

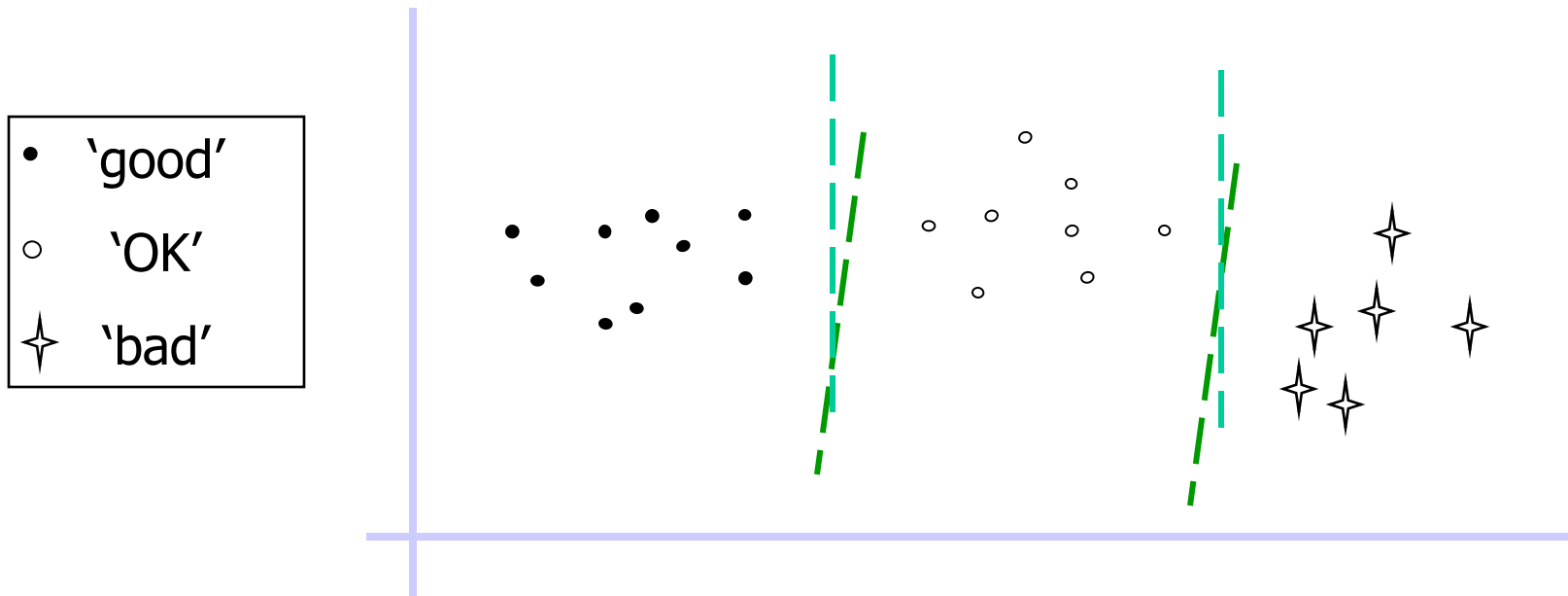
MACHINE LEARNING

Support Vector Machine



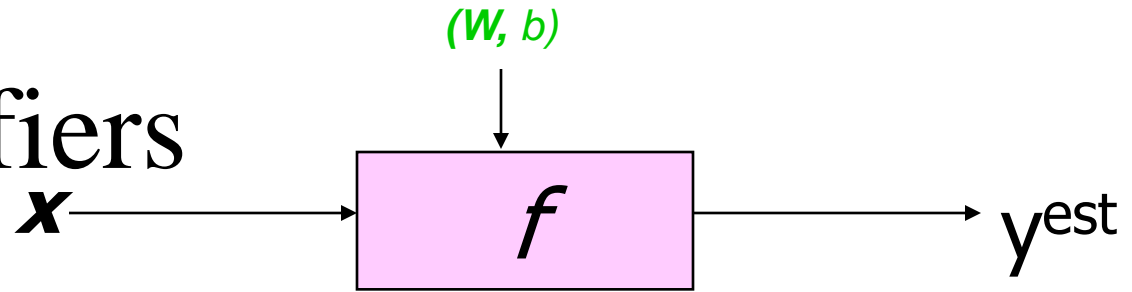
ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Optimal Linear Classification



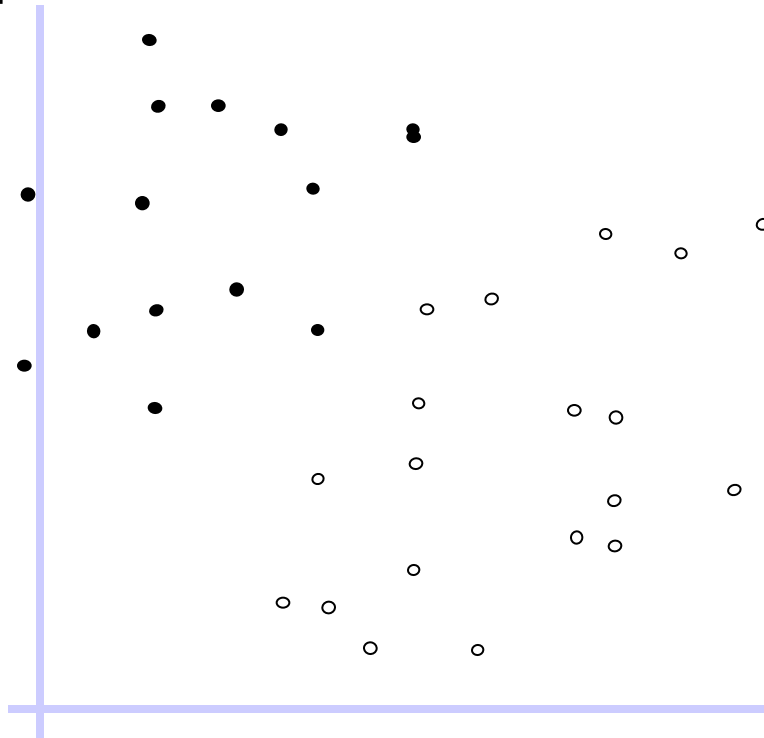
- Which choice is better?
- How could we formulate this problem?

Linear Classifiers



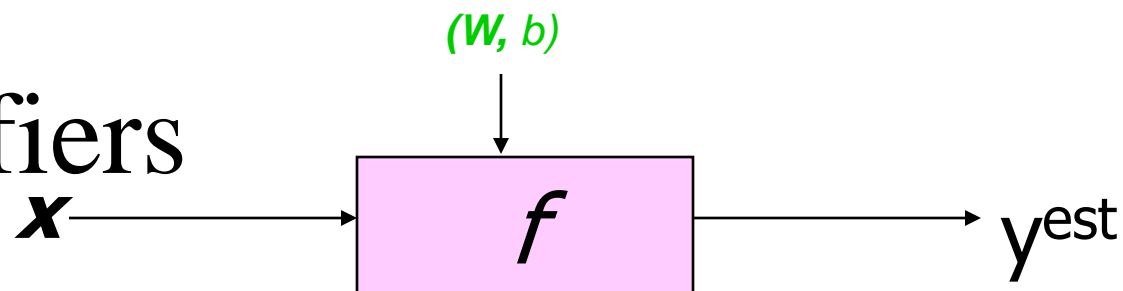
$$f(\mathbf{x}, \mathbf{w}, b) = \text{sign}(\mathbf{w} \cdot \mathbf{x} + b)$$

- denotes +1
- denotes -1



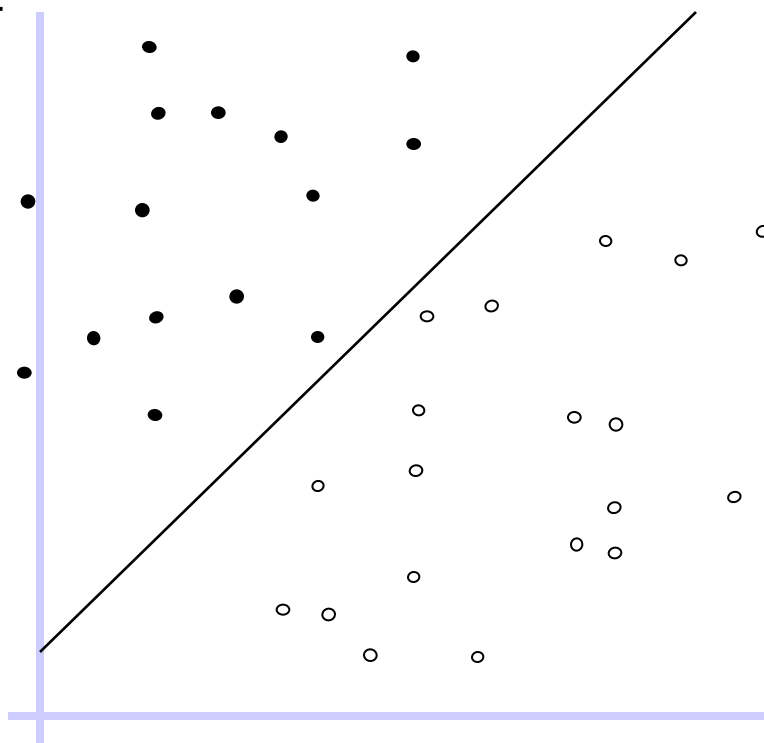
How would you classify this data?

Linear Classifiers



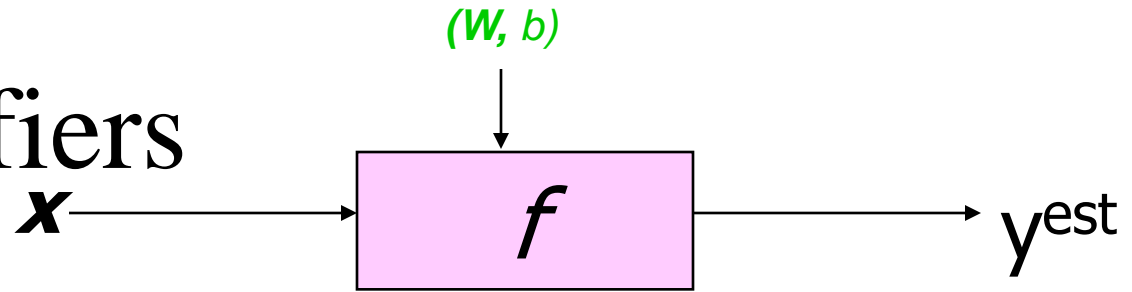
$$f(x, w, b) = \text{sign}(w \cdot x + b)$$

- denotes +1
- denotes -1



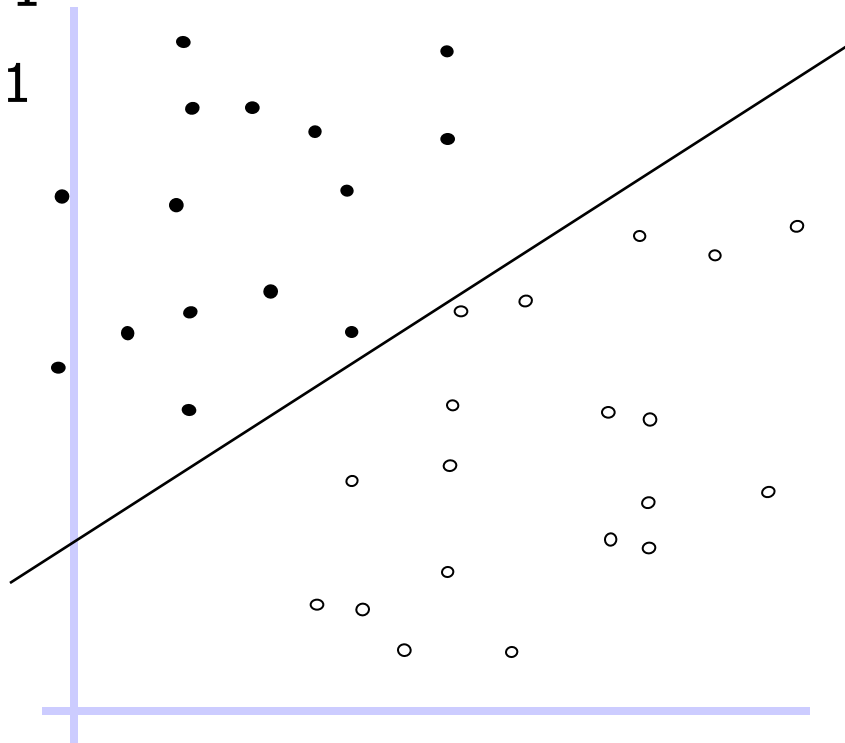
How would you classify this data?

Linear Classifiers



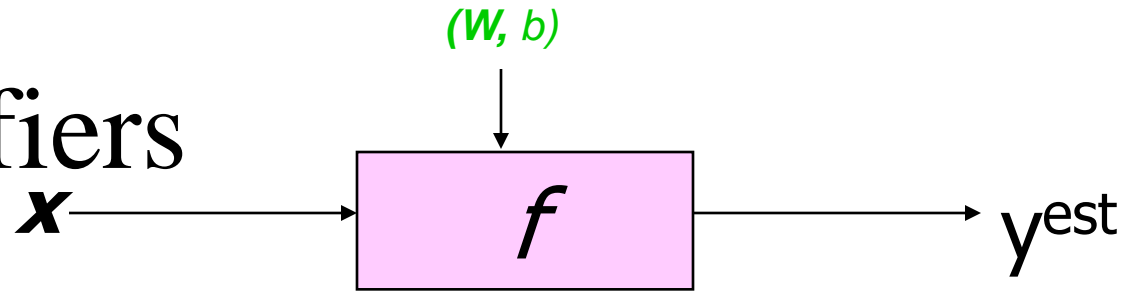
$$f(x, w, b) = \text{sign}(w \cdot x + b)$$

- denotes +1
- denotes -1



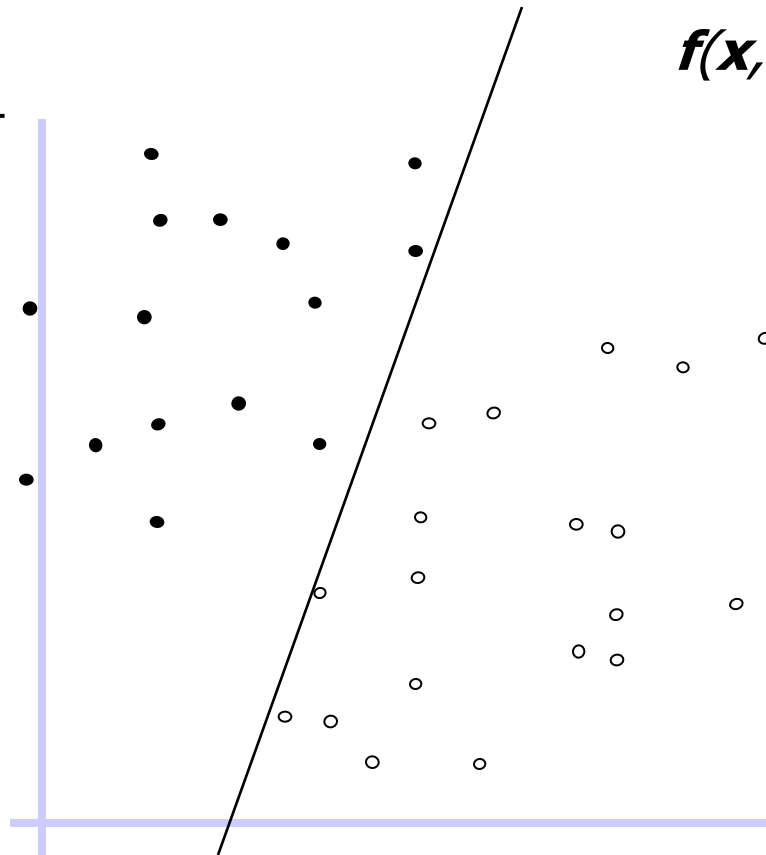
How would you classify this data?

Linear Classifiers



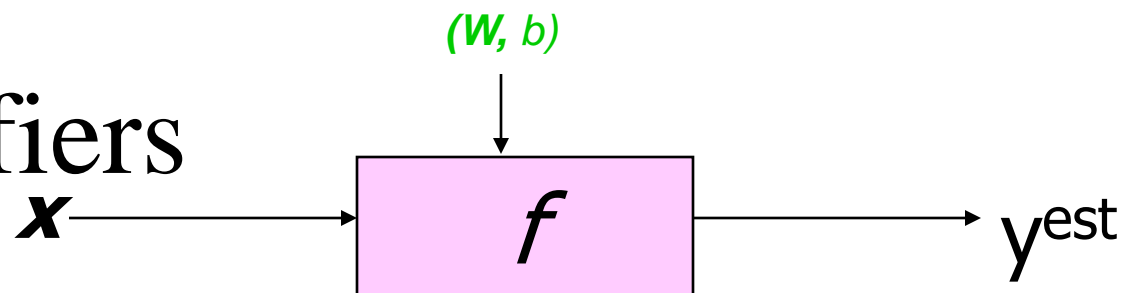
$$f(\mathbf{x}, \mathbf{w}, b) = \text{sign}(\mathbf{w} \cdot \mathbf{x} + b)$$

- denotes +1
- denotes -1

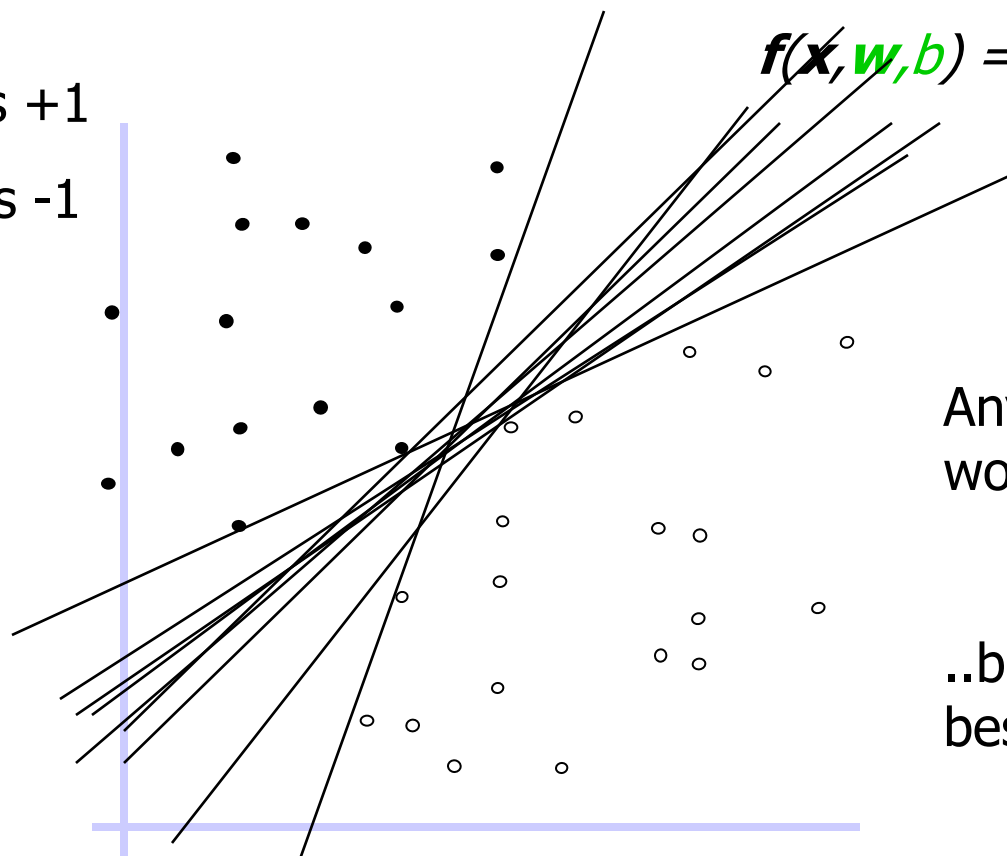


How would you classify this data?

Linear Classifiers



- denotes +1
- denotes -1

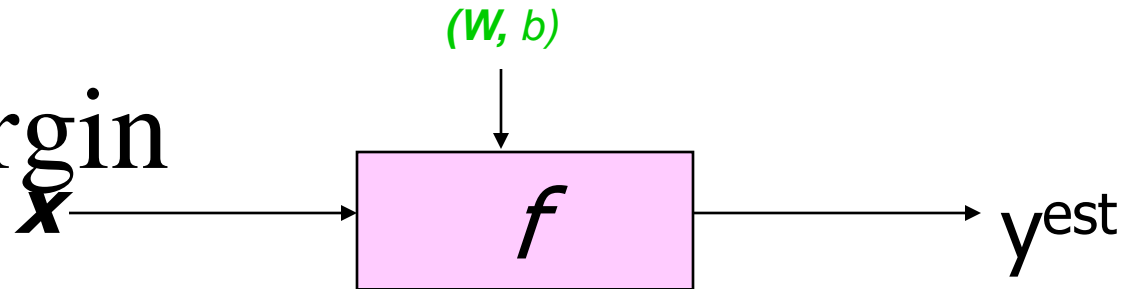


$$f(x, w, b) = \text{sign}(w \cdot x + b)$$

Any of these
would be fine..

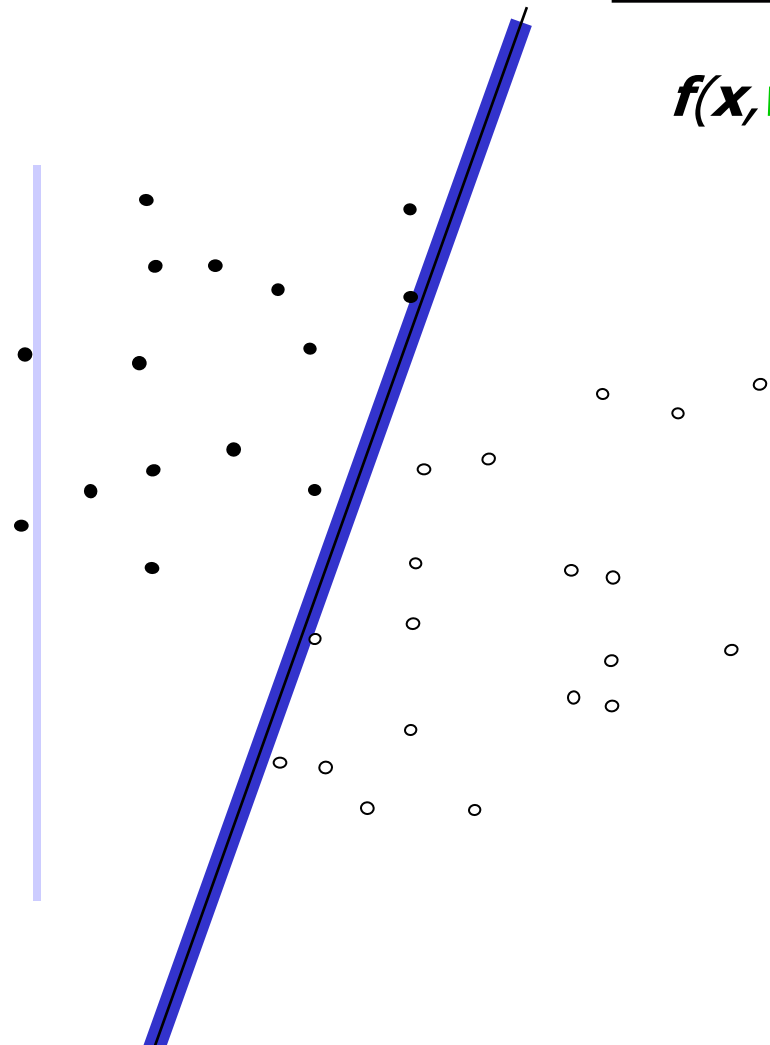
..but which is
best?

Classifier Margin



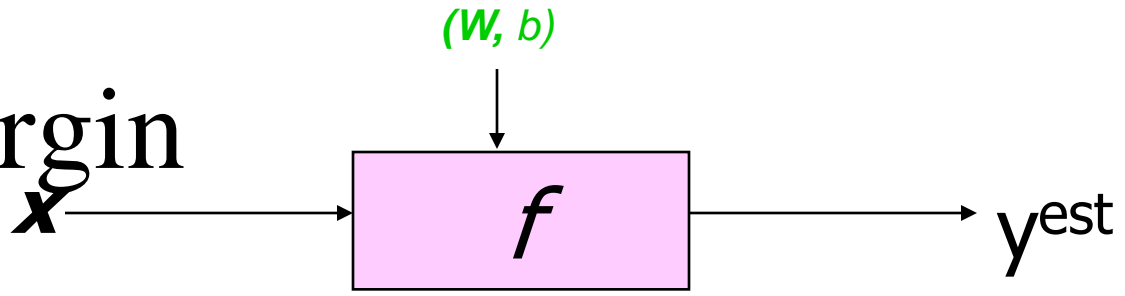
$$f(x, w, b) = \text{sign}(w \cdot x + b)$$

- denotes +1
- denotes -1

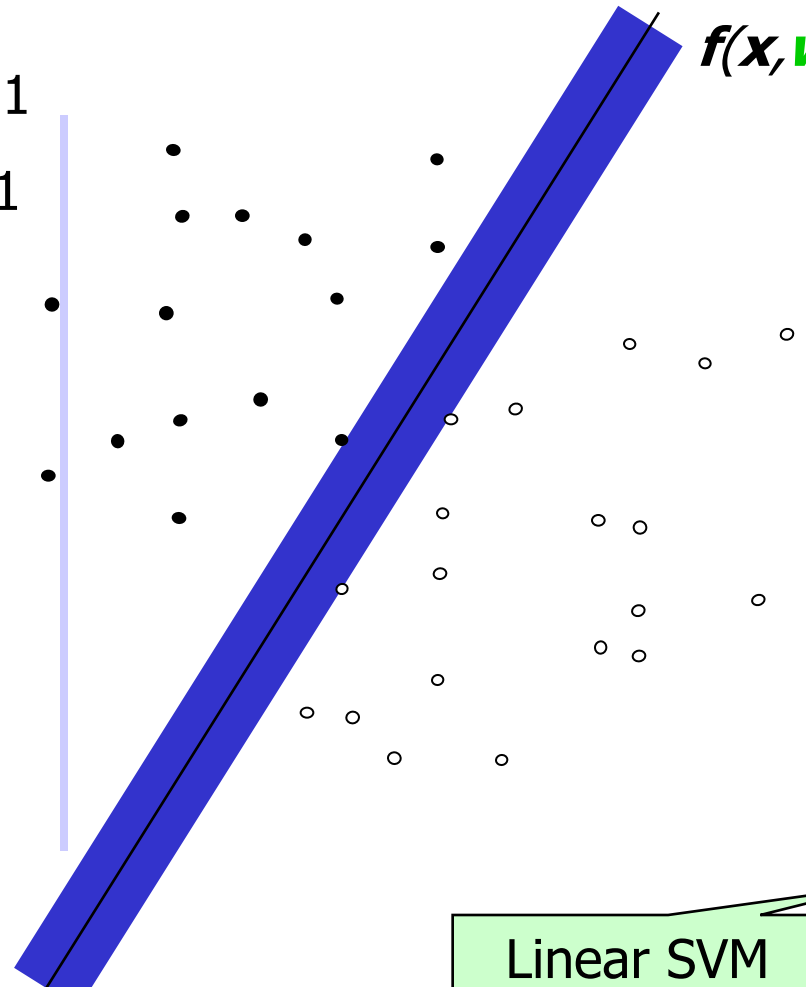


Define the **margin** of a linear classifier as the width that the boundary could be increased by before hitting a datapoint.

Maximum Margin



- denotes +1
- denotes -1



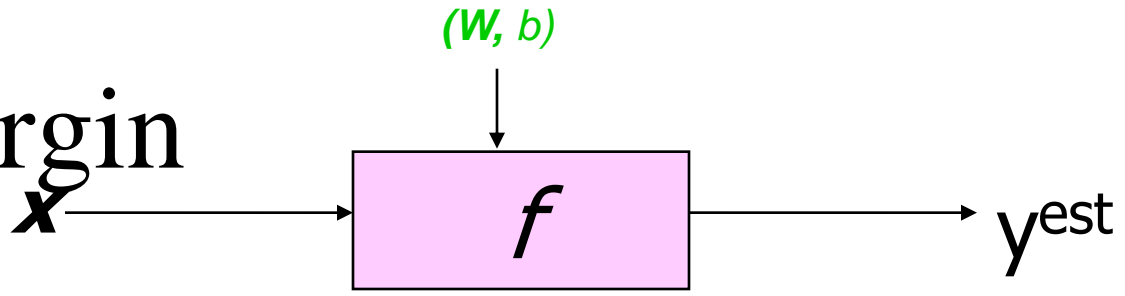
$$f(\mathbf{x}, \mathbf{w}, b) = \text{sign}(\mathbf{w} \cdot \mathbf{x} + b)$$

The **maximum margin linear classifier** is the linear classifier with the maximum margin.

This is the simplest kind of SVM (Called an LSVM)

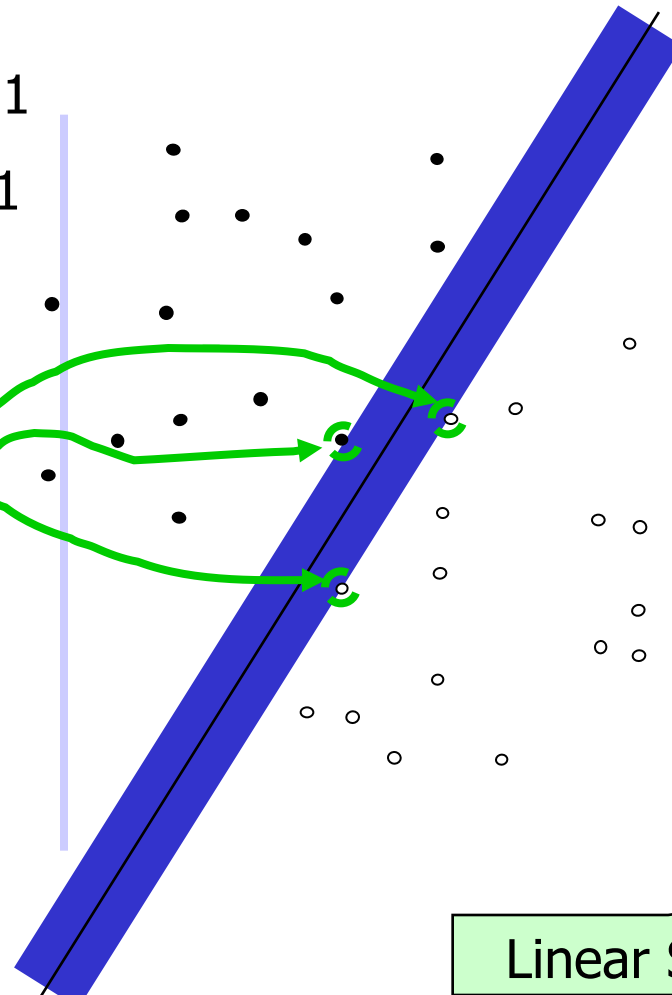
Linear SVM

Maximum Margin



- denotes +1
- denotes -1

Support Vectors are those datapoints that the margin pushes up against



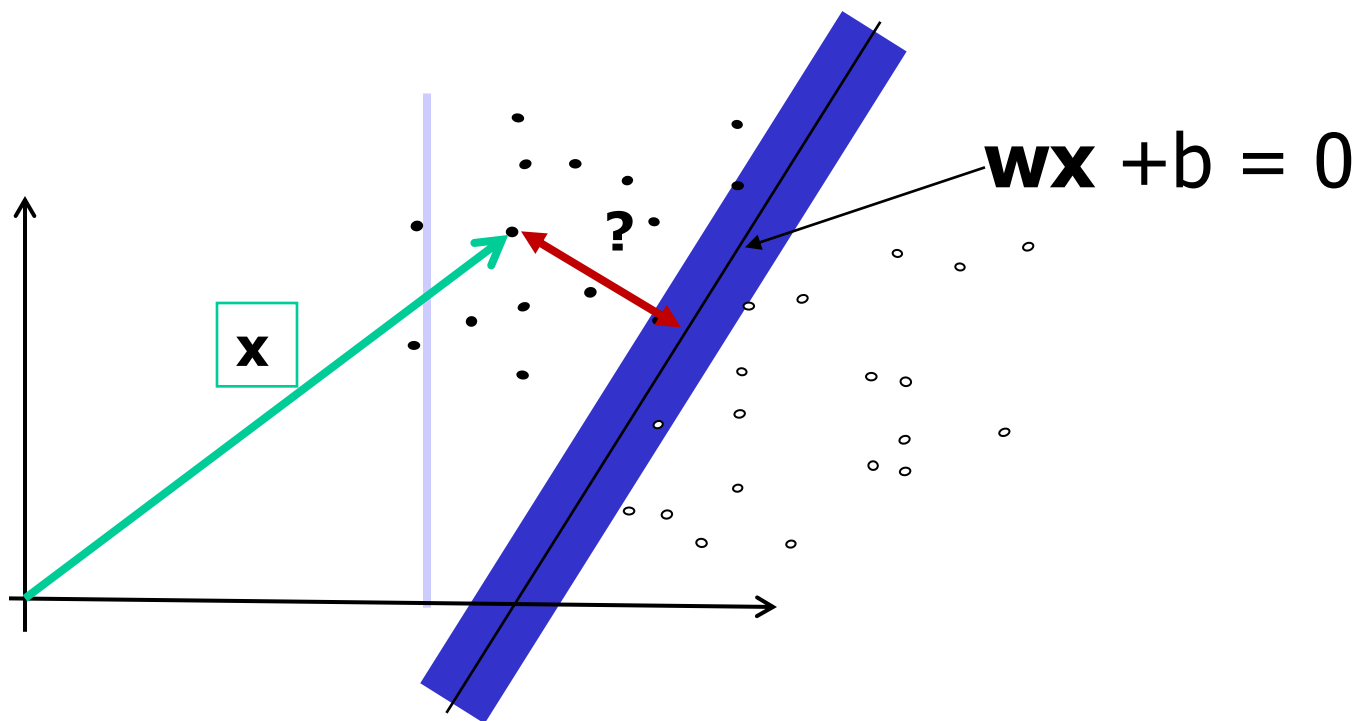
$$f(\mathbf{x}, \mathbf{w}, b) = \text{sign}(\mathbf{w} \cdot \mathbf{x} + b)$$

The **maximum margin linear classifier** is the linear classifier with the, um, maximum margin.

This is the simplest kind of SVM (Called an LSVM)

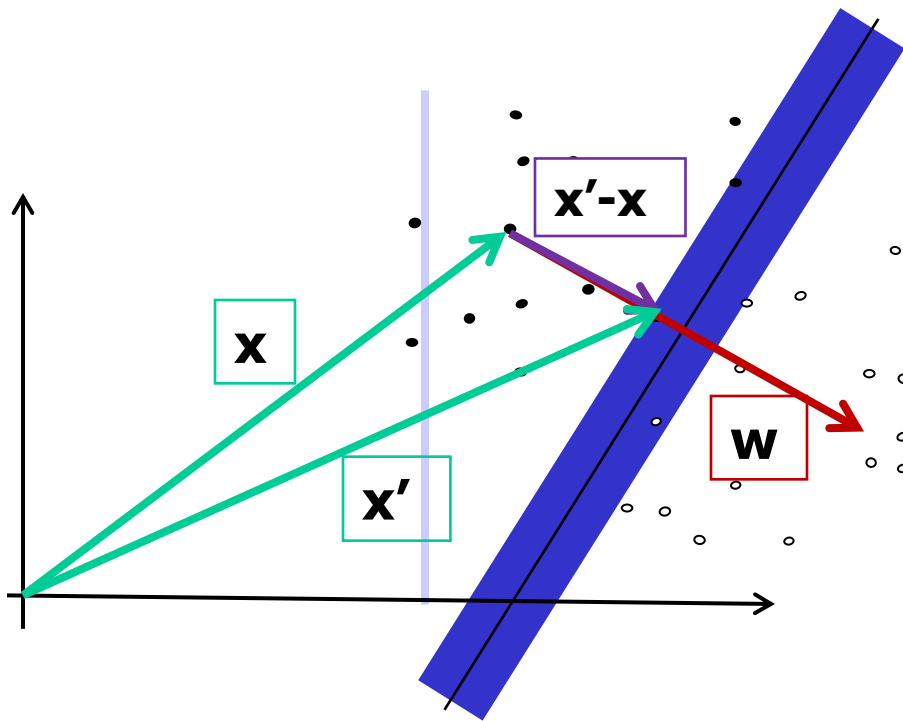
Linear SVM

Determining the Optimal Separating Hyperplane



What is the distance from a point \mathbf{x} to the hyperplane $\langle \mathbf{w}, \mathbf{x} \rangle + b = 0$?

Determining the Optimal Separating Hyperplane



$$x' \text{ s.t. } \langle w, x' \rangle + b = 0$$

$$\langle w, x - x' \rangle = \langle w, x' \rangle - \langle w, x \rangle,$$

$$\langle w, x - x' \rangle = -b - \langle w, x \rangle.$$

Projection of $x - x'$ onto w :

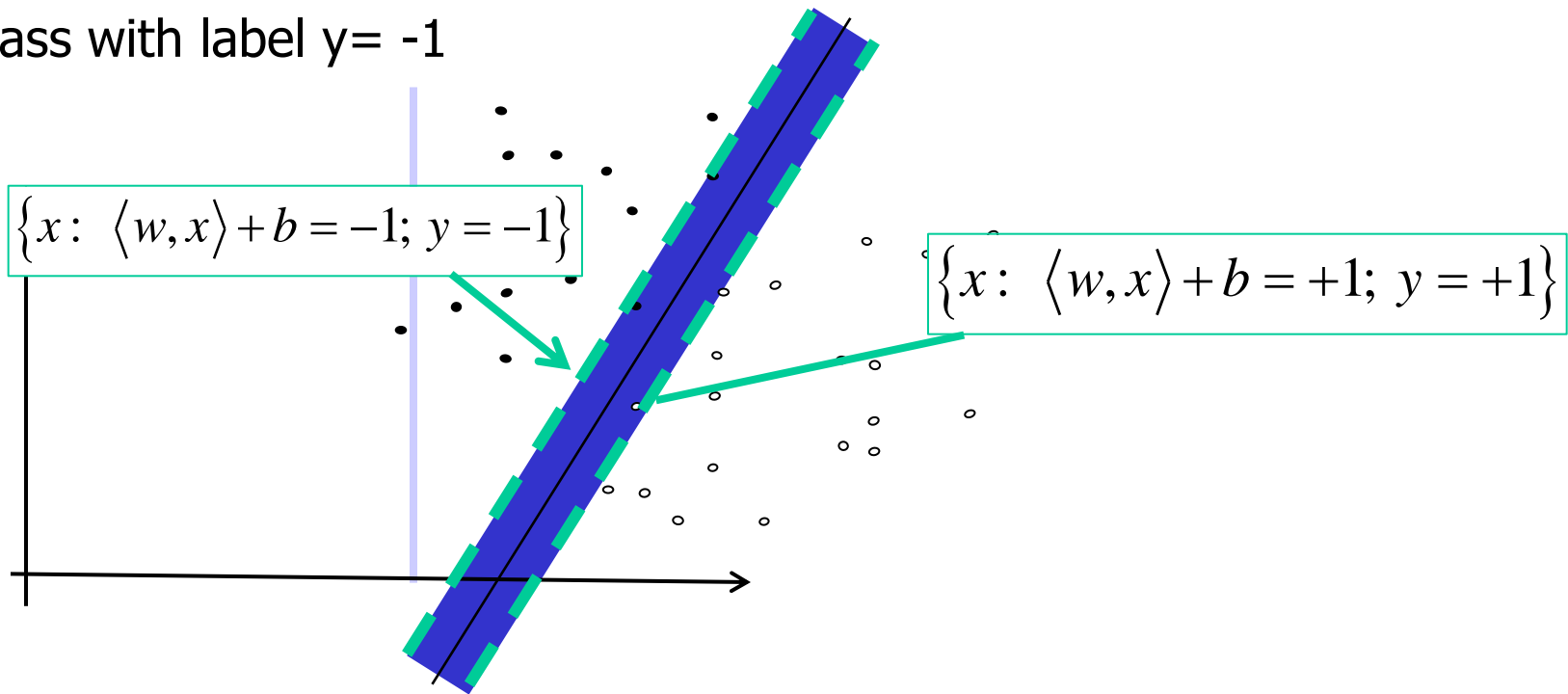
$$\frac{-\langle w, x \rangle - b}{\|w\|^2} w = \frac{\langle w, x \rangle + b}{\|w\|} \frac{w}{\|w\|}$$

unitary vector

$$\text{Distance to hyperplane} = \frac{|\langle w, x \rangle + b|}{\|w\|}$$

Determining the Optimal Separating Hyperplane

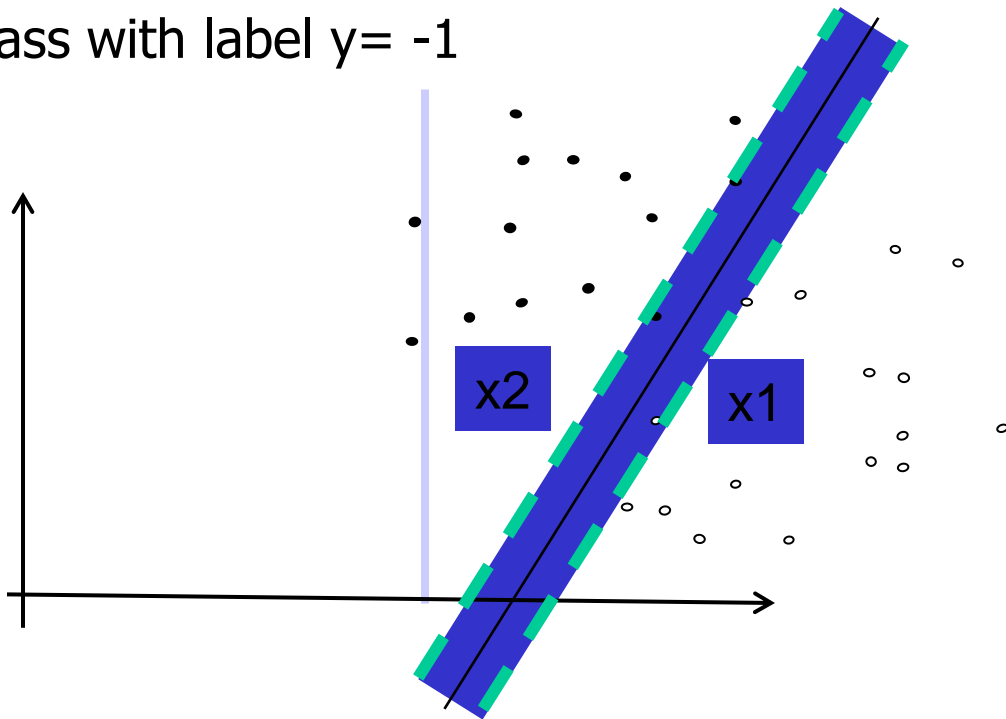
- Class with label $y=+1$
- Class with label $y= -1$



The margin on either side of the hyperplane satisfy $\langle w, x \rangle + b = +/-1$.

Determining the Optimal Separating Hyperplane

- Class with label $y=+1$
- Class with label $y= -1$



Two points on either side of the margin:

$$\langle w, x^1 \rangle + b = +1$$

$$\langle w, x^2 \rangle + b = -1$$

$$\Rightarrow \langle w, (x^1 - x^2) \rangle + b = 2$$

$$\Rightarrow \|x^1 - x^2\| = \frac{2}{\|w\|}$$

The margin between two classes is at least $2/\|w\|$.

Need b to ensure that the two points are on the right side of the separating plane!

Determining the Optimal Separating Hyperplane

Separating condition is measured by $\frac{2}{\|w\|}$.

To maximize this condition is equivalent to minimizing $\frac{\|w\|}{2}$.

Better even is to minimize the convex form $\frac{\|w\|^2}{2}$.

Determining the Optimal Separating Hyperplane

- Finding the Optimal Separating Hyperplane turns out to be an optimization problem of the following form:

$$\begin{array}{l}
 \min_{w,b} \frac{1}{2} \|w\|^2 \\
 \left. \begin{array}{l}
 \langle w, x^i \rangle + b \geq 1 \quad \text{when } y^i = +1 \\
 \langle w, x^i \rangle + b \leq -1 \quad \text{when } y^i = -1
 \end{array} \right\} \Rightarrow y^i (\langle w, x^i \rangle + b) \geq 1, \quad i=1,2,\dots,M.
 \end{array}$$

- $N+1$ parameters (N : dimension of data)
- M constraints (M : nm of datapoints)
- It is called the *primal problem*.

Determining the Optimal Separating Hyperplane

Rephrase the minimization under constraint problem in terms of the Lagrange Multipliers α_i , $i = 1, \dots, M$ (M , # of data points), one for each of the inequality constraints and we get the *dual problem*:

$$L(w, b, \alpha) \equiv \frac{1}{2} \|w\|^2 - \sum_{i=1}^M \alpha_i \left(y^i \left(\langle w, x^i \rangle + b \right) - 1 \right)$$

with $\alpha_i \geq 0$

(Minimization of convex function under linear constraints through Lagrange gives the optimal solution)

Determining the Optimal Separating Hyperplane

The solution of this problem is found when maximizing over α and minimizing over w and b :

$$\max_{\alpha \geq 0} \left(\min_{w, b} L(w, b, \alpha) \right)$$

where

$$L(w, b, \alpha) \equiv \frac{1}{2} \|w\|^2 - \sum_{i=1}^M \alpha_i \left(y^i \left(\langle w, x^i \rangle + b \right) - 1 \right)$$

Determining the Optimal Separating Hyperplane

Requesting that the gradient of L vanishes with w and b

$$\frac{\partial L(w, b, \alpha)}{\partial w} = 0 \Leftrightarrow w = \sum_{i=1}^M \alpha_i y^i x^i$$

$$\frac{\partial L(w, b, \alpha)}{\partial b} = 0 \Leftrightarrow \sum_{i=1}^M \alpha_i y^i = 0$$

The vector defining the hyperplane is determined by the training points.

Note that while w is unique (minimization of convex function), the alpha are not unique.

Determining the Optimal Separating Hyperplane

Complete optimization problem:

$$\frac{\partial L(w, b, \alpha)}{\partial w} = 0 \Leftrightarrow w = \sum_{i=1}^M \alpha_i y^i x^i \quad (\text{Dual feasibility})$$

$$\frac{\partial L(w, b, \alpha)}{\partial b} = 0 \Leftrightarrow \sum_{i=1}^M \alpha_i y^i = 0 \quad (\text{Dual feasibility})$$

$$\frac{\partial L(w, b, \alpha)}{\partial \alpha} \leq 0 \Leftrightarrow y^i (\langle w, x^i \rangle + b) \geq 1 \quad (\text{Primal feasibility})$$

Karush-Kuhn-Tucker conditions:

$$\alpha_i \left(y^i (\langle w, x^i \rangle + b) - 1 \right) = 0 \quad \forall i = 1, \dots, M \quad (\text{Complementarity conditions})$$

$$\alpha_i \geq 0, \quad \forall i = 1, \dots, M$$

Determining the Optimal Separating Hyperplane

Feasibility conditions from KKT require

$\alpha_i \geq 0$ for all points x_i for which the primal constraints are satisfied:

$$y^i \left(\langle w, x^i \rangle + b \right) \geq 1$$

Points correctly classified



Karush-Kuhn-Tucker conditions:

$$\alpha_i \left(y^i \left(\langle w, x^i \rangle + b \right) - 1 \right) = 0 \quad \forall i = 1, \dots, M \quad \text{(Complementarity conditions)}$$

$$\alpha_i \geq 0, \quad \forall i = 1, \dots, M$$

Determining the Optimal Separating Hyperplane

The α_i , $i = 1, \dots, M$ determine the solutions to the constraints

All the pairs of data points (x^i, y^i) for which $\alpha_i > 0$ satisfy the constraints \rightarrow support vectors

All the pairs of data points (x^i, y^i) for which $\alpha_i = 0$ are "irrelevant" when computing the margin.

Karush-Kuhn-Tucker conditions:

$$\alpha_i \left(y^i \left(\langle w, x^i \rangle + b \right) - 1 \right) = 0 \quad \forall i = 1, \dots, M \quad (\text{Complementarity conditions})$$

$$\alpha_i \geq 0, \quad \forall i = 1, \dots, M$$

Determining the Optimal Separating Hyperplane

The α_i , $i = 1, \dots, M$ determine the solutions to the constraints

All the pairs of data points (x^i, y^i) for which $\alpha_i > 0$ satisfy the constraints \rightarrow support vectors

All the pairs of data points (x^i, y^i) for which $\alpha_i = 0$ are "irrelevant" when computing the margin.

Consider 3 cases:

$0 < y^i (w^T x^i + b) < 1$ inside the margin

$1 < y^i (w^T x^i + b)$ outside the margin

$y^i (w^T x^i + b) < 0$ do not satisfy the constraint

Determining the Optimal Separating Hyperplane

The decision function is then expressed in terms of the support vectors:

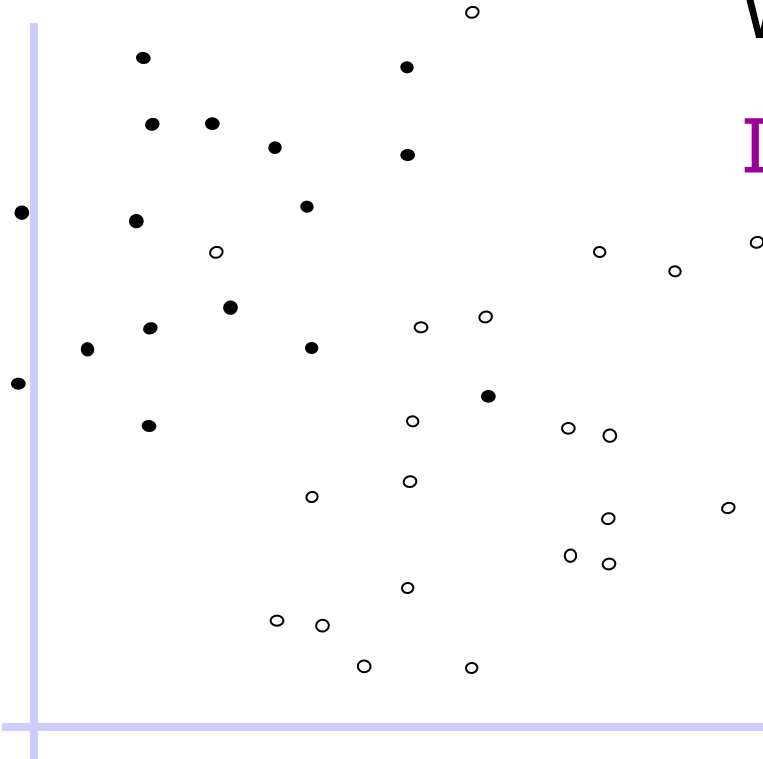
$$f(x) = \text{sgn}(\langle w, x \rangle + b) = \text{sgn}\left(\sum_{i=1}^M \alpha_i y^i \langle x, x^i \rangle + b\right)$$

To determine how good the hyperplane is

→ crossvalidation to get an estimation of the error on the testing set.

Non-Separable Data Sets

- denotes +1
- denotes -1



This is going to be a problem!

What should we do?

Idea :

Introduce some slack on
the constraints

Support Vector Machine for non-separable datasets

The constraints are relaxed by introducing slack variables $\xi_i, i = 1, \dots, M$:

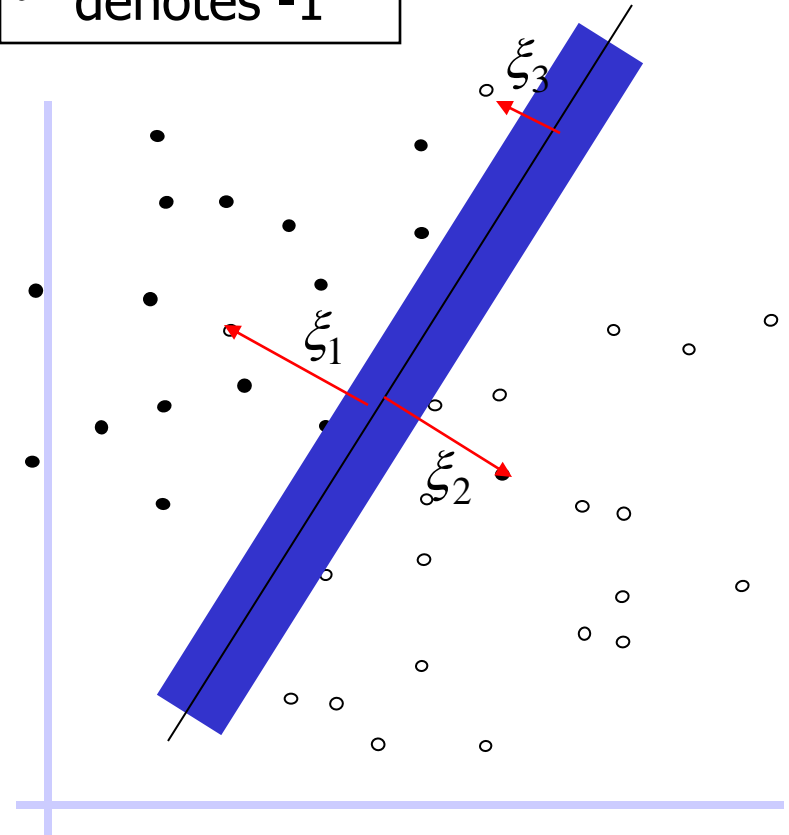
$$y_1 (w \cdot x_1 + b) \geq 1 - \xi_1, \quad \xi_1 \geq 0$$

$$y_2 (w \cdot x_2 + b) \geq 1 - \xi_2, \quad \xi_2 \geq 0$$

...

$$y_M (w \cdot x_M + b) \geq 1 - \xi_M, \quad \xi_M \geq 0$$

- denotes +1
- denotes -1



Support Vector Machine for non-separable datasets

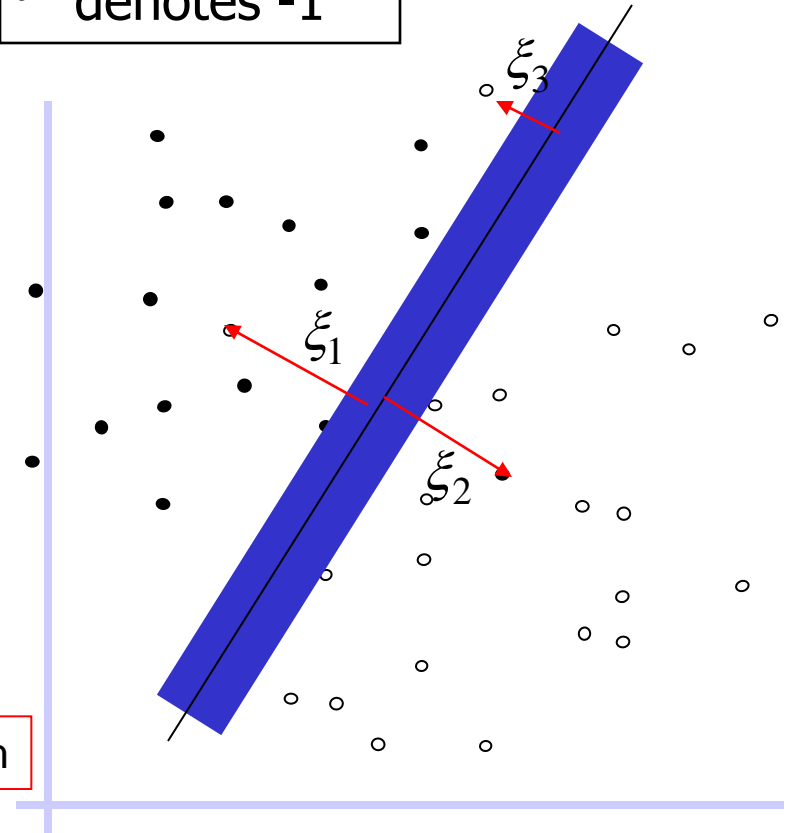
The objective function gives a penalty for too large slack variables:

$$\min_{w, \xi} \left(\frac{1}{2} \|w\|^2 + \frac{C}{M} \sum_{j=1}^M \xi_j \right)$$

Find a trade off between maximizing the margin and minimizing the classification errors.

C weights the influence of the penalty term

- denotes +1
- denotes -1



Support Vector Machine for non-separable datasets

Optimization under constraint problem of the form:

$$\min_{w, \xi} \left(\frac{1}{2} \|w\|^2 + \frac{C}{M} \sum_{j=1}^M \xi_j \right)$$

u.c.

$$y^j \left(w^T \cdot x^j + b \right) \geq 1 - \xi_j,$$

$$\xi_j \geq 0 \quad \forall j=1, \dots, M$$

Support Vector Machine for non-separable datasets

The Dual Form is given by:

$$\max_{\alpha} L_D(\alpha) \equiv \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y^i y^j \langle x^i, x^j \rangle$$

Subject to these constraints:

$$0 \leq \alpha_j \leq \frac{C}{M} \quad \forall j = 1, \dots, M \quad \sum_{j=1}^M \alpha_j y^j = 0$$

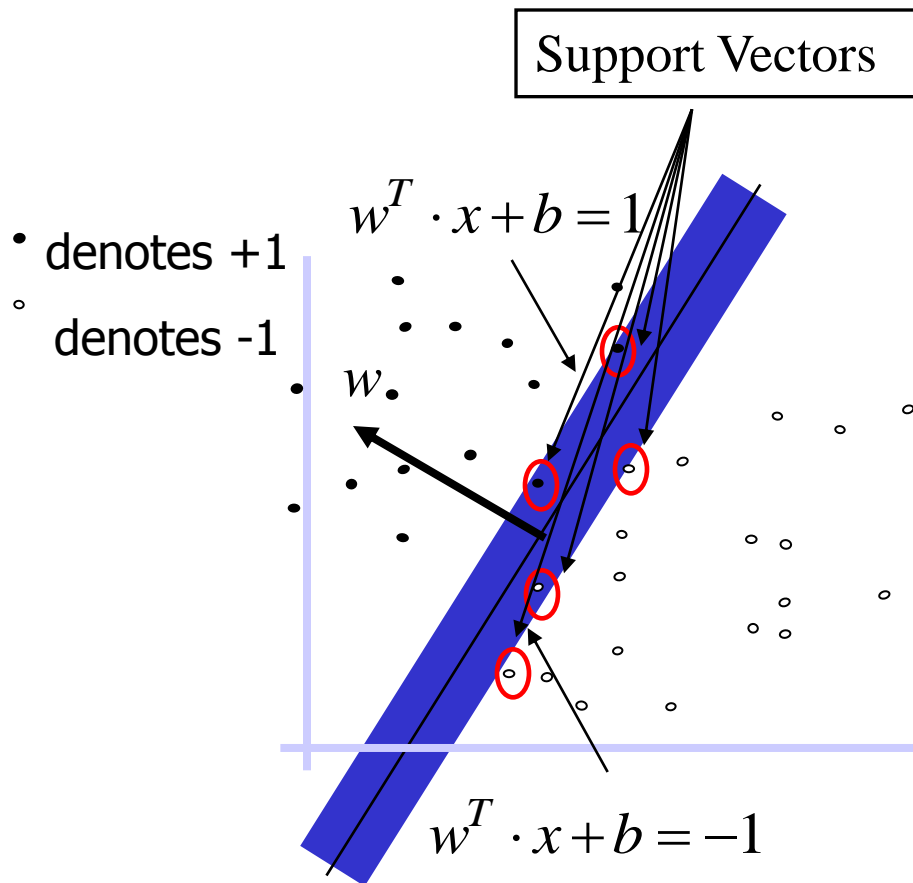
Datapoints with $\alpha_j > 0$ will be the support vectors

The hyperplane has the same solution

..so this sum only needs to be over the support vectors.

$$\mathbf{w} = \sum_{j=1}^M \alpha_j y^j \mathbf{x}^j$$

Support Vector Machine for non-separable datasets



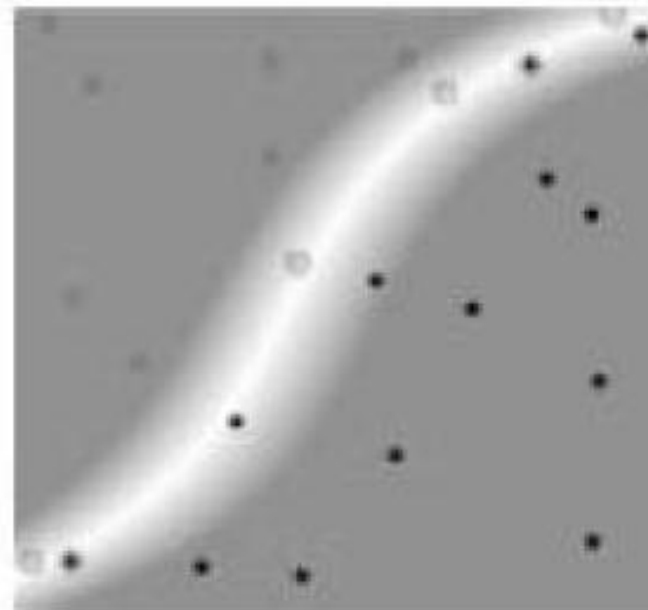
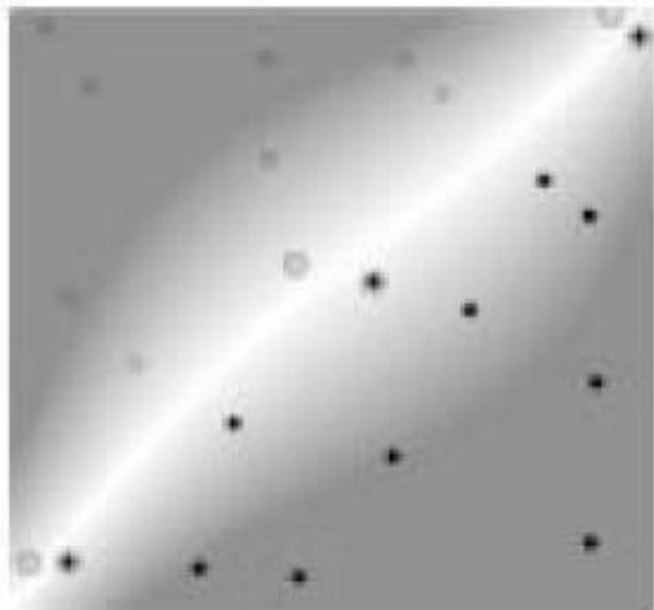
$$\forall i : \alpha_i \left(y^i \left(w^T \cdot x^i + b \right) - (1 - \xi_i) \right) = 0$$

$\alpha_i = 0$ for non-support vectors
 $\alpha_i \neq 0$ for support vectors

$$\mathbf{w} = \sum_{i=1}^M \alpha_i y^i \mathbf{x}^i$$

Decision boundary is
 determined only by those
 support vectors !

Non-Linear Classification



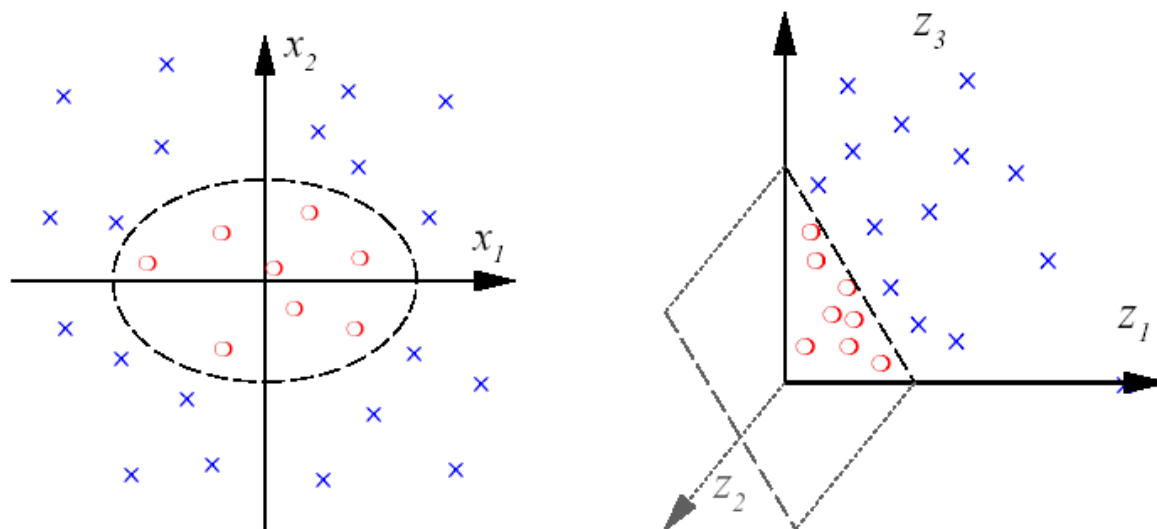
What if the points in the input space cannot be separated by a linear hyperplane?

High Dimension Mapping

Motivation: Linearly inseparable problem become linearly separable in higher dimension space.

$$\Phi : \mathbb{R}^2 \rightarrow \mathbb{R}^3$$

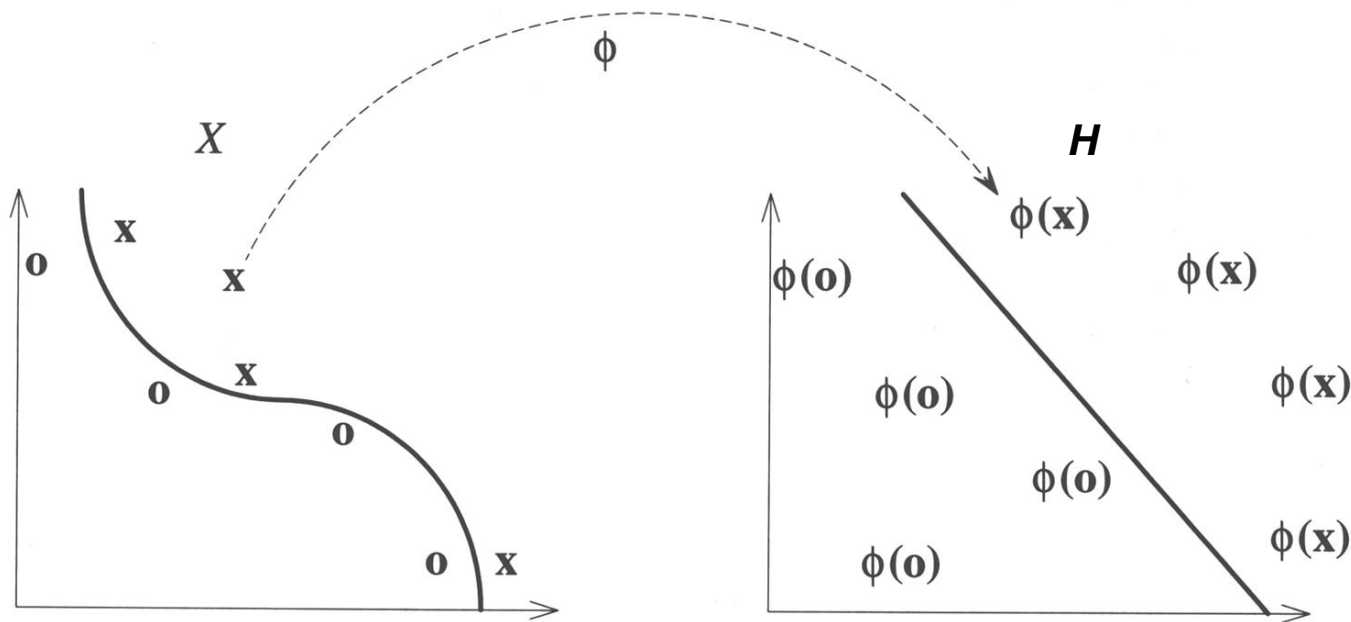
$$(x_1, x_2) \mapsto (z_1, z_2, z_3) := (x_1^2, \sqrt{2} x_1 x_2, x_2^2)$$



Kernel-Induced Feature Space

Idea: Send the data X into a *feature space* H through the *nonlinear map* ϕ .

$$X = \{x^i \in \mathbb{R}^N\}^{i=1\dots M} \mapsto \phi(X) = (\phi(x^1), \dots, \phi(x^M))$$



Kernel-Induced Feature Space

Idea: Send the data X into a *feature space* H through the *nonlinear map* ϕ .

$$X = \{x^i \in \mathbb{R}^N\}_{i=1\dots M} \mapsto \phi(X) = (\phi(x^1), \dots, \phi(x^M))$$

While the dimension of the original space is N , the dimension of the feature space may be greater than N !

→ X is lifted onto H

Determining ϕ is difficult → Kernel Trick

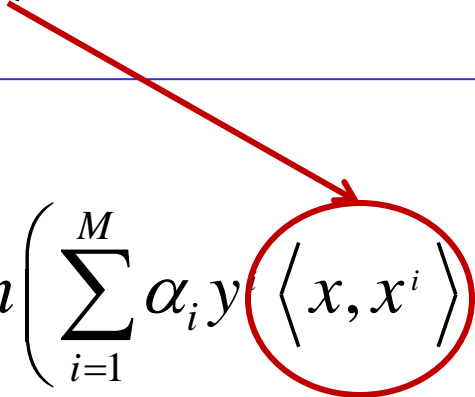
The Kernel Trick

In most cases, determining the transformation ϕ may be difficult.

Linear SVM computes an *inner* product across pairs of observations:

$$\langle x^i, x^j \rangle$$

The decision function:

$$f(x) = \text{sgn}(\langle w, x \rangle + b) = \text{sgn} \left(\sum_{i=1}^M \alpha_i y_i \langle x, x^i \rangle + b \right)$$


The Kernel Trick

In most cases, determining the transformation ϕ may be difficult.

Linear SVM computes an *inner* product across pairs of observations:

$$\langle x^i, x^j \rangle$$

No need to compute the transformation ϕ , if one expresses everything as a function of the inner product in feature space

→ the *kernel function*:

Metric of similarity across datapoints
May extract some features

$$k : X \times X \rightarrow \mathbb{R}$$

$$k(x^i, x^j) \rightarrow \langle \phi(x^i), \phi(x^j) \rangle.$$

Popular Kernels

- Gaussian / RBF Kernel (translation-invariant):

$$k(x, x') = e^{-\frac{\|x-x'\|^2}{2\sigma^2}}, \quad \sigma \in \mathbb{R}.$$

- Homogeneous Polynomial Kernels:

$$k(x, x') = \langle x, x' \rangle^p, \quad p \in \mathbb{N};$$

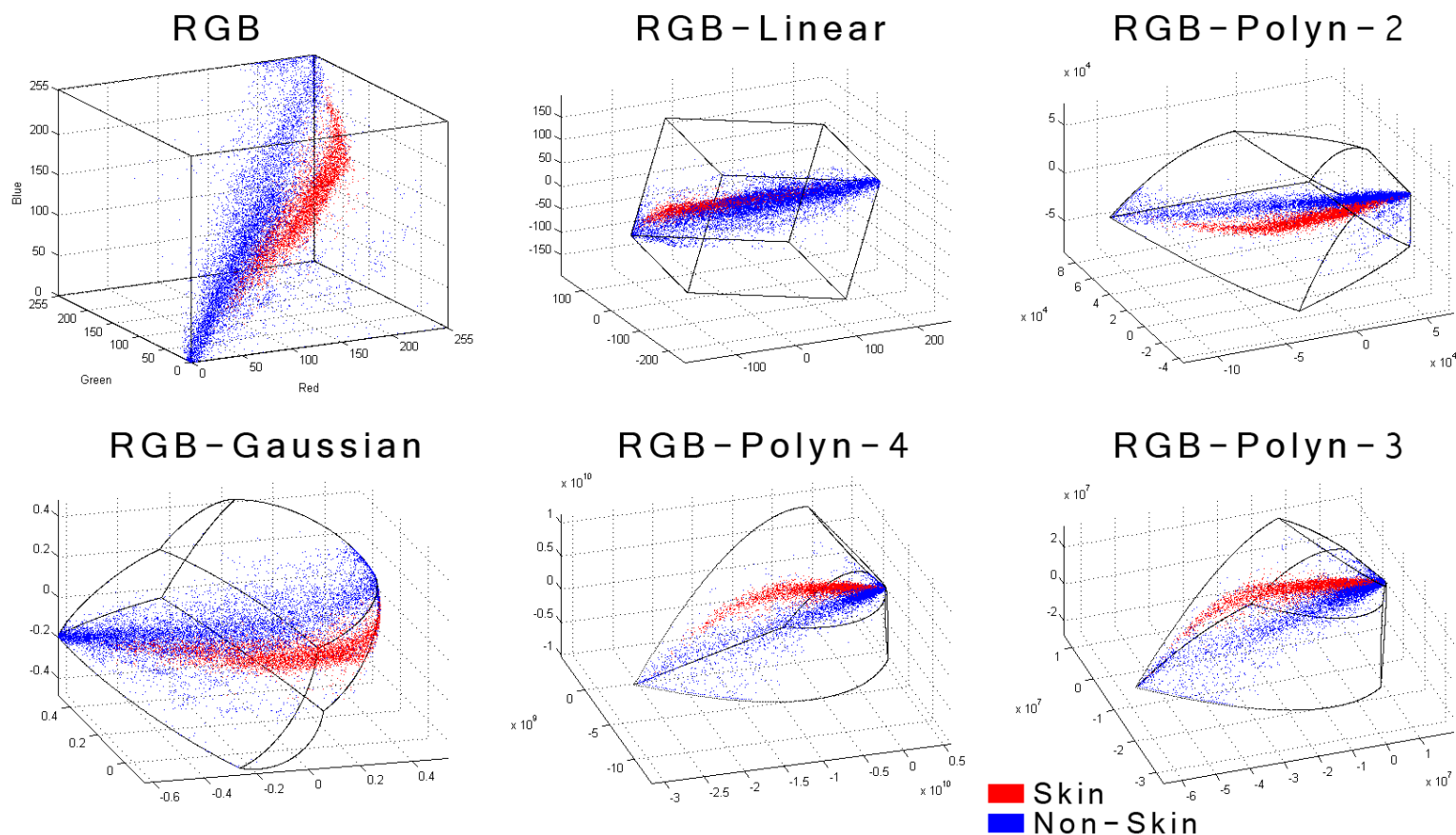
- Inhomogeneous Polynomial Kernels:

$$k(x, x') = \left(\langle x, x' \rangle + c \right)^p, \quad p \in \mathbb{N}, c \geq 0$$

Kernel transformations of the space

The kernel creates a distortion of the metric in original space

→ Can increase distance across features used for separating two classes



Data points representing RGB pixels of skin versus non-skin color

Non-Linear Classification with Support Vector Machines

The decision function in linear classification with SVM was given by:

$$f(x) = \text{sgn}(\langle w, x \rangle + b) = \text{sgn}\left(\sum_{i=1}^M \alpha_i y^i \langle x, x^i \rangle + b\right)$$

Replace linear plane in original space $\langle w, x \rangle + b$

by linear plane in feature space: $x \mapsto \phi(x)$ and $\langle w, \phi(x) \rangle + b$



The decision function becomes:

$$f(x) = \text{sgn}(\langle w, \phi(x) \rangle + b) = \sum_{i=1}^M \alpha_i y^i \langle \phi(x^i), \phi(x) \rangle + b$$

Non-Linear Classification with Support Vector Machines

$$k(x^i, x^j) \rightarrow \langle \phi(x^i), \phi(x^j) \rangle$$

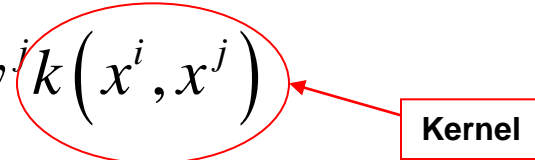
Use the kernel trick by exploiting the fact that the decision function depends on a dot product in feature space

The decision function becomes:

$$f(x) = \text{sgn}(\langle w, \phi(x) \rangle + b) = \sum_{i=1}^M \alpha_i y^i \langle \phi(x^i), \phi(x) \rangle + b$$

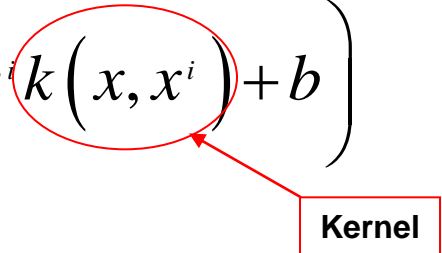
Non-Linear Classification with Support Vector Machines

The optimization problem in feature space becomes:

$$\max_{\alpha \in \mathbb{R}^N} L(\alpha) = \sum_{i=1}^M \alpha_i - \frac{1}{2} \sum_{i,j=1}^M \alpha_i \alpha_j y^i y^j k(x^i, x^j)$$


subject to $\alpha_i \geq 0$ for all $i = 1, \dots, M$ and $\sum_{i=1}^M \alpha_i y^i = 0$.

The decision function in feature space is computed as follows:

$$f(x) = \text{sgn} \left(\sum_{i=1}^M \alpha_i y^i k(x, x^i) + b \right)$$


Non-Linear Classification with Support Vector Machines

The offset b can be estimated from the KKT conditions.

Best is to compute the expectation over the constraints and hence to compute an estimate of b through regression:

$$b = \frac{1}{M} \sum_{j=1}^M \left(y^j - \sum_{i=1}^M y^i \alpha_i k(x^j, x^i) \right)$$

Non-linear classification for non-separable datasets

Generalizable to the non-linear case with optimization on the dual

of the form:
$$\max_{\alpha_1, \dots, \alpha_M} \left(-\frac{1}{2} \sum_{i=1}^M \alpha_i \alpha_j y^i y^j k(x_i, x_j) \right)$$

and with a decision function:

$$f(x) = \text{sgn} \left(\sum_{i=1}^M \alpha_i y^i k(x, x^i) + b \right)$$

To compute b , one must take two sets S_+, S_- of datapoints in each class with equal number s of points in each set:

$$b = -\frac{1}{2s} \sum_{x \in S_+ \cup S_-} \sum_{i=1}^M \alpha_i y^i k(x, x^i)$$

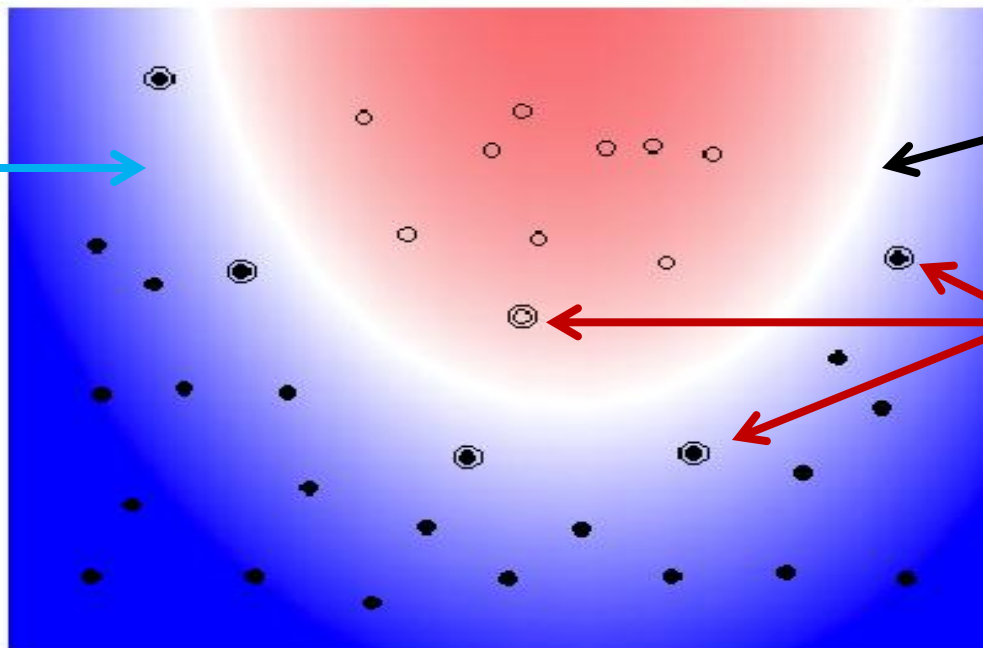
How to read out the result of SVM

Example: SVM with Polynomial of Degree 2

$$\text{Kernel: } K(\vec{x}_i, \vec{x}_j) = [\vec{x}_i \cdot \vec{x}_j + 1]^2$$

plot by Bell SVM applet

Color gradient
= distance to
the hyperplane



Hyperplane

Support vectors

How to read out the result of SVM

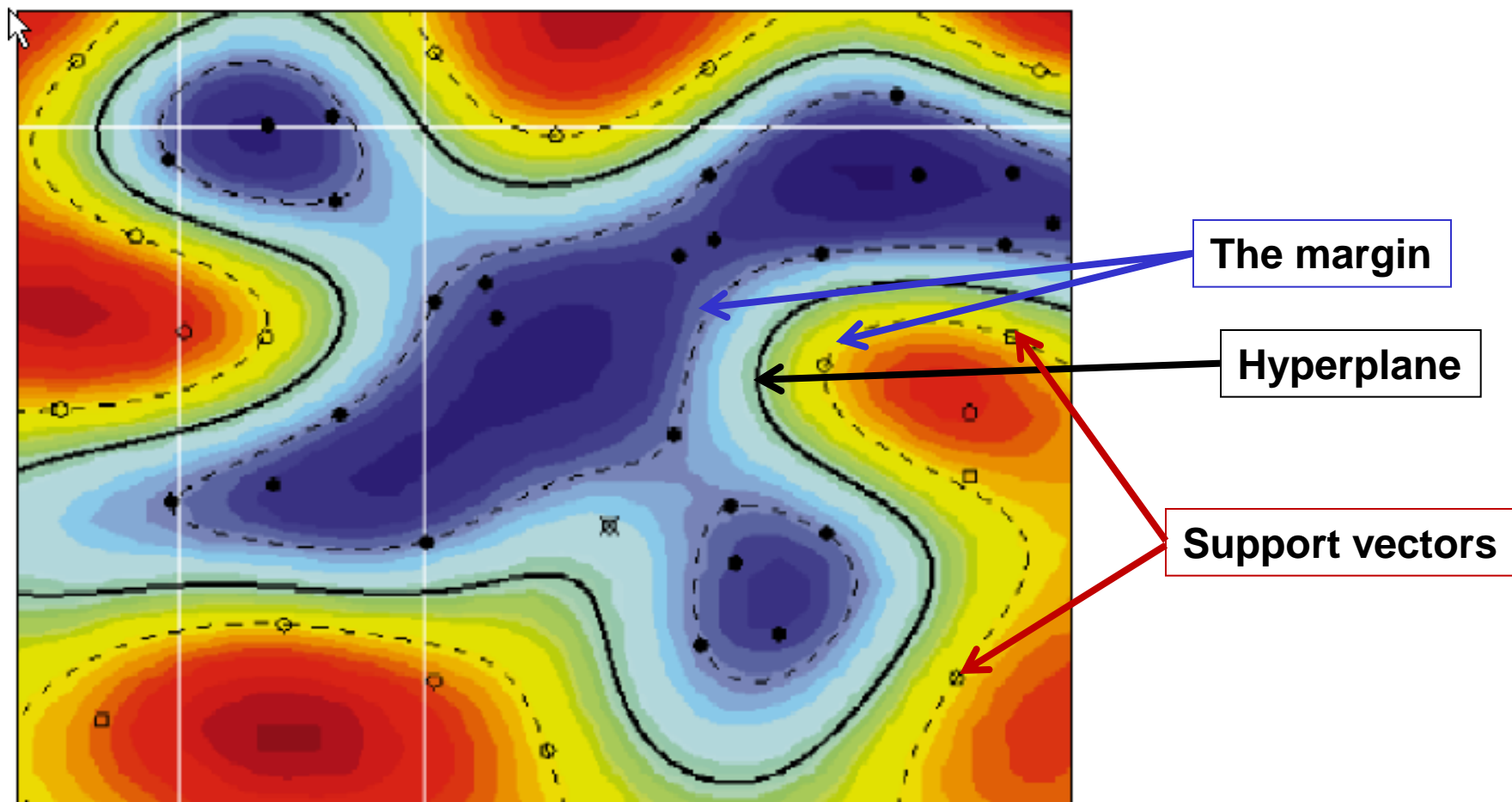


Figure 3. Example of an SV classifier found by using a radial basis function kernel (Equation 8). Circles and disks are two classes of training examples; the solid line is the decision surface; the support vectors found by the algorithm lie on, or between, the dashed lines. Colors code the modulus of the argument $\sum_i v_i \cdot k(\mathbf{x}, \mathbf{x}_i) + b$ of the decision function in Equation 10.

The hyperparameters of SVM

SVM decision function:

$$f(x) = \text{sgn} \left(\sum_{i=1}^M \alpha_i y^i k(x^i, x^i) + b \right)$$

The kernel has several open parameters (Hyperparameters) that need to be determined before running SVM

Gaussian kernel: $k(x, x') = e^{-\frac{\|x-x'\|^2}{2\sigma^2}}$, $\sigma \in \mathbb{R}$.

Kernel width

Order of the polynomial

Inhomogeneous polynomial kernel: $k(x, x') = (\langle x, x' \rangle + c)^p$, $p \in \mathbb{N}$, $c \geq 0$

Usually $c=1$

The hyperparameters of SVM

$$\min_{w, \xi} \left(\frac{1}{2} \|w\|^2 + \frac{C}{M} \sum_{j=1}^M \xi_j \right)$$

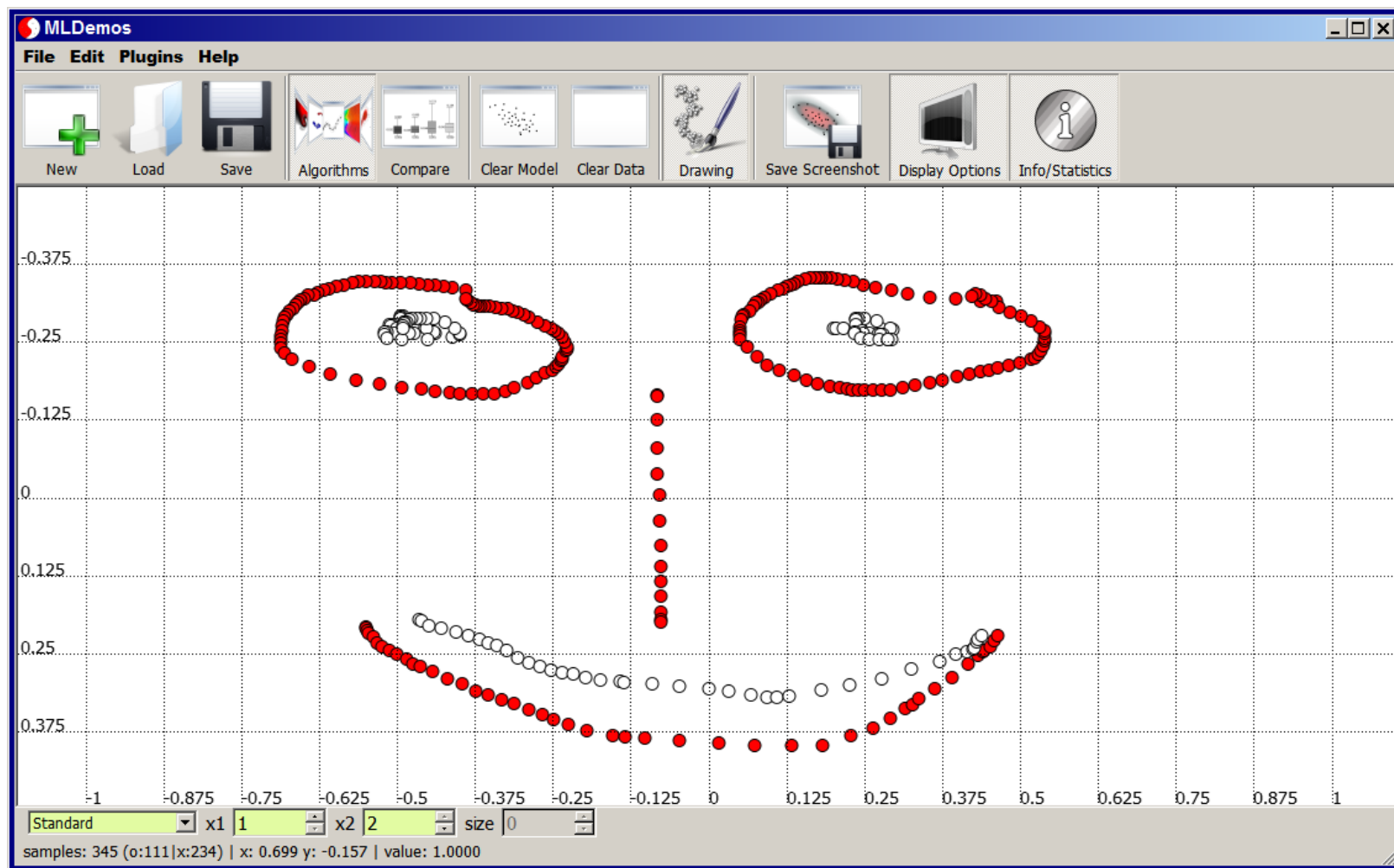
C that determines the costs associated to incorrectly classifying datapoints is an open parameter of the optimization function

u.c.

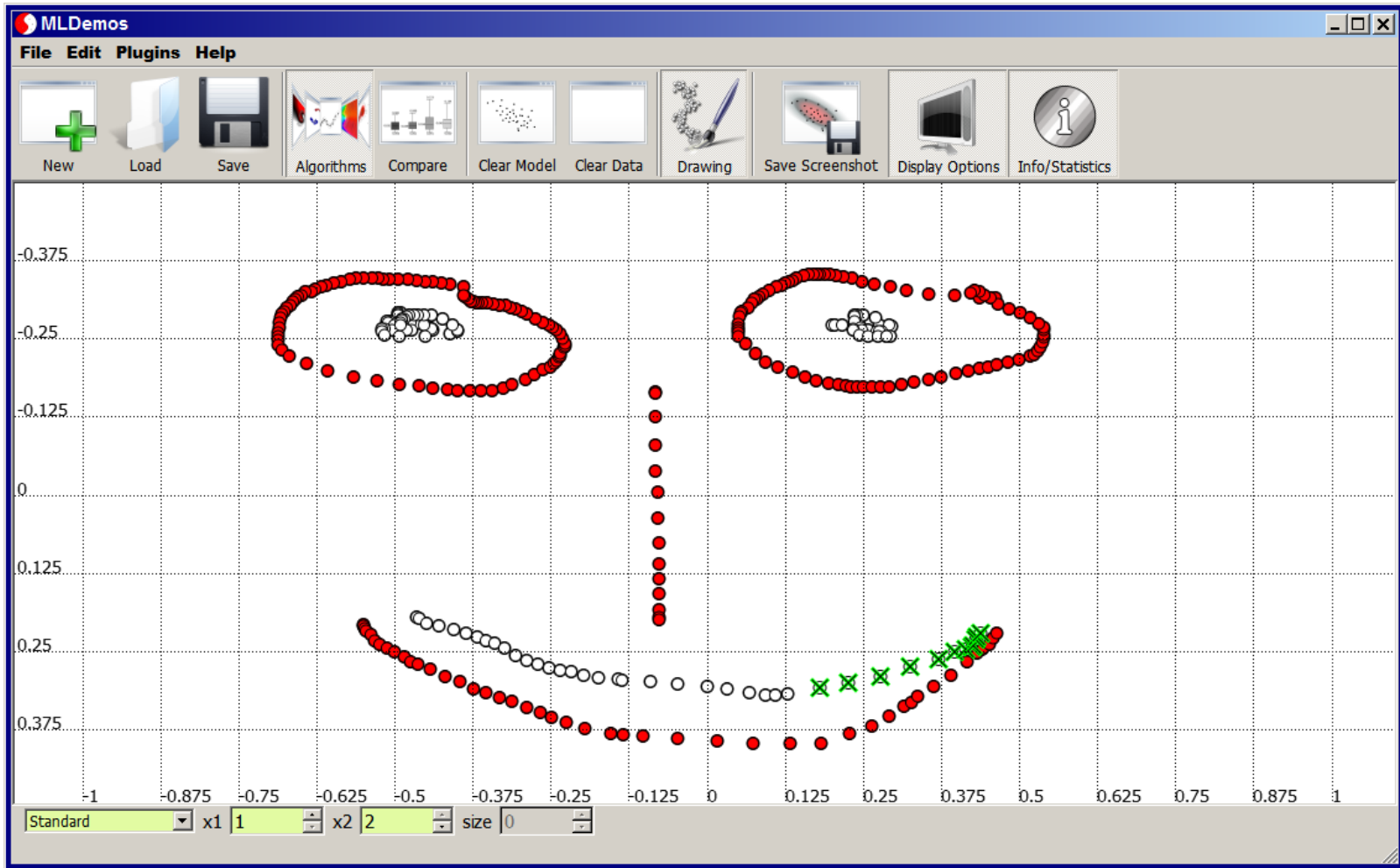
$$y^j (w^T \cdot x^j + b) \geq 1 - \xi_j,$$

$$\xi_j \geq 0 \quad \forall j=1, \dots, M$$

Non-Linear Support Vector Machines: Examples

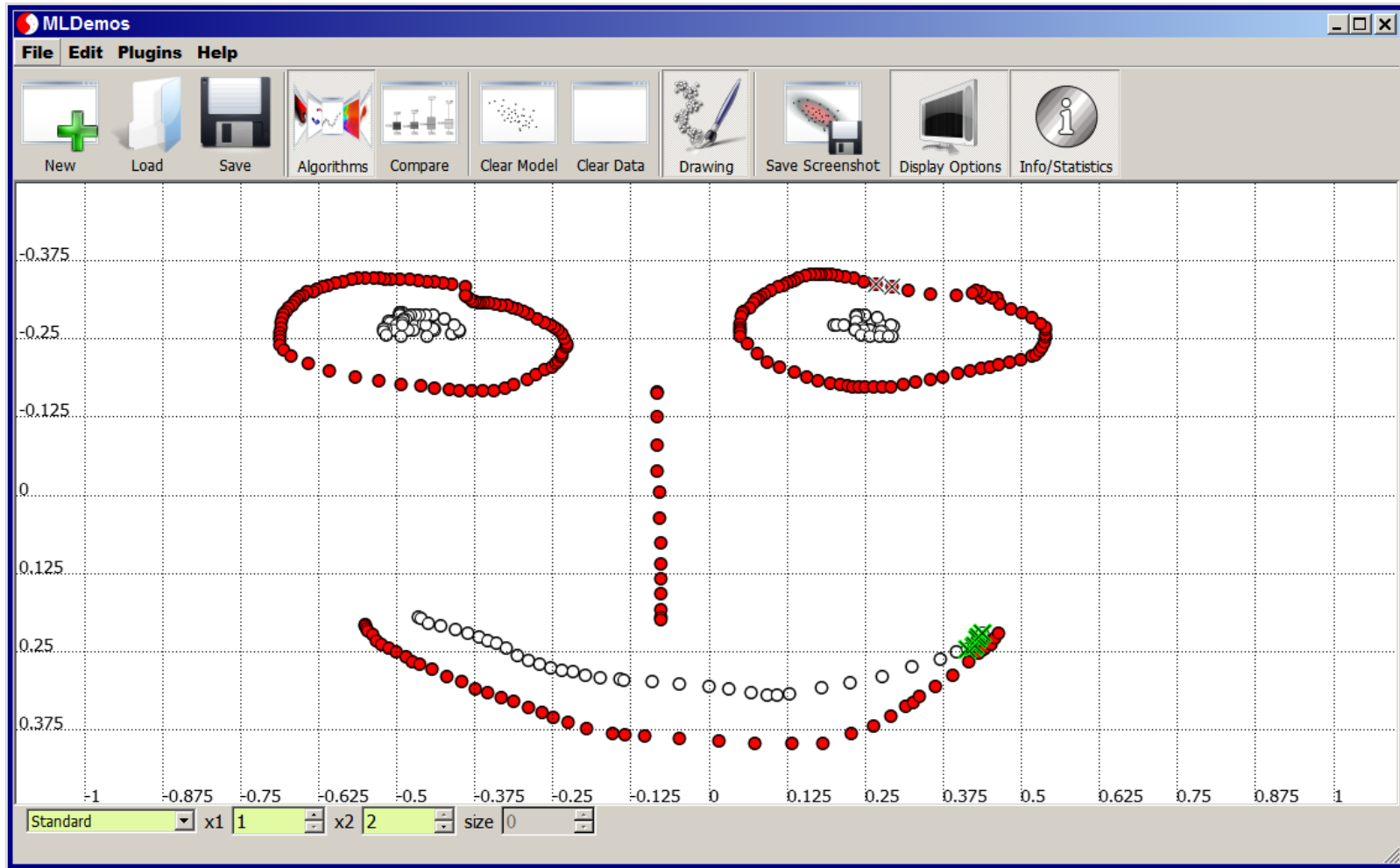


Effect of the penalty factor C



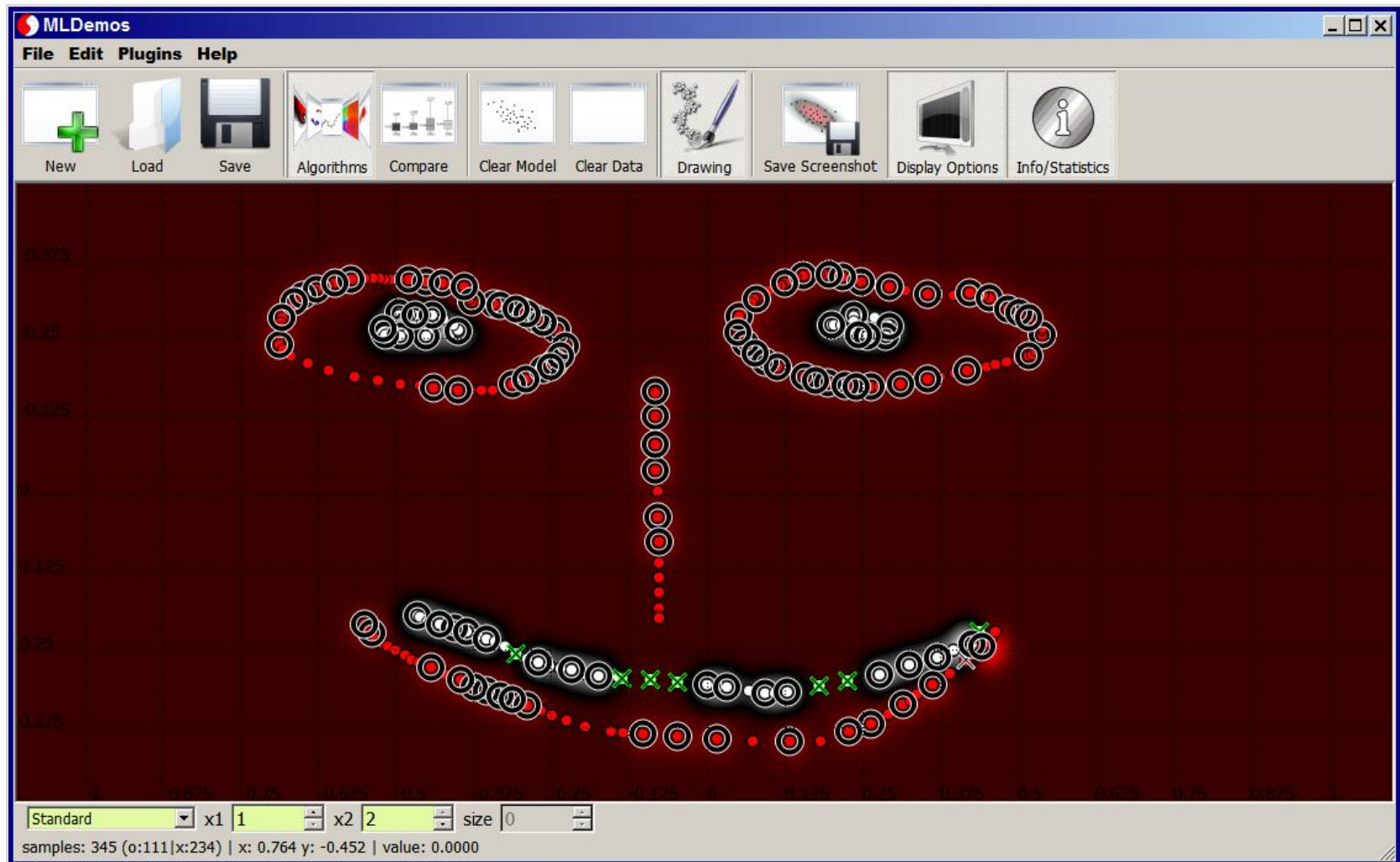
RBF kernel width=0.20; $C=1000$; several misclassified datapoints

Effect of the penalty factor C



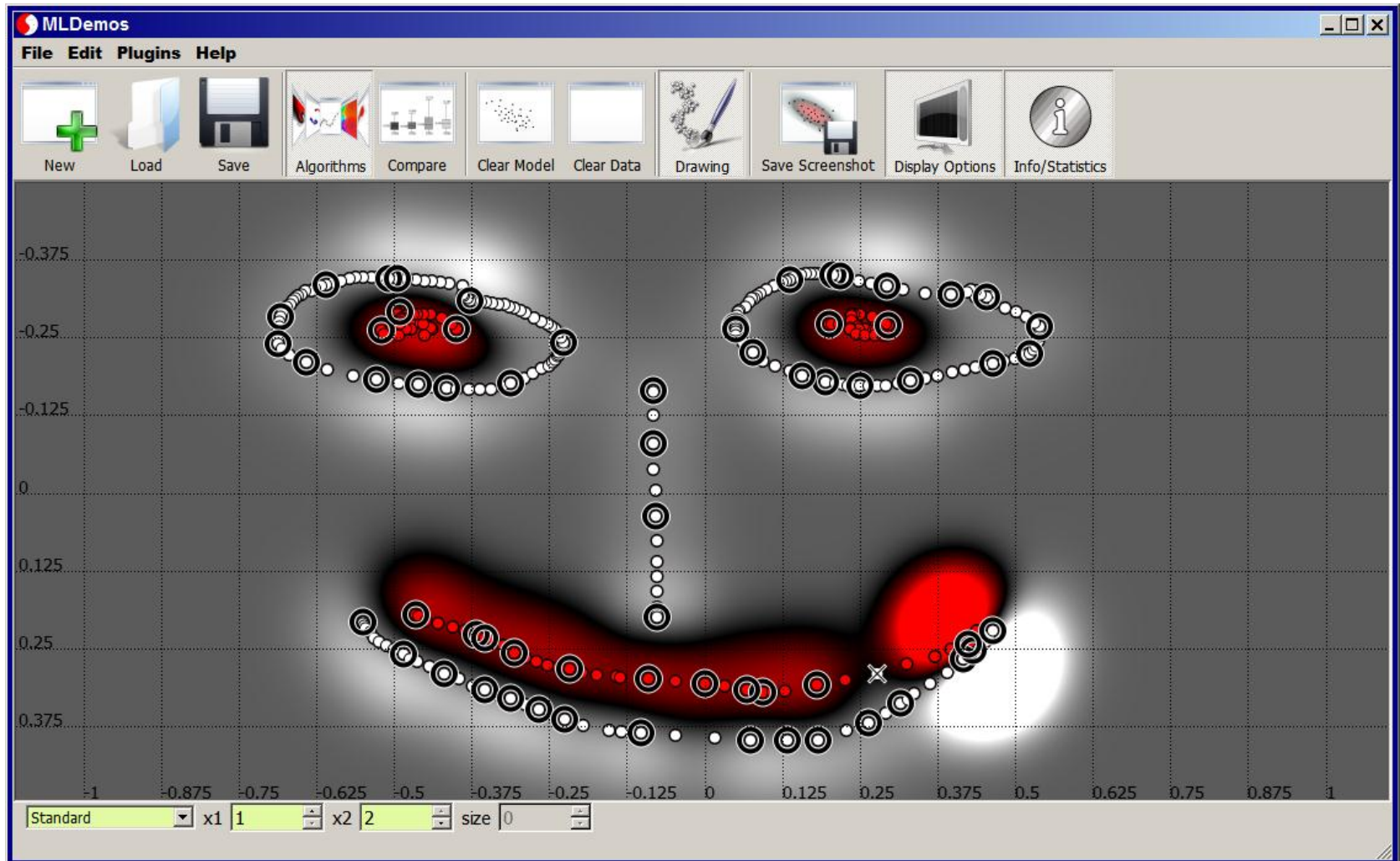
RBF kernel width=0.20; $C=2000$; less misclassified datapoints

Effect of the width of Gaussian kernel



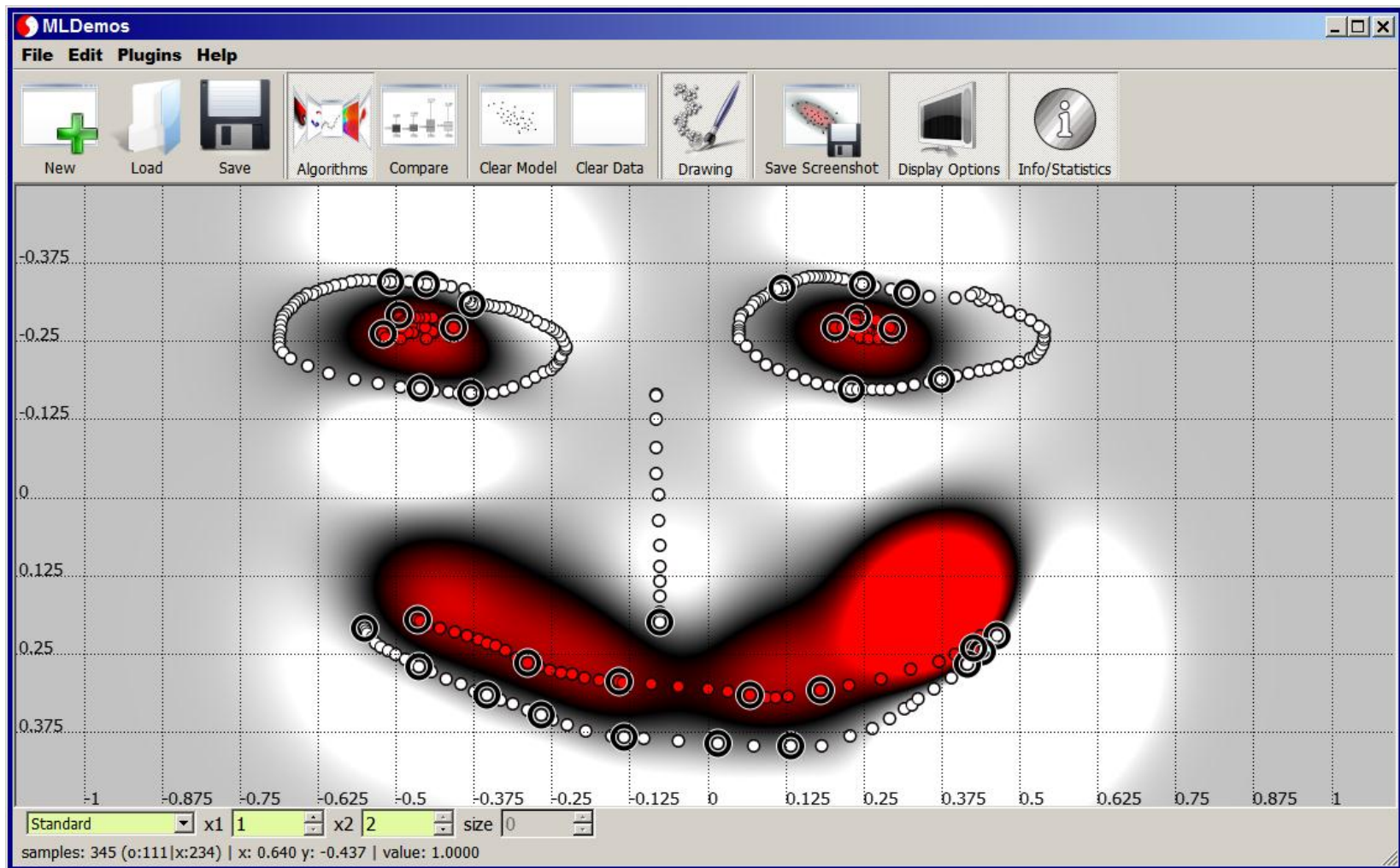
RBF kernel width = 0.001; C=1000; 113 support vectors out of 345 total nm of datapoints

Effect of the width of Gaussian kernel



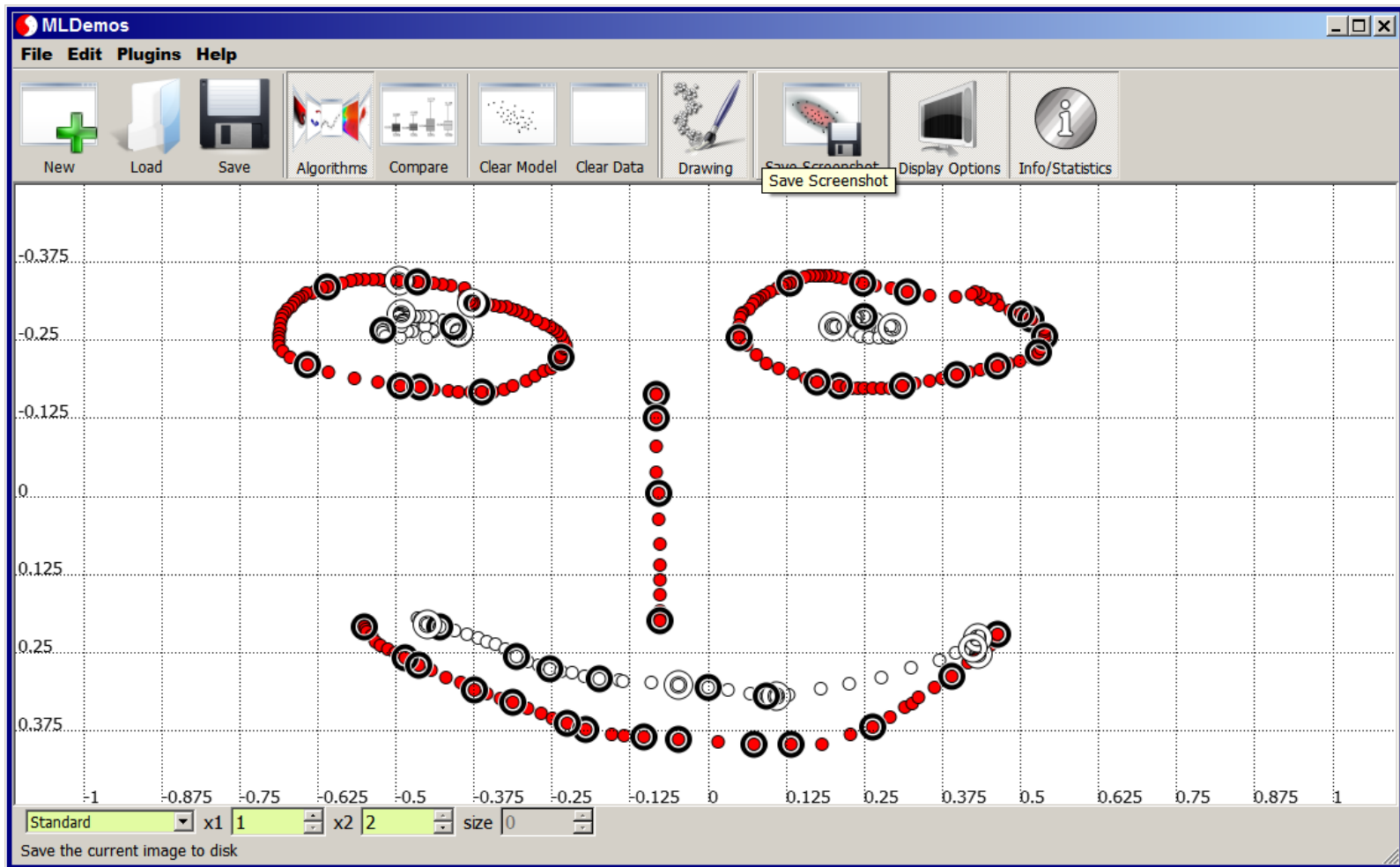
RBF kernel width = 0.008; C=1000; 64 support vectors out of 345 total nm of datapoints

Effect of the width of Gaussian kernel



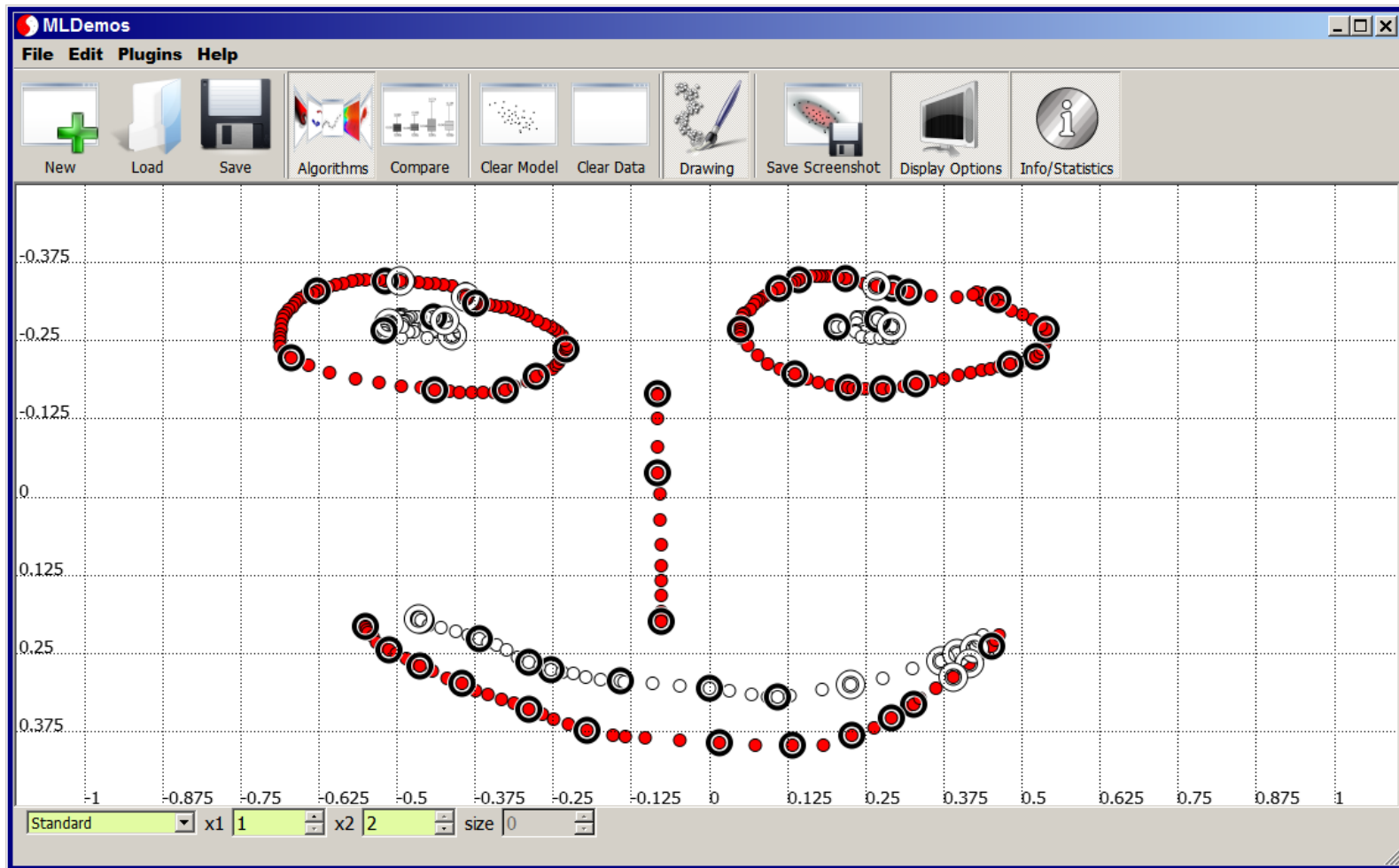
RBF kernel width = 0.02; C=1000; 33 support vectors out of 345 total nm of datapoints

Different optimization runs end up with different solutions



Several combination of the support vectors yield the same optimum

Different optimization runs end up with different solutions



Several combination of the support vectors yield the same optimum

Support Vector Machine for non-separable datasets

The original objective function:

$$\min_{w, \xi} \left(\frac{1}{2} \|w\|^2 + \frac{C}{M} \sum_{j=1}^M \xi_j \right)$$

Determining C may be difficult in practice

Support Vector Machine for non-separable datasets

ν -SVM is an alternative that optimizes for the best tradeoff between model complexity (the largest margin) and penalty on the error automatically.

$$\min_{w, \xi} \left(\|w\|^2 - \nu\rho - \frac{1}{M} \sum_{i=1}^M \xi_i \right),$$

$$\text{subject to } y^i \left(\langle w, x^i \rangle + b \right) \geq \rho - \xi_i$$

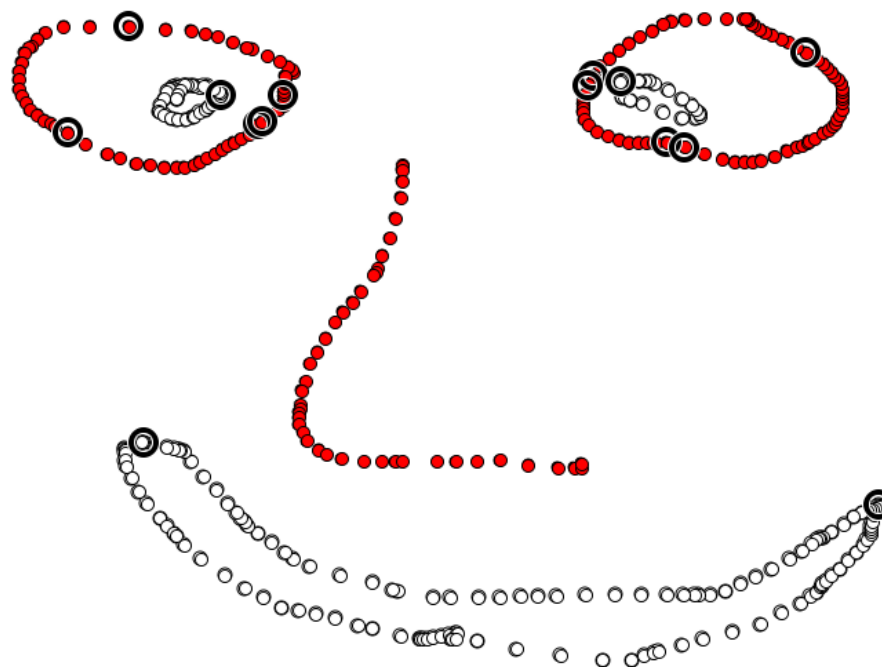
$$\text{and } \xi_i \geq 0, \rho \geq 0.$$

Support Vector Machine for non-separable datasets

ν is an upper bound on the fraction of margin error (i.e. the number of datapoints misclassified in the margin)

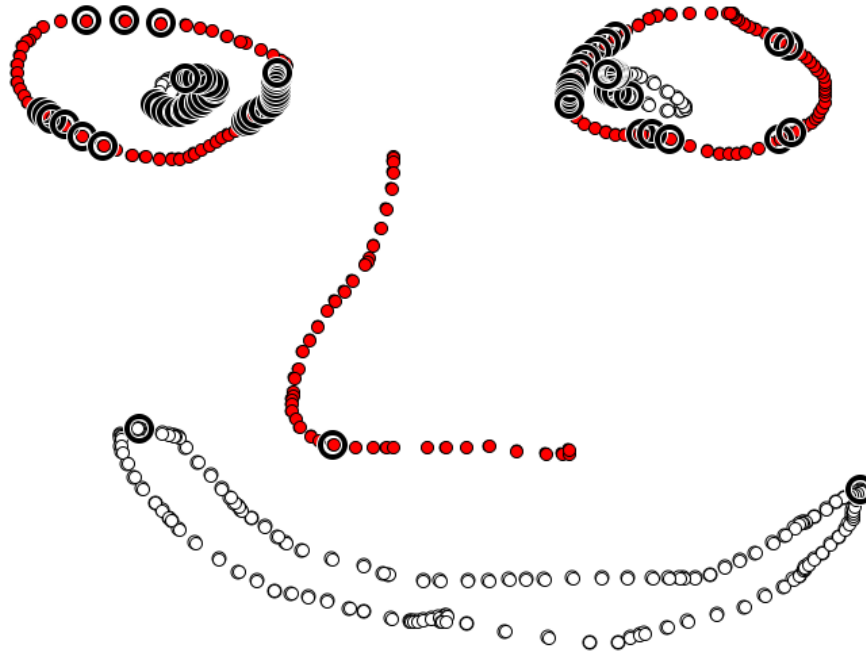
ν is a lower bound on the number of support vectors

Support Vector Machine for non-separable datasets



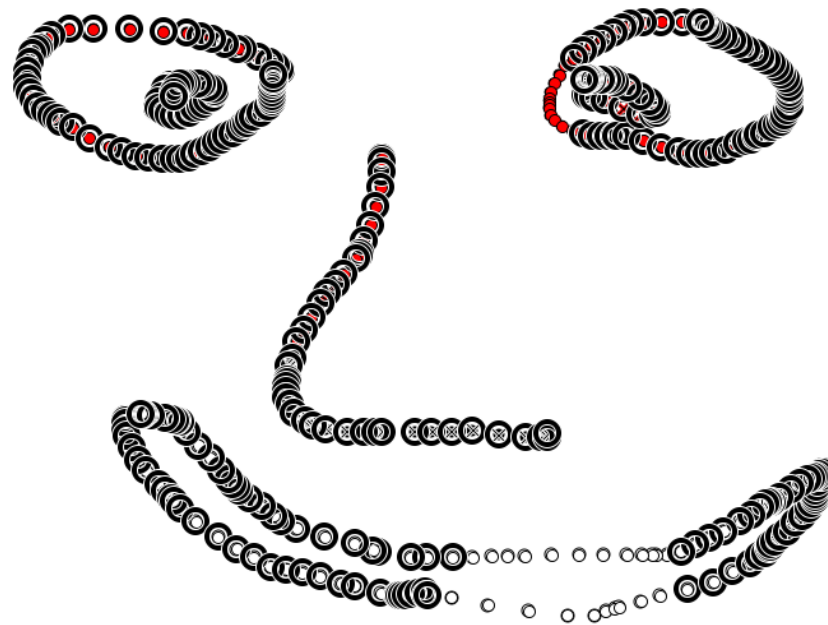
ν -svm $\nu=0.001$, rbf kernel width 0.1

Support Vector Machine for non-separable datasets



Increase in the number of SV-s with $\nu=0.2$

Support Vector Machine for non-separable datasets



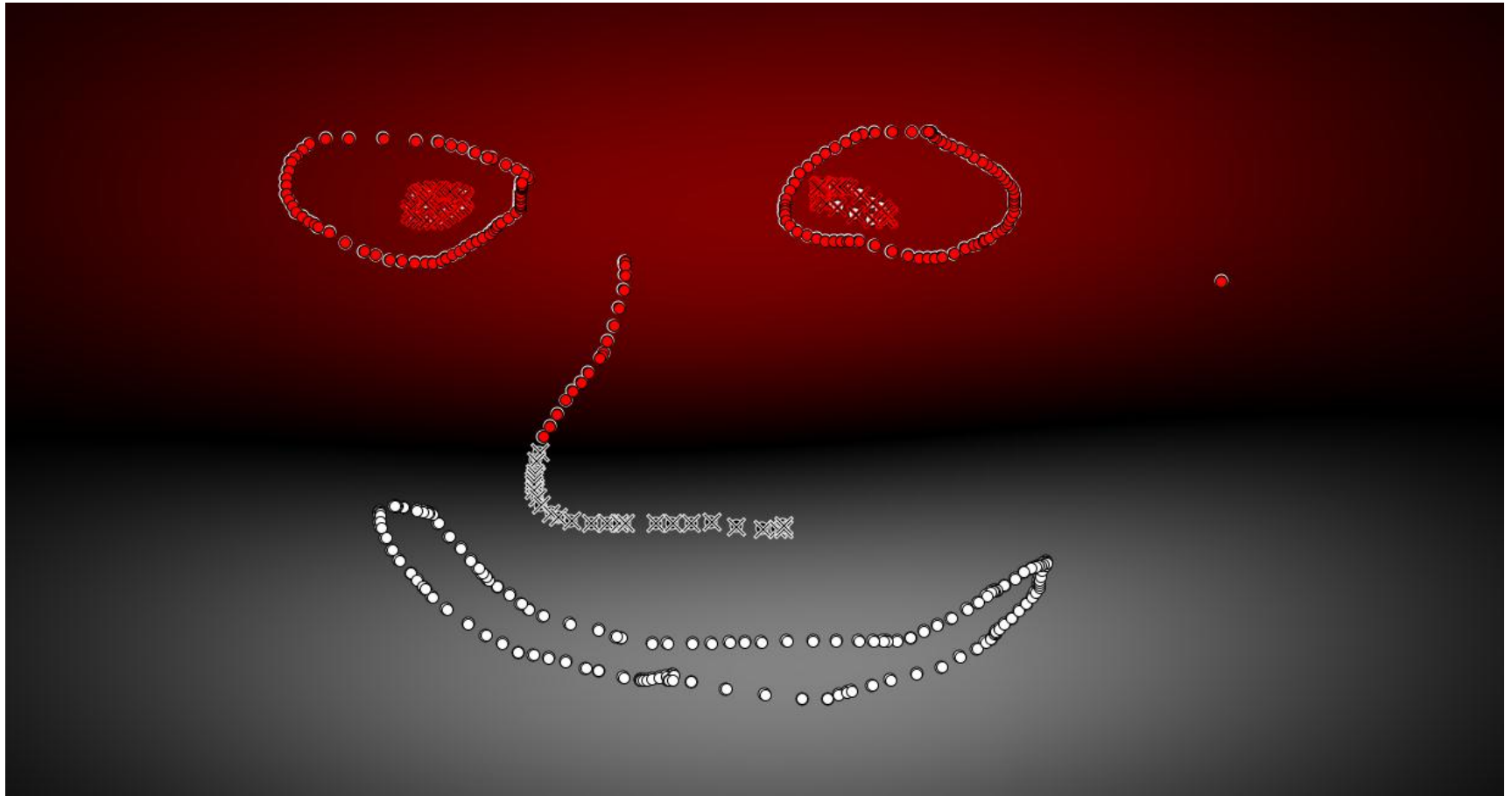
Increase in the number of SV-s with $\nu=0.9$

Support Vector Machine for non-separable datasets



Increase in the error with $\nu=0.2$

Support Vector Machine for non-separable datasets



Increase in the error with $\nu=0.9$

Summary: What is SVM?

- A recently developed learning system (in early 90's, by Vapnik and his coworkers), which could be applied in classification and regression;
- It does the following:
 - **map original input space to higher dimension feature space**, which is implemented implicitly by **kernel function**, such that “linear decision boundaries constructed in the high dimensional feature space correspond to **nonlinear decision boundaries in the input space**”;
 - in feature space, an **optimal separating hyperplane** is constructed, which could be determined by solving an optimization problem;
 - **Lagrange multipliers and dual theory** can then be applied to **convert this optimization problem into a convex quadratic program subject to linear constraints**.
- Advantages:
 - Better generalization
 - Global optimum