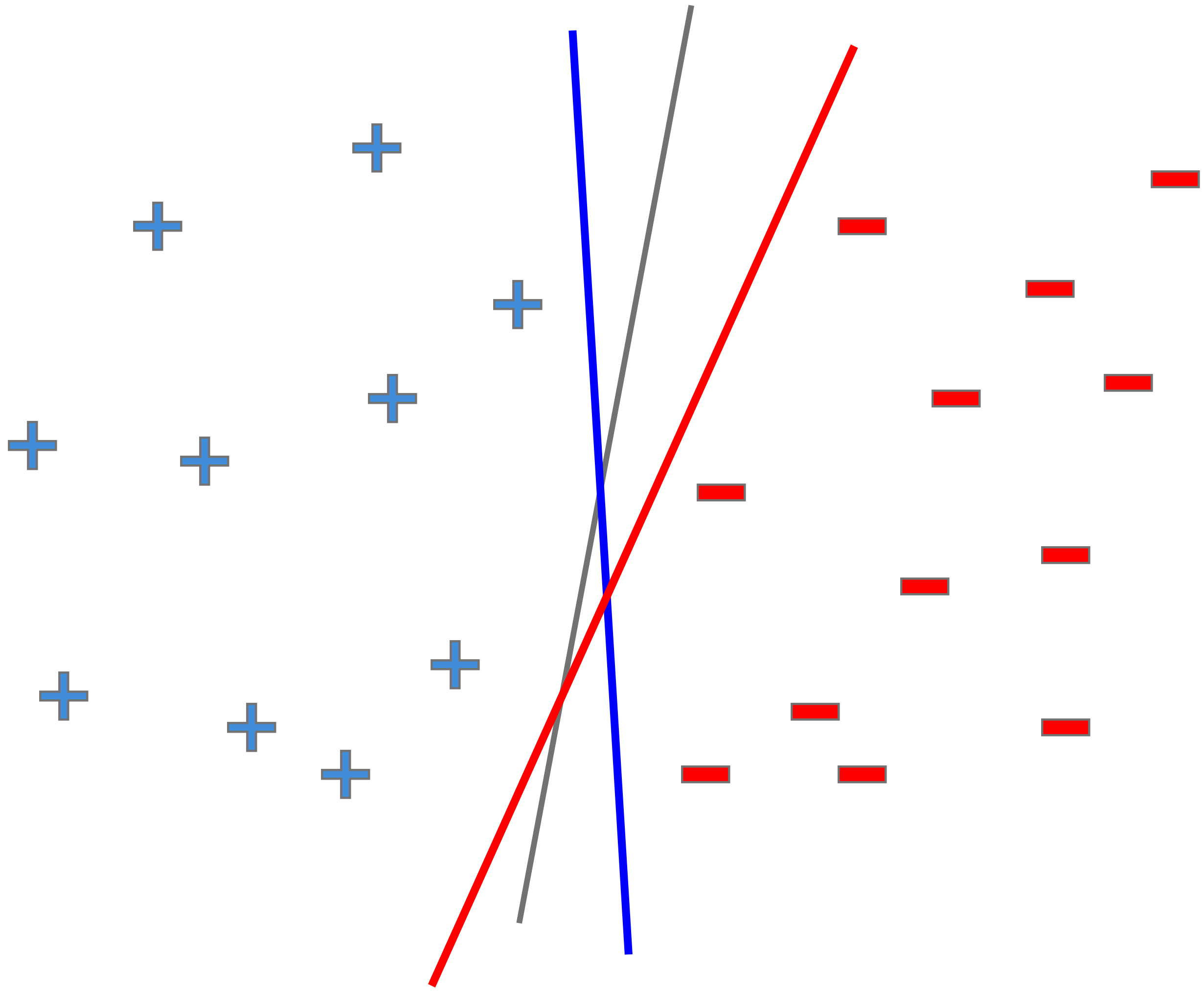
Advanced Introduction to Machine Learning CMU-10715

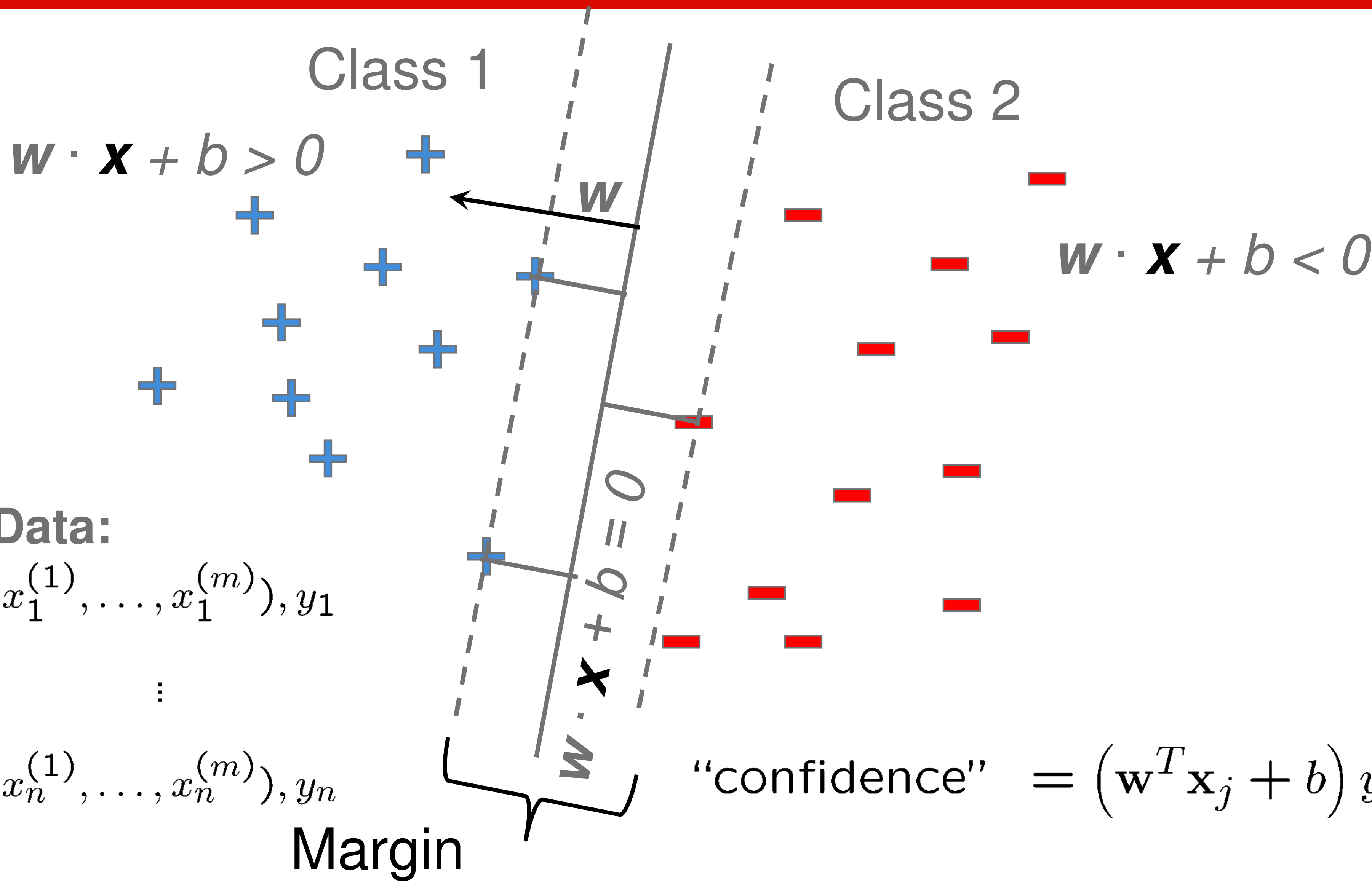
Support Vector Machines

Barnabás Póczos, 2014 Fall

Linear classifiers
which line is better?



Pick the one with the largest margin!



Data:

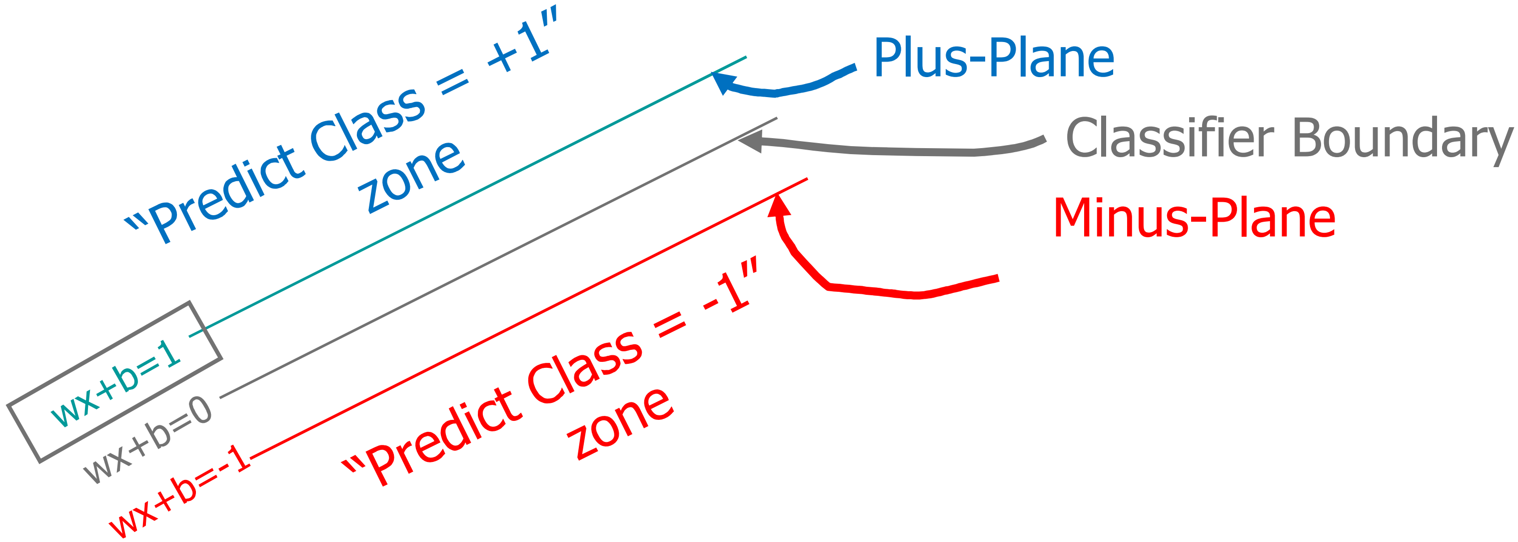
$$(x_1^{(1)}, \ldots, x_1^{(m)}), y_1$$

$$\vdots$$

$$(x_n^{(1)}, \ldots, x_n^{(m)}), y_n$$

"confidence" $= \left(\mathbf{w}^T \mathbf{x}_j + b\right) y_j$

Scaling



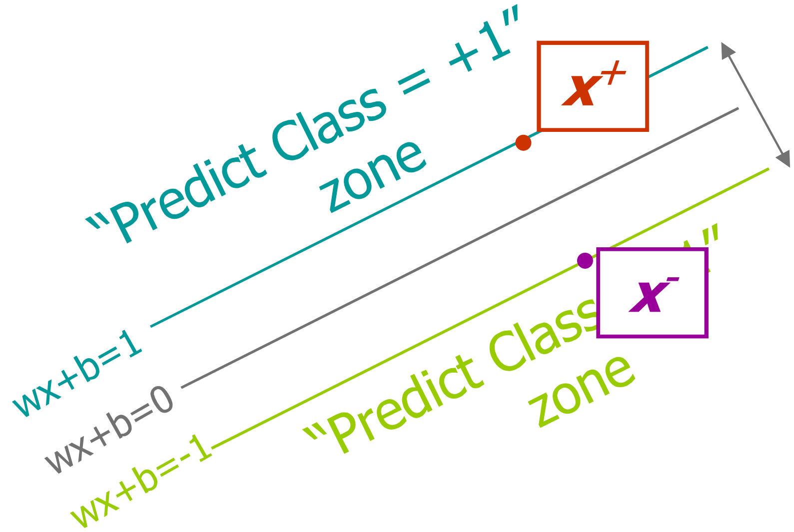
Classification rule:

Classify as..	+1	if	$\mathbf{w} \cdot \mathbf{x} + b \geq 1$
	-1	if	$\mathbf{w} \cdot \mathbf{x} + b \leq -1$
	Universe explodes	if	$-1 < \mathbf{w} \cdot \mathbf{x} + b < 1$

How large is the margin of this classifier?

Goal: Find the maximum margin classifier

Computing the margin width



M = Margin Width $= \dfrac{2}{\sqrt{\mathbf{w}\cdot\mathbf{w}}}$

$$w^T(\underbrace{x^+ - x^-}_{\lambda w}) = 2$$

$$\Downarrow$$

$$\lambda = \frac{2}{w^T w}$$

$$M = |x^+ - x^-| = |\lambda w| = \frac{|2w|}{w^T w} = \frac{2}{\sqrt{w^T w}}$$

Let $\boldsymbol{x}^+$ and $\boldsymbol{x}^-$ be such that

- $\boldsymbol{w} \cdot \boldsymbol{x}^+ + b = +1$
- $\boldsymbol{w} \cdot \boldsymbol{x}^- + b = -1$
- $\boldsymbol{x}^+ = \boldsymbol{x}^- + \lambda\, \boldsymbol{w}$
- $|\boldsymbol{x}^+ - \boldsymbol{x}^-| = M = ?$ *(Margin)*

Maximize M ≡ minimize **w·w !**

The Primal Hard SVM

- Given $D = \{(\mathbf{x}_i, y_i), i = 1, \ldots, n\}$ training data set.
- Assume that D is **linearly separable**.

$$\hat{\mathbf{w}} = \arg \min_{\mathbf{w} \in \mathbb{R}^m} \frac{1}{2} ||\mathbf{w}||^2$$

$$\text{subject to } y_i \langle \mathbf{x}_i, \mathbf{w} \rangle \geq 1, \; \forall i = 1, \ldots, n$$

Prediction: $f_{\hat{\mathbf{w}}}(\mathbf{x}) = \text{sign}(\langle \hat{\mathbf{w}}, \mathbf{x} \rangle)$

This is a QP problem (m-dimensional)
(Quadratic cost function, linear constraints)

Quadratic Programming

Find

$$\underset{w \in \mathbb{R}^n}{\arg\min} \; w^T H w + w^T q + e$$

Subject to

$$A w \leq b$$

$$A \in \mathbb{R}^{n \times m} \quad w \in \mathbb{R}^m \quad b \in \mathbb{R}^n$$

and to

$$C w = d$$

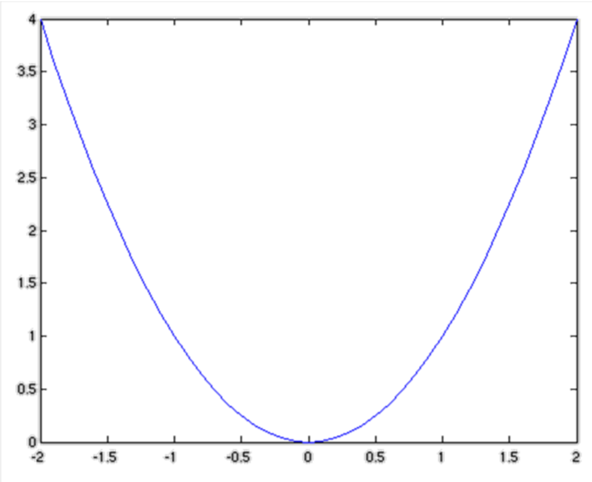
$$C \in \mathbb{R}^{s \times m} \quad d \in \mathbb{R}^s$$

Efficient Algorithms exist for QP.
They often solve the dual problem instead of the primal.

Constrained Optimization

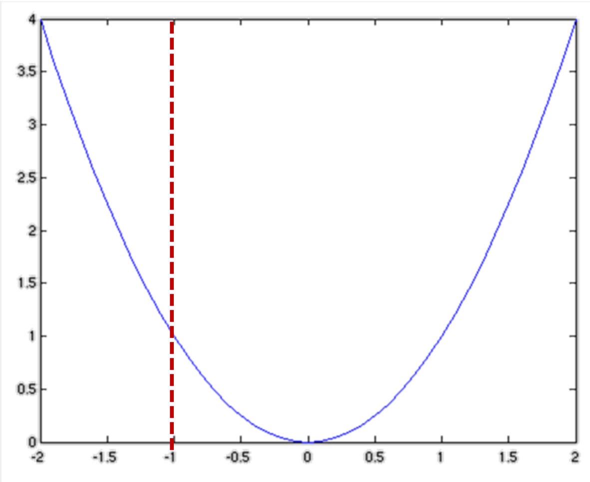
$$\min_x \; x^2$$
$$\text{s.t.} \quad x \geq b$$

$$\min_x \; x^2$$



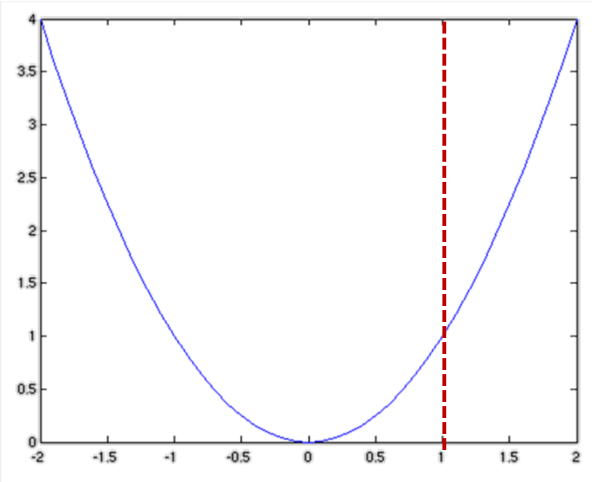
$$x^* = 0$$

$$\min_x \; x^2$$
$$\text{s.t.} \quad x \geq -1$$



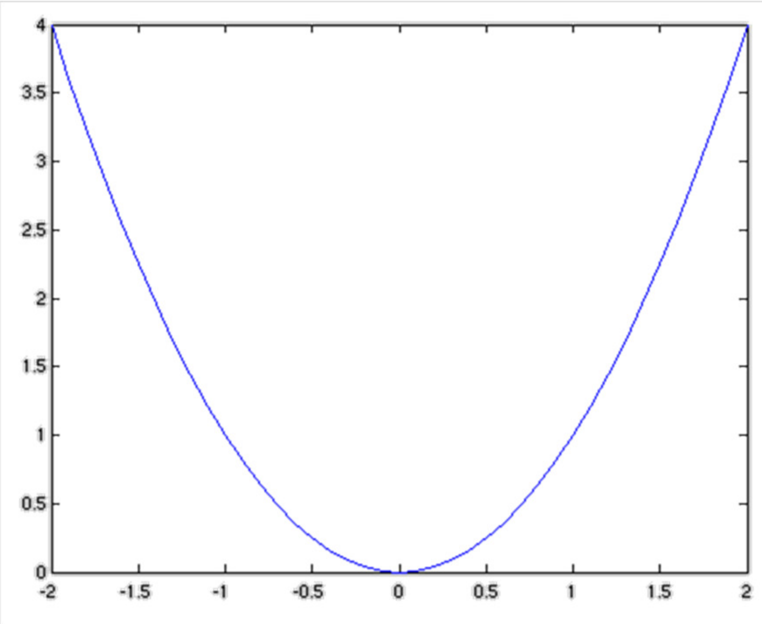
$$x^* = 0$$

$$\min_x \; x^2$$
$$\text{s.t.} \quad x \geq 1$$



$$x^* = 1$$

Lagrange Multiplier



$$\min_x \ x^2$$
$$\text{s.t.} \quad x \geq b$$

Moving the constraint to objective function

Lagrangian:

$$L(x, \alpha) = x^2 - \alpha(x - b)$$
$$\text{s.t.} \quad \alpha \geq 0$$

Solve:

$$\min_x \max_\alpha \ L(x, \alpha)$$
$$\text{s.t.} \quad \alpha \geq 0$$

Constraint is active when $\alpha > 0$

Lagrange Multiplier – Dual Variables

Solving:

$$\overbrace{\min_x \max_\alpha \; x^2 - \alpha(x-b)}^{L(x,\alpha)}$$

$$\text{s.t.} \quad \alpha \geq 0$$

$$\frac{\partial L}{\partial x} = 0 \;\Rightarrow x^* = \frac{\alpha}{2}$$

$$\frac{\partial L}{\partial \alpha} = 0 \;\Rightarrow \alpha^* = \max(2b, 0)$$

$$2x - \alpha = 0 \Rightarrow x^* = \frac{\alpha}{2}$$

$$L(x^*, \alpha) = \frac{\alpha^2}{4} - \alpha\left(\frac{\alpha}{2} - b\right) = -\frac{\alpha^2}{4} + b\alpha$$

$$\frac{\partial L(x^*, \alpha)}{\partial \alpha} = -\frac{\alpha}{2} + b$$

$$\alpha^* = 2b$$

When α > 0, constraint is tight

From Primal to Dual

Primal problem:

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w} \in \mathbb{R}^m} \frac{1}{2}\|\mathbf{w}\|^2$$

$$\text{subject to } y_i\langle \mathbf{x}_i, \mathbf{w}\rangle \geq 1, \ \forall i = 1, \ldots, n$$

Lagrange function:

$$\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_n)^T \geq 0 \text{ Largrange multipliers}$$

$$L(\mathbf{w}, \boldsymbol{\alpha}) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{n} \alpha_i\Big(y_i\langle \mathbf{x}_i, \mathbf{w}\rangle - 1\Big)$$

The Lagrange Problem

$$L(\mathbf{w}, \boldsymbol{\alpha}) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{n} \alpha_i \Big(y_i \langle \mathbf{x}_i, \mathbf{w} \rangle - 1 \Big)$$

The Lagrange problem:

$$(\hat{\mathbf{w}}, \hat{\boldsymbol{\alpha}}) = \arg \min_{\mathbf{w} \in \mathbb{R}^m} \max_{0 \leq \boldsymbol{\alpha} \in \mathbb{R}^n} L(\mathbf{w}, \boldsymbol{\alpha})$$

$$0 = \left. \frac{\partial L(\mathbf{w}, \boldsymbol{\alpha})}{\partial \mathbf{w}} \right|_{\mathbf{w} = \hat{\mathbf{w}}} = \hat{\mathbf{w}} - \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i$$

$$\boxed{\Rightarrow \hat{\mathbf{w}} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i}$$

The Dual Problem

$$L(\mathbf{w}, \boldsymbol{\alpha}) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{n} \alpha_i \Big(y_i \langle \mathbf{x}_i, \mathbf{w} \rangle - 1 \Big)$$

$$\boxed{\Rightarrow \hat{\mathbf{w}} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i}$$

$$\Rightarrow L(\hat{\mathbf{w}}, \boldsymbol{\alpha}) = \frac{1}{2}\|\hat{\mathbf{w}}\|^2 - \sum_{i=1}^{n} \alpha_i \Big(y_i \langle \mathbf{x}_i, \hat{\mathbf{w}} \rangle - 1 \Big)$$

$$= \frac{1}{2} \underbrace{\| \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i \|^2}_{\boldsymbol{\alpha}^T \boldsymbol{Y} \boldsymbol{G} \boldsymbol{Y} \boldsymbol{\alpha}} + \boldsymbol{\alpha}^T \mathbf{1}_n - \underbrace{\sum_{i=1}^{n} \alpha_i y_i \langle \mathbf{x}_i, \sum_{j=1}^{n} \alpha_j y_j \mathbf{x}_j \rangle}_{\boldsymbol{\alpha}^T \boldsymbol{Y} \boldsymbol{G} \boldsymbol{Y} \boldsymbol{\alpha}}$$

$$\boxed{= \boldsymbol{\alpha}^T \mathbf{1}_n - \frac{1}{2} \boldsymbol{\alpha}^T \boldsymbol{Y} \boldsymbol{G} \boldsymbol{Y} \boldsymbol{\alpha}}$$

$$\boldsymbol{Y} \doteq diag(y_1, \ldots, y_n), \;\; y_i \in \{-1, 1\}^n$$

$\boldsymbol{G} \in \mathbb{R}^{n \times n} \doteq \{G_{ij}\}_{i,j}^{n,n}$, where $G_{ij} \doteq \langle \mathbf{x}_i, \mathbf{x}_j \rangle$ Gram matrix.

The Dual Hard SVM

$\boldsymbol{Y} \doteq diag(y_1, \ldots, y_n),\ y_i \in \{-1, 1\}^n$

$\boldsymbol{G} \in \mathbb{R}^{n \times n} \doteq \{G_{ij}\}_{i,j}^{n,n}$, where $G_{ij} \doteq \langle \mathbf{x}_i, \mathbf{x}_j \rangle$ Gram matrix.

$$\hat{\boldsymbol{\alpha}} = \operatorname*{arg\,max}_{\boldsymbol{\alpha} \in \mathbb{R}^n} \boldsymbol{\alpha}^T \mathbf{1}_n - \frac{1}{2} \boldsymbol{\alpha}^T \boldsymbol{Y} \boldsymbol{G} \boldsymbol{Y} \boldsymbol{\alpha}$$

subject to $\alpha_i \geq 0,\ \forall i = 1, \ldots, n$

Quadratic Programming (n-dimensional)

Lemma $\hat{\mathbf{w}} = \sum_{i=1}^{n} \hat{\alpha}_i y_i \mathbf{x}_i$

Prediction: $f_{\hat{\mathbf{w}}}(x) = \text{sign}(\langle \hat{\mathbf{w}}, \mathbf{x} \rangle) = \text{sign}(\sum_{i=1}^{n} \hat{\alpha}_i y_i \underbrace{\langle \mathbf{x}_i, \mathbf{x} \rangle}_{k(\mathbf{x}_i, \mathbf{x})})$

The Problem with Hard SVM

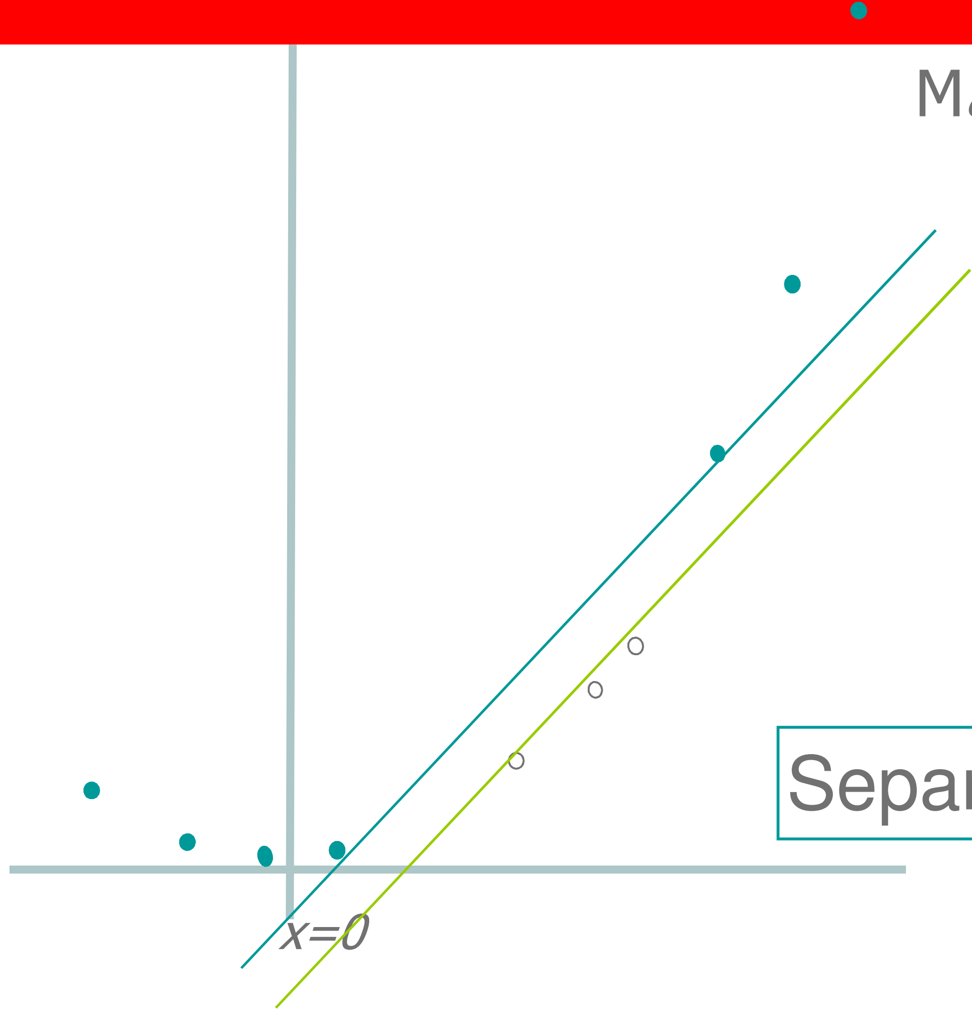
It assumes samples are linearly separable...

What can we do if data is not linearly separable???

Hard 1-dimensional Dataset

If the data set is **not** linearly separable, then adding new features (mapping the data to a larger feature space) the data might become linearly separable

Hard 1-dimensional Dataset



Make up a new feature!

Sort of…
… computed from original feature(s)

$$\mathbf{z}_k = (x_k, x_k^2)$$

Separable! MAGIC!

Now drop this "augmented" data into our linear SVM.

Feature mapping

- *n* general! points in an *n-1* dimensional space is always linearly separable by a hyperspace!
  $\Rightarrow$ it is good to map the data to high dimensional spaces

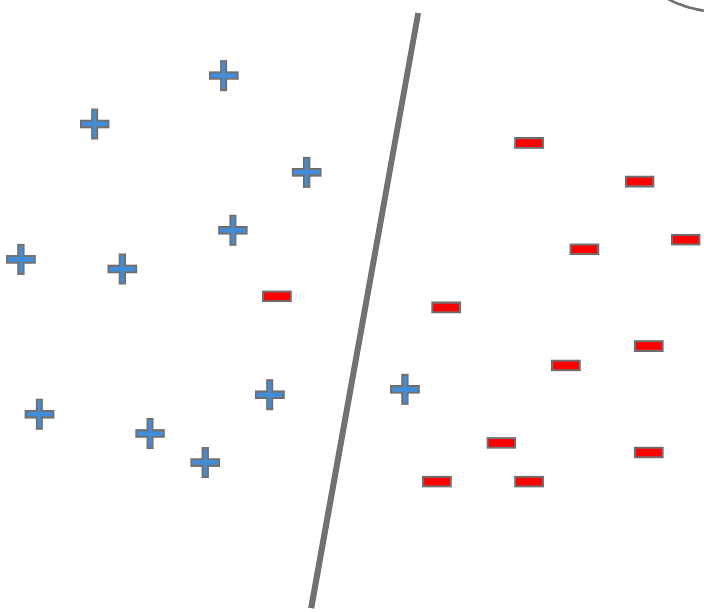
- Having *n* training data, is it always enough to map the data into a feature space with dimension *n-1*?
  - *Nope... We have to think about the test data as well! Even if we don't know how many test data we have and what they are...*
  - *We might want to map our data to a huge (∞) dimensional feature space*
  - *Overfitting? Generalization error?... We don't care now...*

How to do feature mapping?

Let us have n training objects: $\vec{x}_i = [\vec{x}_{i,1}, \vec{x}_{i,2}] \in \mathbb{R}^2$, $i = 1, \ldots, n$

The possible test objects are denoted by $\vec{x} = [\vec{x}_1, \vec{x}_2] \in \mathbb{R}^2$

$$\text{Let } \phi(\vec{x}) \doteq \underbrace{[sin(\vec{x}_2), \exp(\vec{x}_2 + \vec{x}_1), \vec{x}_1, \vec{x}_2^{\tan(\vec{x}_1)}, \ldots]}_{\infty}$$



Use features of features of features of features….

The Problem with Hard SVM

It assumes samples are linearly separable...

Solutions:

1. Use feature transformation to a larger space
   $\Rightarrow$ each training samples are linearly separable in the feature space
   $\Rightarrow$ Hard SVM can be applied ☺
   $\Rightarrow$ overfitting... ☹

2. **Soft margin** SVM instead of Hard SVM
   - Slack variables... We will discuss them now

Hard SVM

The Hard SVM problem can be rewritten:

$$\hat{\mathbf{w}}_{hard} = \arg\min_{\mathbf{w}\in\mathbb{R}^m} \frac{1}{2}\|\mathbf{w}\|^2$$

$$\text{subject to } y_i\langle \mathbf{x}_i, \mathbf{w}\rangle \geq 1,\ \forall i = 1, \ldots, n$$

$$\Downarrow$$

$$\boxed{\hat{\mathbf{w}}_{hard} = \arg\min_{\mathbf{w}\in\mathbb{R}^m} \sum_{i=1}^{n} l_{0-\infty}(\langle \mathbf{x}_i, \mathbf{w}\rangle, y_i) + \frac{1}{2}\|\mathbf{w}\|^2}$$

where

$$l_{0-\infty}(a, b) \doteq \begin{cases} \infty : ab < 1 & \text{Misclassification, or inside the margin} \\ 0 : ab \geq 1 & \text{Correct classification and outside of the margin} \end{cases}$$

From Hard to Soft constraints

Instead of using hard constraints (points are linearly separable)

$$\hat{\mathbf{w}}_{hard} = \arg\min_{\mathbf{w}\in\mathbb{R}^m} \sum_{i=1}^{n} l_{0-\infty}(\langle \mathbf{x}_i, \mathbf{w}\rangle, y_i) + \frac{1}{2}\|\mathbf{w}\|^2$$

We can try solve the soft version of it:. Introduce a λ parameter!

(Your loss is only 1 instead of ∞ if you misclassify an instance)

$$\hat{\mathbf{w}}_{soft} = \arg\min_{\mathbf{w}\in\mathbb{R}^m} \sum_{i=1}^{n} l_{0-1}(\langle \mathbf{x}_i, \mathbf{w}\rangle, y_i) + \frac{\lambda}{2}\|\mathbf{w}\|^2$$

where

$$l_{0-1}(y, f(\mathbf{x})) = \begin{cases} 1 : yf(\mathbf{x}) < 0 & \text{Misclassification} \\ 0 : yf(\mathbf{x}) > 0 & \text{Correct classification} \end{cases}$$

Problems with $l_{0\text{-}1}$ loss

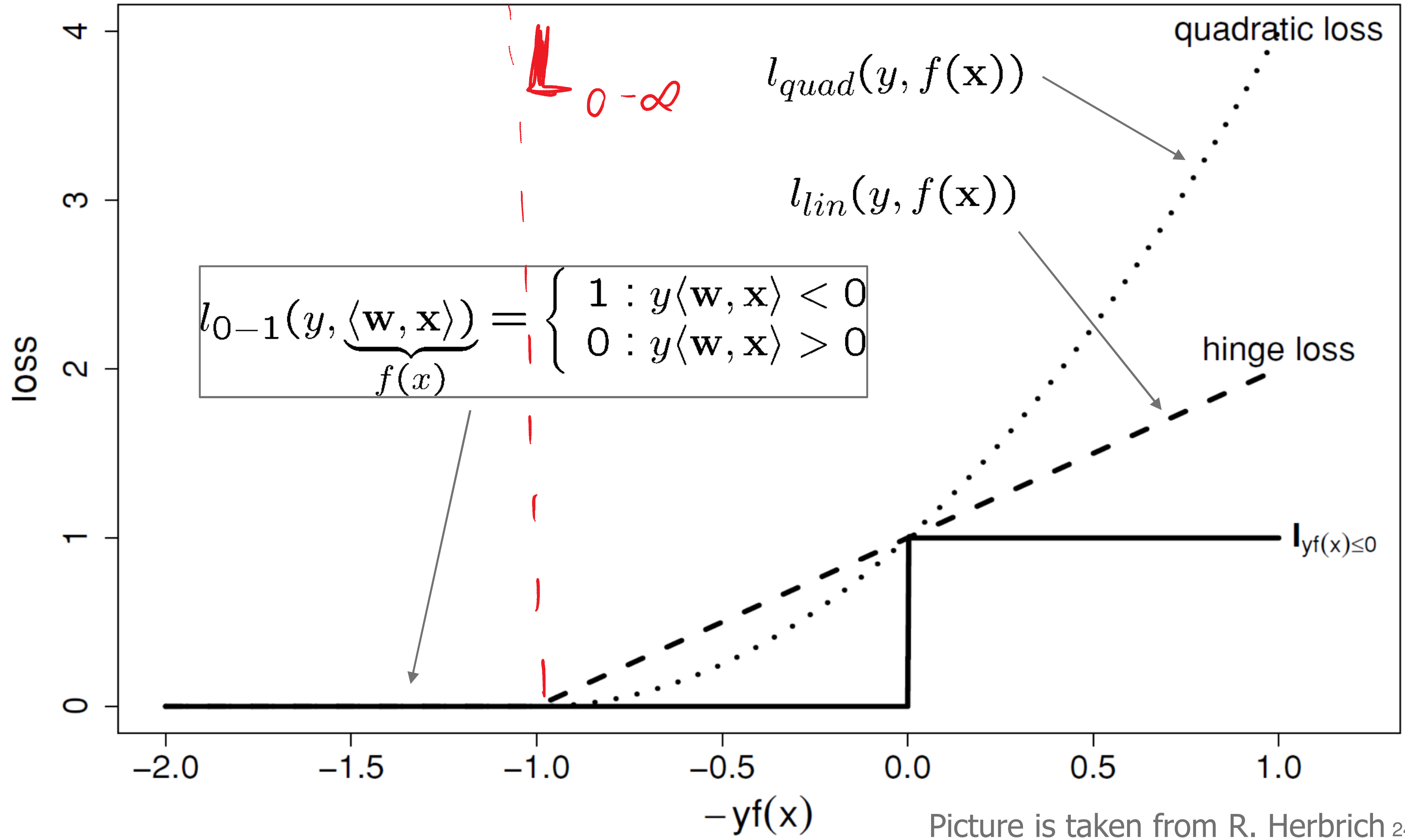
$$\hat{\mathbf{w}}_{soft} = \arg\min_{\mathbf{w}\in\mathbb{R}^m} \sum_{i=1}^{n} l_{0-1}(\langle \mathbf{x}_i, \mathbf{w}\rangle, y_i) + \frac{\lambda}{2}\|\mathbf{w}\|^2$$

$$l_{0-1}(y, f(\mathbf{x})) = \begin{cases} 1 : yf(\mathbf{x}) < 0 \\ 0 : yf(\mathbf{x}) > 0 \end{cases}$$

It is not convex in $yf(\mathbf{x})$ $\Rightarrow$ It is not convex in **w**, either...
... and we only like convex functions...

Let us approximate it with convex functions!

Approximation of the Heaviside step function



Picture is taken from R. Herbrich

Approximations of $l_{0\text{-}1}$ loss

- Piecewise linear approximations (hinge loss, l_{lin})

$$l_{lin}(f(\mathbf{x}), y) = \max\{1 - yf(\mathbf{x}), 0)\}$$

[We want $yf(\mathbf{x}) > 1$]

- Quadratic approximation (l_{quad})

$$l_{quad}(f(\mathbf{x}), y) = \max\{1 - yf(\mathbf{x}), 0)\}^2$$

The hinge loss approximation of $l_{0\text{-}1}$

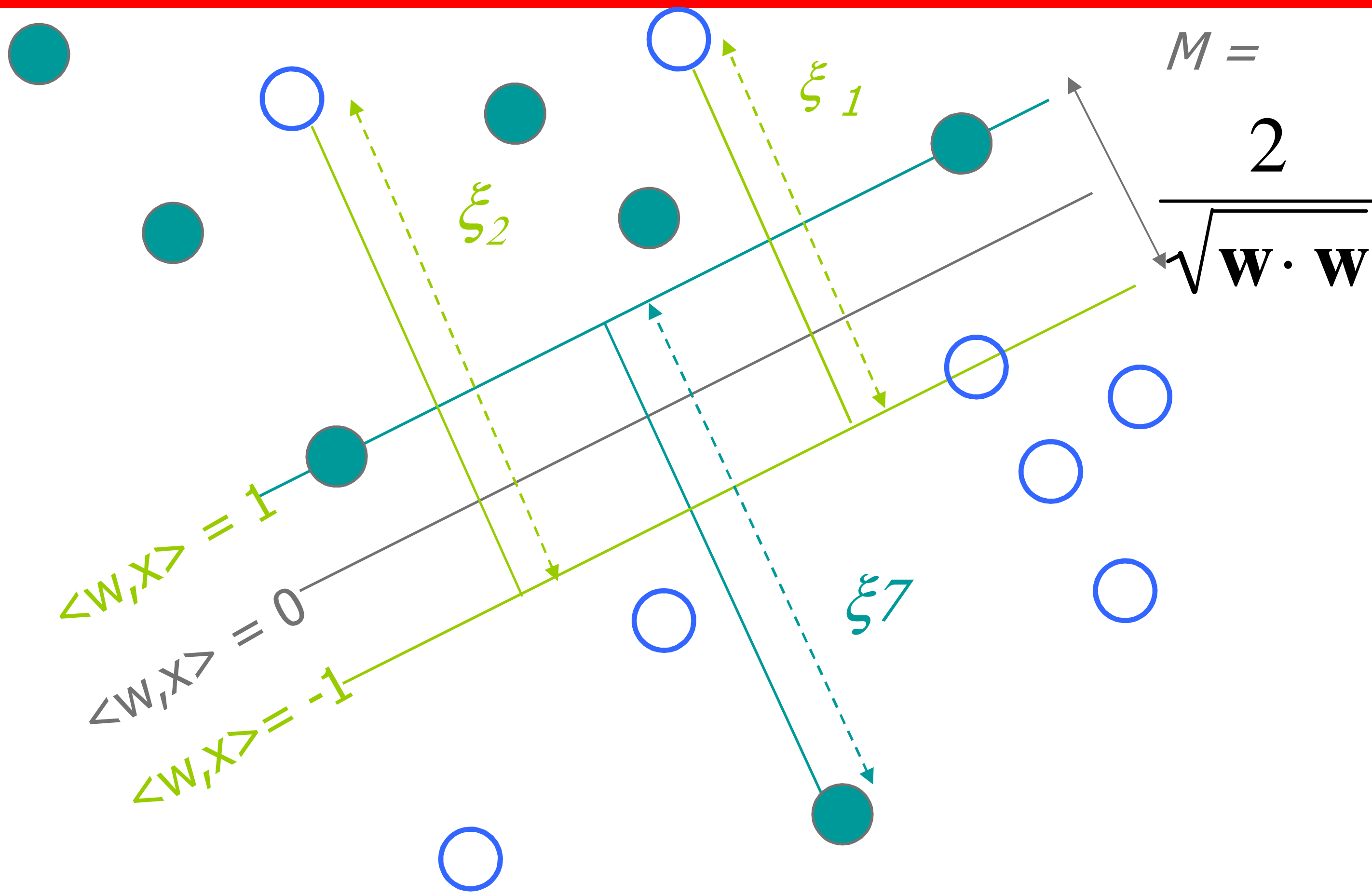
$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w}\in\mathbb{R}^m} \sum_{i=1}^{n} \underbrace{l_{lin}(\langle \mathbf{x}_i, \mathbf{w}\rangle, y_i)}_{\xi_i \geq 0} + \frac{\lambda}{2}\|\mathbf{w}\|^2$$

Where,

$$\xi_i \doteq l_{lin}(f(\mathbf{x}_i), y_i) = \max\{1 - y_i f(\mathbf{x}_i), 0)\}$$

$$\geq 1 - y_i \underbrace{\langle \mathbf{w}, \mathbf{x}_i \rangle}_{f(\mathbf{x}_i)} \geq l_{0-1}(y_i, f(\mathbf{x}_i))$$

The Slack Variables



$$\xi_i \doteq l_{lin}(f(\mathbf{x}_i), y_i) = \max\{1 - y_i f(\mathbf{x}_i), 0)\}$$
$$= \max\{1 - y_i(\mathbf{w}^T \mathbf{x}_i), 0\}$$

The Primal Soft SVM problem

$$\hat{\mathbf{w}}_{soft} = \arg\min_{\mathbf{w}\in\mathbb{R}^m} \sum_{i=1}^{n} \underbrace{l_{lin}(\langle \mathbf{x}_i, \mathbf{w}\rangle, y_i)}_{\xi_i \geq 0} + \frac{\lambda}{2}\|\mathbf{w}\|^2$$

where

$$\xi_i \doteq l_{lin}(f(\mathbf{x}_i), y_i) = \max\{1 - y_i(\mathbf{w}^T\mathbf{x}_i), 0\}$$

Equivalently,

$$\hat{\mathbf{w}}_{soft} = \arg\min_{\mathbf{w}\in\mathbb{R}^m, \boldsymbol{\xi}\in\mathbb{R}^n} \sum_{i=1}^{n} \xi_i + \frac{\lambda}{2}\|\mathbf{w}\|^2$$

$$\text{subject to } y_i\langle \mathbf{x}_i, \mathbf{w}\rangle \geq 1 - \xi_i,\ \forall i = 1, \ldots, n$$

$$\xi_i \geq 0,\ \forall i = 1, \ldots, n$$

ξ_i: Slack variables

The Primal Soft SVM problem

$$\hat{\mathbf{w}}_{soft} = \arg \min_{\mathbf{w}\in\mathbb{R}^m, \boldsymbol{\xi}\in\mathbb{R}^n} \sum_{i=1}^{n} \xi_i + \frac{\lambda}{2}\|\mathbf{w}\|^2$$

$$\text{subject to } y_i\langle \mathbf{x}_i, \mathbf{w}\rangle \geq 1 - \xi_i, \ \forall i = 1, \ldots, n$$

Equivalently,

$$\xi_i \geq 0, \ \forall i = 1, \ldots, n$$

We can use this form, too... where $C = \frac{1}{\lambda}$

$$\hat{\mathbf{w}}_{soft} = \arg \min_{\mathbf{w}\in\mathcal{R}^m, \boldsymbol{\xi}\in\mathbb{R}^n} C \underbrace{\sum_{i=1}^{n} \xi_i}_{\boldsymbol{\xi}^T \mathbf{1}_n} + \frac{1}{2}\|\mathbf{w}\|^2$$

$$\text{subject to } y_i\langle \mathbf{x}_i, \mathbf{w}\rangle \geq 1 - \xi_i, \ \forall i = 1, \ldots, n$$

$$\xi_i \geq 0, \ \forall i = 1, \ldots, n$$

What is the dual form of primal soft SVM?

The Dual Soft SVM (using hinge loss)

$$\hat{\mathbf{w}}_{soft} = \arg \min_{\mathbf{w}\in\mathcal{R}^m, \boldsymbol{\xi}\in\mathbb{R}^n} C \sum_{i=1}^{n} \xi_i + \frac{1}{2}\|\mathbf{w}\|^2$$

$$\text{subject to } y_i\langle \mathbf{x}_i, \mathbf{w}\rangle \geq 1 - \xi_i,\ \forall i = 1, \ldots, n$$

$$\xi_i \geq 0,\ \forall i = 1, \ldots, n$$

$\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_n)^T \geq 0$ Largrange multipliers

$\boldsymbol{\beta} = (\beta_1, \ldots, \beta_n)^T \geq 0$ Largrange multipliers

$$(\hat{\mathbf{w}}, \hat{\boldsymbol{\xi}}, \hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\beta}}) = \arg \min_{\substack{\mathbf{w} \in \mathbb{R}^m \\ \mathbf{0} \leq \boldsymbol{\xi} \in \mathbb{R}^n}} \max_{\substack{0 \leq \boldsymbol{\alpha} \\ 0 \leq \boldsymbol{\beta}}} L(\mathbf{w}, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta})$$

where

$$L(\mathbf{w}, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n} \xi_i - \sum_{i=1}^{n} \alpha_i\Big(y_i\langle \mathbf{x}_i, \mathbf{w}\rangle - 1 + \xi_i\Big) - \sum_{i=1}^{n} \beta_i\xi_i$$

The Dual Soft SVM (using hinge loss)

$$L(\mathbf{w}, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \frac{1}{2}\|\mathbf{w}\|^2 + C\boldsymbol{\xi}^T\mathbf{1}_n - \sum_{i=1}^{n} \alpha_i y_i \langle \mathbf{x}_i, \mathbf{w} \rangle + \boldsymbol{\alpha}^T\mathbf{1}_n - \boldsymbol{\xi}^T(\boldsymbol{\alpha} + \boldsymbol{\beta})$$

$$0 = \left.\frac{\partial L(\mathbf{w}, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta})}{\partial \mathbf{w}}\right|_{\mathbf{w}=\hat{\mathbf{w}}} = \hat{\mathbf{w}} - \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i \Rightarrow \boxed{\hat{\mathbf{w}} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i}$$

$$0 = \left.\frac{\partial L(\mathbf{w}, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta})}{\partial \boldsymbol{\xi}}\right|_{\boldsymbol{\xi}=\hat{\boldsymbol{\xi}}} = C\mathbf{1}_n - \boldsymbol{\alpha} - \boldsymbol{\beta} \Rightarrow \boldsymbol{\beta} = C\mathbf{1}_n - \boldsymbol{\alpha} \geq 0$$

$$\boxed{\Rightarrow 0 \leq \boldsymbol{\alpha} \leq C}$$

$$\Rightarrow (\hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\beta}}) = \arg \max_{\substack{0 \leq \boldsymbol{\alpha} \leq C \\ 0 \leq \boldsymbol{\beta}}} L(\hat{\mathbf{w}}, \hat{\boldsymbol{\xi}}, \boldsymbol{\alpha}, \boldsymbol{\beta})$$

$$\boxed{\Rightarrow \hat{\boldsymbol{\alpha}} = \arg \max_{0 \leq \boldsymbol{\alpha} \leq C} \boldsymbol{\alpha}^T\mathbf{1}_n - \tfrac{1}{2}\boldsymbol{\alpha}^T \boldsymbol{Y}\boldsymbol{G}\boldsymbol{Y}\boldsymbol{\alpha}}$$

The Dual Soft SVM (using hinge loss)

$\boldsymbol{Y} \doteq diag(y_1, \ldots, y_n) \in \{-1, 1\}^n$

$\boldsymbol{G} \in \mathbb{R}^{n \times n} \doteq \{G_{ij}\}_{i,j}^{n,n}$, where $G_{ij} \doteq \overbrace{\langle \mathbf{x}_i, \mathbf{x}_j \rangle}^{k(\mathbf{x}_i, \mathbf{x}_j)}$, Gram matrix.

$$\hat{\boldsymbol{\alpha}} = \arg\max_{\boldsymbol{\alpha} \in \mathbb{R}^n} \boldsymbol{\alpha}^T \mathbf{1}_n - \frac{1}{2} \boldsymbol{\alpha}^T \boldsymbol{Y} \boldsymbol{G} \boldsymbol{Y} \boldsymbol{\alpha}$$

subject to $0 \le \alpha_i \le C$

where $C = \frac{1}{\lambda}$

If $\lambda \to \mathbf{0} \Rightarrow$ soft-SVM $\to$ hard-SVM

This is the same as the dual hard-SVM problem,
but now we have the additional $0 \le \alpha_i \le C$ constraints.

SVM classification in the dual space

Solve the dual problem

$$\hat{\boldsymbol{\alpha}} = \underset{\boldsymbol{\alpha} \in \mathbb{R}^n}{\text{arg max}}\ \boldsymbol{\alpha}^T \mathbf{1}_n - \frac{1}{2} \boldsymbol{\alpha}^T \boldsymbol{Y} \boldsymbol{G} \boldsymbol{Y} \boldsymbol{\alpha}$$

$$\text{subject to } 0 \leq \alpha_i \leq C$$

where $C = \frac{1}{\lambda}$. Let $\hat{\mathbf{w}} = \sum_{i=1}^{n} \hat{\alpha}_i y_i \mathbf{x}_i$.

On test data $\mathbf{x}$: $f_{\hat{\mathbf{w}}}(\mathbf{x}) = \langle \hat{\mathbf{w}}, \mathbf{x} \rangle = \sum_{i=1}^{n} \hat{\alpha}_i y_i \underbrace{\langle \mathbf{x}_i, \mathbf{x} \rangle}_{k(\mathbf{x}_i, \mathbf{x})}$

Why is it called Support Vector Machine?

$\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_n)^T \geq 0$ Lagrange multipliers

$$L(\mathbf{w}, \boldsymbol{\alpha}) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{n} \alpha_i \left(y_i \langle \mathbf{x}_i, \mathbf{w} \rangle - 1 \right)$$

0

KKT conditions

COMPLEMENTARY SLACKNESS CONDITION

$\alpha_i > 0 \Rightarrow y_i \langle x_i, w \rangle = 1$

$\langle x_i, w \rangle = \begin{matrix} +1 \\ -1 \end{matrix}$

EITHER $\alpha_i = 0$ OR $(y_i \langle x_i, w \rangle - 1) = 0$ AND $\alpha_i > 0$

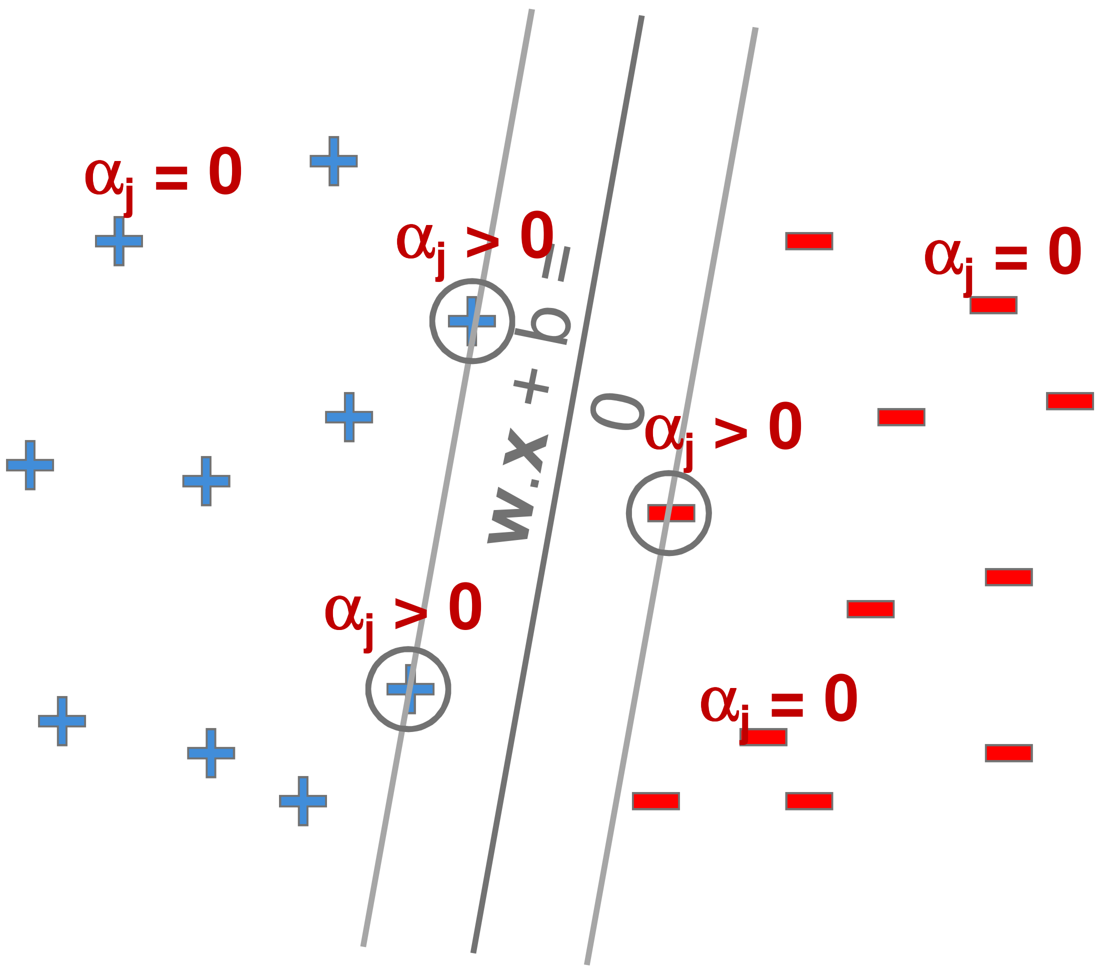
$(y_i \langle x_i, w \rangle - 1) \neq 0$

$\Downarrow$

x_i IS ON THE MARGIN LINES

SUPPORT VECTORS

Dual SVM Interpretation: Sparsity



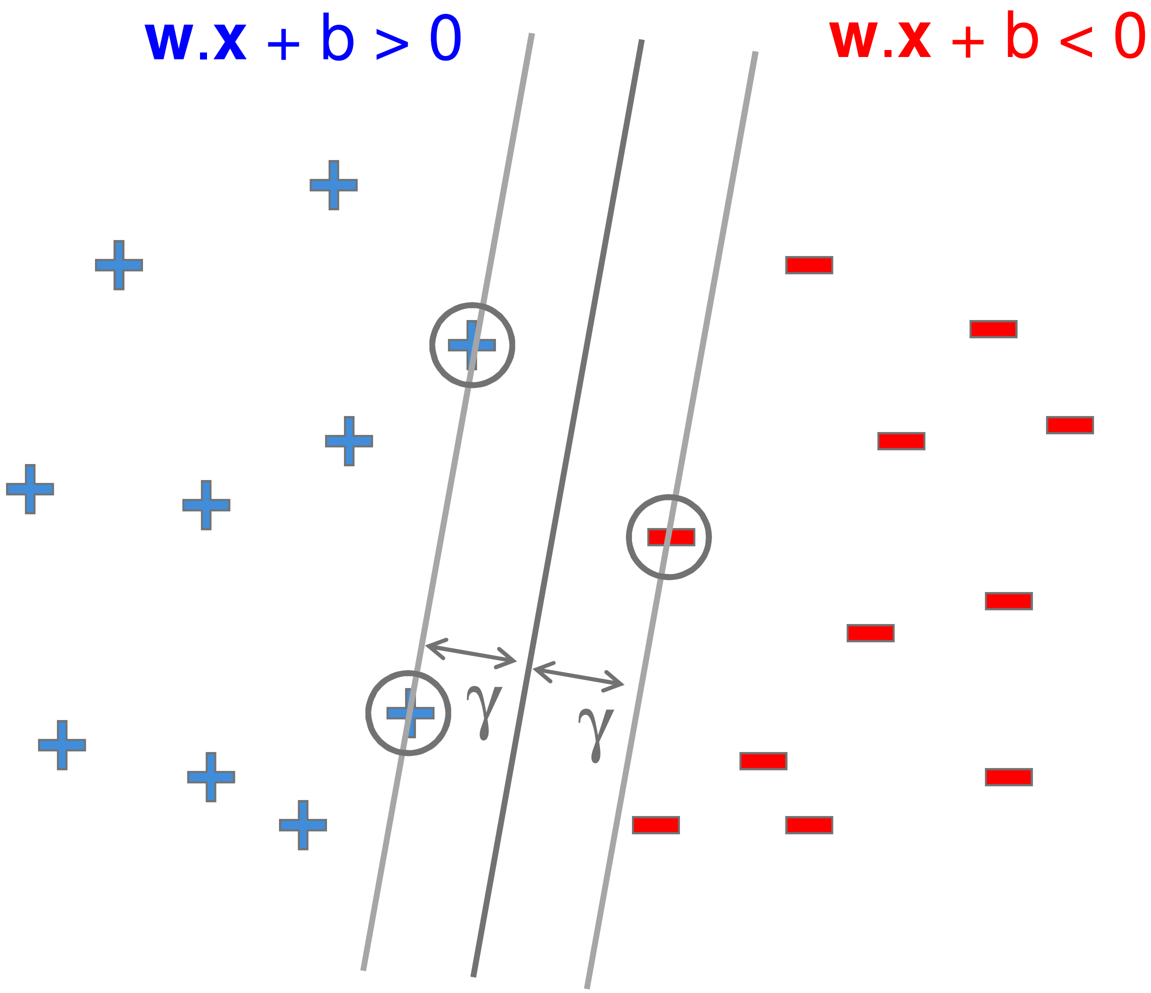
$$\mathbf{w} = \sum_j \alpha_j y_j \mathbf{x}_j$$

Only few α_js can be non-zero : where constraint is tight

$(\langle \mathbf{w}, \mathbf{x}_j \rangle + b) y_j = 1$

Support vectors – training points j whose α_js are non-zero

Support Vectors



Linear hyperplane defined by "support vectors"

Moving other points a little doesn't effect the decision boundary

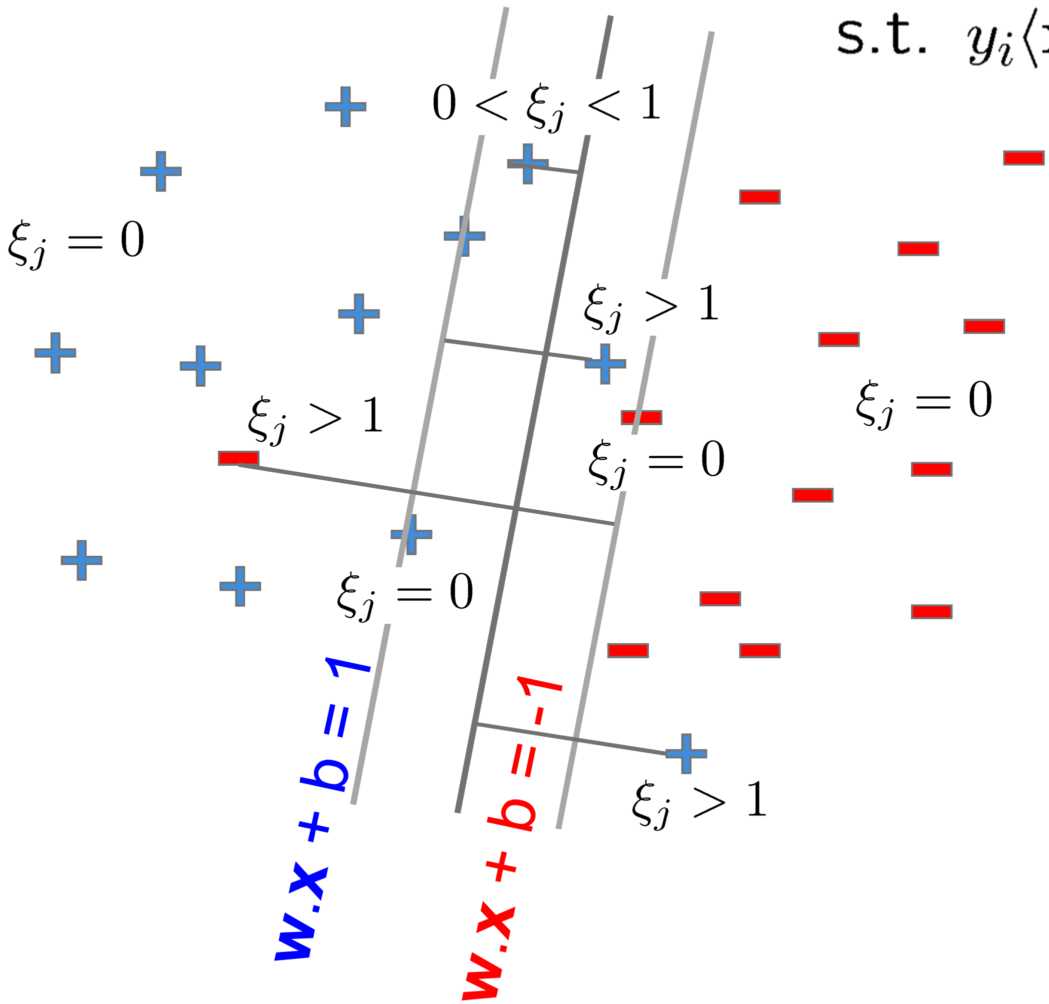
only need to store the support vectors to predict labels of new points

Support vectors in Soft SVM

$$\widehat{\mathbf{w}}_{soft} = \arg \min_{\mathbf{w} \in \mathbb{R}^m, \boldsymbol{\xi} \in \mathbb{R}^n} C \sum_{i=1}^{n} \xi_i + \frac{1}{2}\|\mathbf{w}\|^2$$

$$\text{s.t. } y_i \langle \mathbf{x}_i, \mathbf{w} \rangle \geq 1 - \xi_i, \ \forall i = 1, \ldots, n$$

$$\xi_i \geq 0, \ \forall i = 1, \ldots, n$$

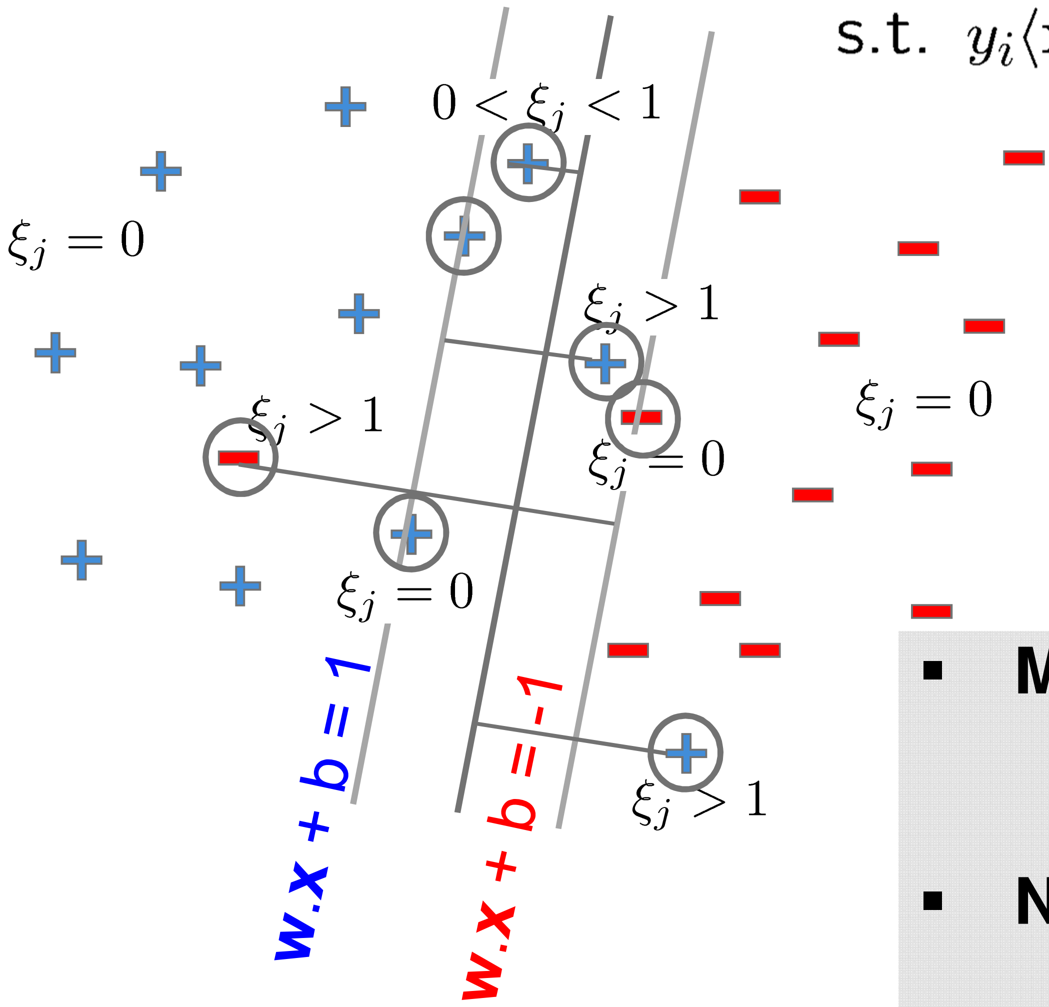


Support vectors in Soft SVM

$$\hat{\mathbf{w}}_{soft} = \arg \min_{\mathbf{w}\in\mathbb{R}^m, \boldsymbol{\xi}\in\mathbb{R}^n} C\sum_{i=1}^{n} \xi_i + \frac{1}{2}\|\mathbf{w}\|^2$$

$$\text{s.t. } y_i\langle \mathbf{x}_i, \mathbf{w}\rangle \geq 1 - \xi_i, \ \forall i = 1, \ldots, n$$

$$\xi_i \geq 0, \ \forall i = 1, \ldots, n$$



- **Margin support vectors**
  $y_i\langle \mathbf{x}_i, \mathbf{w}\rangle = 1$
- **Nonmargin support vectors**
  $\xi_i > 0$

SVM classification in the dual space

"Without b"

$$\hat{\boldsymbol{\alpha}} = \arg\max_{\boldsymbol{\alpha}\in\mathbb{R}^m} \boldsymbol{\alpha}^T \mathbf{1}_m - \frac{1}{2}\boldsymbol{\alpha}^T \boldsymbol{Y}\boldsymbol{G}\boldsymbol{Y}\boldsymbol{\alpha}$$

subject to $0 \le \alpha_i \le C$

"With b"

$$L(\mathbf{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{n} \alpha_i \Big(y_i(\mathbf{x}_i \cdot \mathbf{w} + b) - 1 \Big)$$

$$\hat{\boldsymbol{\alpha}} = \arg\max_{\boldsymbol{\alpha}\in\mathbb{R}^n} \boldsymbol{\alpha}^T \mathbf{1}_n - \frac{1}{2}\boldsymbol{\alpha}^T \boldsymbol{Y}\boldsymbol{G}\boldsymbol{Y}\boldsymbol{\alpha}$$

subject to $0 \le \alpha_i \le C$

$$\sum_i \alpha_i y_i = 0$$

SVM with Linear Programs

QP:

Max margin

$$\min_{\mathbf{w}\in\mathbb{R}^m, \boldsymbol{\xi}\in\mathbb{R}^n} C\sum_{i=1}^{n} \xi_i + \frac{1}{2}\|\mathbf{w}\|^2$$

$$\text{subject to } y_i\langle \mathbf{x}_i, \mathbf{w}\rangle \geq 1 - \xi_i,\ \forall i = 1, \ldots, n$$

$$\xi_i \geq 0,\ \forall i = 1, \ldots, n$$

Min support vectors

LP:

$$\min_{\boldsymbol{\alpha}\in\mathbb{R}^n, \boldsymbol{\xi}\in\mathbb{R}^n} C\sum_{i=1}^{n} \xi_i + \sum_{i=1}^{n} \alpha_i$$

$$\text{subject to } y_i\langle \mathbf{x}_i, \mathbf{w}\rangle \geq 1 - \xi_i,\ \forall i = 1, \ldots, n$$

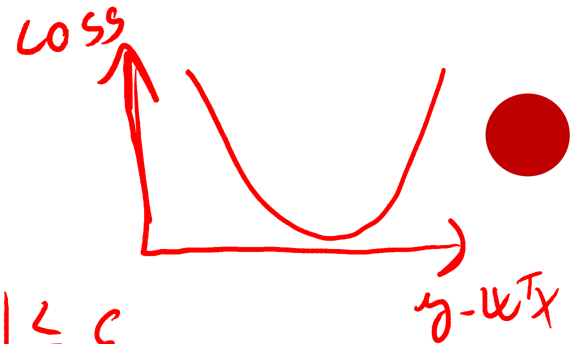
$$\xi_i \geq 0,\ \forall i = 1, \ldots, n$$

$$\alpha_i \geq 0,\ \forall i = 1, \ldots, n$$

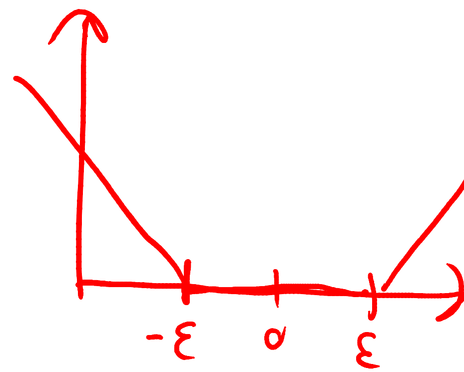
$$\mathbf{w} = \sum_{j=1}^{n} \alpha_j y_j \mathbf{x}_j$$

SVM for Regression

$$\text{LOSS}(y, w^T x) = (y - w^T x)^2$$



$$\begin{cases} 0 & \text{IF } |y - w^T x| \le \varepsilon \\ |y - w^T x| & \text{OTHERWISE} \end{cases}$$



Ridge Regression

Linear regression: $f(x) = \langle \mathbf{w}, \phi(x) \rangle$

Primal:

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w} \in \mathcal{K}} \sum_{i=1}^{n} \xi_i^2$$

$$\text{subject to } y_i - \langle \underbrace{\phi(x_i)}_{\mathbf{x}_i}, \mathbf{w} \rangle = \xi_i, \ \forall i = 1, \ldots, n$$

$$\text{and } \|\mathbf{w}\| \leq B$$

$$L(\mathbf{w}, \boldsymbol{\xi}, \boldsymbol{\alpha}, \beta) = \sum_{i=1}^{n} \xi_i^2 + \sum_{i=1}^{n} \alpha_i (y_i - \langle \phi(x_i), \mathbf{w} \rangle - \xi_i) + \lambda(\|\mathbf{w}\|^2 - B^2)$$

$$\lambda > 0$$

Dual for a given λ: …after some calculations…

$$\hat{\boldsymbol{\alpha}} = \arg\max_{\boldsymbol{\alpha} \in \mathbb{R}^n} \lambda \sum_{i=1}^{n} \alpha_i^2 - 2 \sum_{i=1}^{n} \alpha_i y_i + \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j k(x_i, x_j)$$

This can be solved in closed form:

Kernel Ridge Regression Algorithm

Given $D = \{(x_i, y_i), i = 1, \ldots, n\}$ training data set.

$k(\cdot,\cdot)$ kernel, $\lambda > 0$ parameter. $\mathbf{y} \doteq (y_1, \ldots, y_n)^T \in \{-1, 1\}^n$

- $\boldsymbol{G} \in \mathbb{R}^{n\times n} \doteq \{G_{ij}\}_{i,j}^{n,n}$,

  where $G_{ij} \doteq \overbrace{\langle \underbrace{\mathbf{x}_i}_{\phi(x_i)}, \underbrace{\mathbf{x}_j}_{\phi(x_j)} \rangle_{\mathcal{K}}}^{k(x_i,x_j)}$, Gram matrix.
- $\hat{\boldsymbol{\alpha}} = (\boldsymbol{G} + \lambda \boldsymbol{I}_n)^{-1}\mathbf{y}$
- $\hat{\mathbf{w}} = \sum_{i=1}^{n} \hat{\alpha}_i \phi(x_i)$.
- $f(x) = \langle \hat{\mathbf{w}}, \phi(x) \rangle = \sum_{i=1}^{n} \hat{\alpha}_i k(x_i, x)$

SVM vs. Logistic Regression

SVM : **Hinge loss**

$$\text{loss}(f(x_j), y_j) = (1 - (\mathbf{w} \cdot x_j + b)y_j))_+$$

Logistic Regression : **Log loss** (log conditional likelihood)

$$\text{loss}(f(x_j), y_j) = -\log P(y_j \mid x_j, \mathbf{w}, b) = \log(1 + e^{-(\mathbf{w} \cdot x_j + b)y_j})$$

Difference between SVMs and Logistic Regression

	SVMs	**Logistic Regression**
Loss function	Hinge loss	Log-loss
High dimensional features with kernels	**Yes!**	**No** (but there is kernel logistic regression too)
Solution sparse	Often yes!	Almost always no!
Semantics of output	“Margin”	“Real probabilities”

Constructing Kernels

Common Kernels

- Polynomials of degree d

$$K(\mathbf{u},\mathbf{v}) = (\mathbf{u}\cdot\mathbf{v})^d$$

- Polynomials of degree up to d

$$K(\mathbf{u},\mathbf{v}) = (\mathbf{u}\cdot\mathbf{v}+1)^d$$

- Gaussian/Radial kernels

$$K(\mathbf{u},\mathbf{v}) = \exp\left(-\frac{||\mathbf{u}-\mathbf{v}||^2}{2\sigma^2}\right)$$

- Sigmoid

$$K(\mathbf{u},\mathbf{v}) = \tanh(\eta\mathbf{u}\cdot\mathbf{v}+\nu)$$

Designing new kernels from kernels

$k_1 : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, $k_2 : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ are kernels $\Rightarrow$

1. $k(x, \tilde{x}) = k_1(x, \tilde{x}) + k_2(x, \tilde{x})$,
2. $k(x, \tilde{x}) = c \cdot k_1(x, \tilde{x})$, *for all* $c \in \mathbb{R}^+$,
3. $k(x, \tilde{x}) = k_1(x, \tilde{x}) + c$, *for all* $c \in \mathbb{R}^+$,
4. $k(x, \tilde{x}) = k_1(x, \tilde{x}) \cdot k_2(x, \tilde{x})$,
5. $k(x, \tilde{x}) = f(x) \cdot f(\tilde{x})$, *for any function* $f : \mathcal{X} \to \mathbb{R}$

are also kernels.

Picture is taken from R. Herbrich

Designing new kernels from kernels

1. $k(x,\tilde{x}) = (k_1(x,\tilde{x}) + \theta_1)^{\theta_2}$, *for all* $\theta_1 \in \mathbb{R}^+$ *and* $\theta_2 \in \mathbb{N}$
2. $k(x,\tilde{x}) = \exp\left(\frac{k_1(x,\tilde{x})}{\sigma^2}\right)$, *for all* $\sigma \in \mathbb{R}^+$,
3. $k(x,\tilde{x}) = \exp\left(-\frac{k_1(x,x) - 2k_1(x,\tilde{x}) + k_1(\tilde{x},\tilde{x})}{2\sigma^2}\right)$, *for all* $\sigma \in \mathbb{R}^+$
4. $k(x,\tilde{x}) = \frac{k_1(x,\tilde{x})}{\sqrt{k_1(x,x) \cdot k_1(\tilde{x},\tilde{x})}}$

Picture is taken from R. Herbrich

Designing new kernels from kernels

The meaning of

$$k(x, \tilde{x}) = \frac{k_1(x, \tilde{x})}{\sqrt{k_1(x, x) k_1(\tilde{x}, \tilde{x})}}$$

is that we can normalize the data in the feature space without performing the explicit mapping.

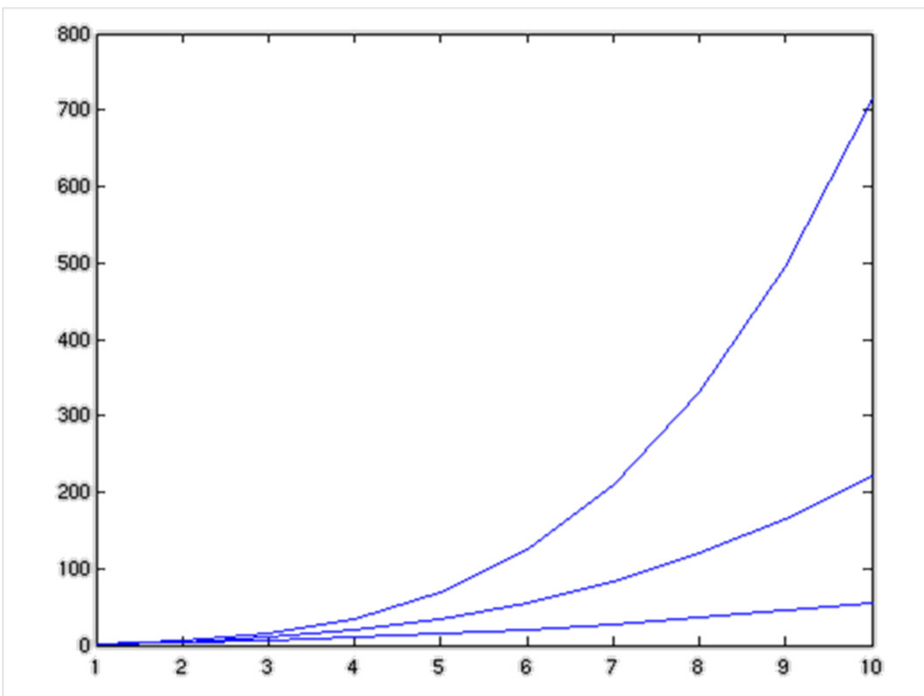
Use the normailzed kernel k_{norm}:

$$k_{norm}(x, \tilde{x}) \doteq \frac{k(x, \tilde{x})}{\sqrt{k(x, x) k(\tilde{x}, \tilde{x})}} = \frac{\langle x, \tilde{x} \rangle}{\sqrt{\|x\|^2 \|\tilde{x}\|^2}} = \langle \frac{x}{\|x\|^2}, \frac{\tilde{x}}{\|\tilde{x}\|^2} \rangle$$

Higher Order Polynomials

m – input features d – degree of polynomial

$$\text{num. terms } = \binom{d+m-1}{d} = \frac{(d+m-1)!}{d!(m-1)!} \sim m^d$$



grows fast!
d = 6, m = 100
about 1.6 billion terms

Dot Product of Polynomials

$\Phi(\mathbf{x}) =$ polynomials of degree exactly d

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \quad \mathbf{z} = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}$$

d=1 $\Phi(\mathbf{x}) \cdot \Phi(\mathbf{z}) = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \cdot \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = x_1 z_1 + x_2 z_2 = \mathbf{x} \cdot \mathbf{z}$

d=2 $\Phi(\mathbf{x}) \cdot \Phi(\mathbf{z}) = \begin{bmatrix} x_1^2 \\ x_1 x_2 \\ x_2^2 \end{bmatrix} \cdot \begin{bmatrix} z_1^2 \\ z_1 z_2 \\ z_2^2 \end{bmatrix}$

$$\begin{aligned} &= x_1^2 z_1^2 + x_2^2 z_2^2 + x_1 x_2 z_1 z_2 \\ &= (x_1 z_1 + x_2 z_2)^2 \\ &= (\mathbf{x} \cdot \mathbf{z})^2 \end{aligned}$$

d $\Phi(\mathbf{x}) \cdot \Phi(\mathbf{z}) = K(\mathbf{x}, \mathbf{z}) = (\mathbf{x} \cdot \mathbf{z})^d$

$dim(\mathcal{X}) = N$

Name	Kernel function	dim $(\mathcal{K})$
pth degree polynomial	$k(\vec{u}, \vec{v}) = (\langle \vec{u}, \vec{v} \rangle_{\mathcal{X}})^p$ $p \in \mathbb{N}^+$	$\binom{N+p-1}{p}$
complete polynomial	$k(\vec{u}, \vec{v}) = (\langle \vec{u}, \vec{v} \rangle_{\mathcal{X}} + c)^p$ $c \in \mathbb{R}^+, \ \ p \in \mathbb{N}^+$	$\binom{N+p}{p}$
RBF kernel	$k(\vec{u}, \vec{v}) = \exp\left(-\frac{\lVert \vec{u}-\vec{v} \rVert_{\mathcal{X}}^2}{2\sigma^2}\right)$ $\sigma \in \mathbb{R}^+$	∞
Mahalanobis kernel	$k(\vec{u}, \vec{v}) = \exp\left(-(\vec{u} - \vec{v})' \, \mathbf{\Sigma} \, (\vec{u} - \vec{v})\right)$ $\mathbf{\Sigma} = \operatorname{diag}\left(\sigma_1^{-2}, \ldots, \sigma_N^{-2}\right)$, $\sigma_1, \ldots, \sigma_N \in \mathbb{R}^+$	∞

Picture is taken from R. Herbrich

The RBF kernel

Note:

The RBF kernel maps the input space $\mathcal{X}$ onto the surface of an infinite dimensional hypersphere.

Proof:

$$\|\phi(x)\| = \sqrt{k(x,x)} = \sqrt{\exp(0)} = 1$$

Note:

The RBF kernel is shift invariant:

$$k(u+a, v+a) = k(u,v), \ \forall a$$

Overfitting

- Huge feature space with kernels, what about overfitting???
  - Maximizing margin leads to sparse set of support vectors
  - Some interesting theory says that SVMs search for simple hypothesis with large margin
  - Often robust to overfitting

String kernels

P-spectrum kernel:

P=3: s="statistics" t="computation"

They contain the following substrings of length 3

"sta", "tat", "ati", "tis", "ist", "sti", "tic", "ics"
"com", "omp", "mpu", "put", "uta", "tat", "ati", "tio", "ion"

Common substrings: "tat", "ati"

k(s,t)=2

Distribution kernels

Euclidean: $K(p, q) = \int p(x)\, q(x)\, dx$

Bhattacharyya's affinity: $K(p, q) = \int \sqrt{p(x)}\, \sqrt{q(x)}\, dx$

Mean map: $\Phi(p) = \mathbb{E}_{x \sim p} K(\cdot, x)$

$$K(p, q) = \mathbb{E}_{x \sim p} \mathbb{E}_{y \sim q} K(x, y)$$

Set kernels

Mean map:

$$\frac{1}{nm}\sum_{i=1}^{n}\sum_{j=1}^{m} k(x_i, y_j) \stackrel{\circ}{=} \langle \{x_1 .. x_n\}, \{y_1 .. y_m\} \rangle$$

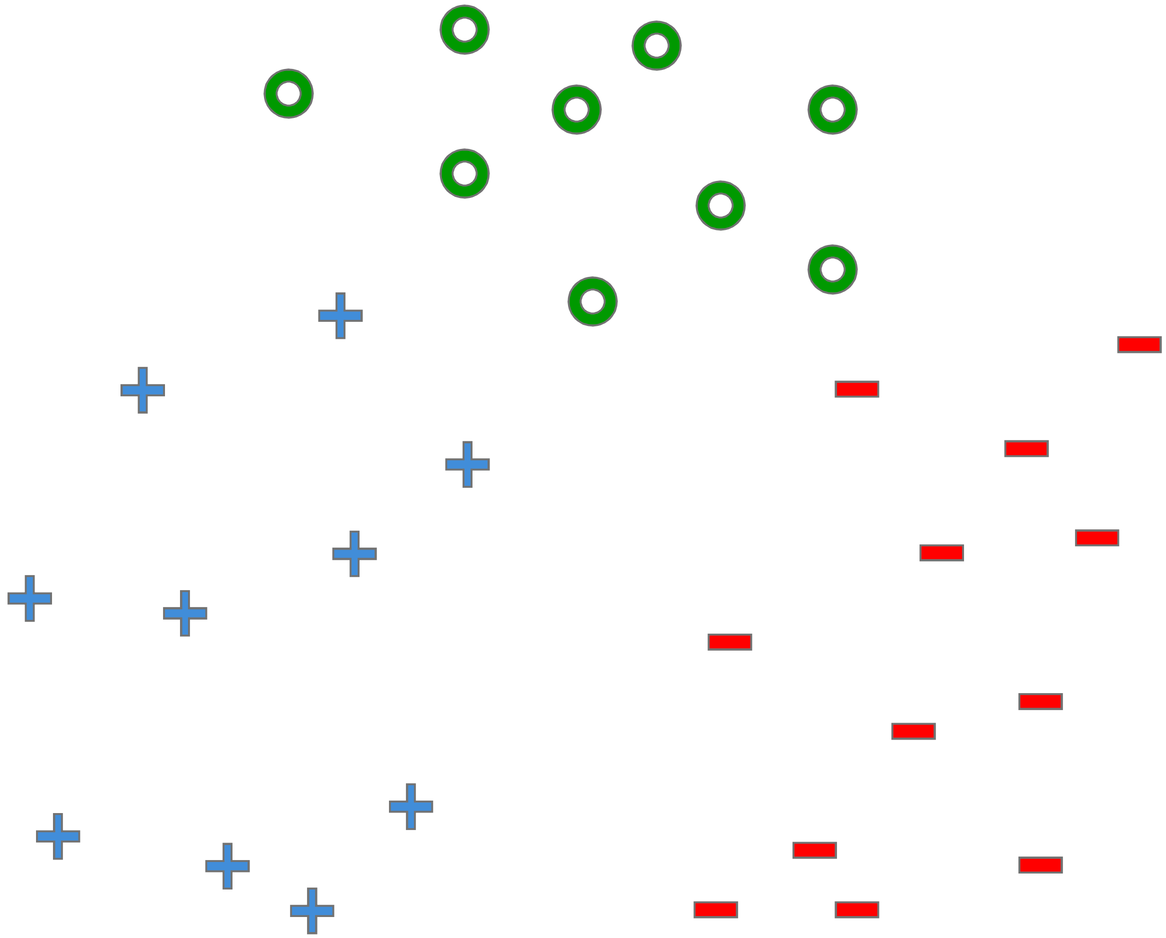
Intersection kernel:

$$k(A_1, A_2) = \int I_{A_1 \cap A_2}(x)\, d\mu(x)$$
$$= \mu(A_1 \cap A_2)$$

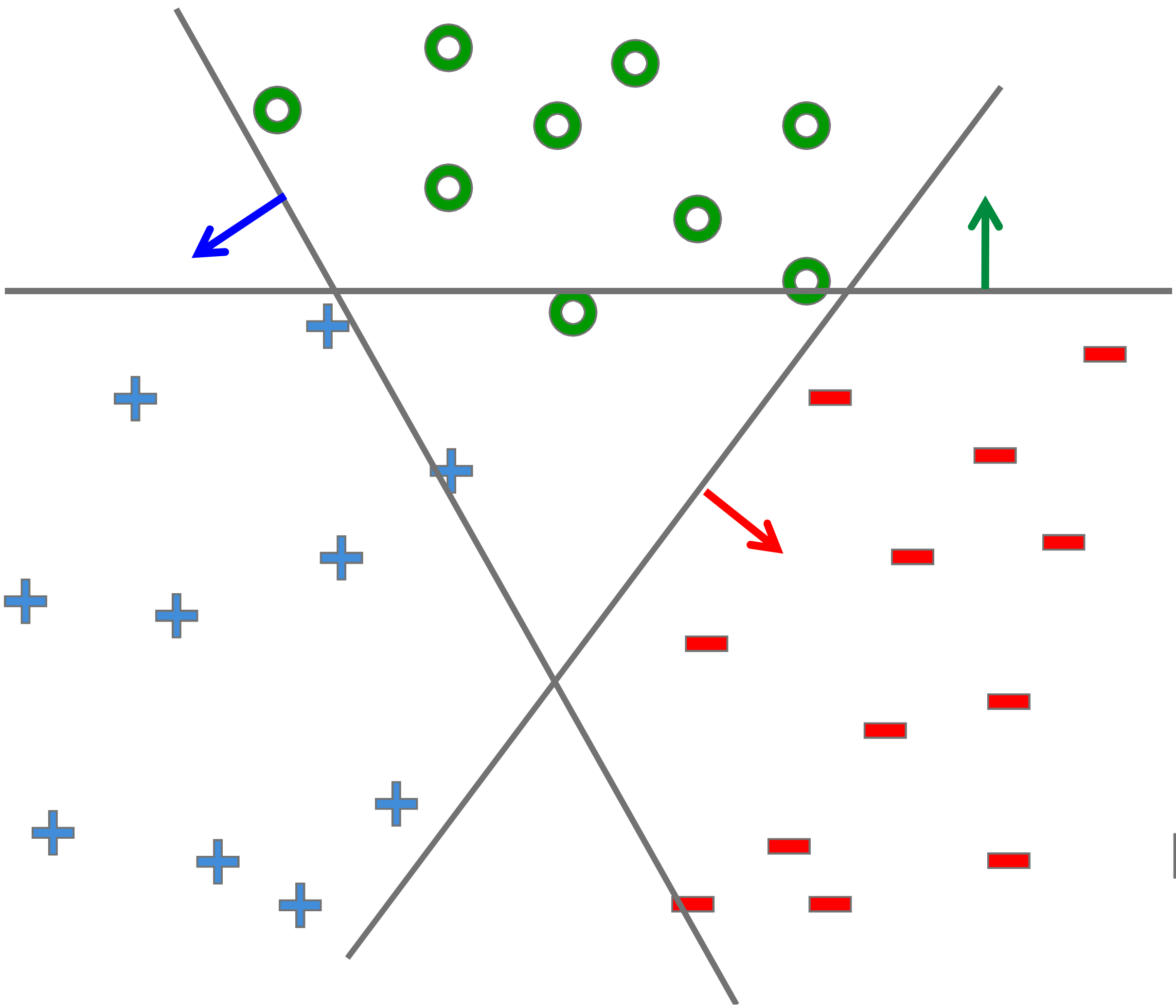
Union complement kernel:

$$1 - \mu(A_1 \cup A_2) \quad \mu(\Omega) = 1$$

What about multiple classes?



One against all



Learn 3 classifiers separately:
Class k vs. rest
$(\mathbf{w}_k, b_k)_{k=1,2,3}$

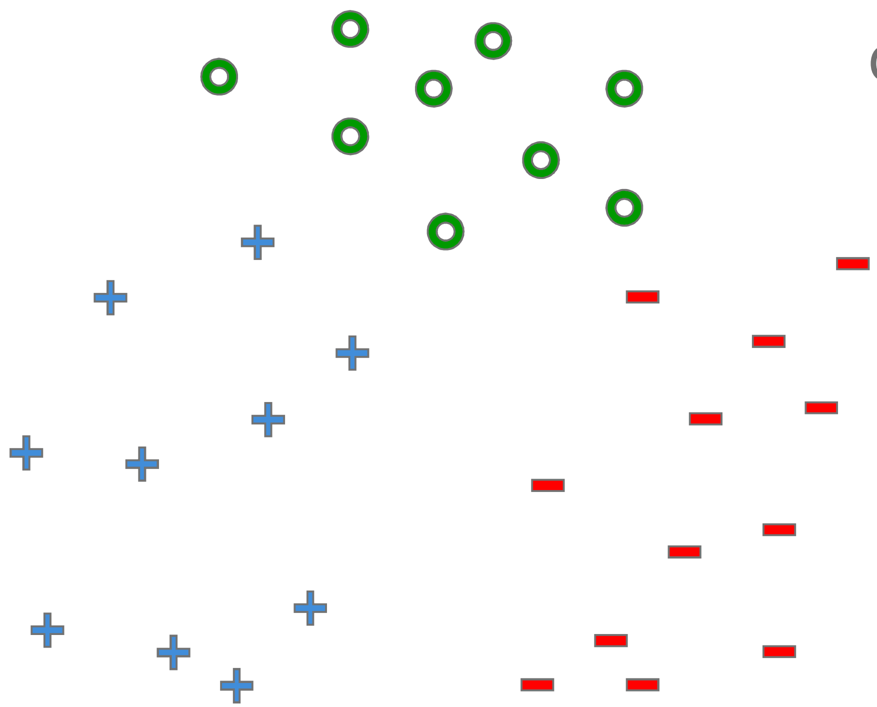
$$y = \arg\max_k \mathbf{w}_k.x + b_k$$

But $\mathbf{w}_k$s may not be based on the same scale.
Note: $(a\mathbf{w}).x + (ab)$ is also a solution

Learn 1 classifier: Multi-class SVM

Simultaneously learn 3 sets of weights

$$\mathbf{w}^{(y_j)}.\mathbf{x}_j + b^{(y_j)} \geq \mathbf{w}^{(y')}.\mathbf{x}_j + b^{(y')} + 1, \ \forall y' \neq y_j, \ \forall j$$



Margin - gap between correct class and nearest other class

y = arg max $\mathbf{w}^{(k)}.x + b^{(k)}$

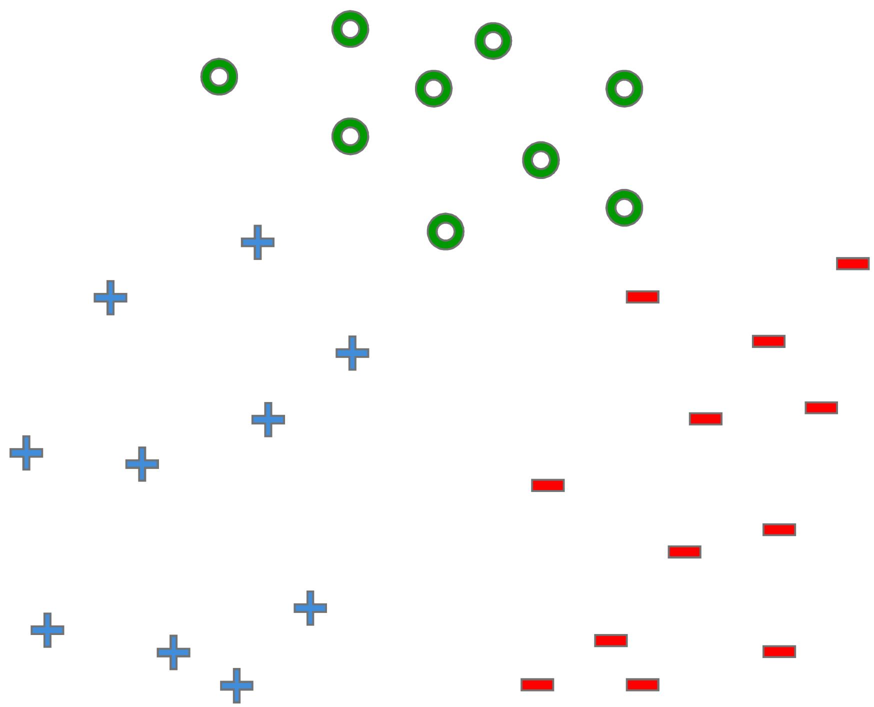
Learn 1 classifier: Multi-class SVM

Simultaneously learn 3 sets of weights

$$\text{minimize}_{\mathbf{w},b} \quad \sum_y \mathbf{w}^{(y)}.\mathbf{w}^{(y)} + C \sum_j \sum_{y \neq y_j} \xi_j^{(y)}$$

$$\mathbf{w}^{(y_j)}.\mathbf{x}_j + b^{(y_j)} \geq \mathbf{w}^{(y)}.\mathbf{x}_j + b^{(y)} + 1 - \xi_j^{(y)}, \ \forall y \neq y_j, \ \forall j$$

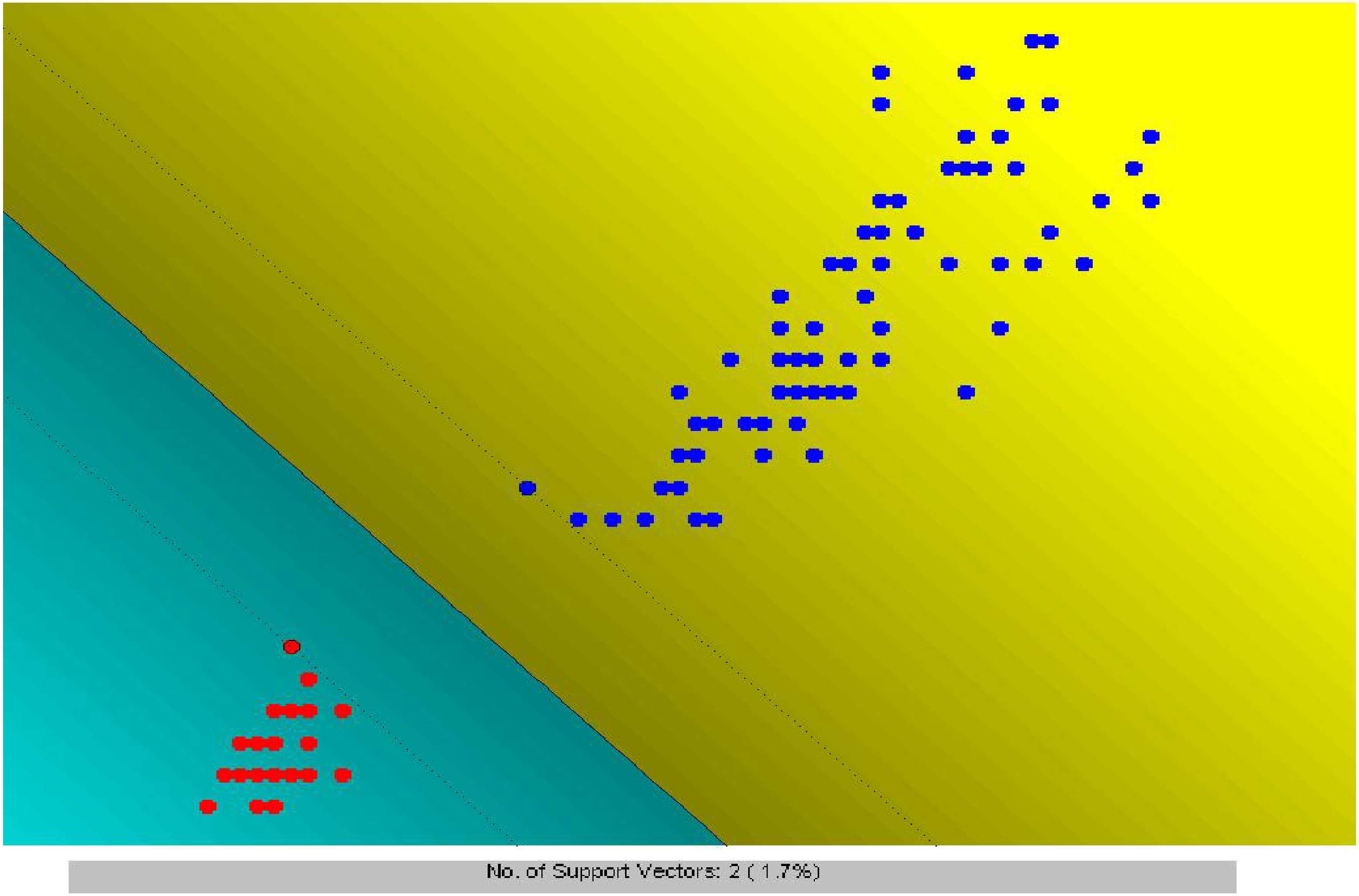
$$\xi_j^{(y)} \geq 0 \quad , \ \forall y \neq y_j, \ \forall j$$



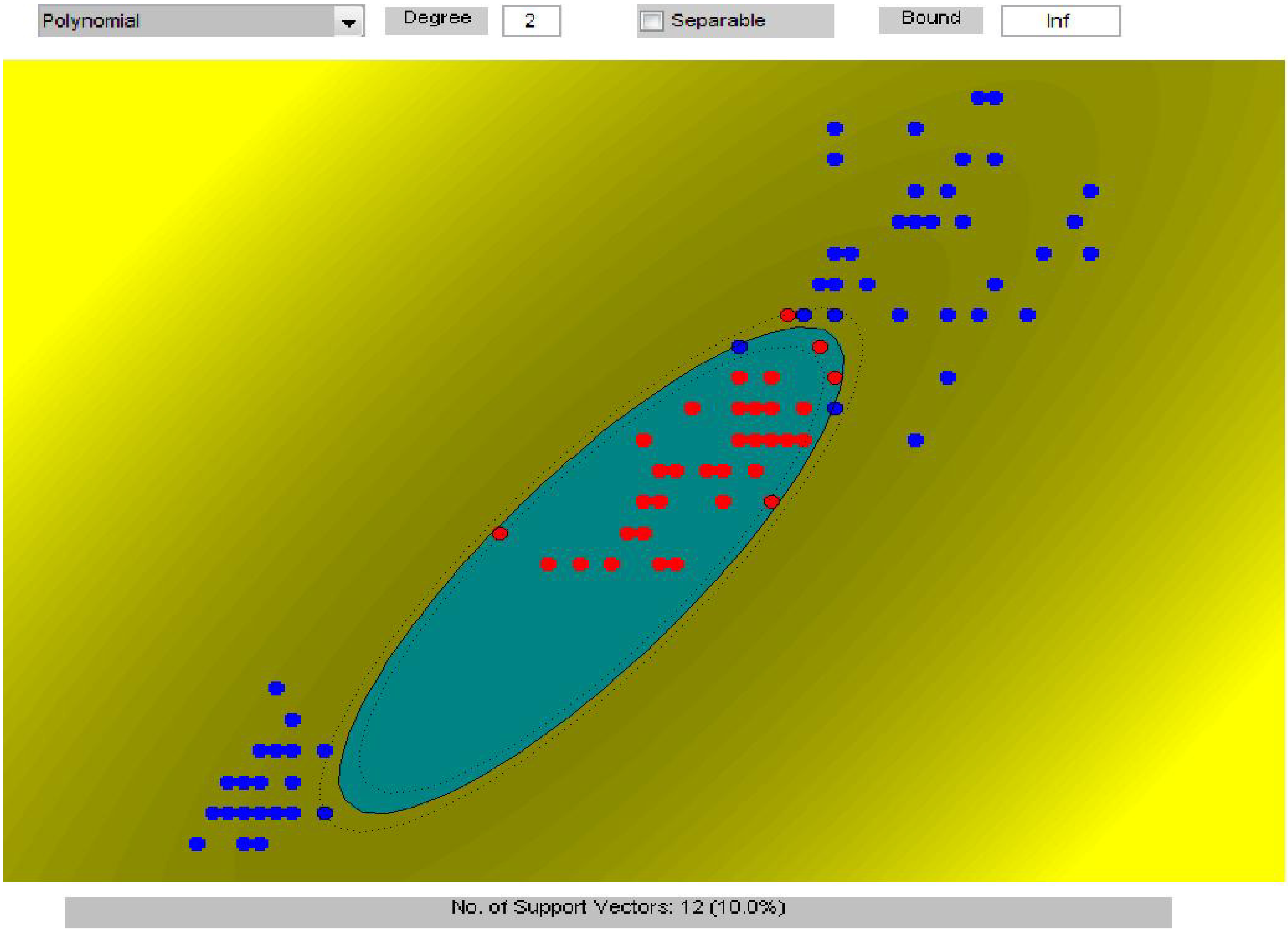
y = arg max $\mathbf{w}^{(k)}.x + b^{(k)}$

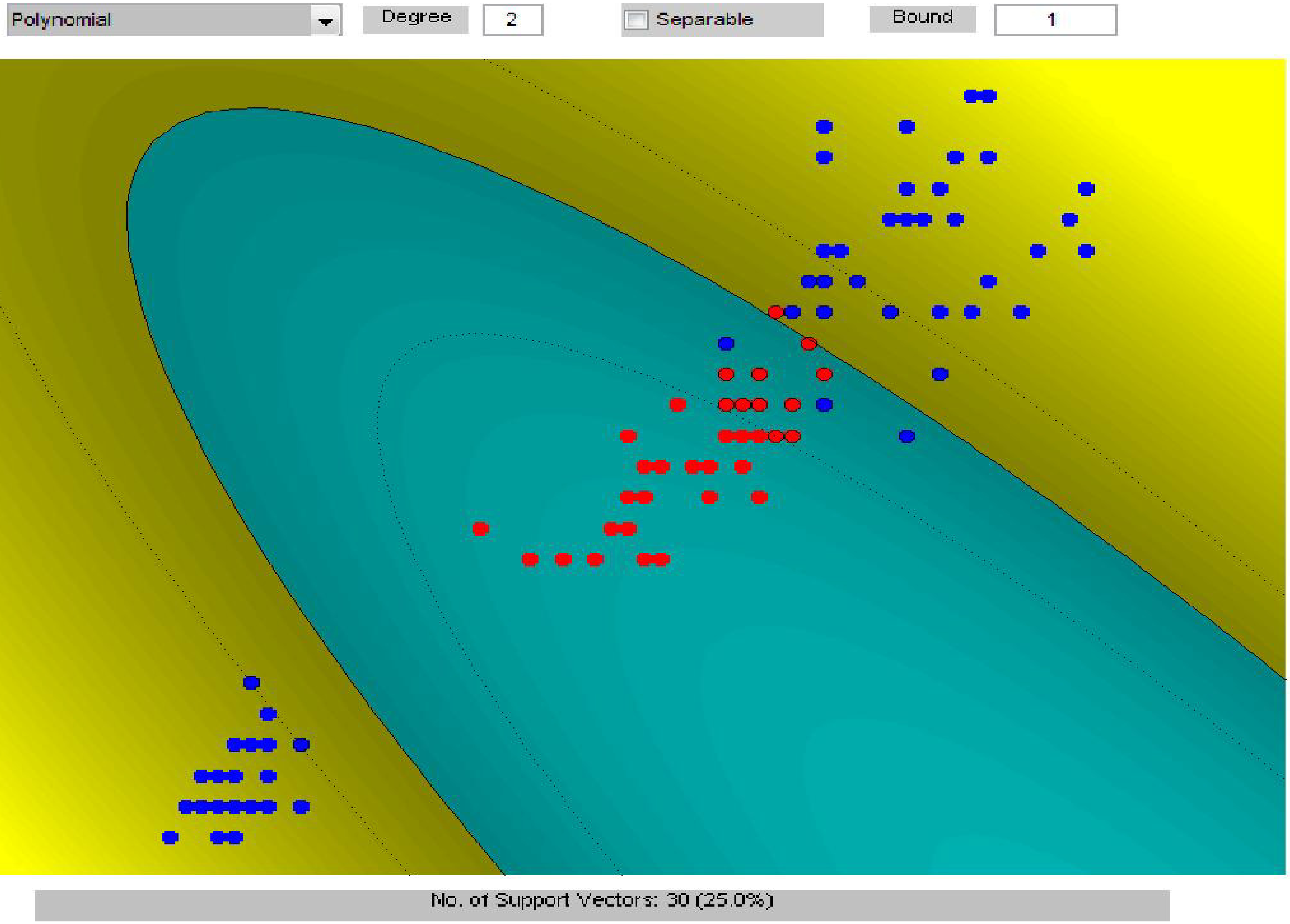
Joint optimization: $\mathbf{w}_k$s have the same scale.

Steve Gunn's svm toolbox
Results, Iris 2vs13, Linear kernel



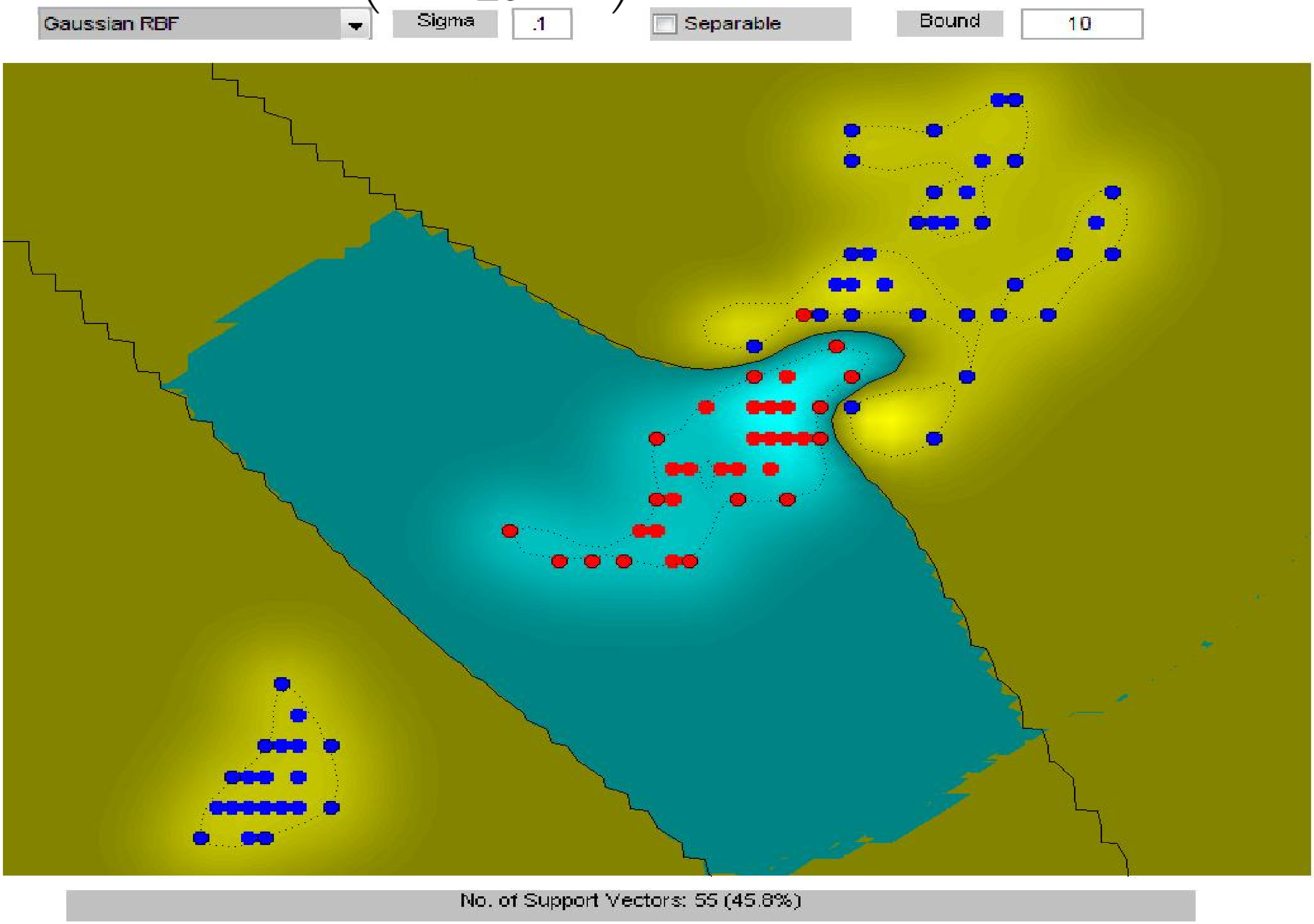
Results, Iris 1vs23, 2nd order kernel

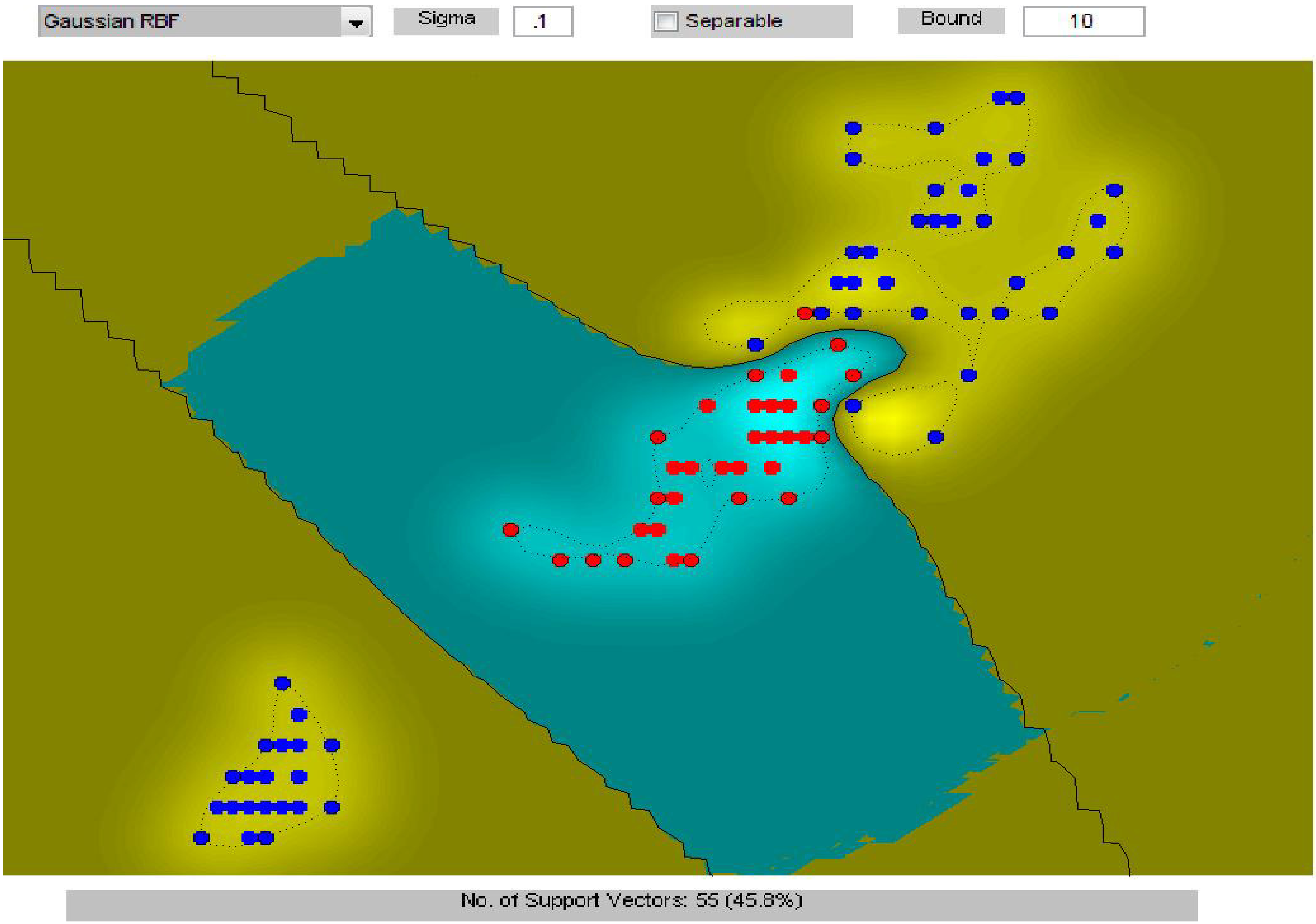


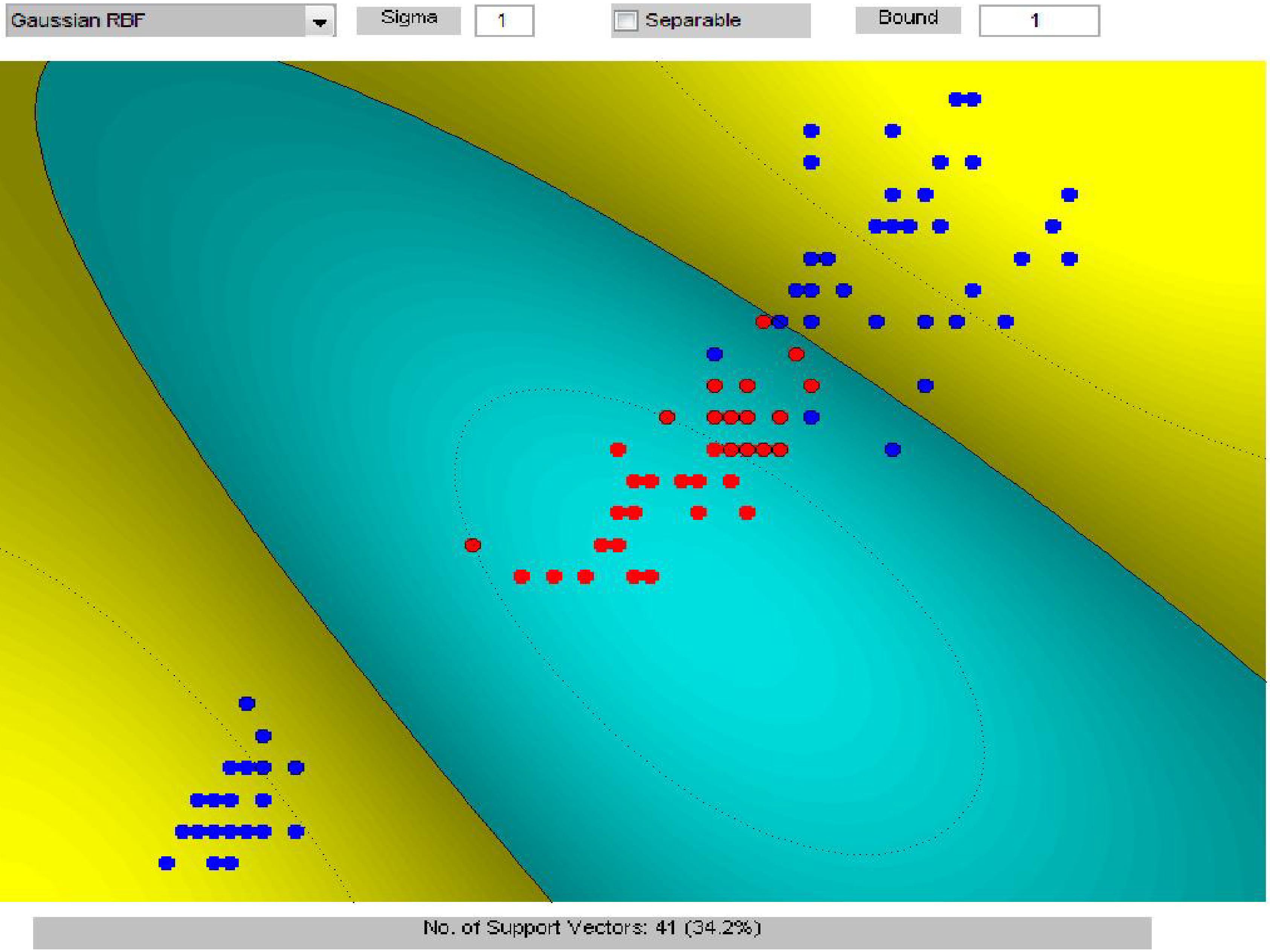

Polynomial
Degree
2
Separable
Bound
1
No. of Support Vectors: 30 (25.0%)

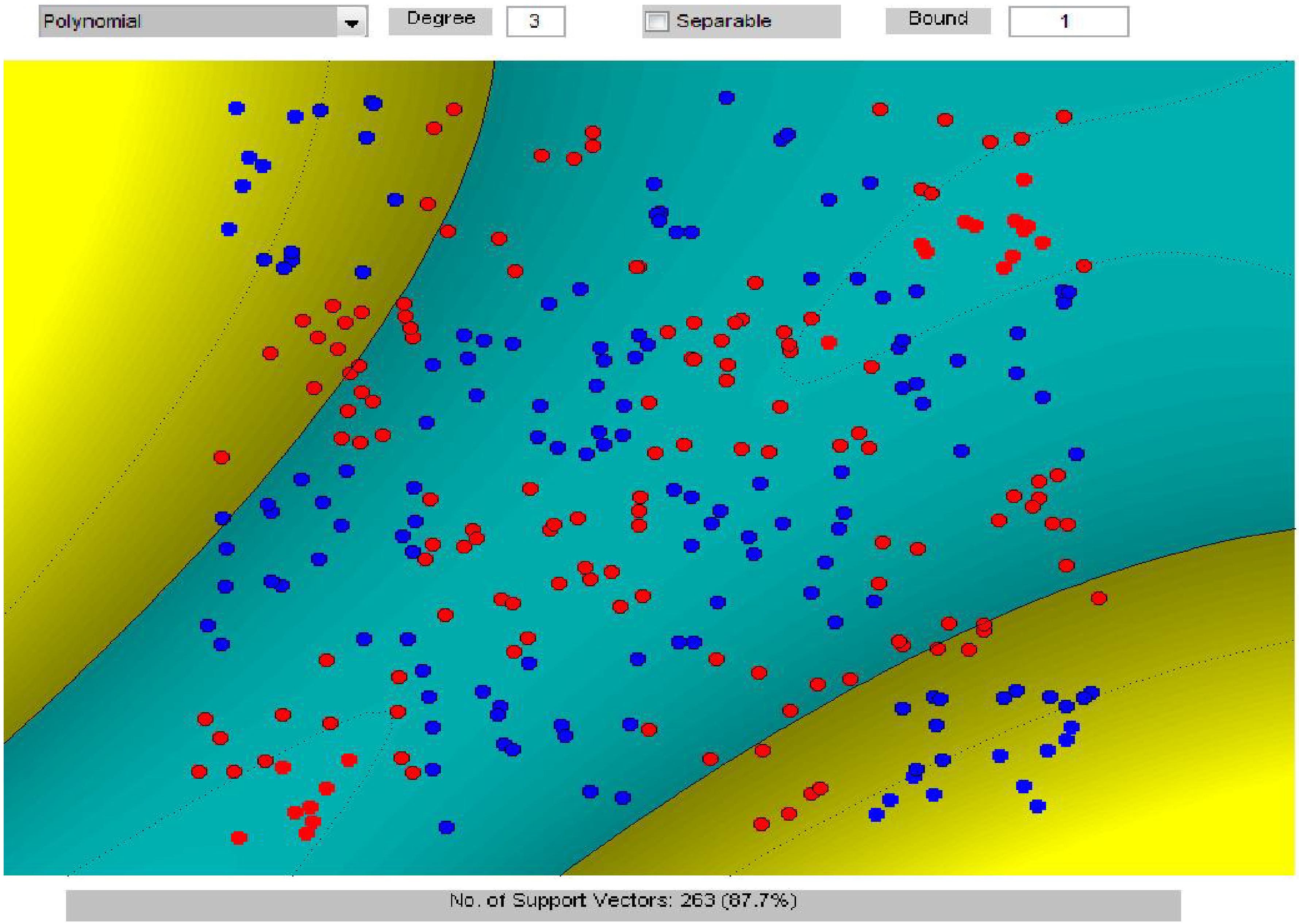
$$K(\mathbf{u}, \mathbf{v}) = \exp\left(-\frac{||\mathbf{u} - \mathbf{v}||^2}{2\sigma^2}\right)$$

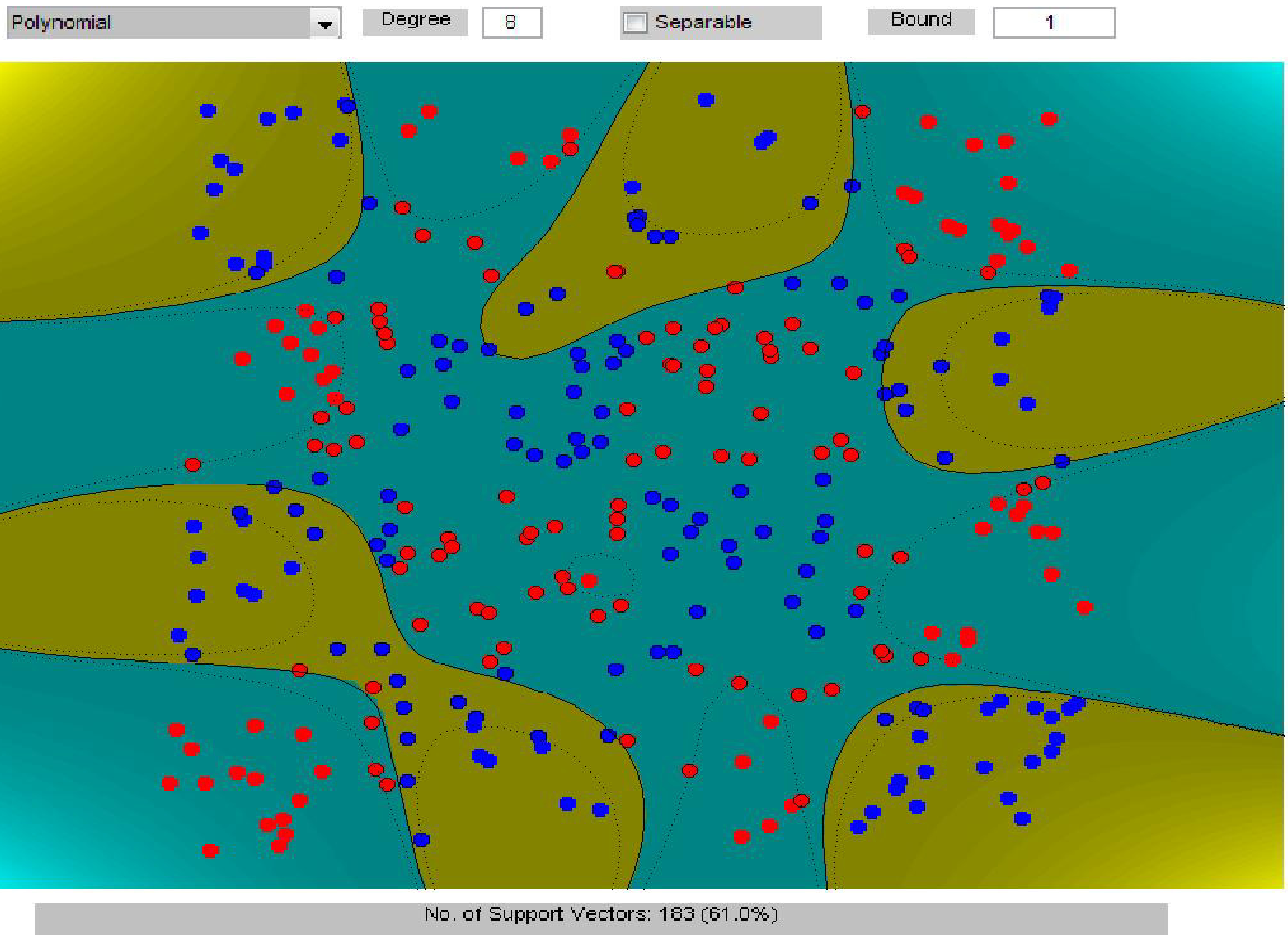
$\sigma \to 0 \Rightarrow$ MORE SUPPORT VECTORS

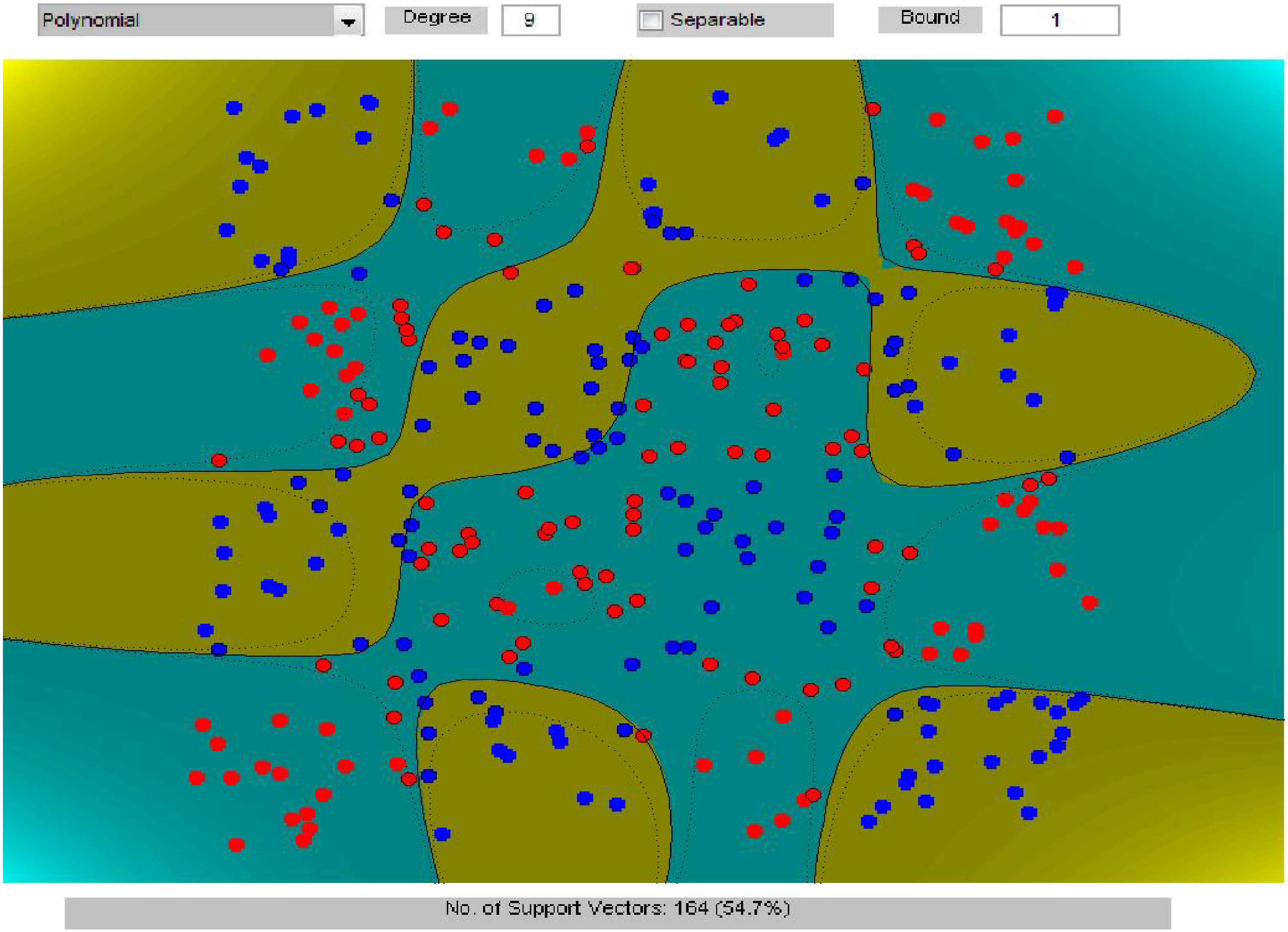


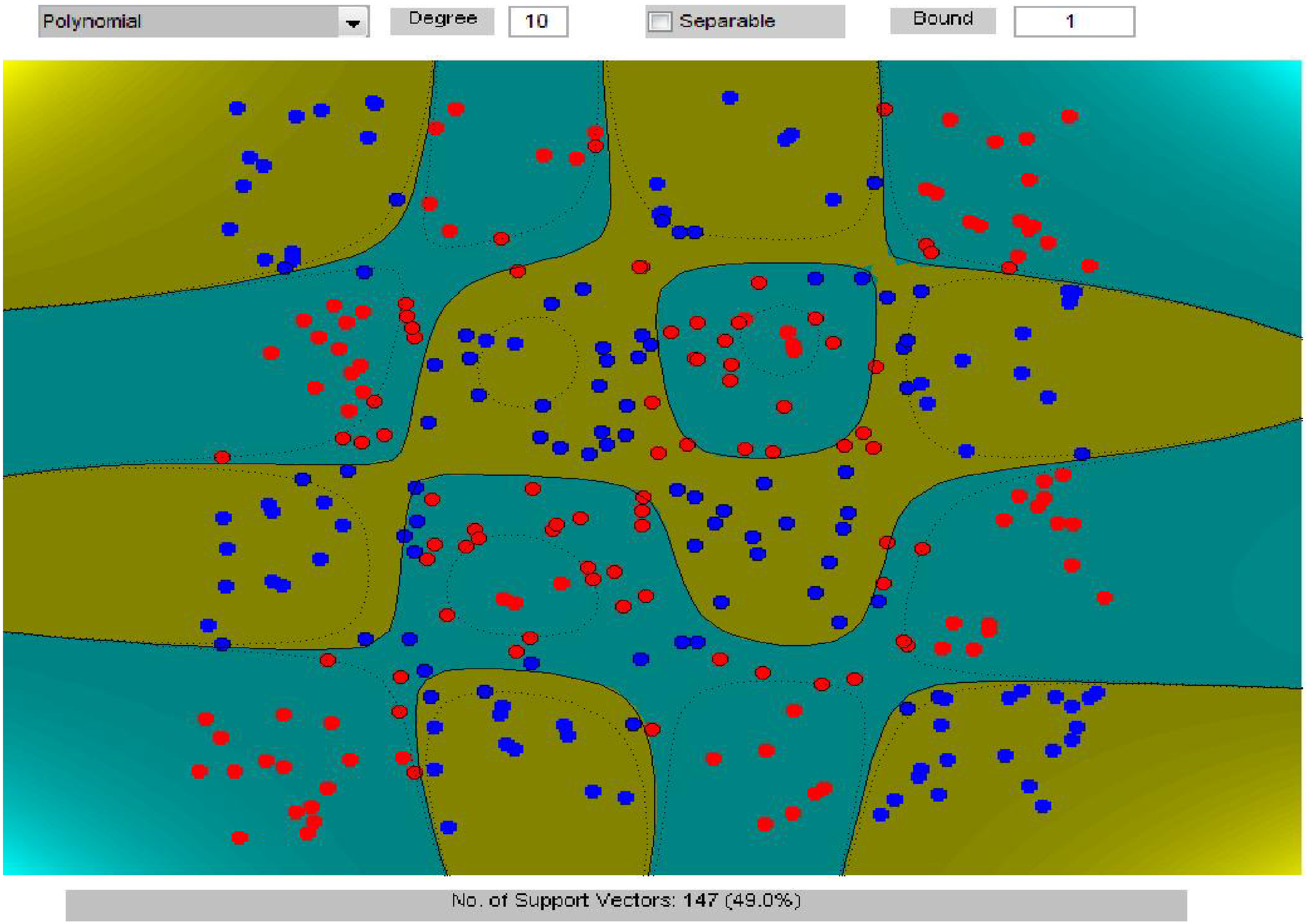

Gaussian RBF
Sigma
.1
Separable
Bound
10
No. of Support Vectors: 55 (45.8%)

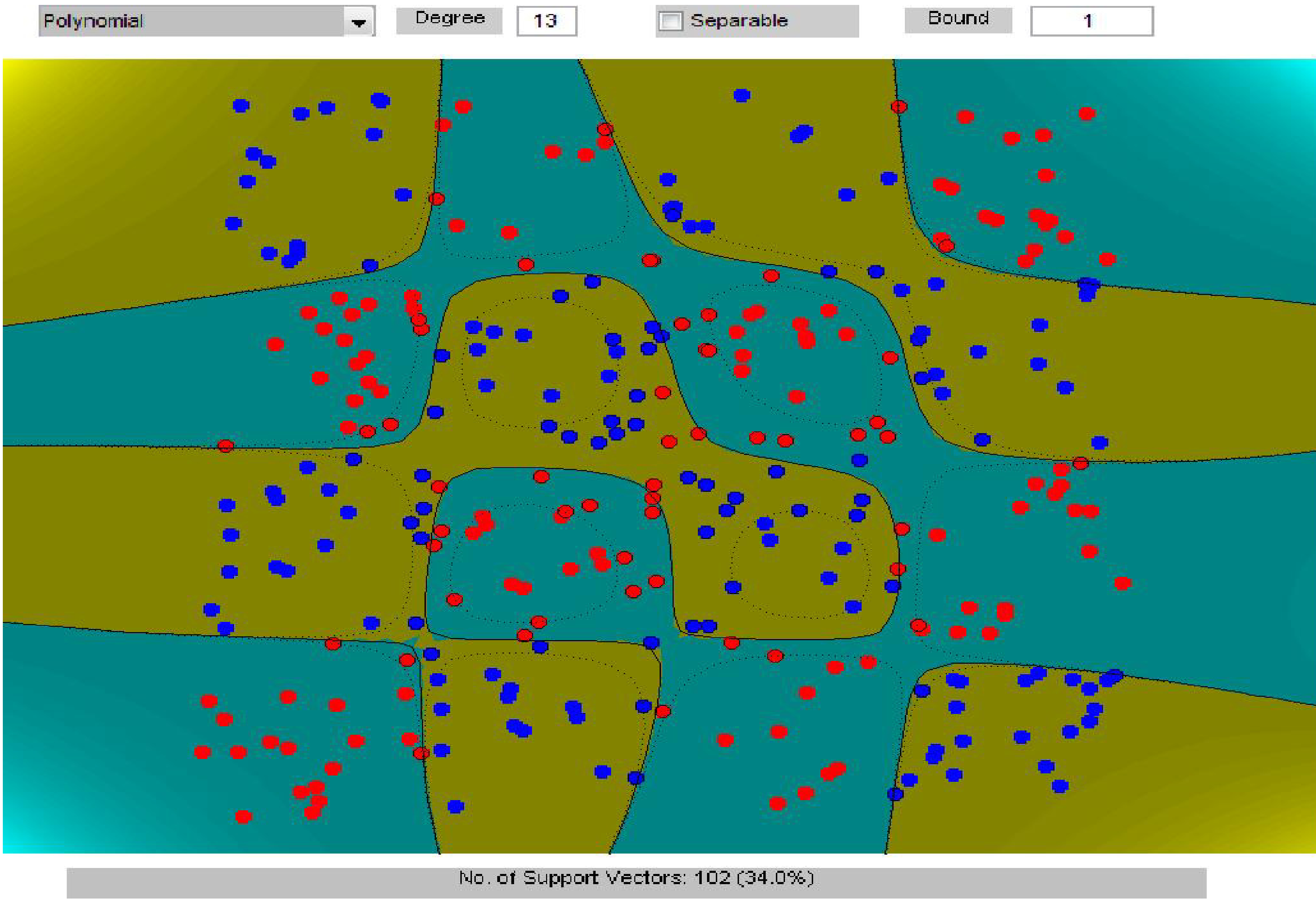

Gaussian RBF
Sigma
1
Separable
Bound
1
No. of Support Vectors: 41 (34.2%)


Polynomial
Degree
3
Separable
Bound
1
No. of Support Vectors: 263 (87.7%)

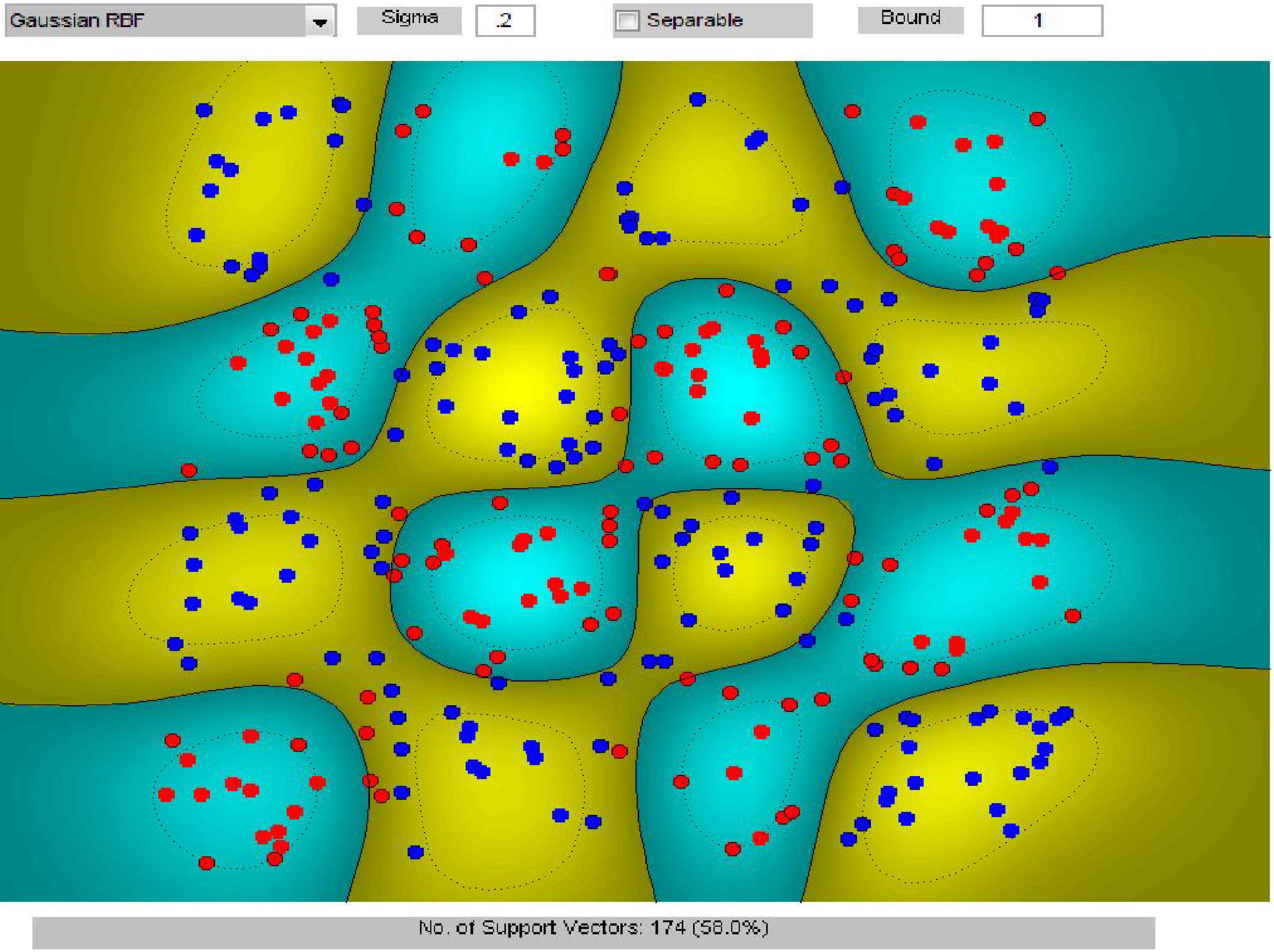

Polynomial
Degree
8
Separable
Bound
1
No. of Support Vectors: 183 (61.0%)


Polynomial
Degree
9
Separable
Bound
1
No. of Support Vectors: 164 (54.7%)

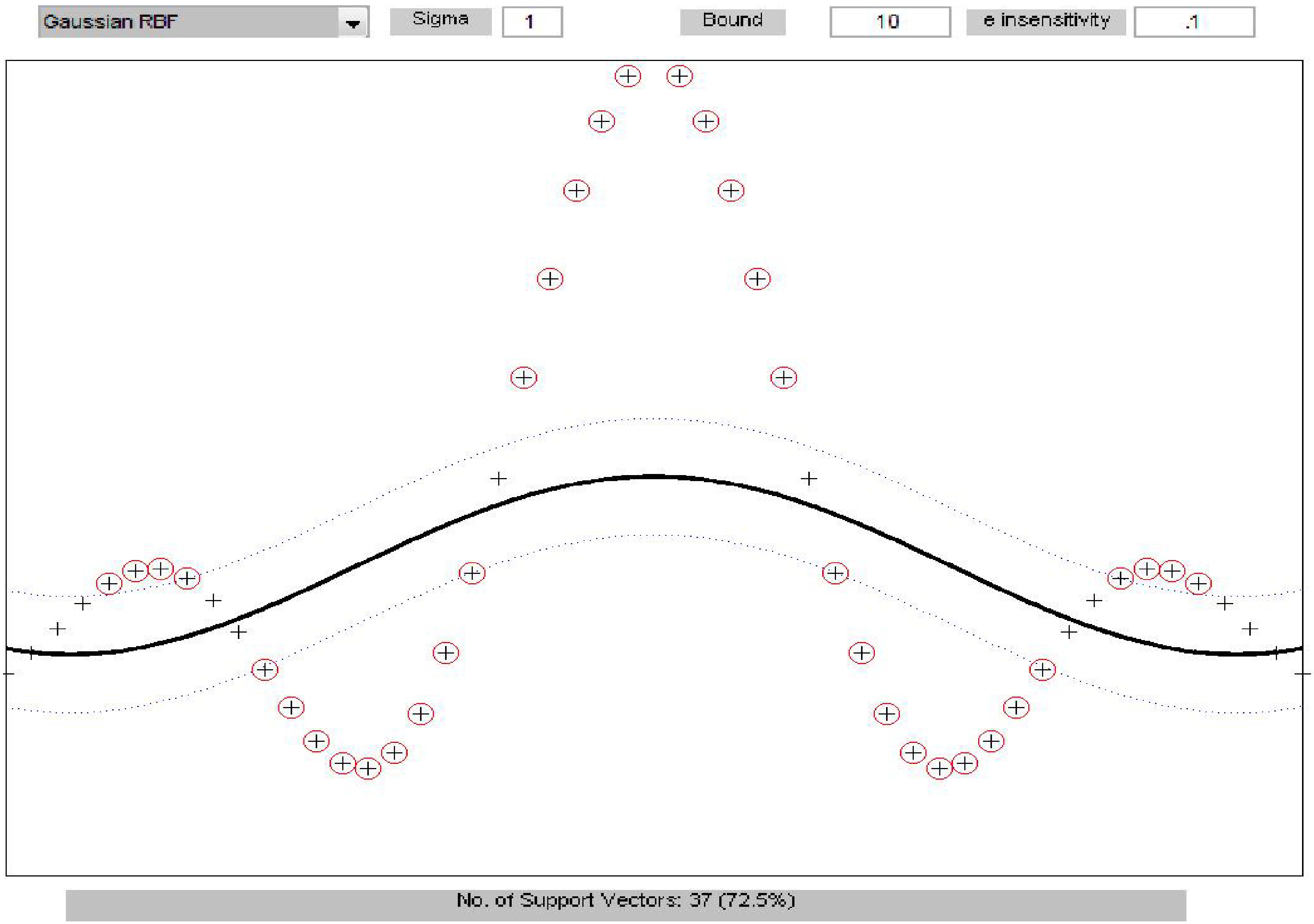

Polynomial
Degree
10
Separable
Bound
1
No. of Support Vectors: 147 (49.0%)


Polynomial
Degree
13
Separable
Bound
1
No. of Support Vectors: 102 (34.0%)

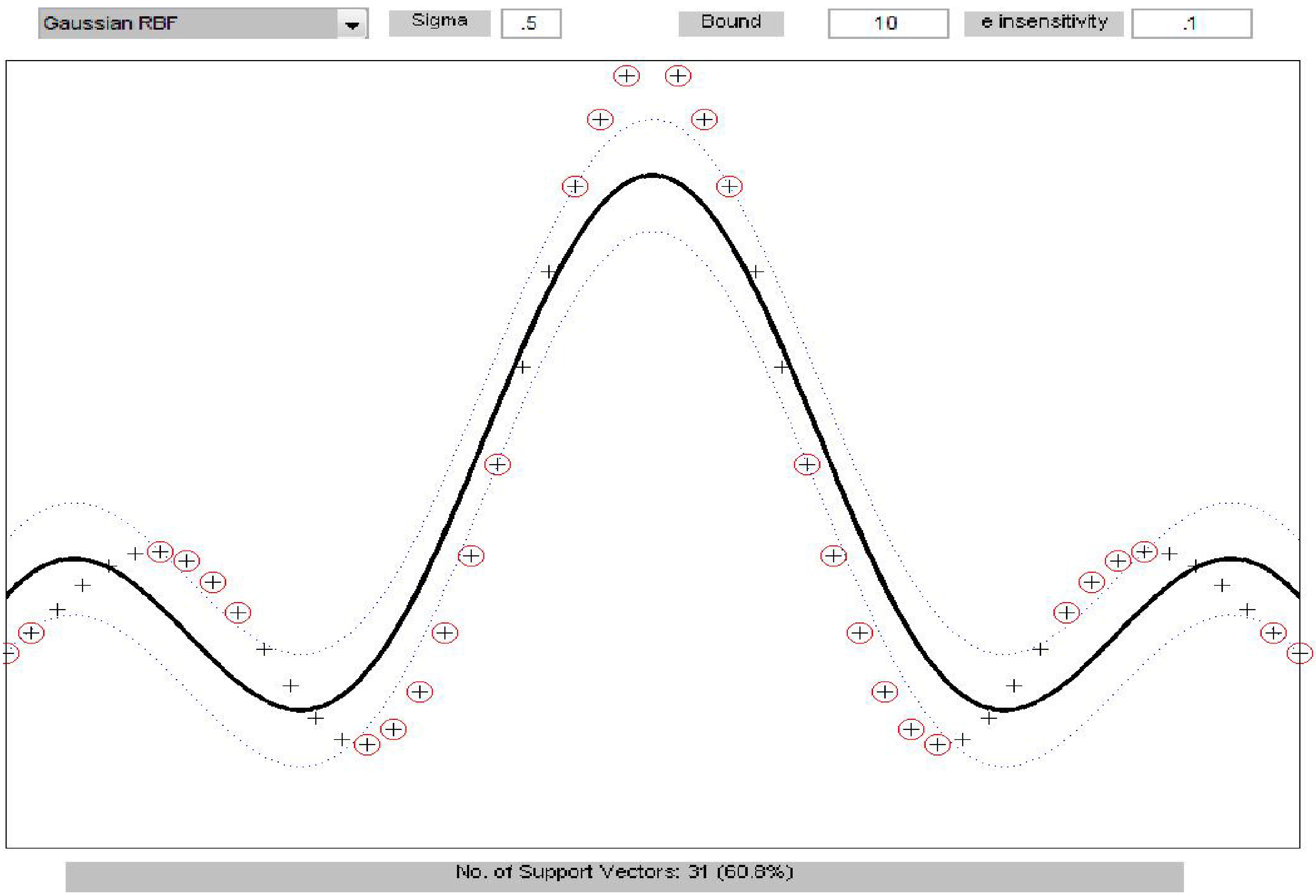
Results, Chessboard, RBF kernel



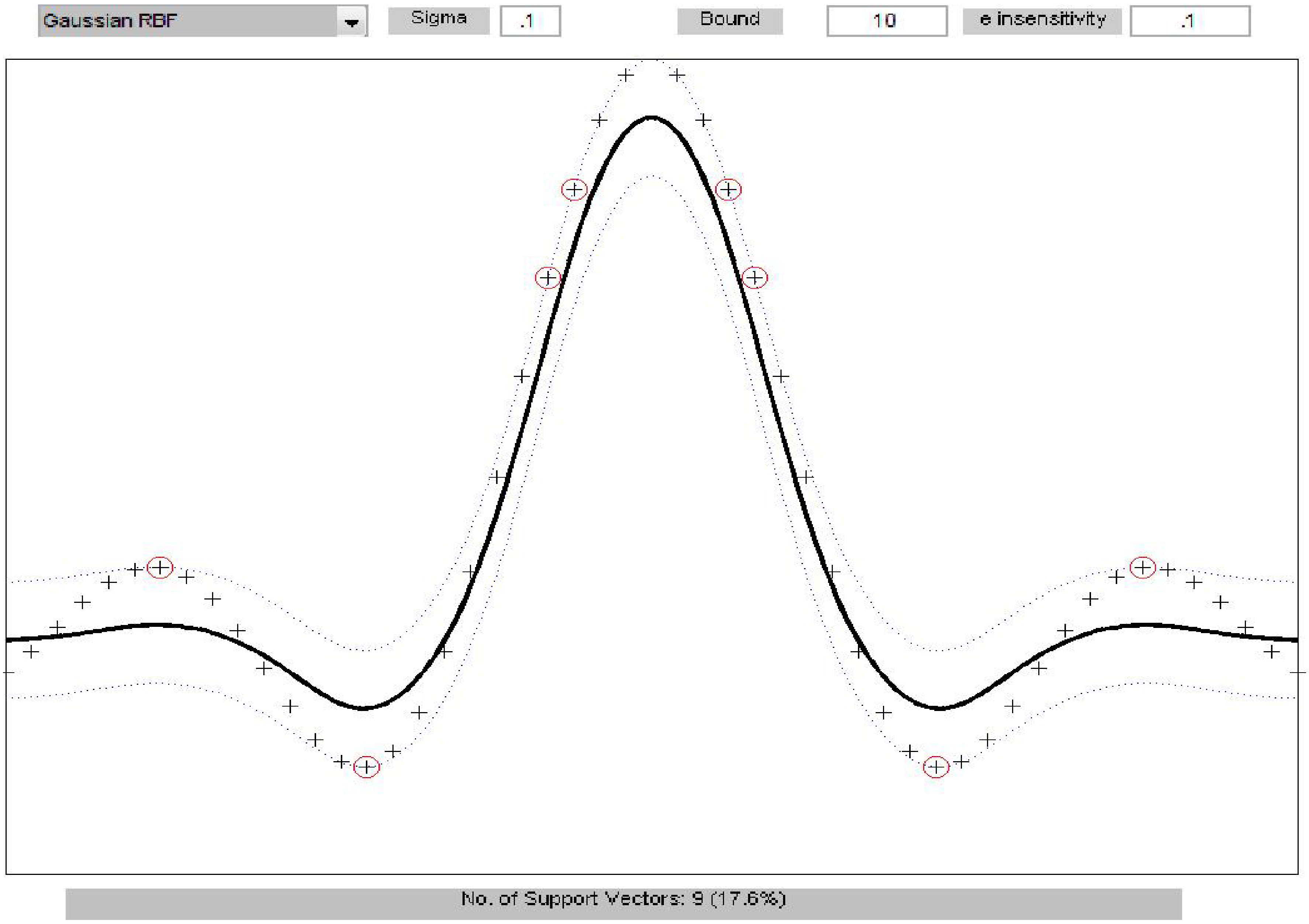
Sinc=sin(π x)/ (π x), RBF kernel



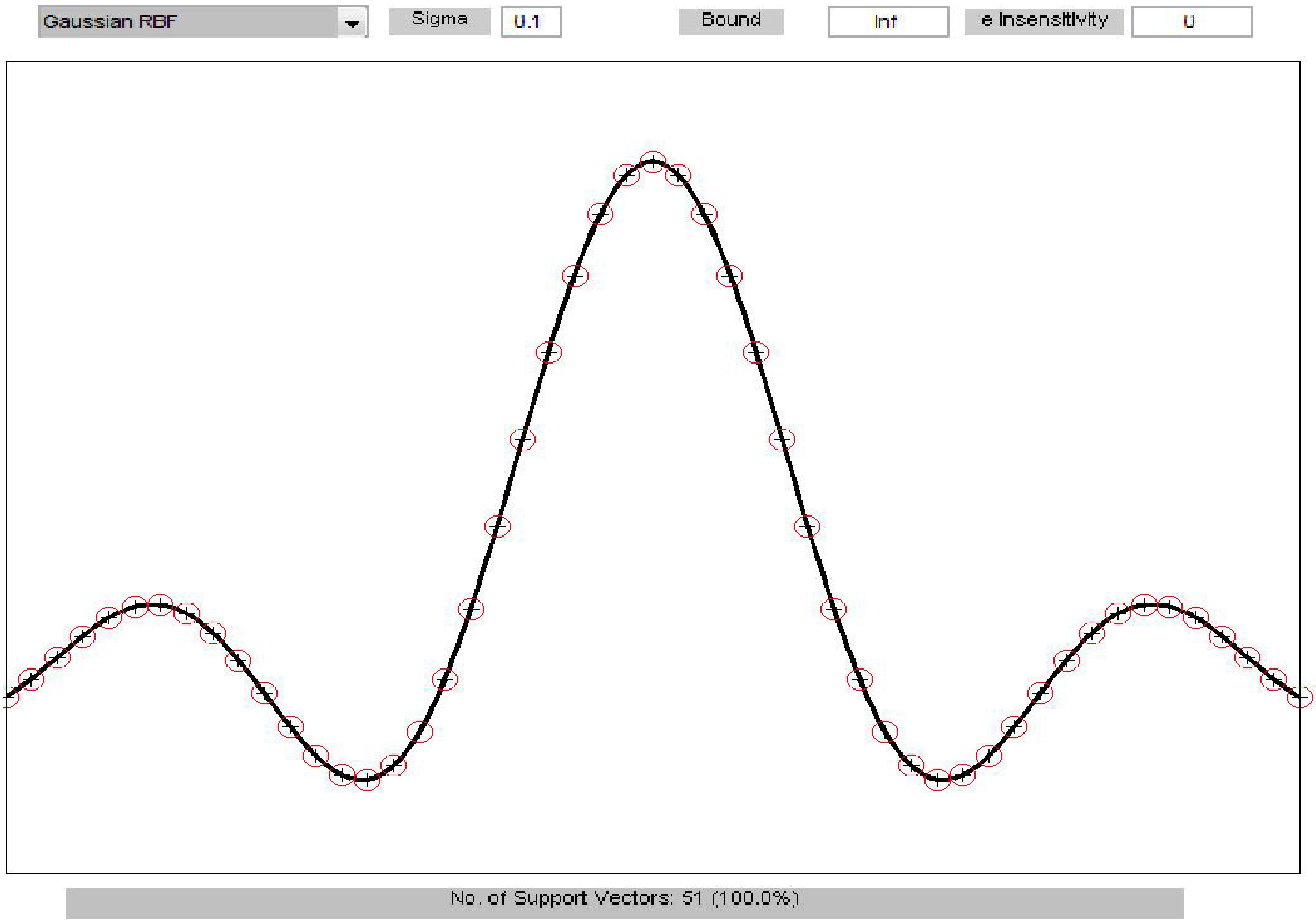
Sinc=sin(π x)/ (π x), RBF kernel



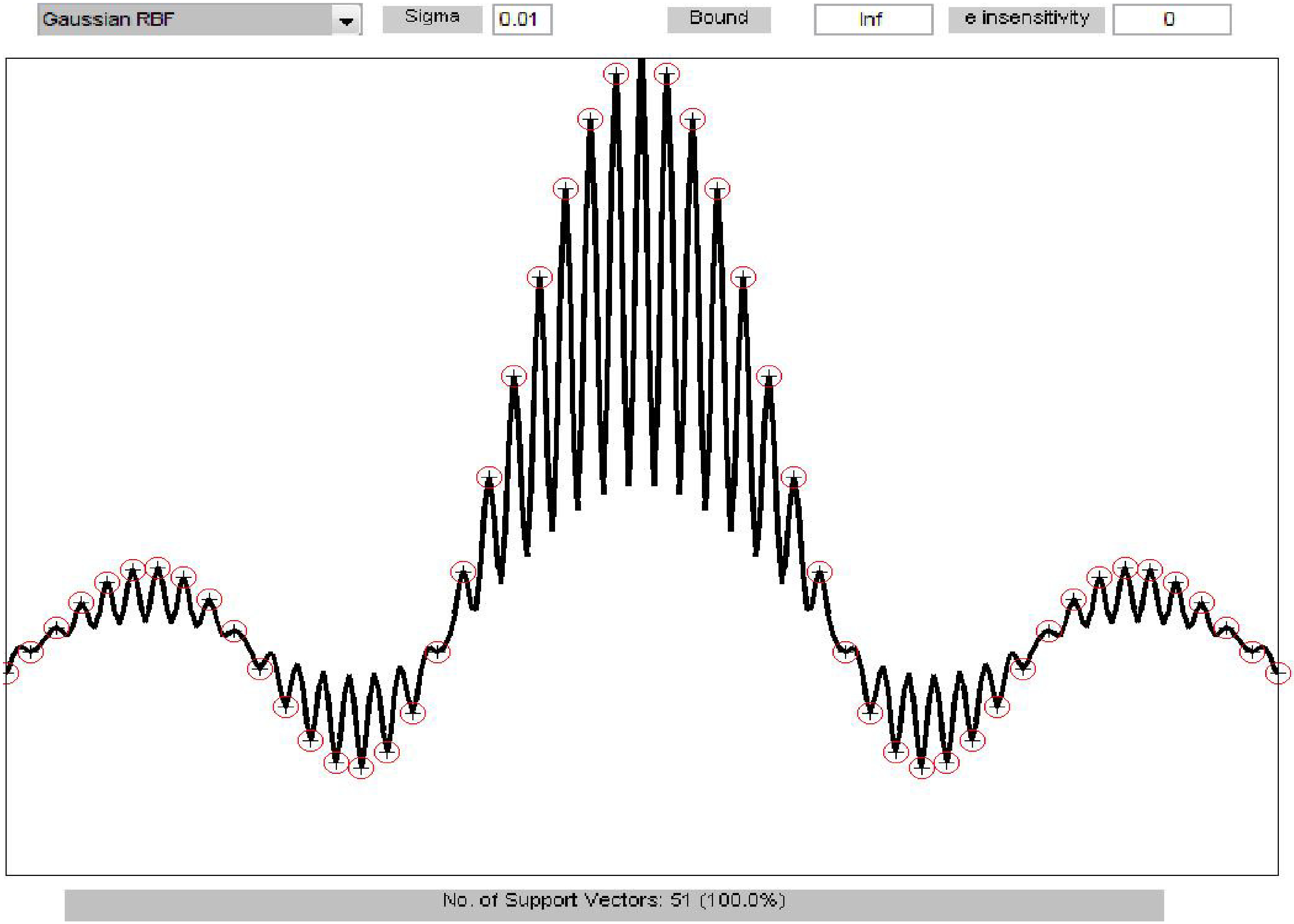
Sinc=sin(π x)/ (π x), RBF kernel



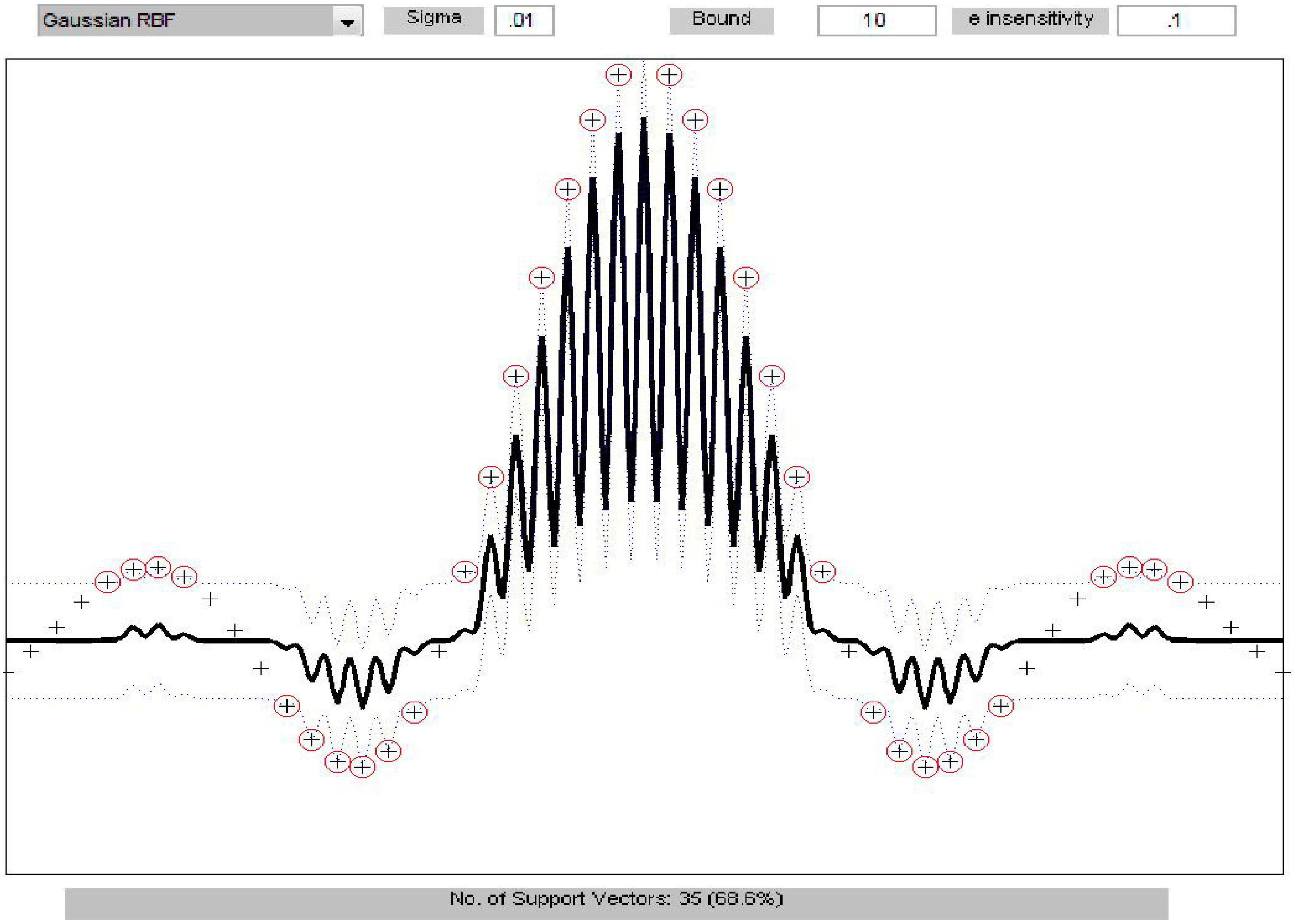
Sinc=sin(π x)/ (π x), RBF kernel



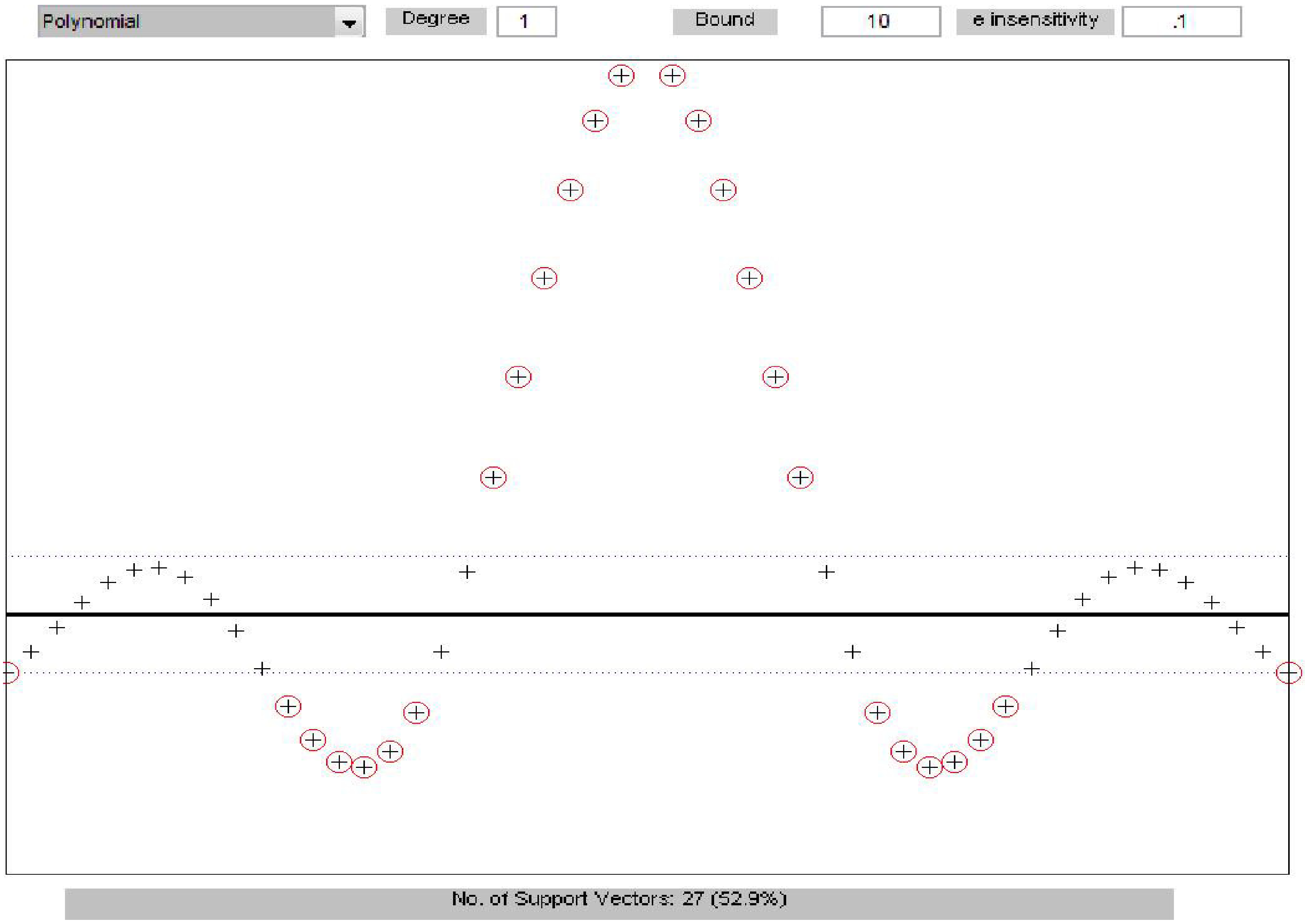
Sinc=sin(π x)/ (π x), RBF kernel



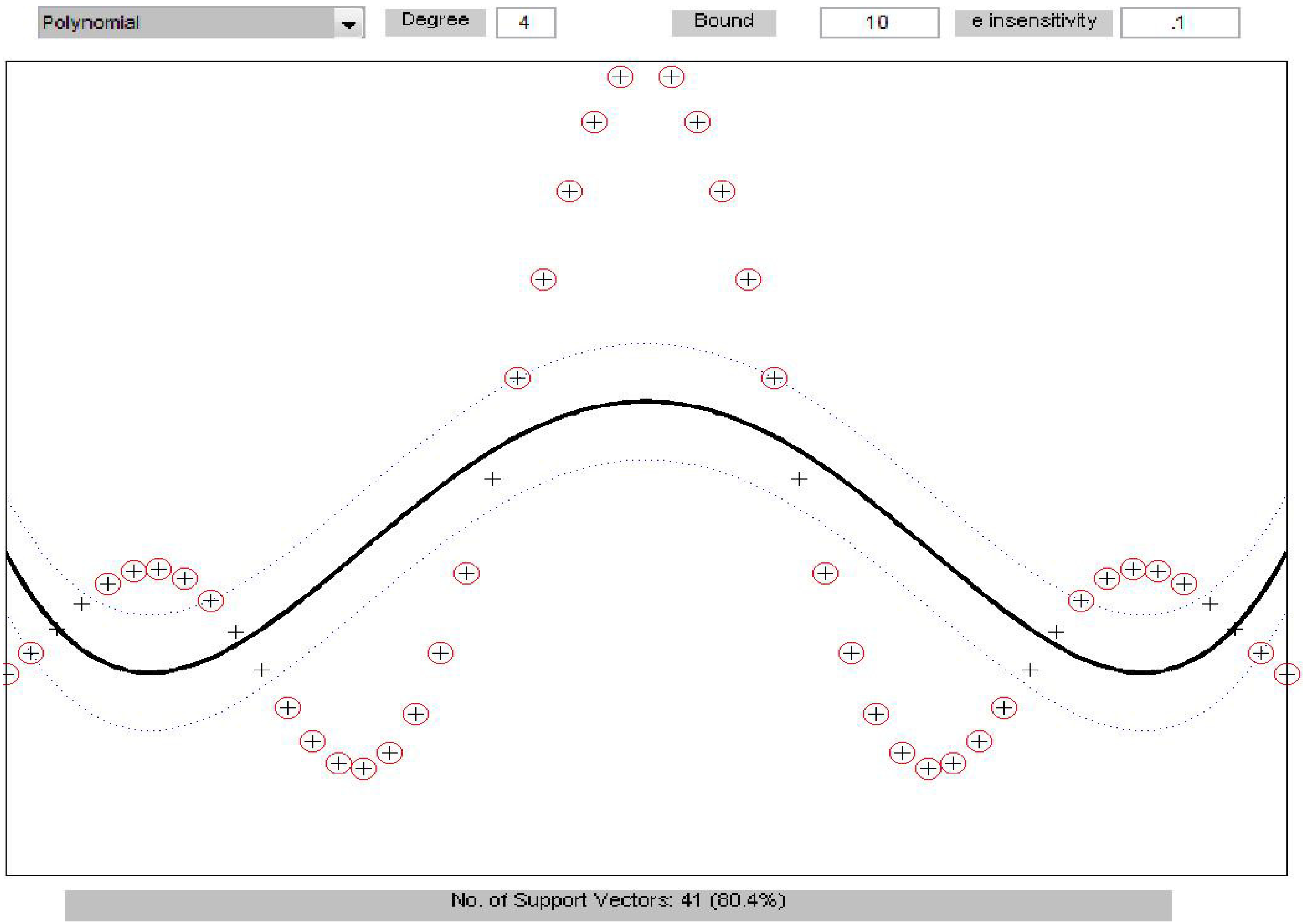
Sinc=sin(π x)/ (π x), RBF kernel



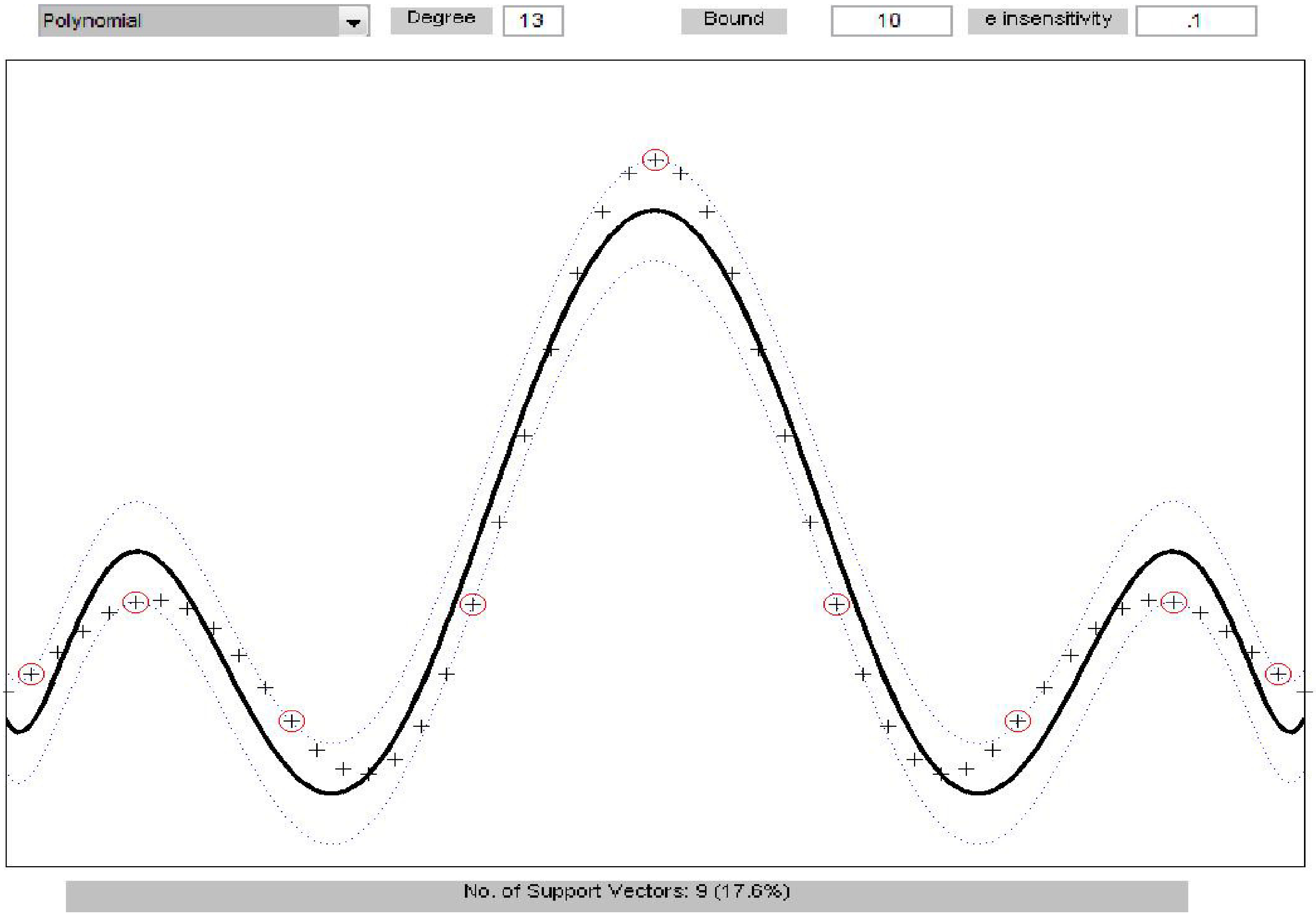
Sinc=sin(π x)/ (π x), poly kernel



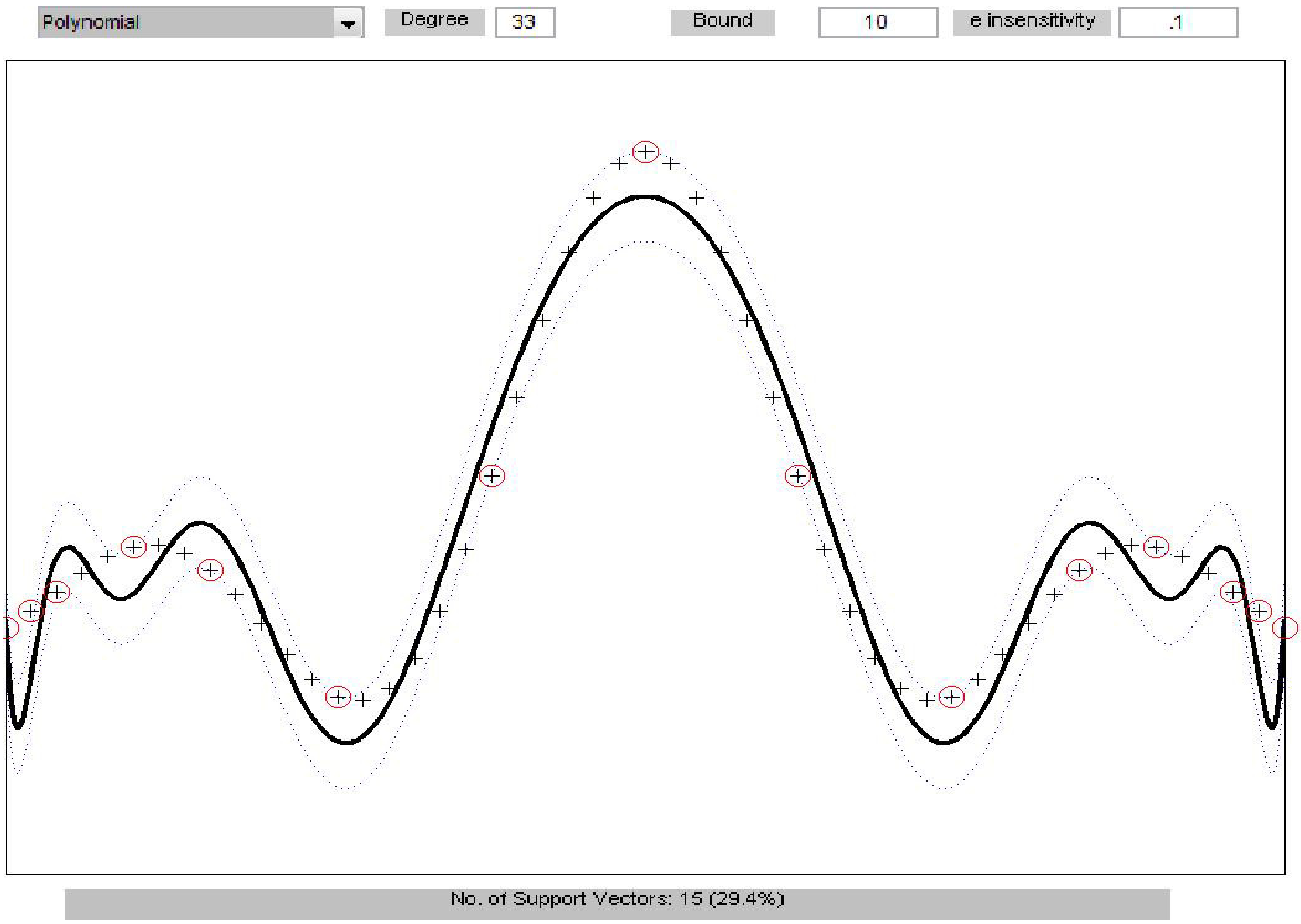
Sinc=sin(π x)/ (π x), poly kernel



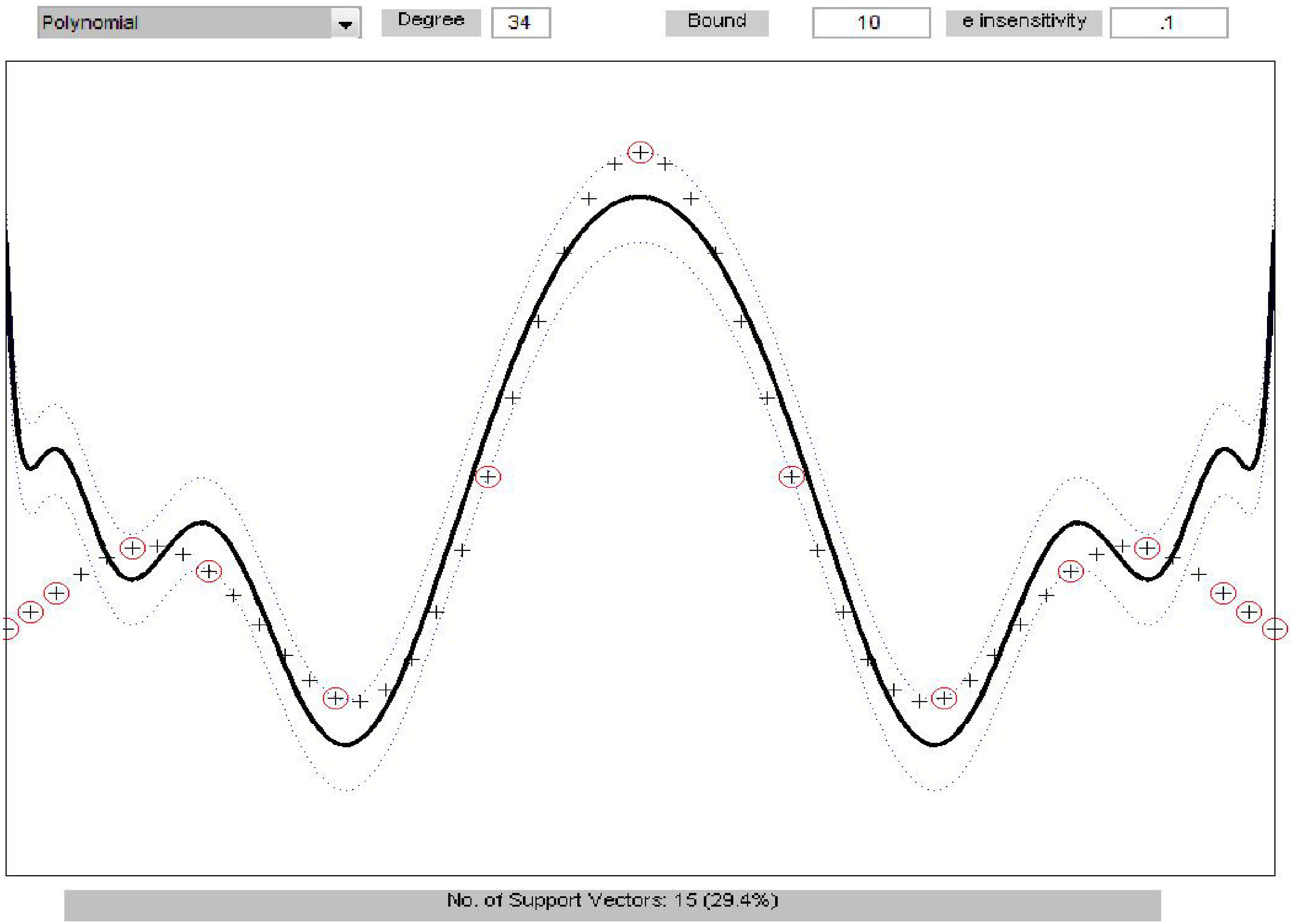
Sinc=sin(π x)/ (π x), poly kernel



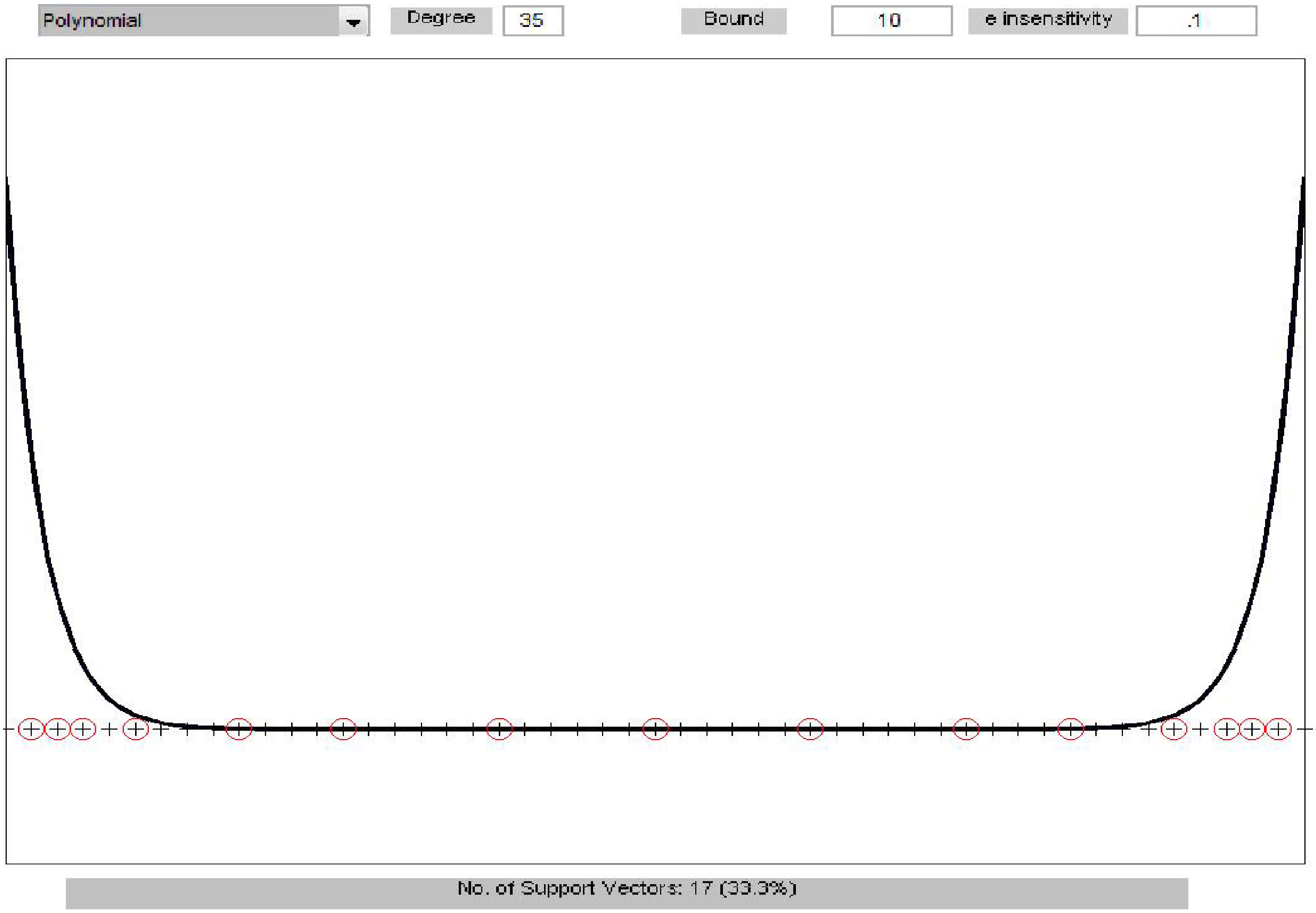
Sinc=sin(π x)/ (π x), poly kernel



Sinc=sin(π x)/ (π x), poly kernel



Sinc=sin(π x)/ (π x), poly kernel



What you need to know…

- Dual SVM formulation
  - How it's derived
- Common kernels
- Differences between SVMs and logistic regression

Thanks for your attention ☺