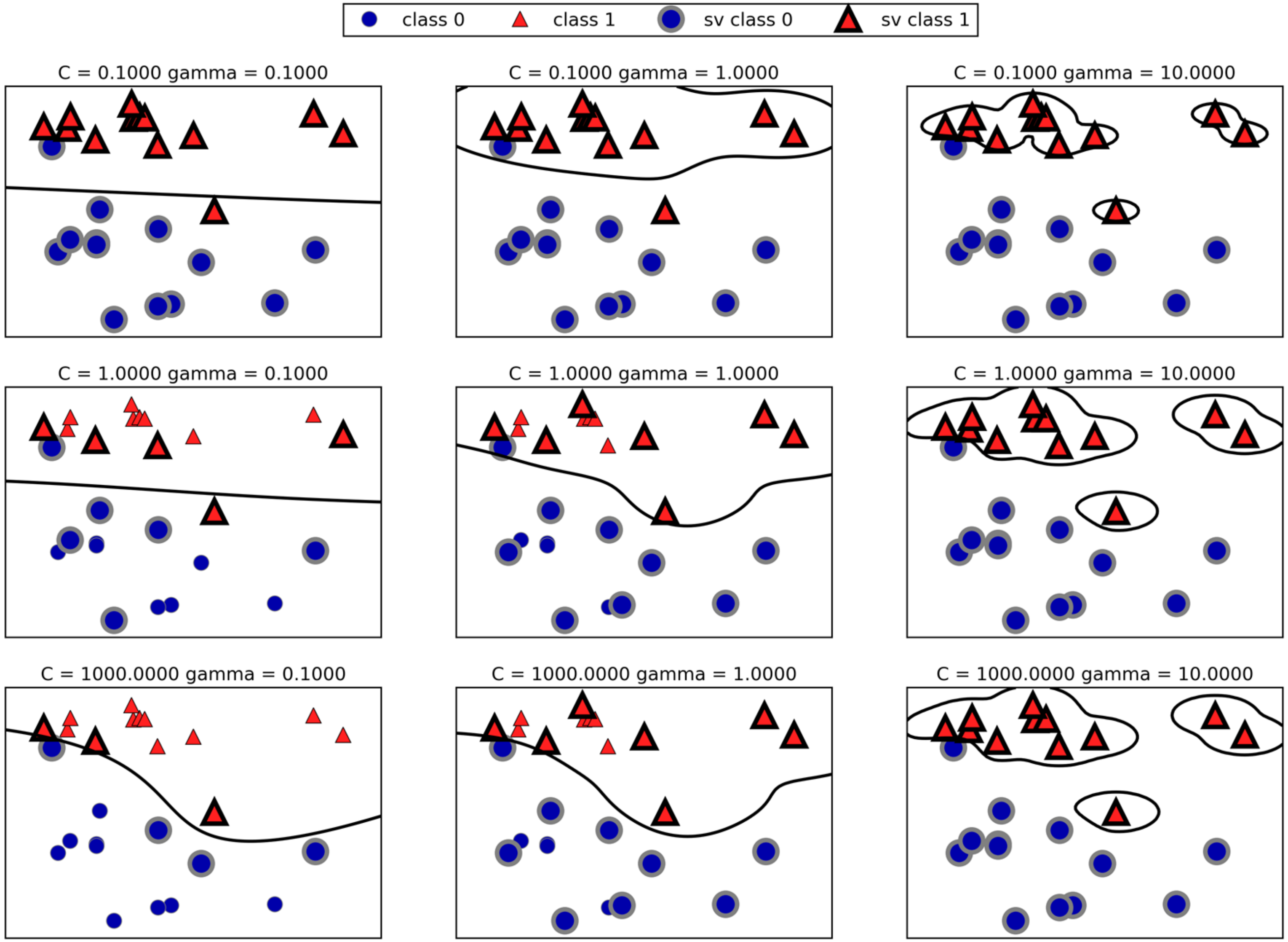
gamma = 0.01



gamma = 1.0



gamma = 10



A low value of gamma means that the decision boundary will vary slowly, which yields a model of low complexity, while a high value of gamma yields a more complex model.

At the top left the decision boundary looks nearly linear, with the misclassified points barely having any influence on the line.

Increasing C, as shown on the bottom right, allows these points to have a stronger influence on the model and makes the decision boundary bend to correctly classify them.

Defining kernels

- Kernels are generally considered to be measures of similarity in the sense that *K(**x**,**y**)* takes a larger value as ***x*** and ***y*** are more “similar,” from the point of view of the application.
- There are string kernels, tree kernels, graph kernels, and so on depending on how we represent the data and how we measure similarity in that representation.

Multiple Kernel Learning

- Fixed kernel combination

$$K(\mathbf{x},\mathbf{y}) = \begin{cases} cK(\mathbf{x},\mathbf{y}) \\ K_1(\mathbf{x},\mathbf{y}) + K_2(\mathbf{x},\mathbf{y}) \\ K_1(\mathbf{x},\mathbf{y}) K_2(\mathbf{x},\mathbf{y}) \end{cases}$$

- Adaptive kernel combination

$$K(\mathbf{x},\mathbf{y}) = \sum_{i=1}^{m} \eta_i K_i(\mathbf{x},\mathbf{y})$$

$$L_d = \sum_t \alpha^t - \frac{1}{2} \sum_t \sum_s \alpha^t \alpha^s r^t r^s \sum_i \eta_i K_i(\mathbf{x}^t, \mathbf{x}^s)$$

$$g(\mathbf{x}) = \sum_t \alpha^t r^t \sum_i \eta_i K_i(\mathbf{x}^t, \mathbf{x})$$

- Localized kernel combination

$$g(\mathbf{x}) = \sum_t \alpha^t r^t \sum_i \eta_i(\mathbf{x} \mid \theta) K_i(\mathbf{x}^t, \mathbf{x})$$

Multiclass Kernel Machines

- One-vs-all
  - learn *K* support vector machines $g_i(x)$ ($C_i = +1, C_k = -1\ k \neq i$)
  - Testing: calculate all $g_i(x)$ choose maximum
- Pairwise
  - learn *K(K-1)* support vector machines $g_{ij}(x)$ ($C_i = +1, C_j = -1$) not using examples of other classes
- Single multiclass optimization

$$\min \frac{1}{2}\sum_{i=1}^{K}\|\mathbf{w}_i\|^2 + C\sum_i\sum_t \xi_i^t$$

subject to

$$\mathbf{w}_{z^t}{}^T\mathbf{x}^t + w_{z^t 0} \geq \mathbf{w}_i{}^T\mathbf{x}^t + w_{i0} + 2 - \xi_i^t, \forall i \neq z^t, \xi_i^t \geq 0$$

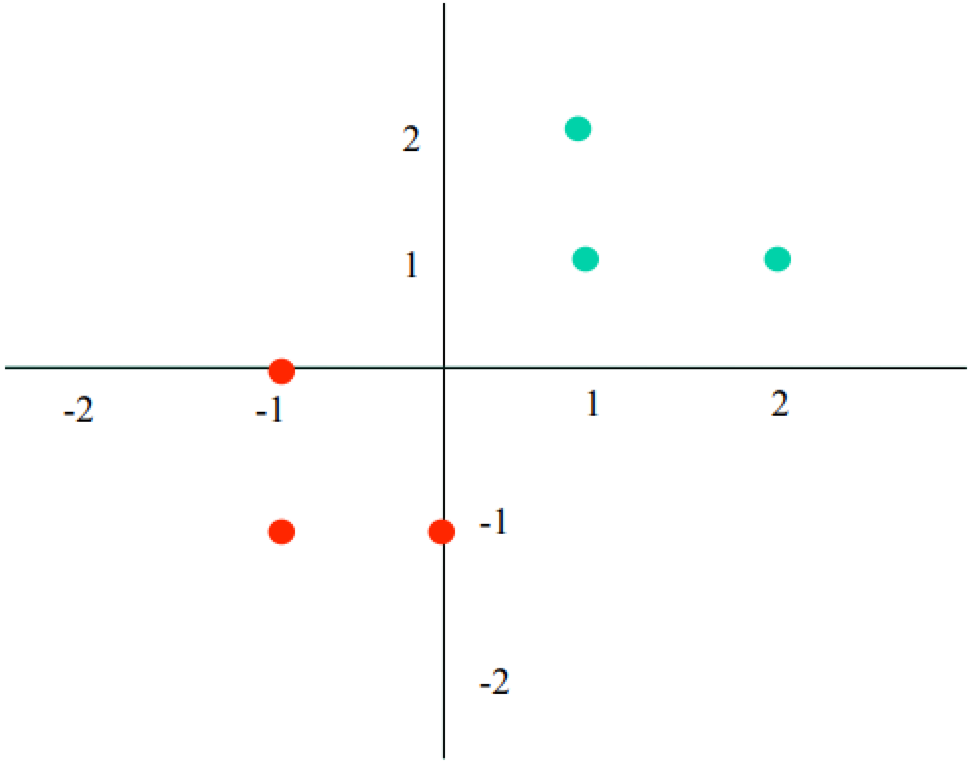
Scikit learn

- `class sklearn.svm.SVC(C=1.0, kernel=’rbf’, …)`
  - The implementation is based on libsvm
- `class sklearn.svm.LinearSVC(…)`
  - Linear Support Vector Classification.
  - Similar to SVC with parameter kernel=’linear’,
  - but implemented in terms of liblinear rather than libsvm
  - https://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVC.html

Example

- Input

x_1	x_2	*class*
1	1	1
1	2	1
2	1	1
−1	0	−1
0	−1	−1
−1	−1	−1

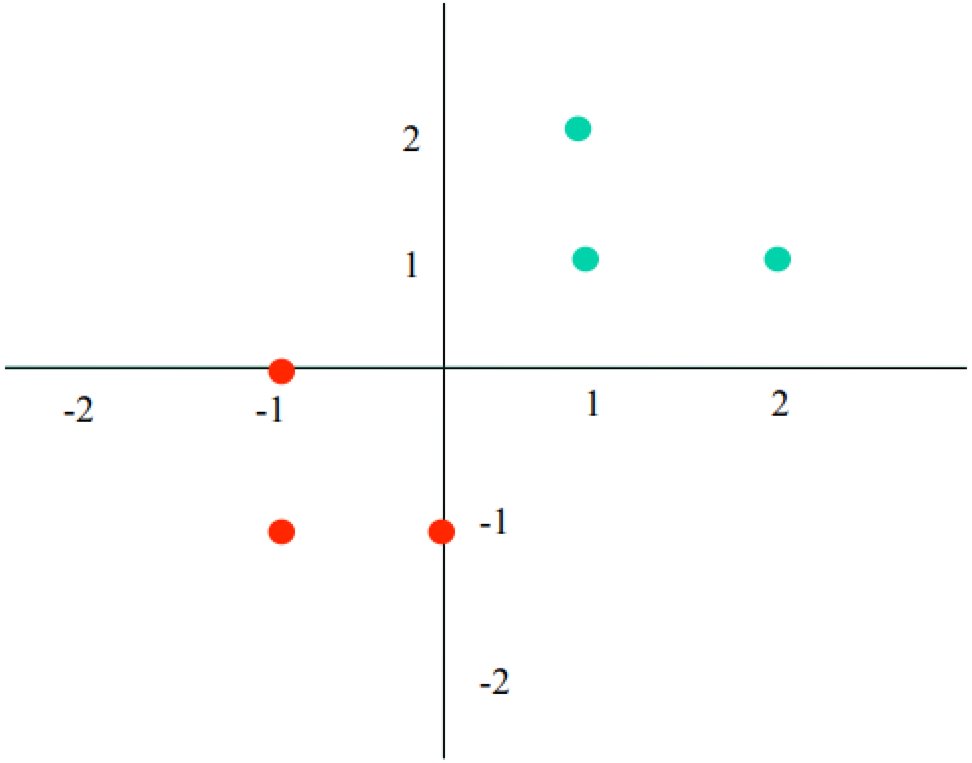


- Output from SVM optimizer
- $w_0 = -0.376$
- Support vectors and Lagrange multipliers

$(1,1) \rightarrow \alpha = 0.416$
$(0,-1) \rightarrow \alpha = 0.208$
$(-1,0) \rightarrow \alpha = 0.208$

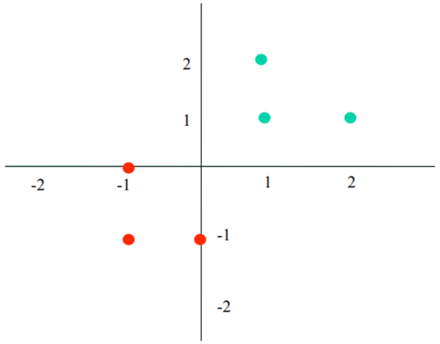
Example

- Draw the decision boundary
- Classify the point (2,2)



Example

$(1,1) \to \alpha = 0.416$
$(0,-1) \to \alpha = 0.208$
$(-1,0) \to \alpha = 0.208$



$$w = \sum_t \alpha^t r^t x^t = 0.208 * (-1) * \begin{bmatrix}-1\\0\end{bmatrix} + 0.208 * (-1) * \begin{bmatrix}0\\-1\end{bmatrix} + 0.416 * (1) * \begin{bmatrix}1\\1\end{bmatrix} = \begin{bmatrix}0.208\\0\end{bmatrix} + \begin{bmatrix}0\\0.208\end{bmatrix} + \begin{bmatrix}0.416\\0.416\end{bmatrix} = \begin{bmatrix}0.624\\0.624\end{bmatrix}$$

$w_1 = 0.624, w_2 = 0.624, w_0 = -0.376$

$$-0.376 + 0.624x_1 + 0.624x_2 = 0$$

$$x_2 = -x_1 + 0.6$$

$(1,1) \rightarrow \alpha = 0.416$
$(0,-1) \rightarrow \alpha = 0.208$
$(-1,0) \rightarrow \alpha = 0.208$

Example

$x_2 = -x_1 + 0.6$
$g(x) = x_1 + x_2 - 0.6 = 0$

$g((2,2)) = 2 + 2 - 0.6 = 3.4$
$g(x) > 0 \quad \rightarrow assign\ x\ to\ class + 1$