Support Vector Machines (SVMs)

Jalal A. Nasiri
Tarbiat Modares University
j.nasiri@modares.ac.ir

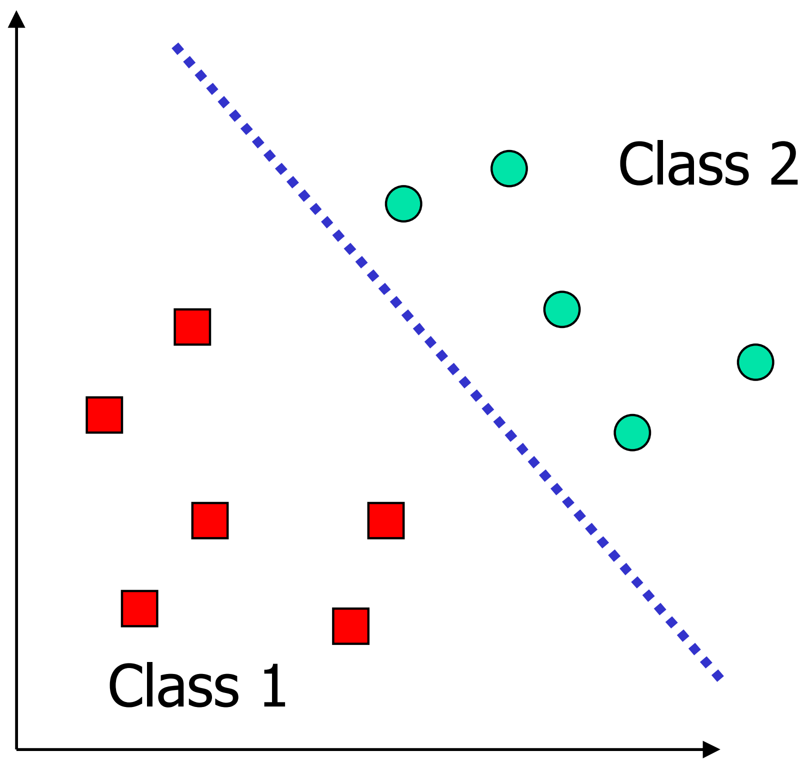
History of SVMs

- SVMs were first introduced in 1995 by V.Vapnik
- SVMs became popular in many various aplications.
- In recent many extended version of SVM such as FSVM,RSVM,FRSVM proposed.
- It has been introduced as a powerful tool for solving classification problems in recent years.

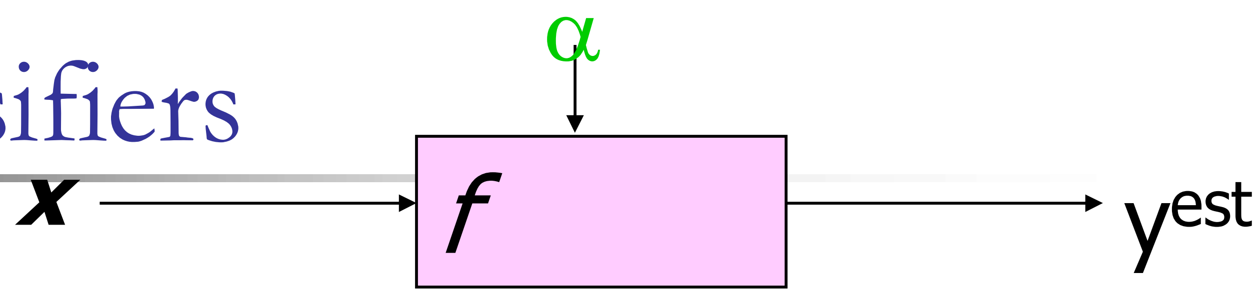
(HAO TANG, LIANG-SHENG QU, 2008)

What is a good Decision Boundary?

- Consider a two-class, linearly separable classification problem
- Many decision boundaries!
  - The Perceptron algorithm can be used to find such a boundary
  - Different algorithms have been proposed for this goal.
- Are all decision boundaries equally good?

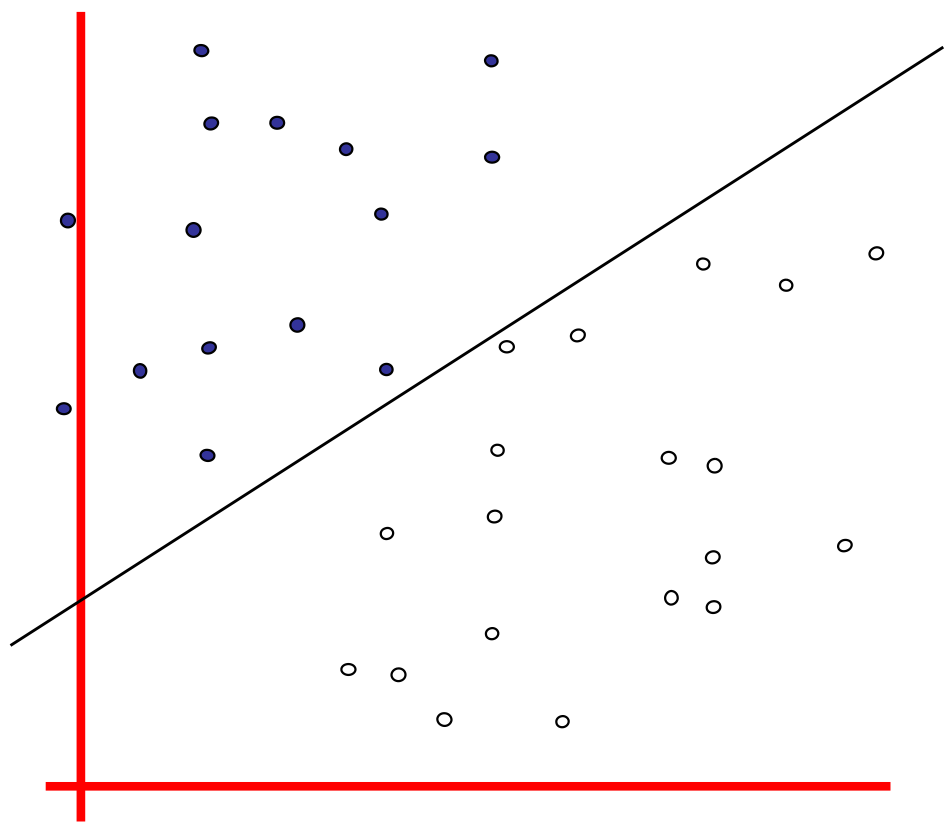


Linear Classifiers



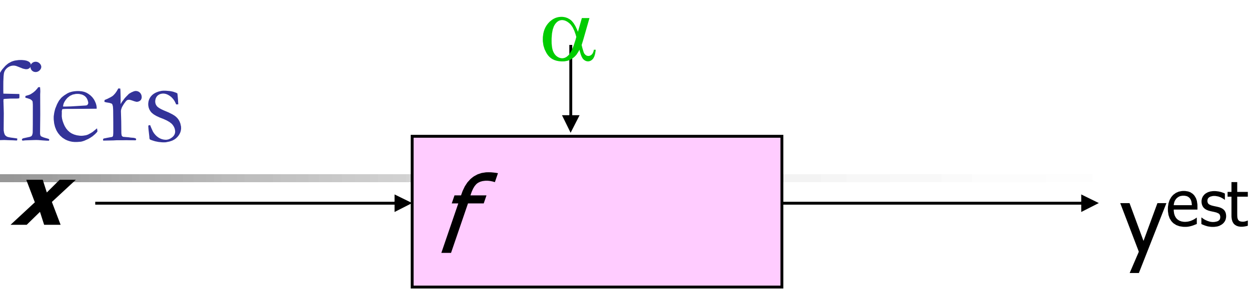
$$f(\mathbf{x},\mathbf{w},b) = sign(\mathbf{w}\ \mathbf{x} + b)$$

- denotes +1
- denotes -1



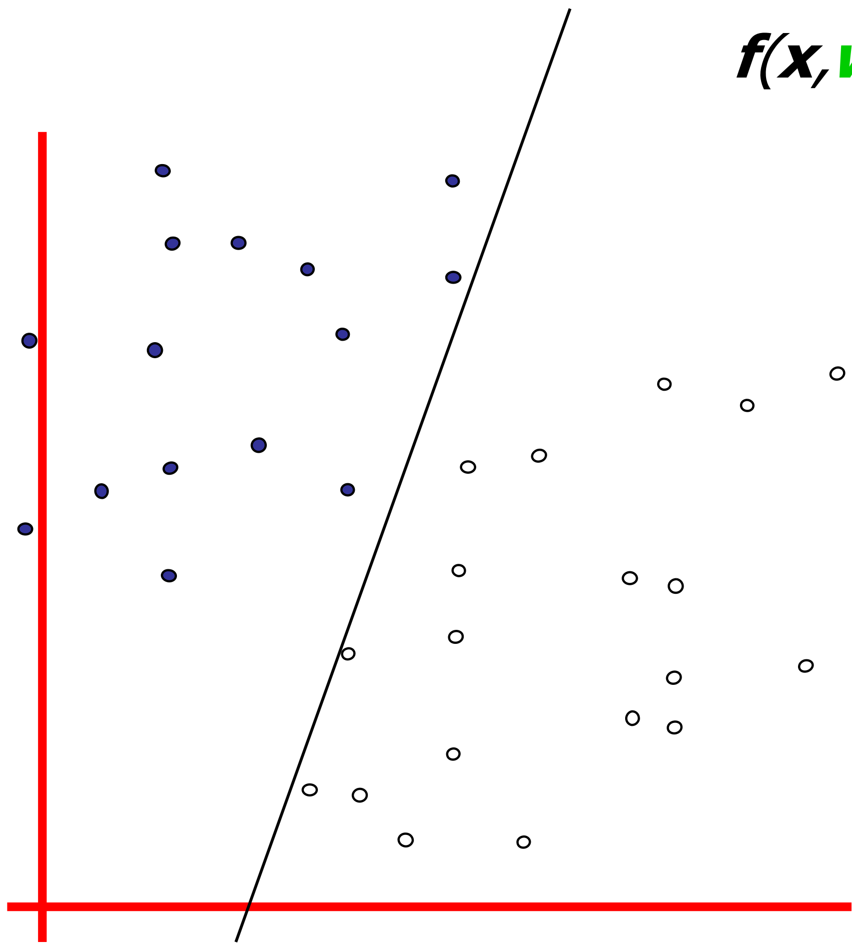
How would you classify this data?

Linear Classifiers



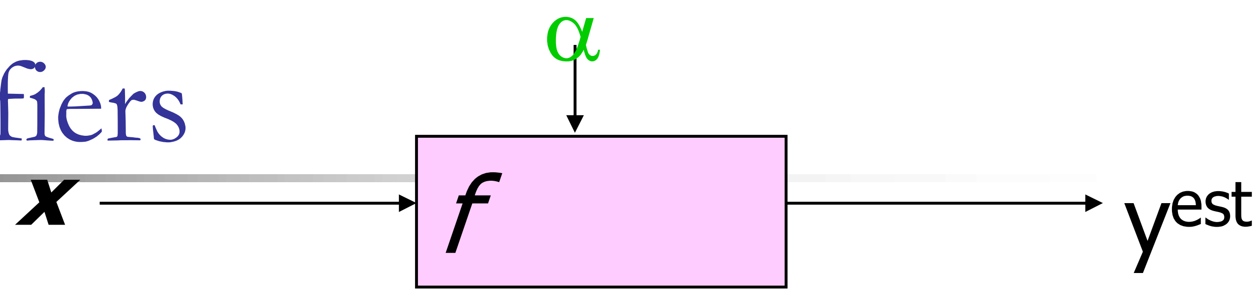
$f(\mathbf{x},\mathbf{w},b) = sign(\mathbf{w}\ \mathbf{x} + b)$

- denotes +1
- denotes -1



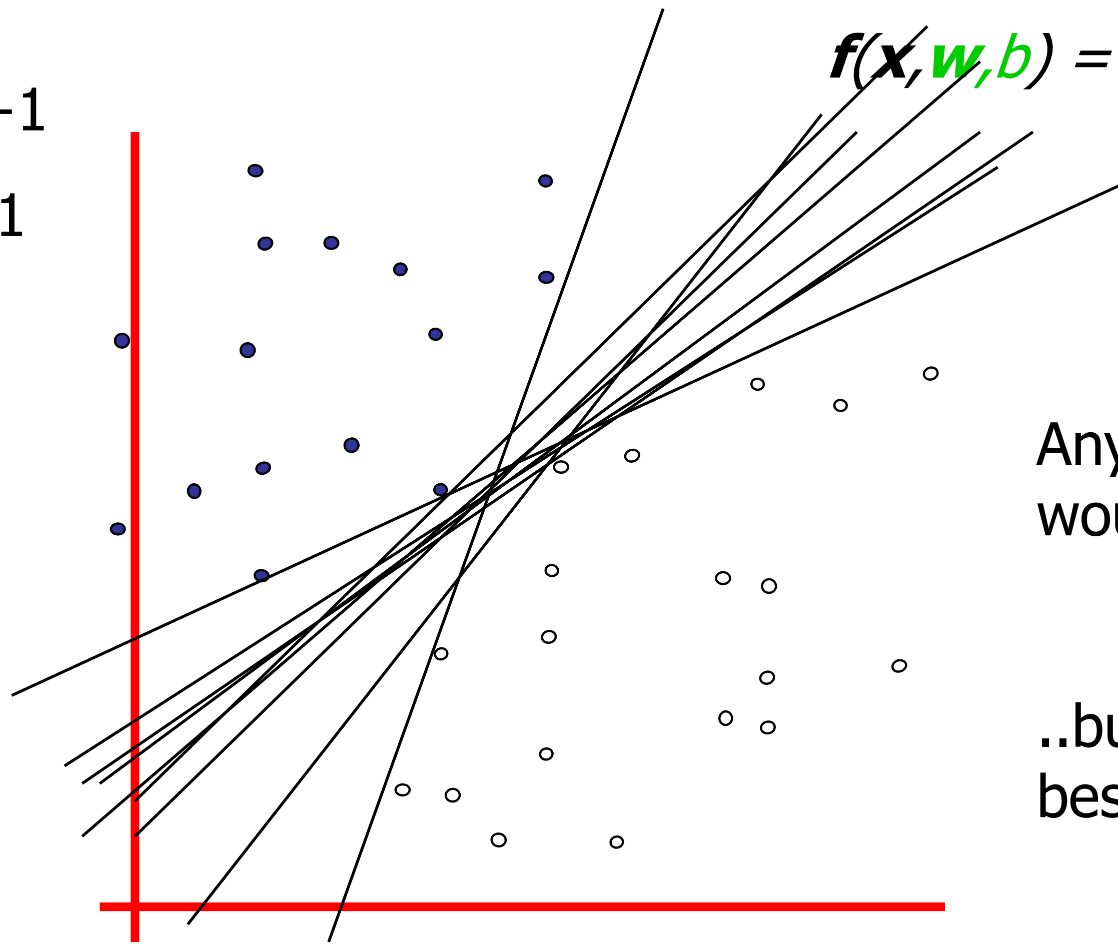
How would you classify this data?

Linear Classifiers



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

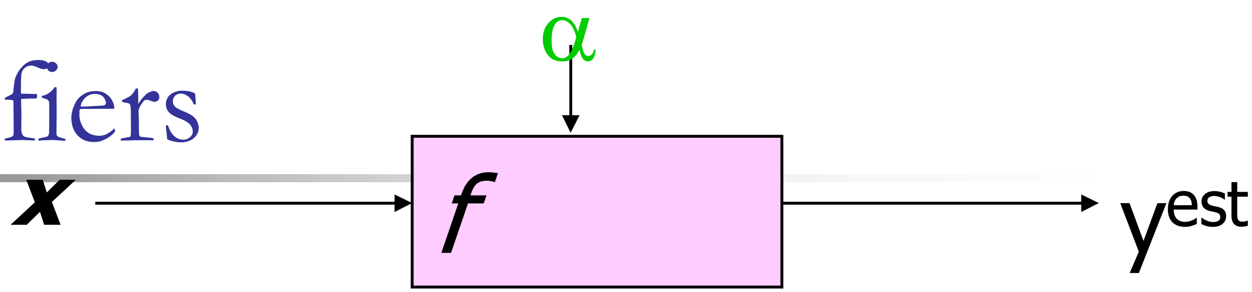
- denotes +1
- denotes -1



Any of these would be fine..

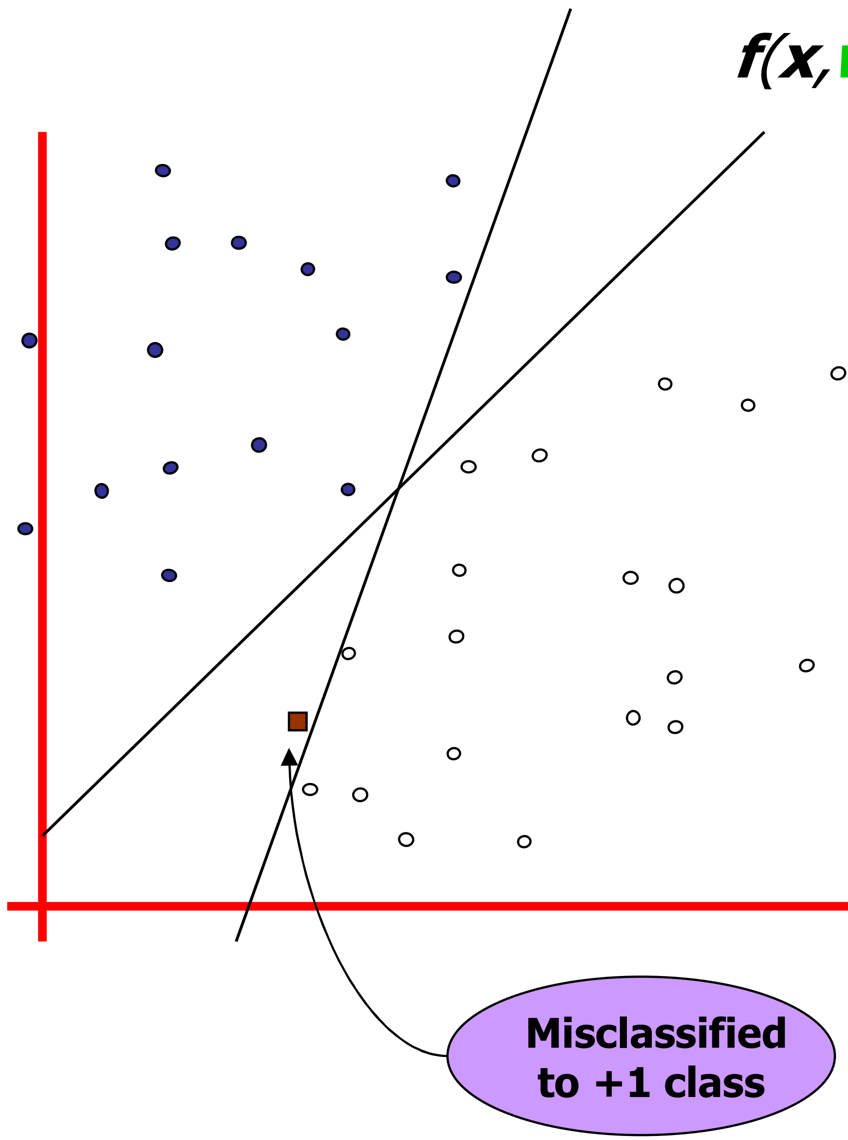
..but which is best?

Linear Classifiers



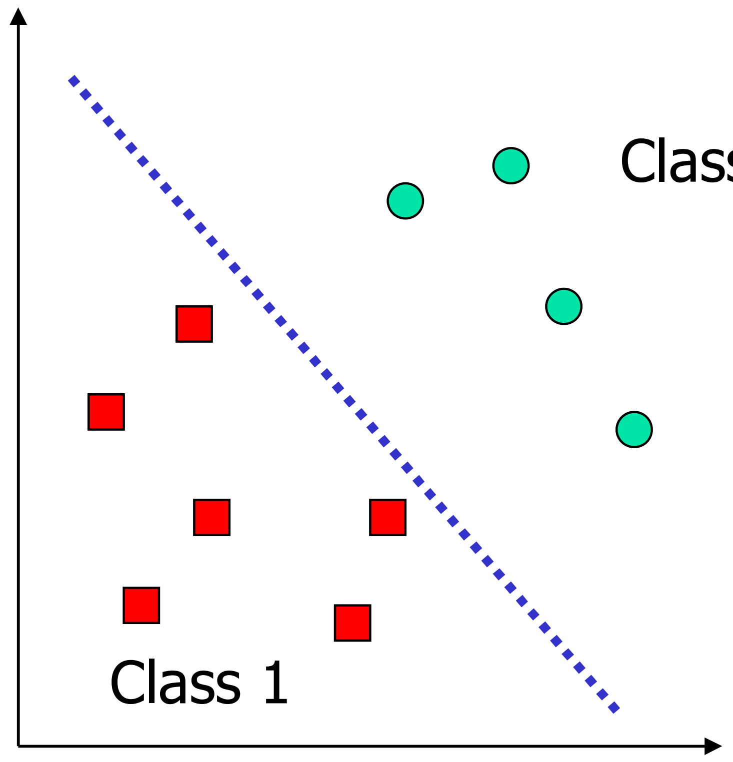
$f(\mathbf{x},\mathbf{w},b) = sign(\mathbf{w}\ \mathbf{x} + b)$

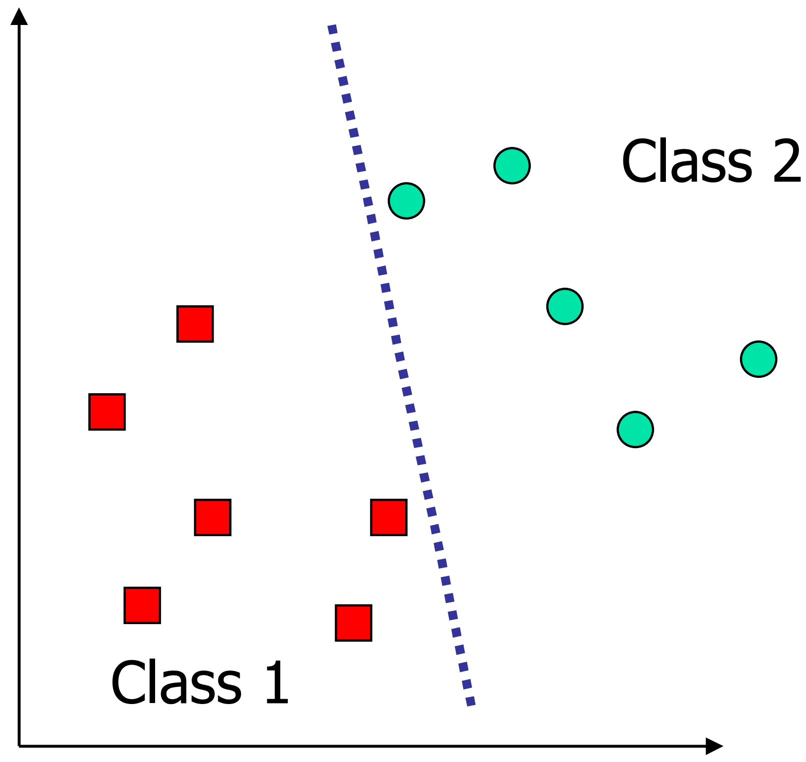
- denotes +1
- denotes -1



How would you classify this data?

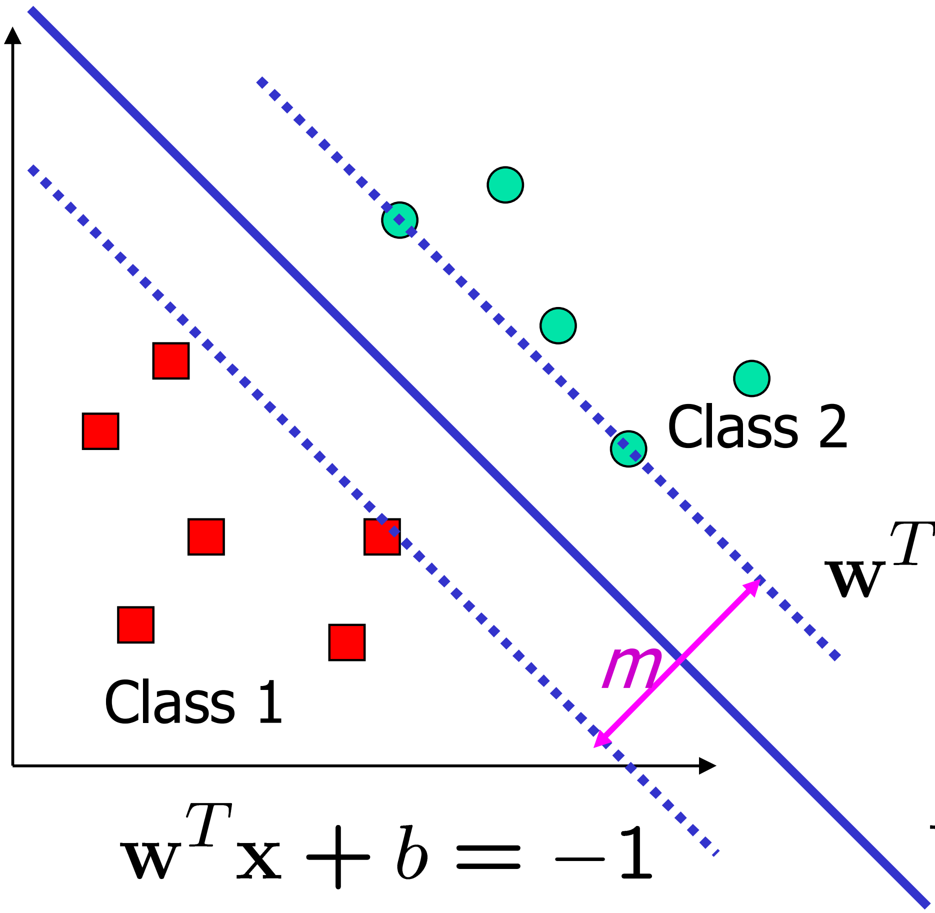
Examples of Bad Decision Boundaries





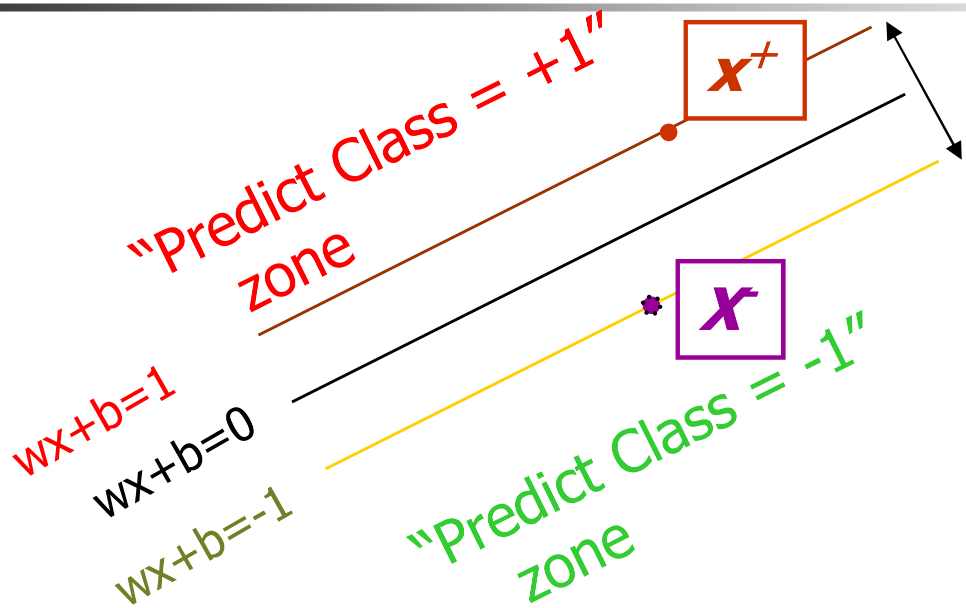
Large-margin Decision Boundary

- The decision boundary should be as far away from the data of both classes as possible
  - We should maximize the margin, *m*



$$m = \frac{2}{||\mathbf{w}||}$$

Linear SVM Mathematically



M=Margin Width

What we know:

- ***w*** *.* ***x***$^+$ *+ b = +1*
- ***w*** *.* ***x***$^-$ *+ b = -1*
- ***w*** *. (****x***$^+$***-x***$^-$*) = 2*

$$M = \frac{(x^+ - x^-)\cdot w}{|w|} = \frac{2}{|w|}$$

Linear SVM Mathematically

- Goal: **1) Correctly classify all training data**

$$wx_i + b \geq 1 \qquad \textit{if } y_i = +1$$

$$wx_i + b \leq -1 \qquad \textit{if } y_i = -1$$

$$y_i(wx_i + b) \geq 1 \qquad \text{for all i}$$

2) Maximize the Margin $M = \frac{2}{|w|}$

same as minimize $\frac{1}{2} w^t w$

- **We can formulate a Quadratic Optimization Problem and solve for w and b**

- Minimize $\Phi(w) = \frac{1}{2} w^t w$

  subject to $y_i(wx_i + b) \geq 1 \quad \forall i$

Finding the Decision Boundary

- Let $\{x_1, ..., x_n\}$ be our data set and let $y_i \in \{1,-1\}$ be the class label of x_i
- The decision boundary should classify all points correctly $\Rightarrow \quad y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1, \qquad \forall i$
- The decision boundary can be found by solving the following constrained optimization problem

$$\text{Minimize } \frac{1}{2}||\mathbf{w}||^2$$
$$\text{subject to } y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 \qquad \forall i$$

- This is a constrained optimization problem. Solving it requires some new tools.

Recap of Constrained Optimization

- The case for inequality constraint $g_i(\mathbf{x}) \leq 0$ is similar, except that the Lagrange multiplier α_i should be positive
- If $\mathbf{x}_0$ is a solution to the constrained optimization problem

$$\min_{\mathbf{x}} f(\mathbf{x}) \quad \text{subject to} \quad g_i(\mathbf{x}) \leq 0 \quad \text{for } i = 1, \ldots, m$$

- There must exist $\alpha_i \geq 0$ for i=1, …, m such that $\mathbf{x}_0$ satisfy

$$\begin{cases} \frac{\partial}{\partial \mathbf{x}} \left(f(\mathbf{x}) + \sum_i \alpha_i g_i(\mathbf{x}) \right) \Big|_{\mathbf{x}=jx_0} = \mathbf{0} \\ g_i(\mathbf{x}) \leq 0 \quad \text{for } i = 1, \ldots, m \end{cases}$$

- The function $f(\mathbf{x}) + \sum_i \alpha_i g_i(\mathbf{x})$ is also known as the Lagrangian; we want to set its gradient to **0**

Back to the Original Problem

$$\text{Minimize } \frac{1}{2}||\mathbf{w}||^2$$

$$\text{subject to } 1-y_i(\mathbf{w}^T\mathbf{x}_i+b) \le 0 \qquad \text{for } i = 1, \ldots, n$$

- The Lagrangian is

$$\mathcal{L} = \frac{1}{2}\mathbf{w}^T\mathbf{w} + \sum_{i=1}^{n} \alpha_i \left(1 - y_i(\mathbf{w}^T\mathbf{x}_i + b)\right)$$

  - Note that $||\mathbf{w}||^2 = \mathbf{w}^T\mathbf{w}$

- Setting the gradient of $\mathcal{L}$ w.r.t. **w** and b to zero, we have

$$\mathbf{w} + \sum_{i=1}^{n} \alpha_i(-y_i)\mathbf{x}_i = \mathbf{0} \quad \Rightarrow \quad \mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i$$

$$\sum_{i=1}^{n} \alpha_i y_i = 0$$

The Dual Problem

- If we substitute $\mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i$ to $\mathcal{L}$, we have

$$\mathcal{L} = \frac{1}{2} \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i^T \sum_{j=1}^{n} \alpha_j y_j \mathbf{x}_j + \sum_{i=1}^{n} \alpha_i \left(1 - y_i \left(\sum_{j=1}^{n} \alpha_j y_j \mathbf{x}_j^T \mathbf{x}_i + b \right) \right)$$

$$= \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j + \sum_{i=1}^{n} \alpha_i - \sum_{i=1}^{n} \alpha_i y_i \sum_{j=1}^{n} \alpha_j y_j \mathbf{x}_j^T \mathbf{x}_i - b \sum_{i=1}^{n} \alpha_i y_i$$

$$= -\frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j + \sum_{i=1}^{n} \alpha_i$$

- Note that $\sum_{i=1}^{n} \alpha_i y_i = 0$

- This is a function of α_i only

The Dual Problem

- The new objective function is in terms of α_i only
- It is known as the dual problem: if we know **w**, we know all α_i; if we know all α_i, we know **w**
- The original problem is known as the primal problem
- The objective function of the dual problem needs to be maximized!
- The dual problem is therefore:

$$\text{max. } W(\boldsymbol{\alpha}) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1,j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$$

$$\text{subject to } \alpha_i \geq 0, \qquad \sum_{i=1}^{n} \alpha_i y_i = 0$$

Properties of α_i when we introduce the Lagrange multipliers

The result when we differentiate the original Lagrangian w.r.t. b

The Dual Problem

$$\text{max. } W(\boldsymbol{\alpha}) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1,j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$$

$$\text{subject to } \alpha_i \geq 0, \sum_{i=1}^{n} \alpha_i y_i = 0$$

- This is a quadratic programming (QP) problem
  - A global maximum of α_i can always be found
- **w** can be recovered by $\mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i$

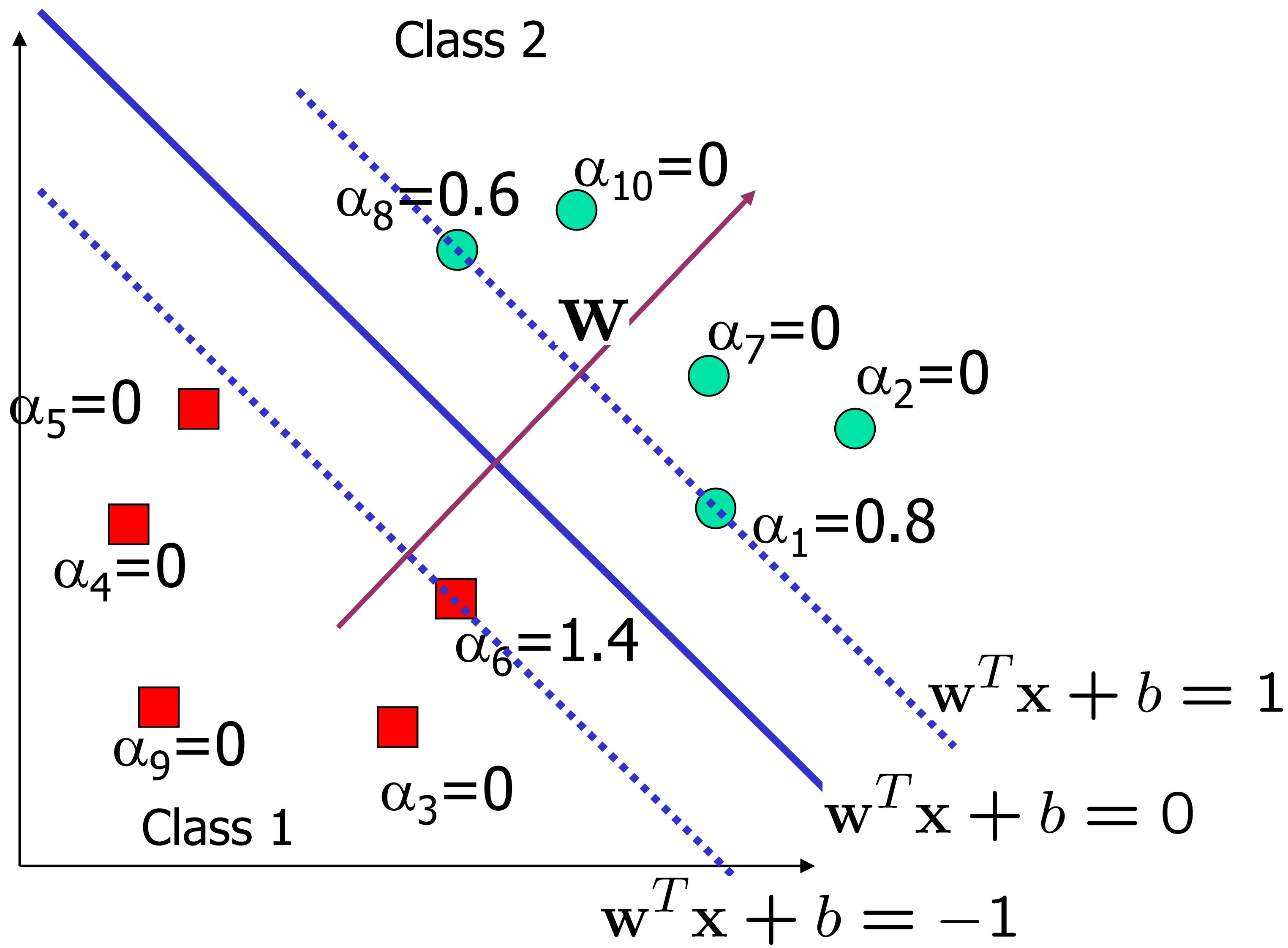
Characteristics of the Solution

- Many of the α_i are zero
  - **w** is a linear combination of a small number of data points
  - This "sparse" representation can be viewed as data compression as in the construction of knn classifier
- $\mathbf{x}_i$ with non-zero α_i are called support vectors (SV)
  - The decision boundary is determined only by the SV
  - Let t_j (j=1, ..., s) be the indices of the s support vectors. We can write $\mathbf{w} = \sum_{j=1}^{s} \alpha_{t_j} y_{t_j} \mathbf{x}_{t_j}$
- For testing with a new data **z**
  - Compute $\mathbf{w}^T\mathbf{z} + b = \sum_{j=1}^{s} \alpha_{t_j} y_{t_j} (\mathbf{x}_{t_j}^T \mathbf{z}) + b$ and
  - classify **z** as class 1 if the sum is positive, and class 2 otherwise
  - Note: **w** need not be formed explicitly

The Quadratic Programming Problem

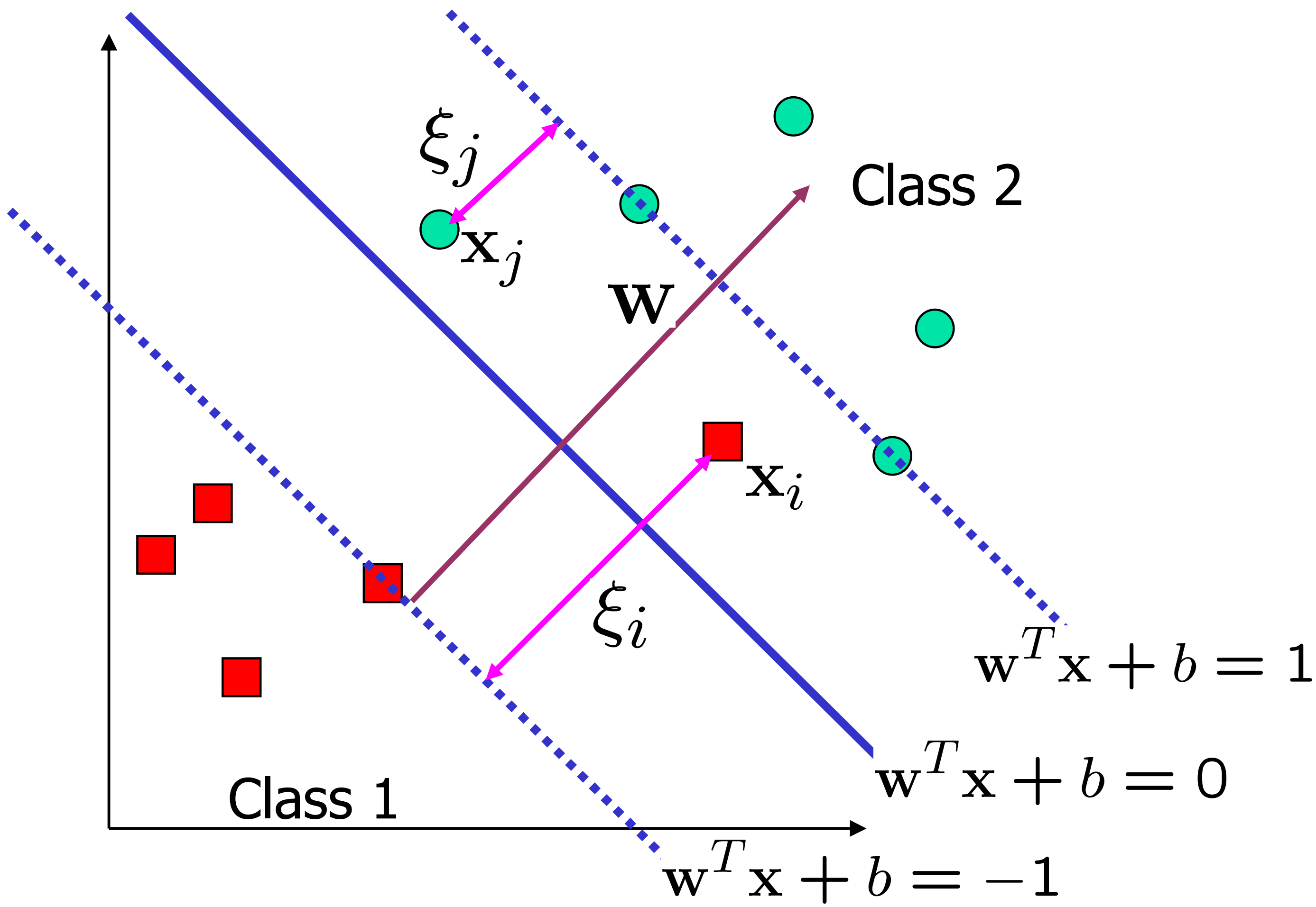
- Many approaches have been proposed
  - Loqo, cplex, etc. (see http://www.numerical.rl.ac.uk/qp/qp.html)
- Most are "interior-point" methods
  - Start with an initial solution that can violate the constraints
  - Improve this solution by optimizing the objective function and/or reducing the amount of constraint violation
- For SVM, sequential minimal optimization (SMO) seems to be the most popular
  - A QP with two variables is trivial to solve
  - Each iteration of SMO picks a pair of (α_i,α_j) and solve the QP with these two variables; repeat until convergence
- In practice, we can just regard the QP solver as a "black-box" without bothering how it works

A Geometrical Interpretation



Non-linearly Separable Problems

- We allow "error" ξ_i in classification; it is based on the output of the discriminant function $\mathbf{w}^T\mathbf{x}$+b
- ξ_i approximates the distance of misclassified samples



Soft Margin Hyperplane

- If we minimize $\sum_i \xi_i$, ξ_i can be computed by

$$\begin{cases} \mathbf{w}^T\mathbf{x}_i + b \geq 1 - \xi_i & y_i = 1 \\ \mathbf{w}^T\mathbf{x}_i + b \leq -1 + \xi_i & y_i = -1 \\ \xi_i \geq 0 & \forall i \end{cases}$$

  - ξ_i are "slack variables" in optimization
  - Note that ξ_i=0 if there is no error for $\mathbf{x}_i$
  - ξ_i is an upper bound of the number of errors
- We want to minimize $\frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{n}\xi_i$
  - *C*: tradeoff parameter between error and margin
- The optimization problem becomes

$$\text{Minimize } \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{n}\xi_i$$
$$\text{subject to } y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0$$

The Optimization Problem

- The dual of this new constrained optimization problem is

$$\text{max. } W(\boldsymbol{\alpha}) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1,j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$$

$$\text{subject to } C \geq \alpha_i \geq 0, \sum_{i=1}^{n} \alpha_i y_i = 0$$

Upper bound

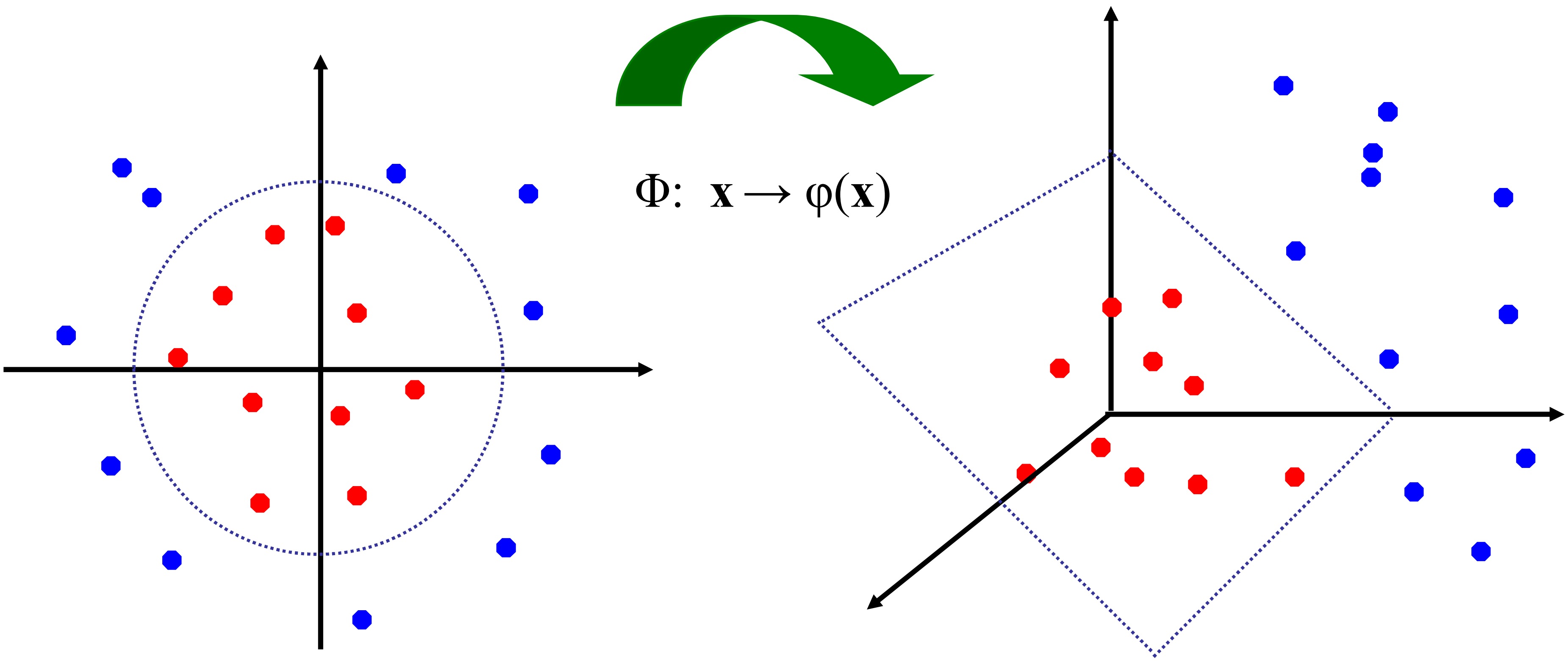
- **w** is recovered as $\mathbf{w} = \sum_{j=1}^{s} \alpha_{t_j} y_{t_j} \mathbf{x}_{t_j}$
- This is very similar to the optimization problem in the linear separable case, except that there is an upper bound *C* on α_i now
- Once again, a QP solver can be used to find α_i

Extension to Non-linear Decision Boundary

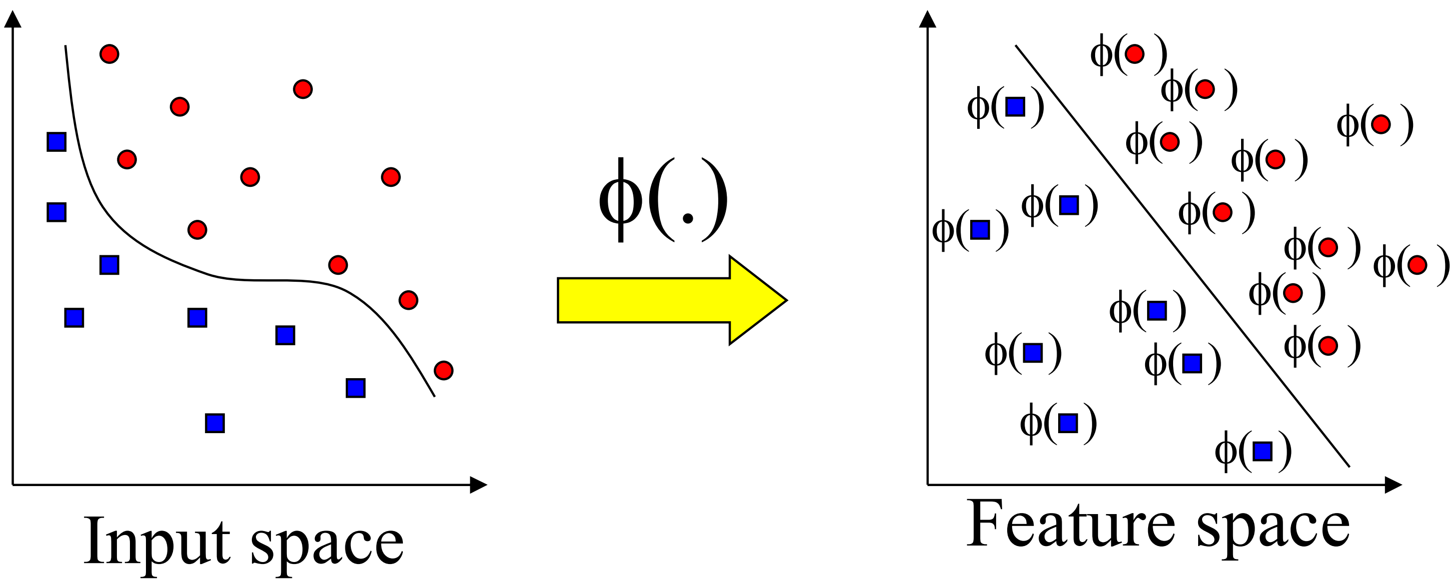
- So far, we have only considered large-margin classifier with a linear decision boundary
- How to generalize it to become nonlinear?
- Key idea: transform $\mathbf{x}_i$ to a higher dimensional space to "make life easier"
  - Input space: the space the point $\mathbf{x}_i$ are located
  - Feature space: the space of $\phi(\mathbf{x}_i)$ after transformation
- Why transform?
  - Linear operation in the feature space is equivalent to non-linear operation in input space
  - Classification can become easier with a proper transformation. In the XOR problem, for example, adding a new feature of $x_1 x_2$ make the problem linearly separable

Non-linear SVMs: Feature spaces

- General idea: the original input space can always be mapped to some higher-dimensional feature space where the training set is separable:



Transforming the Data



Note: feature space is of higher dimension than the input space in practice

- Computation in the feature space can be costly because it is high dimensional
  - The feature space is typically infinite-dimensional!
- The kernel trick comes to rescue

The Kernel Trick

- Recall the SVM optimization problem

$$\text{max. } W(\boldsymbol{\alpha}) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1,j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$$

$$\text{subject to } C \geq \alpha_i \geq 0, \sum_{i=1}^{n} \alpha_i y_i = 0$$

- The data points only appear as inner product
- As long as we can calculate the inner product in the feature space, we do not need the mapping explicitly
- Many common geometric operations (angles, distances) can be expressed by inner products
- Define the kernel function *K* by

$$K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$$

An Example for $\phi(.)$ and $K(.,.)$

- Suppose $\phi(.)$ is given as follows

$$\phi(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}) = (1, \sqrt{2}x_1, \sqrt{2}x_2, x_1^2, x_2^2, \sqrt{2}x_1x_2)$$

- An inner product in the feature space is

$$\langle \phi(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}), \phi(\begin{bmatrix} y_1 \\ y_2 \end{bmatrix}) \rangle = (1 + x_1y_1 + x_2y_2)^2$$

- So, if we define the kernel function as follows, there is no need to carry out $\phi(.)$ explicitly

$$K(\mathbf{x}, \mathbf{y}) = (1 + x_1y_1 + x_2y_2)^2$$

- This use of kernel function to avoid carrying out $\phi(.)$ explicitly is known as the kernel trick

Kernel Functions

- In practical use of SVM, the user specifies the kernel function; the transformation $\phi(.)$ is not explicitly stated
- Another view: kernel function, being an inner product, is really a similarity measure between the objects

Examples of Kernel Functions

- Polynomial kernel with degree *d*

$$K(\mathbf{x},\mathbf{y}) = (\mathbf{x}^T\mathbf{y}+1)^d$$

- Radial basis function kernel with width σ

$$K(\mathbf{x},\mathbf{y}) = \exp(-||\mathbf{x}-\mathbf{y}||^2/(2\sigma^2))$$

  - Closely related to radial basis function neural networks
  - The feature space is infinite-dimensional
- Sigmoid with parameter κ and θ

$$K(\mathbf{x},\mathbf{y}) = \tanh(\kappa\mathbf{x}^T\mathbf{y}+\theta)$$

  - It does not satisfy the Mercer condition on all κ and θ

Modification Due to Kernel Function

- Change all inner products to kernel functions
- For training,

Original

$$\text{max. } W(\boldsymbol{\alpha}) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1,j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$$

$$\text{subject to } C \geq \alpha_i \geq 0, \sum_{i=1}^{n} \alpha_i y_i = 0$$

With kernel function

$$\text{max. } W(\boldsymbol{\alpha}) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1,j=1}^{n} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)$$

$$\text{subject to } C \geq \alpha_i \geq 0, \sum_{i=1}^{n} \alpha_i y_i = 0$$

Modification Due to Kernel Function

- For testing, the new data **z** is classified as class 1 if $f \geq 0$, and as class 2 if $f < 0$

Original

$$\mathbf{w} = \sum_{j=1}^{s} \alpha_{t_j} y_{t_j} \mathbf{x}_{t_j}$$

$$f = \mathbf{w}^T \mathbf{z} + b = \sum_{j=1}^{s} \alpha_{t_j} y_{t_j} \mathbf{x}_{t_j}^T \mathbf{z} + b$$

With kernel function

$$\mathbf{w} = \sum_{j=1}^{s} \alpha_{t_j} y_{t_j} \phi(\mathbf{x}_{t_j})$$

$$f = \langle \mathbf{w}, \phi(\mathbf{z}) \rangle + b = \sum_{j=1}^{s} \alpha_{t_j} y_{t_j} K(\mathbf{x}_{t_j}, \mathbf{z}) + b$$

More on Kernel Functions

- Since the training of SVM only requires the value of $K(\mathbf{x}_i, \mathbf{x}_j)$, there is no restriction of the form of $\mathbf{x}_i$ and $\mathbf{x}_j$
  - $\mathbf{x}_i$ can be a sequence or a tree, instead of a feature vector
- $K(\mathbf{x}_i, \mathbf{x}_j)$ is just a similarity measure comparing $\mathbf{x}_i$ and $\mathbf{x}_j$
- For a test object **z**, the discriminat function essentially is a weighted sum of the similarity between z and a pre-selected set of objects (the support vectors)

$$f(\mathbf{z}) = \sum_{\mathbf{x}_i \in \mathcal{S}} \alpha_i y_i K(\mathbf{z}, \mathbf{x}_i) + b$$

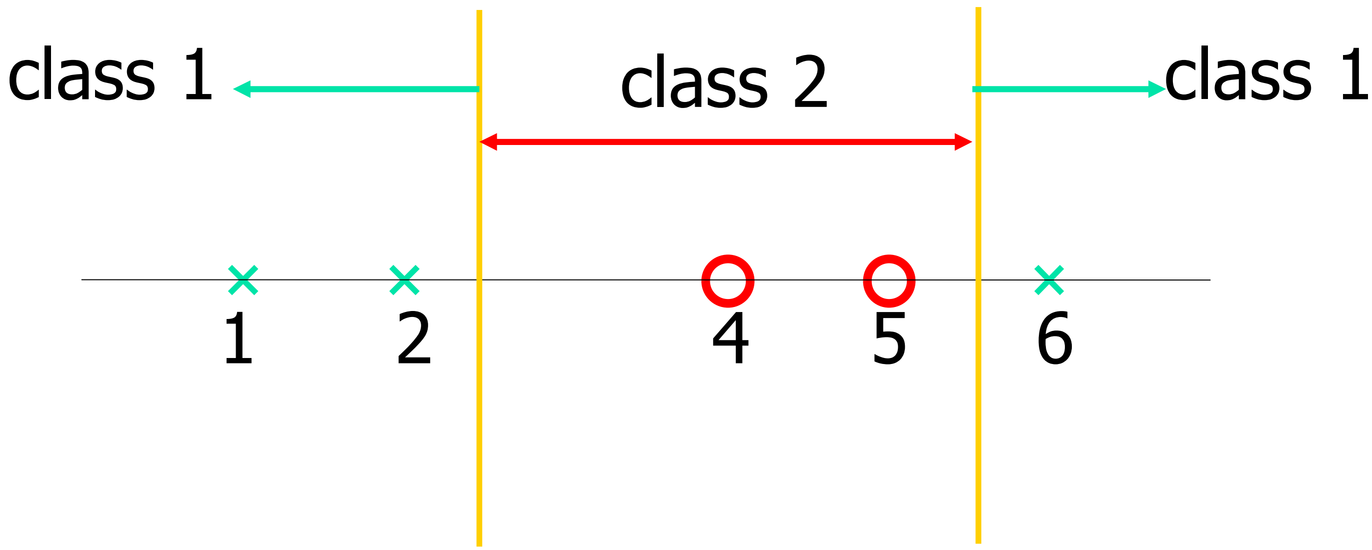
$\mathcal{S}$: the set of support vectors

More on Kernel Functions

- Not all similarity measure can be used as kernel function, however
  - The kernel function needs to satisfy the Mercer function, i.e., the function is "positive-definite"
  - This implies that the *n* by *n* kernel matrix, in which the (i,j)-th entry is the $K(\mathbf{x}_i, \mathbf{x}_j)$, is always positive definite
  - This also means that the QP is convex and can be solved in polynomial time

Example

- Suppose we have 5 1D data points
  $x_1=1$, $x_2=2$, $x_3=4$, $x_4=5$, $x_5=6$,
  with 1, 2, 6 as class 1 and
  4, 5 as class 2
  $\Rightarrow y_1=1, y_2=1, y_3=-1, y_4=-1, y_5=1$



Our samples and labels

Example

- We use the polynomial kernel of degree 2
  - $K(x,y) = (xy+1)^2$
  - C is set to 100
- We first find α_i (i=1, ..., 5) by

$$\text{max.} \quad \sum_{i=1}^{5} \alpha_i - \frac{1}{2} \sum_{i=1}^{5} \sum_{j=1}^{5} \alpha_i \alpha_j y_i y_j (x_i x_j + 1)^2$$

$$\text{subject to } 100 \geq \alpha_i \geq 0, \sum_{i=1}^{5} \alpha_i y_i = 0$$

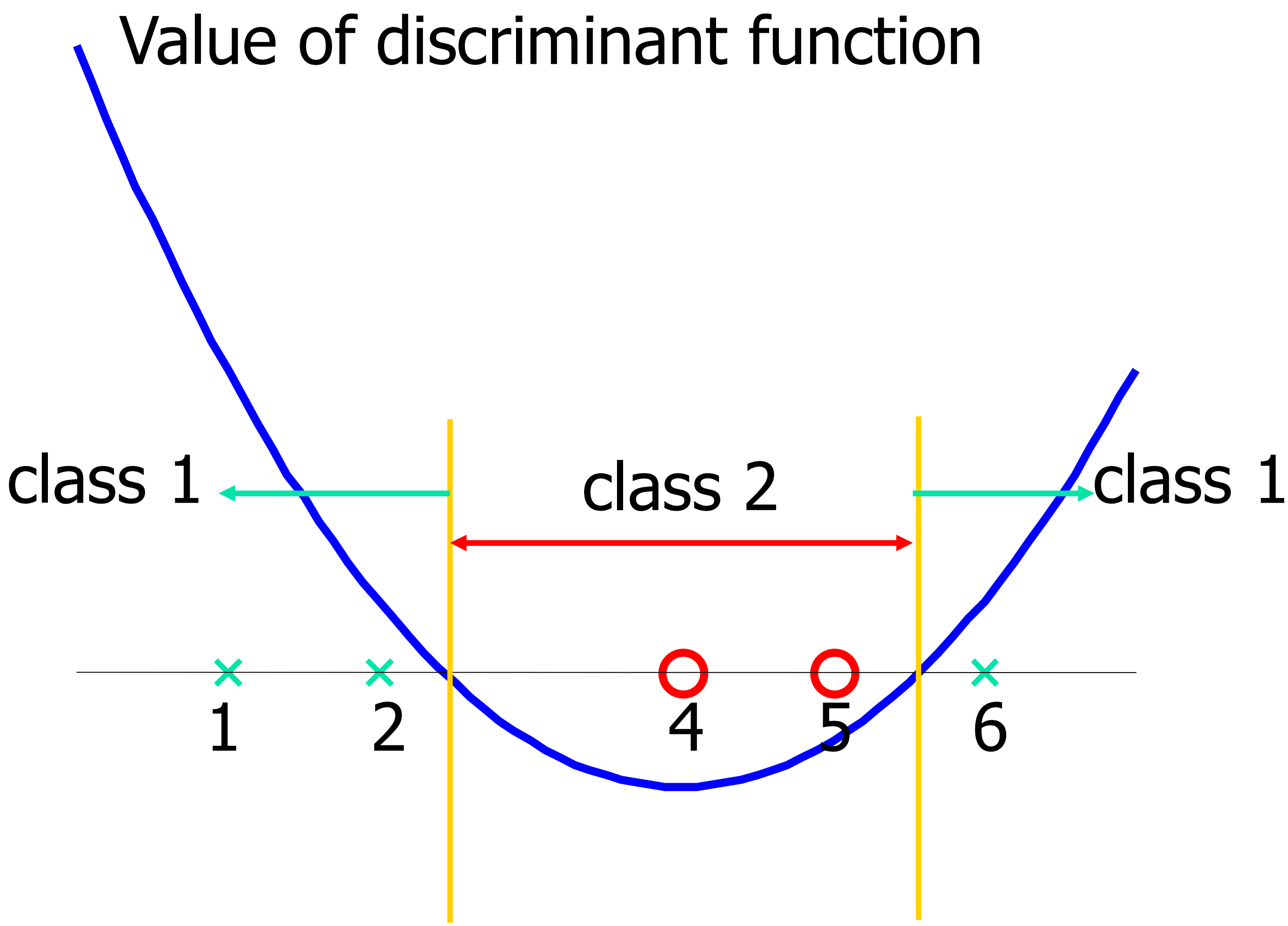
Example

- By using a QP solver, we get
  - $\alpha_1=0$, $\alpha_2=2.5$, $\alpha_3=0$, $\alpha_4=7.333$, $\alpha_5=4.833$
  - Note that the constraints are indeed satisfied
  - The support vectors are $\{x_2=2, x_4=5, x_5=6\}$
- The discriminant function is

$$
\begin{aligned}
& f(z) \\
&= 2.5(1)(2z+1)^2 + 7.333(-1)(5z+1)^2 + \overbrace{4.833}^{\alpha_5}(\overbrace{1}^{y_5})\overbrace{(6z+1)^2}^{K(z,x_5)} + b \\
&= 0.6667z^2 - 5.333z + b
\end{aligned}
$$

- *b* is recovered by solving f(2)=1 or by f(5)=-1 or by f(6)=1, as x_2 and x_5 lie on the line $\phi(\mathbf{w})^T\phi(\mathbf{x})+b=1$ and x_4 lies on the line $\phi(\mathbf{w})^T\phi(\mathbf{x})+b=-1$
- All three give b=9 $\Rightarrow$ $f(z)=0.6667z^2-5.333z+9$

Example



Value of discriminant function
class 1
class 2
class 1
1
2
4
5
6

Choosing the Kernel Function

- Probably the most tricky part of using SVM.
- The kernel function is important because it creates the kernel matrix, which summarizes all the data
- Many principles have been proposed (diffusion kernel, Fisher kernel, string kernel, …)
- There is even research to estimate the kernel matrix from available information
- In practice, a low degree polynomial kernel or RBF kernel with a reasonable width is a good initial try
- Note that SVM with RBF kernel is closely related to RBF neural networks

Summary: Steps for Classification

- Prepare the pattern matrix
- Select the kernel function to use
- Select the parameter of the kernel function and the value of *C*
- Execute the training algorithm and obtain the α_i
- Unseen data can be classified using the α_i and the support vectors

Strengths and Weaknesses of SVM

- Strengths
  - Training is relatively easy
    - No local optimal, unlike in neural networks
  - It scales relatively well to high dimensional data
  - Tradeoff between classifier complexity and error can be controlled explicitly
- Weaknesses
  - Need to choose a “good” kernel function
  - Since the optimal hyperplane obtained by the SVM depends on only a small part of the data points, it may become sensitive to noises or outliers in the training set

Fuzzy SVM(2002)IEEE Transaction on Neural Network

- SVM is sensitive to noises or outliers in the training set.
- each training point no more exactly belongs to one of the two classes. It may 90% belong to one class and 10% be meaningless.
- In FSVM there is a fuzzy membership associated with each training point.

$$Minimize\ \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n} s_i\xi_i$$

$$subject\ to\ y_i(w^T.z_i + b) \geq 1 - \xi_i,\ \ i = 1, \dots, n$$

$$\xi_i \geq 0,\quad i = 1, \dots, n$$

Fuzzy membership

- Various method consider for fuzzy member ship.
- The original FSVM used as follow:

$$s_i = \begin{cases} 1 - (\|x_+ - x_i\|/(r_+ + \delta)) & \text{if } x_i \in \text{class1} \\ 1 - (\|x_- - x_i\|/(r_- + \delta)) & \text{if } x_i \in \text{class2} \end{cases}$$

- Where x_+ is mean of class1
- And r_+ is max radius of class1

$$r_+ = \max_{\{x_i : x_i \in \text{Class1}\}} \|x_+ - x_i\|$$

$$r_- = \max_{\{x_i : x_i \in \text{Class2}\}} \|x_- - x_i\|$$

FSVM

- We construct the lagrangian:

$$L(w,b,\xi,\alpha,\beta)=\frac{1}{2}\|w\|^2+C\sum_{i=1}^{n}s_i\xi_i-\sum_{i=1}^{n}\alpha_i(y_i(w.z_i+b)-1+\xi_i)-\sum_{i=1}^{n}\beta_i\xi_i$$

- The parameters must satisfy the following conditions:

$$\frac{\partial L(w,b,\xi,\alpha,\beta)}{\partial w}=w-\sum_{i=1}^{n}\alpha_i y_i\, z_i=0$$

$$\frac{\partial L(w,b,\xi,\alpha,\beta)}{\partial b}=-\sum_{i=1}^{n}\alpha_i y_i=0$$

$$\frac{\partial L(w,b,\xi,\alpha,\beta)}{\partial b}=s_iC-\alpha_i-\beta_i=0.$$

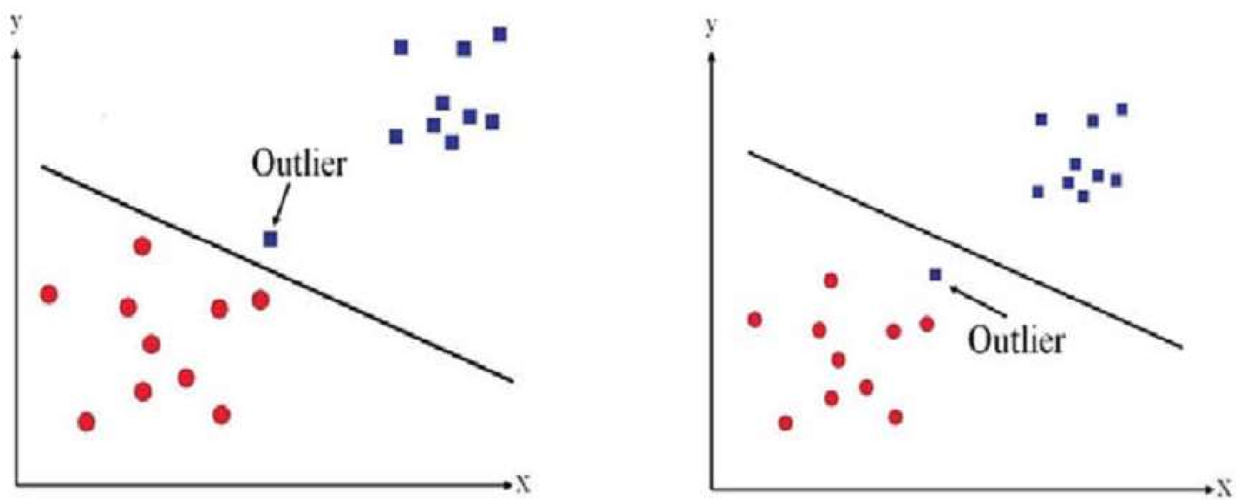
- The dual of this new constrained optimization problem is

$$Maximize\ L(\alpha)=\sum_{i=1}^{n}\alpha_i-\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j K(x_i,x_j)$$

$$subject\ to\ \sum_{i=1}^{n}y_i\alpha_i=0,\quad 0\le\alpha_i\le s_iC,\quad i=1,\ldots n$$

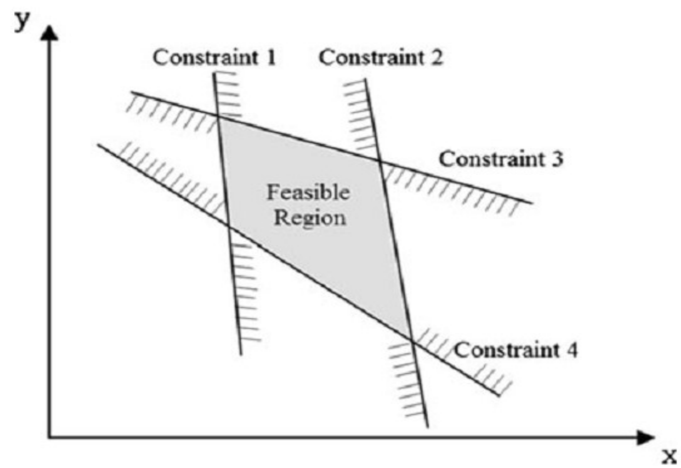
FSVM

- The membership with low value means this data is not very important then in optimization may be have low effect.
- In original type “slack variables” didn’t have any user knowledge but in FSVM we help optimizer with extra information.



Relax Constraint SVM(2010)Neural computing & application

- Constrains play a major role in finding answer.



- In subject of RSVM use fuzzy instead of crisp for relaxation

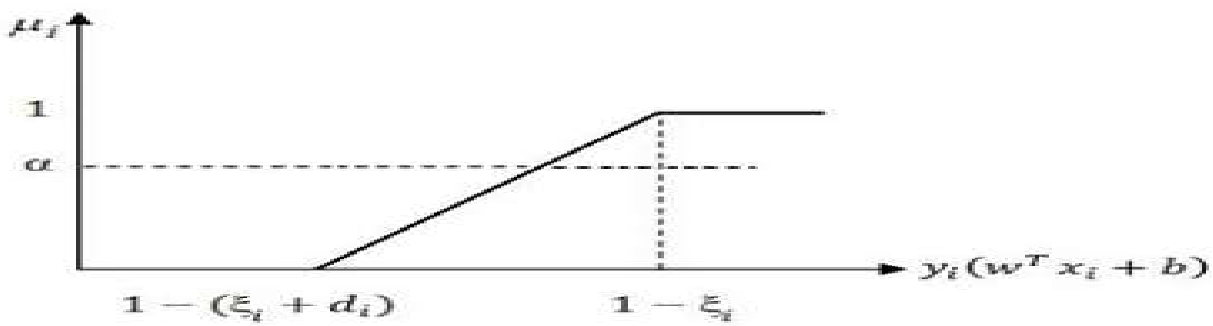
$$Minimize\ Q(w,b,\xi) = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i$$

$$subject\ to\ \ y_i(w^T x_i + b) \gtrsim 1 - \xi_i, \quad i = 1, \dots, n$$

$$\xi_i \geq 0, i = 1, \dots, n$$

RSVM parameter

- Linear membership functions for fuzzy greater than or equal inequality can be defined as follows:



- In other word:

$$\mu_i : \Re^{m+1+n} \rightarrow [0,1),\ i = 1,2,\ldots,n,$$

$$\mu_i(w,b,\xi)$$

$$= \begin{cases} 1, & \text{if } \ y_i(w^T x_i + b) \geq 1 - \xi_i \\ \frac{(w^T x_i + b) - 1 + \xi_i + d_i}{d_i}, & \text{if } \ 1 - (\xi_i + d_i) \leq y_i(w^T x_i + b) < 1 - \xi_i \\ 0, & \text{if } \ y_i(w^T x_i + b) < 1 - (\xi_i + d_i) \end{cases}$$

RSVM optimization problem

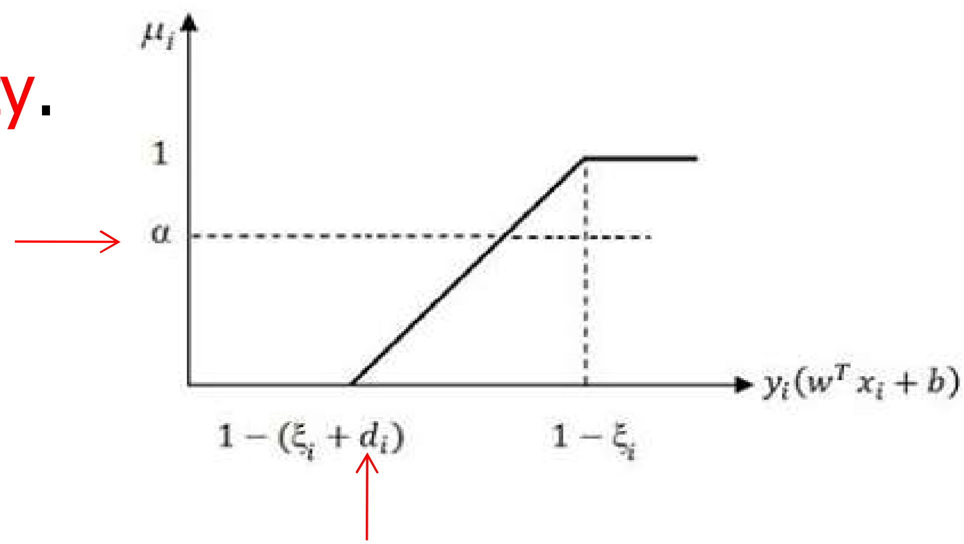
- New formulation of RSVM:

$$\text{Minimize} \quad \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i$$

$$\text{subject to } y_i(w^T x_i + b) \geq 1 - \xi_i - d_i(1-\alpha)$$

$$\xi_i \geq 0, \quad i = 1, 2, \ldots, n$$

- Di named tolerance.
- Alpha named certainty.



Some results

Table 1 The recognition rate of SVM, FSVM, RSVM, and fuzzy RSVM on BUPA liver disorders dataset

	SVM	FSVM	RSVM	Fuzzy RSVM
$C = 100, \ \alpha = 0.9$	65.5556	68.8889	71.1111	**72.222**
$C = 1{,}000, \ \alpha = 0.9$	63.3333	68.8889	71.1111	**72.222**

Table 2 The recognition rate of SVM, FSVM, RSVM, and fuzzy RSVM on Statlog dataset

	SVM	FSVM
	81.1111	80
	RSVM	Fuzzy RSVM
$\alpha = 0.9$	82.2222	84.4444
$\alpha = 0.8$	84.4444	**86.6667**
$\alpha = 0.7$	82.2222	83.3333
$\alpha = 0.6$	78.8889	81.1111
$\alpha = 0.5$	78.8889	78.8889
$\alpha = 0.4$	67.7778	70
$\alpha = 0.3$	73.3333	74.4444
$\alpha = 0.2$	72.2222	73.3333
$\alpha = 0.1$	74.4444	75.5556

Extended on SVM for large dataset

- Another type of improvement on svm focus on time and memory needs for optimization.
- Two type of improvement:
  1) modify SVM algorithm so that it could be applied to large data sets.
  2) select representative training data from a large data set so that a normal SVM could handle.
- In this review we going on second type briefly.

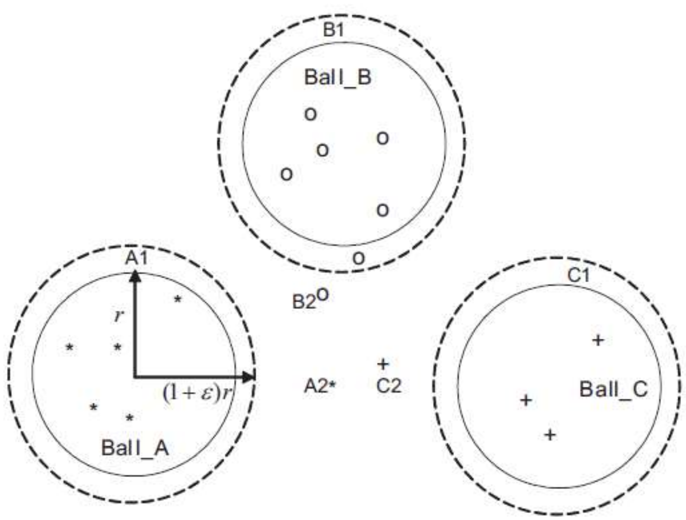
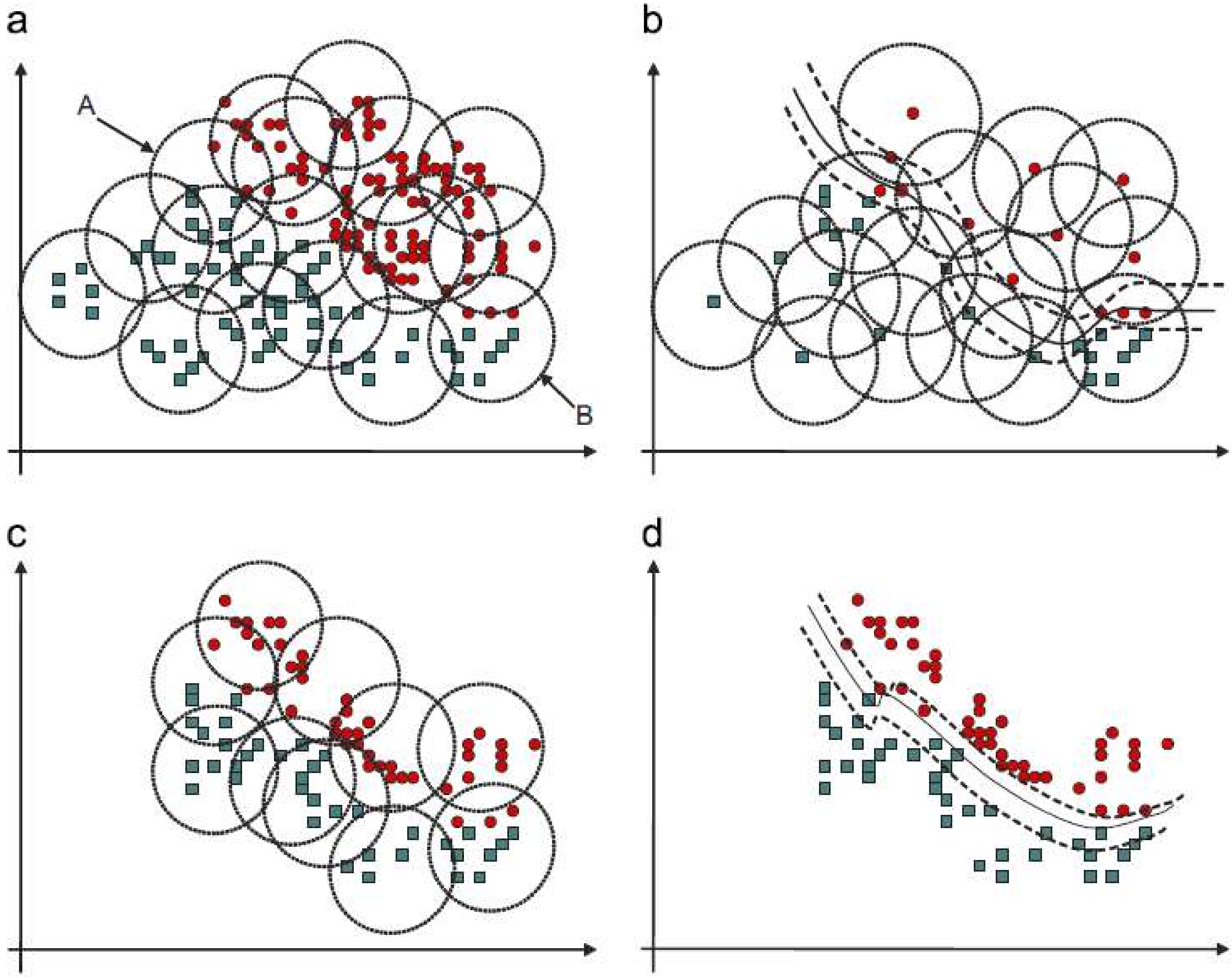


Fig. 1. MED clustering.


a
A
B
b
c
d

Flowchart

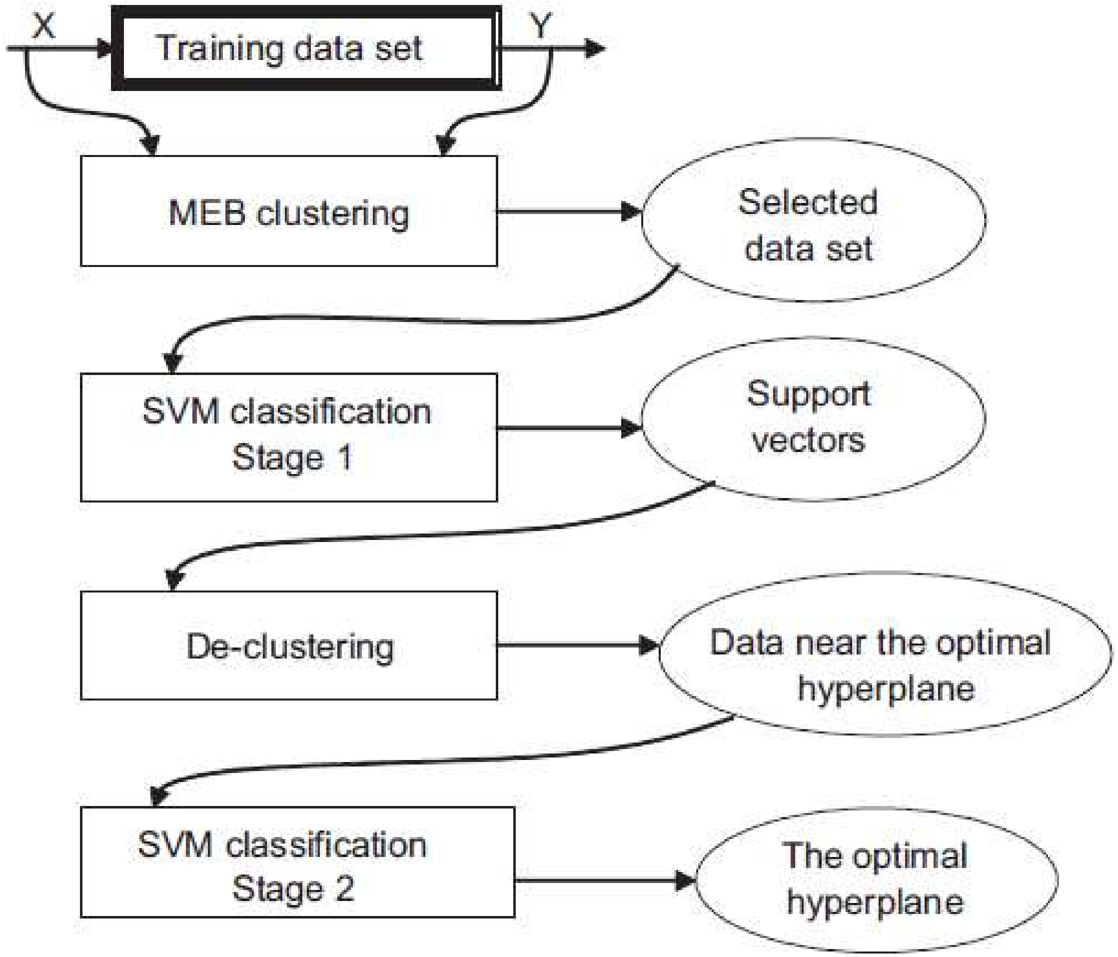


Fig. 2. SVM classification via MEB clustering.

Comparison of time and accuracy

Table 1
Comparison with the other SVM classification

RNA sequence 1												
MEB two stages				LIBSVM			SMO			Simple SVM		
#	t	Acc	K	#	t	Acc	#	t	Acc	#	t	Acc
500	45.04	79.6	300	500	0.23	84.88	500	26.125	85.6	500	2.563	85.38
1000	103.6	82.5	300	1000	0.69	85.71	1000	267.19	87.5	1000	9.40	87.21
5000	163.2	85.7	300	5000	10.28	86.40				5000	539.88	88.65
23,605	236.9	88.5	300	23,605	276.9	87.57						

Table 2
Comparison with the other SVM classification

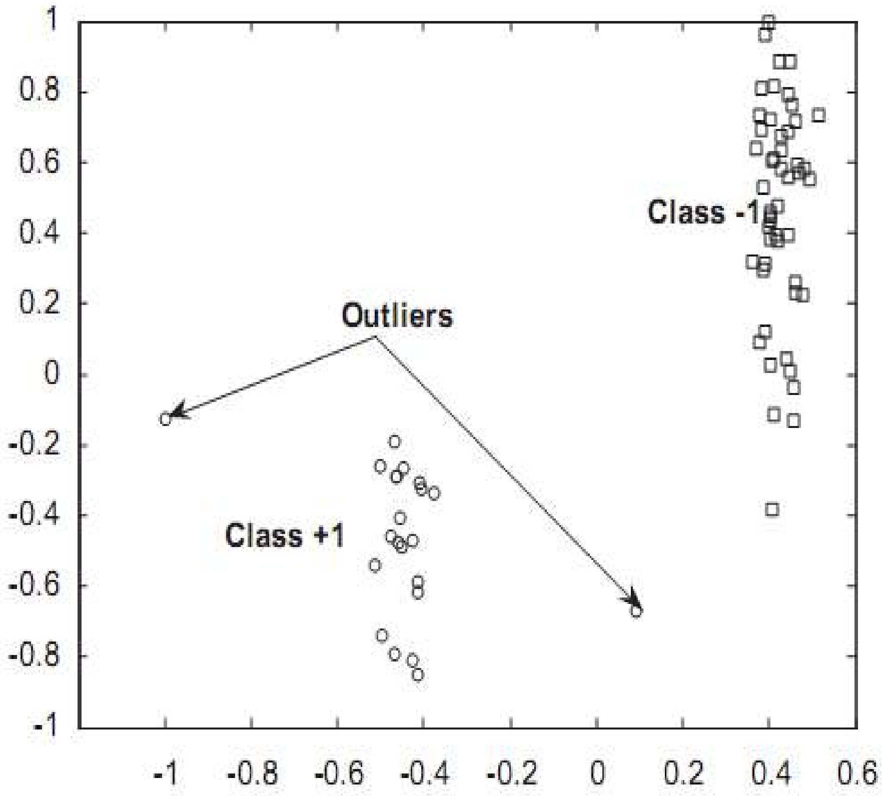
RNA sequence 2												
MEB two stages				LIBSVM			SMO			Simple SVM		
#	t	Acc	K	#	t	Acc	#	t	Acc	#	t	Acc
2000	17.18	75.9	400	2000	8.71	73.15	2000	29.42	78.7	2000	27.35	59.15
2000	7.81	71.7	100									

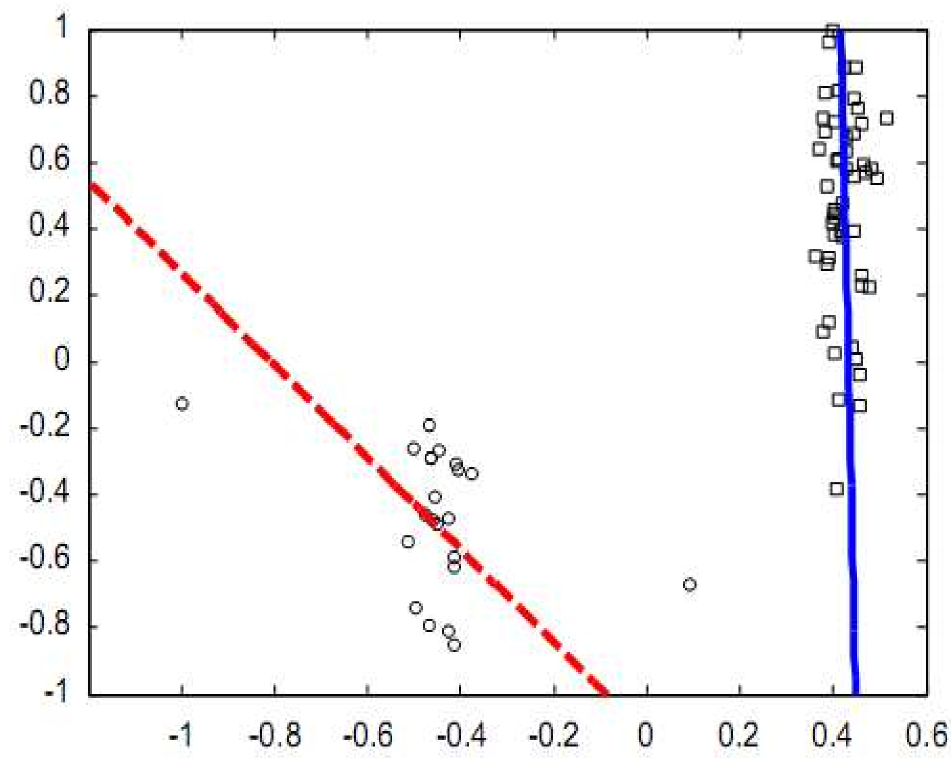
SVDD(one class)

- ارائه روشي مبتني بر مرز براي توصيف داده ها و کاربرد آن در تشخيص نفوذ به شبکه هاي کامپيوتري

Twin support vector machine(2007)IEEE trans

- find two nonparallel hyperplanes around which the data points of the corresponding class get clustered
- solving two independent optimization problems. In each of these twin QPPs, constraints involve patterns from only one class.





Geometric result

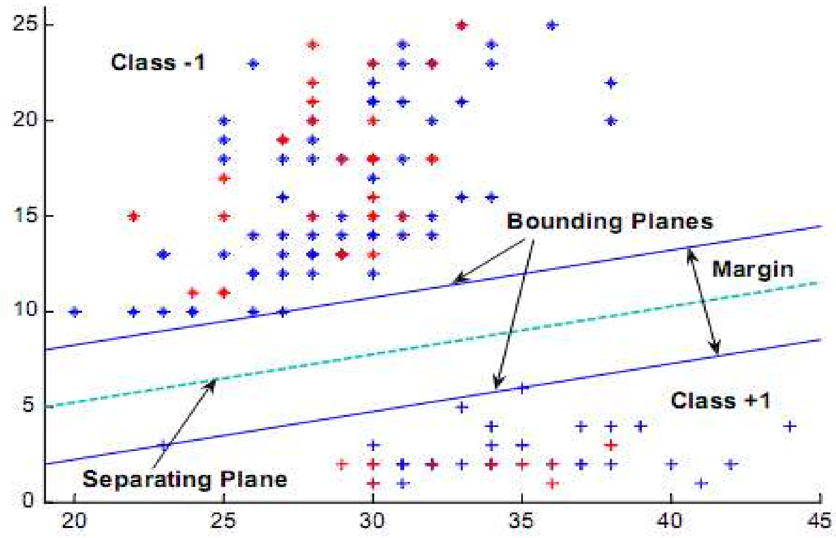


Fig. 1. Geometric interpretation of standard SVM.

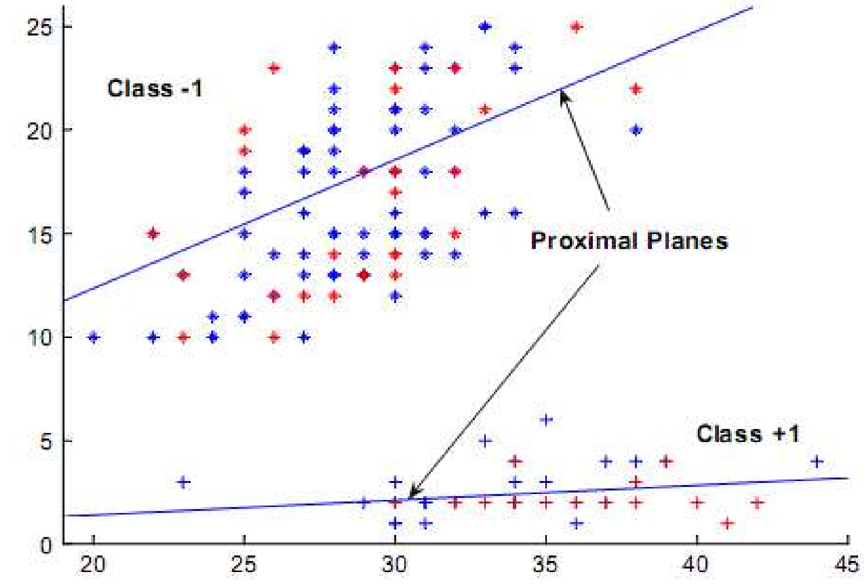


Fig. 2. Geometric interpretation of TSVM.

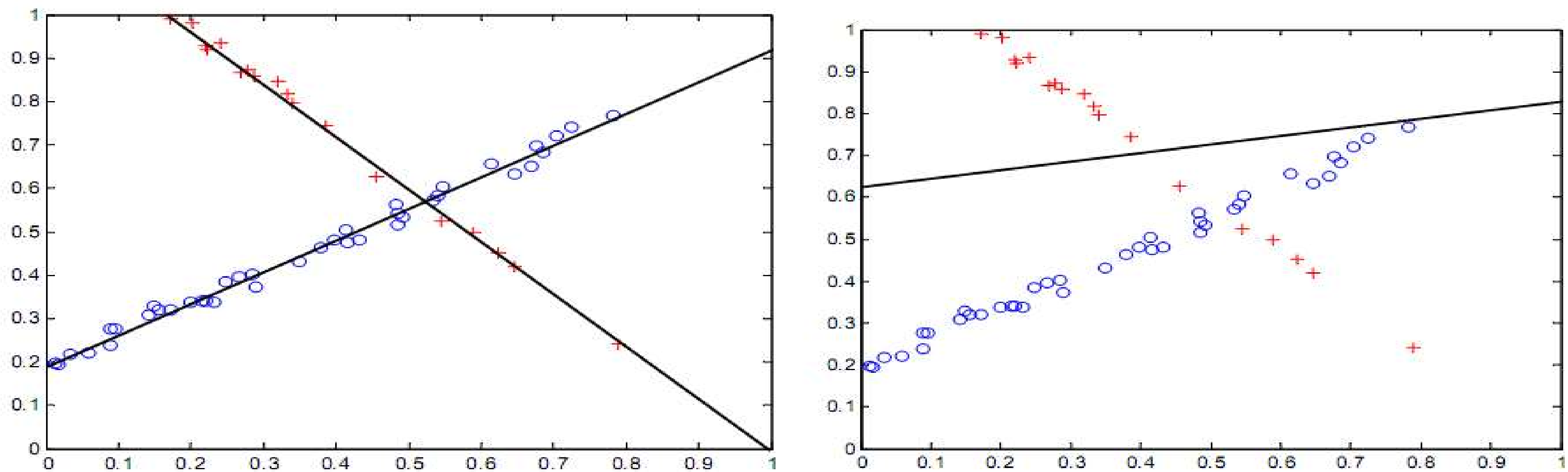


Fig. 3. Classification results of GEPSVM/TSVM/LSTSVM (left) and PSVM (right) for "cross planes" dataset.

TSVM formulation

$$\min_{w^{(1)},b^{(1)},q_1} \frac{1}{2}(Aw^{(1)}+e_1b^{(1)})^T(Aw^{(1)}+e_1b^{(1)})+c_1e_2^Tq_1$$

$$\text{subject to} \quad -(Bw^{(1)}+e_2b^{(1)})+q_1 \geq e_2, q_1 \geq 0,$$

- c1 is a trade off parameter
- e1 is a vector of ones of dimension n1*1
- A sample of one class with dimension n1*m
- B another class with dimension n2*m
- q1 non-negative slack variables.

TSVM(in another notation)

$$\begin{aligned} \min \quad & \frac{1}{2}\sum_{i\in I^+}(w_+^T x_i + b_+)^2 + C_1 \sum_{j\in I^-}\xi_j \\ \text{s.t.} \quad & -w_+^T x_j - b_+ \geqslant 1 - \xi_j, \\ & \xi_j \geqslant 0, \quad j \in I^-, \end{aligned}$$

$$\begin{aligned} \min \quad & \frac{1}{2}\sum_{j\in I^-}(w_-^T x_j + b_-)^2 + C_2 \sum_{i\in I^+}\xi_i \\ \text{s.t.} \quad & w_-^T x_i + b_- \geqslant 1 - \xi_i, \\ & \xi_i \geqslant 0, \quad i \in I^+, \end{aligned}$$

Fuzzy TSVM(2008)

S1

$$\min_{w^{(1)},b^{(1)},q_1} \frac{1}{2}(Aw^{(1)}+e_1b^{(1)})^T(Aw^{(1)}+e_1b^{(1)})+c_1e_2^Tq_1$$

$$\text{subject to} \quad -(Bw^{(1)}+e_2b^{(1)})+q_1 \geq e_2, q_1 \geq 0,$$

- c1 is a trade off parameter
- e1 is a vector of ones of dimension n1*1
- A sample of one class with dimension n1*m
- B another class with dimension n2*m
- q1 non-negative slack variables.

IS-TSVM(2009-exper system.)

$$\underset{w^{(1)},b^{(1)}}{\text{Min}} \quad \frac{1}{2}(Aw^{(1)}+eb^{(1)})'(Aw^{(1)}+eb^{(1)})+\frac{C_1}{2}y'y$$

$$\text{subject to} \quad -(Bw^{(1)}+eb^{(1)})+y=e,$$

$$A'(Aw^{(1)}+eb^{(1)})+C_1B'(Bw^{(1)}+eb^{(1)}+e)=0e,$$

$$e'(Aw^{(1)}+eb^{(1)})+C_1e'(Bw^{(1)}+eb^{(1)}+e)=0.$$

$$\begin{bmatrix} B'B & B'e \\ e'B & m_2 \end{bmatrix}\begin{bmatrix} w^{(1)} \\ b^{(1)} \end{bmatrix}+\frac{1}{C_1}\begin{bmatrix} A'A & A'e \\ e'A & m_1 \end{bmatrix}\begin{bmatrix} w^{(1)} \\ b^{(1)} \end{bmatrix}+\begin{bmatrix} B'e \\ m_2 \end{bmatrix}=0e$$

$$\begin{bmatrix} w^{(1)} \\ b^{(1)} \end{bmatrix}=\begin{bmatrix} B'B+\frac{1}{C_1}A'A & B'e+\frac{1}{C_1}A'e \\ e'B+\frac{1}{C_1}e'A & m_2+\frac{1}{C_1}m_1 \end{bmatrix}^{-1}\begin{bmatrix} -B'e \\ -m_2 \end{bmatrix}$$

$$\begin{bmatrix} w^{(1)} \\ b^{(1)} \end{bmatrix}=\left[\begin{bmatrix} B' \\ e' \end{bmatrix}[B \quad e]+\frac{1}{C_1}\begin{bmatrix} A' \\ e' \end{bmatrix}[A \quad e]\right]^{-1}\begin{bmatrix} -B'e \\ -m_2 \end{bmatrix}$$

Defining $E=[A \quad e]$ and $F=[B \quad e]$, the solution becomes:

$$\begin{bmatrix} w^{(1)} \\ b^{(1)} \end{bmatrix}=-\left(F'F+\frac{1}{C_1}E'E\right)^{-1}F'e.$$

result

Table 1
Classification accuracy for linear kernel

Dataset $l \times n$	LSTSVM	TSVM	GEPSVM	PSVM
Hepatitis (155 × 19)	86.42 ± 9.78	85.71 ± 6.73	85.0 ± 9.19	85.71 ± 5.83
WPBC (198 × 34)	83.88 ± 5.52	83.68 ± 6.24	81.11 ± 7.94	83.3 ± 4.53
Sonar (208 × 60)	80.47 ± 6.7	80.52 ± 4.9	79.47 ± 7.6	78.94 ± 4.43
Heart-statlog (270 × 14)	85.55 ± 4.07	86.66 ± 6.8	85.55 ± 6.1	85.55 ± 7.27
Cross Planes (300 × 7)	98.12 ± 4.67	98.02 ± 3.92	98.2 ± 5.1	60.71 ± 4.19
Heart-c (303 × 14)	85.86 ± 6.17	85.86 ± 6.9	85.51 ± 5.08	85.51 ± 5.08
Bupa Liver (345 × 7)	70.90 ± 6.09	70.5 ± 6.6	66.36 ± 4.39	70.15 ± 8.82
Ionosphere (351 × 34)	89.70 ± 5.58	88.23 ± 3.10	84.11 ± 3.2	89.11 ± 2.79
Votes (435 × 16)	95.23 ± 1.94	95.9 ± 2.2	95.0 ± 2.36	95.0 ± 3.06
Australian (690 × 14)	86.61 ± 4.0	86.91 ± 3.5	80.00 ± 3.99	85.43 ± 3.0
Pima-Indian (768 × 8)	79.4 ± 2.65	78 ± 6.29	76.66 ± 4.62	77.86 ± 3.67
CMC (1473 × 9)	68.84 ± 2.77	68.84 ± 2.39	68.76 ± 2.98	68.98 ± 3.95
Mean accuracy	84.24	84.06	82.14	80.52

twin bounded SVM(2011-IEEE transc.)

$$\min_{w_1,b_1,\xi,\xi^*} \quad \frac{1}{2}c_3(||w_1||^2 + b_1^2) + \frac{1}{2}\xi^{*\top}\xi^* + c_1 e_2^\top \xi$$
$$\text{s.t.} \quad Aw_1 + e_1 b_1 = \xi^*$$
$$-(Bw_1 + e_2 b_1) + \xi \geq e_2, \quad \xi \geq 0$$

and

$$\min_{w_2,b_2,\eta,\eta^*} \quad \frac{1}{2}c_4(||w_2||^2 + b_2^2) + \frac{1}{2}\eta^{*\top}\eta^* + c_2 e_1^\top \eta$$
$$\text{s.t.} \quad Bw_2 + e_2 b_2 = \eta^*$$
$$(Aw_2 + e_1 b_2) + \eta \geq e_1, \quad \eta \geq 0$$

where c_1, c_2, c_3, and c_4 are positive parameters.

margin between two classes can be measured by some kind of distances between the proximal hyperplane $w_1^\top x + b_1 = 0$ and the bounding hyperplane $w_1^\top x + b_1 = -1$ here. Now we show that one of the reasonable distances can be expressed by

$$\frac{1}{\sqrt{||w_1||^2 + b_1^2}} \tag{16}$$

result

Datasets	TBSVM Accuracy % Time (s) c_3/c_4	TWSVM Accuracy % Time (s) $\epsilon = 10^{-6}$	SVC Accuracy % Time (s)
Hepatitis (155 × 19)	83.23 ± 5.94 0.011 0.0039/0.0039	82.89 ± 6.30 0.012/0.281	**84.13 ± 5.58** 1.170
BUPA liver (345 × 6)	**70.12 ± 7.94** 0.010 0.0078/8	66.40 ± 7.74 0.011/0.840	67.78 ± 5.51 3.540
Heart-Statlog (270 × 14)	**85.27 ± 4.95** 0.025 0.0078/64	84.44 ± 6.80 0.023/0.454	83.12 ± 5.41 1.584
Heart-c (303 × 14)	**85.02 ± 8.04** 0.034 32/256	84.86 ± 6.27 0.042/0.516	83.33 ± 5.64 2.193
Votes (435 × 16)	**96.33 ± 4.62** 0.062 64/4	95.85 ± 2.75 0.797/1.851	95.80 ± 2.65 3.192
WPBC (198 × 34)	**84.14 ± 3.33** 0.012 0.0156/0.0039	83.68 ± 5.73 0.012/0.560	83.30 ± 4.53 2.094
Sonar	78.94 ± 5.54	77.00 ± 6.10	**80.13 ± 5.43**

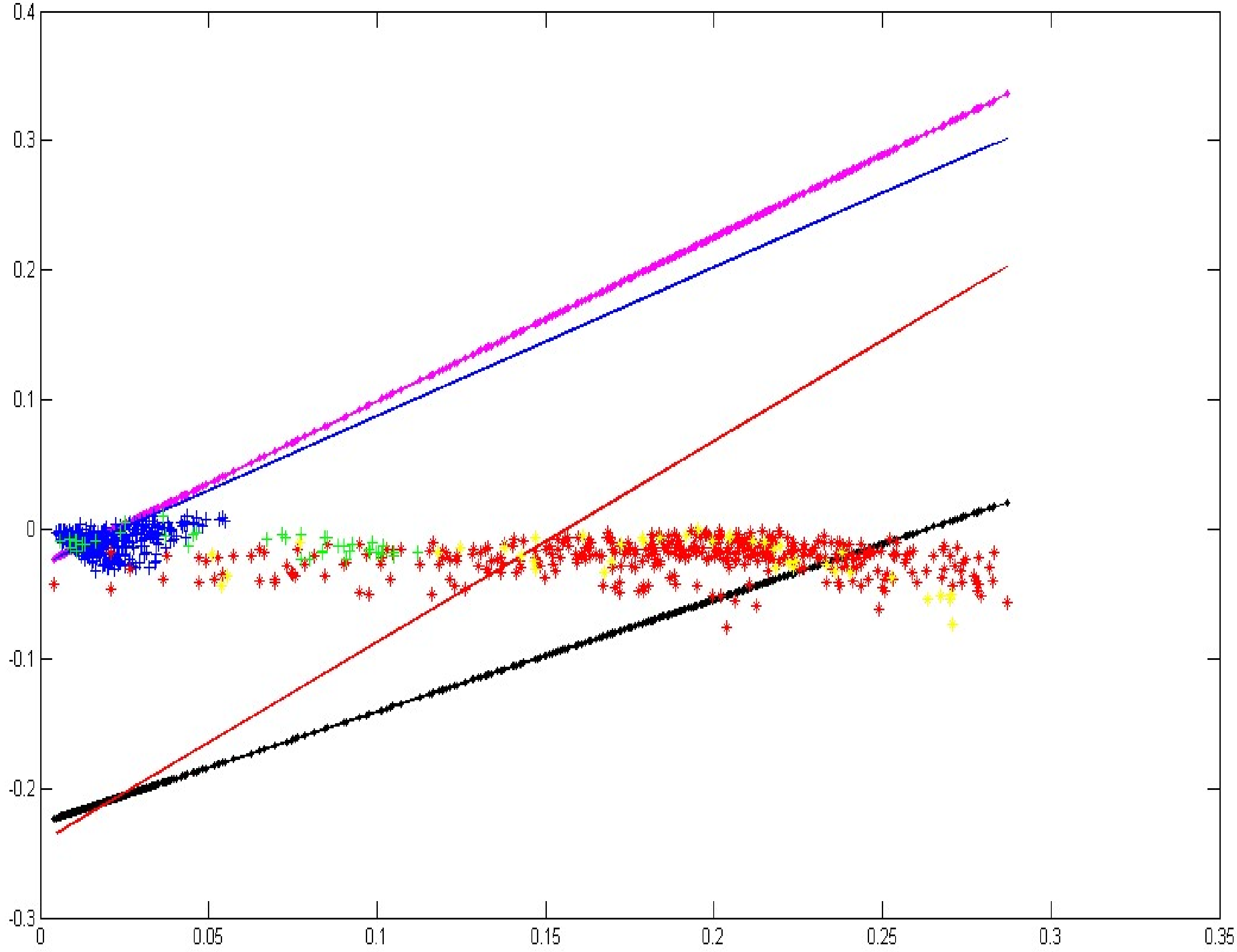
Relax constraint TSVM(TC-TSVM)

$$\begin{aligned}\min \quad & \frac{1}{2}\sum_{i\in I^+}(w_+^T x_i + b_+)^2 + C_1 \sum_{j\in I^-}\xi_j \\ \text{s.t.} \quad & -w_+^T x_j - b_+ \gtrsim 1 - \xi_i, \quad i = 1, \ldots, \\ & \xi_j \geqslant 0, \quad j \in I^-,\end{aligned}$$

$$\text{Minimize} \quad \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i$$

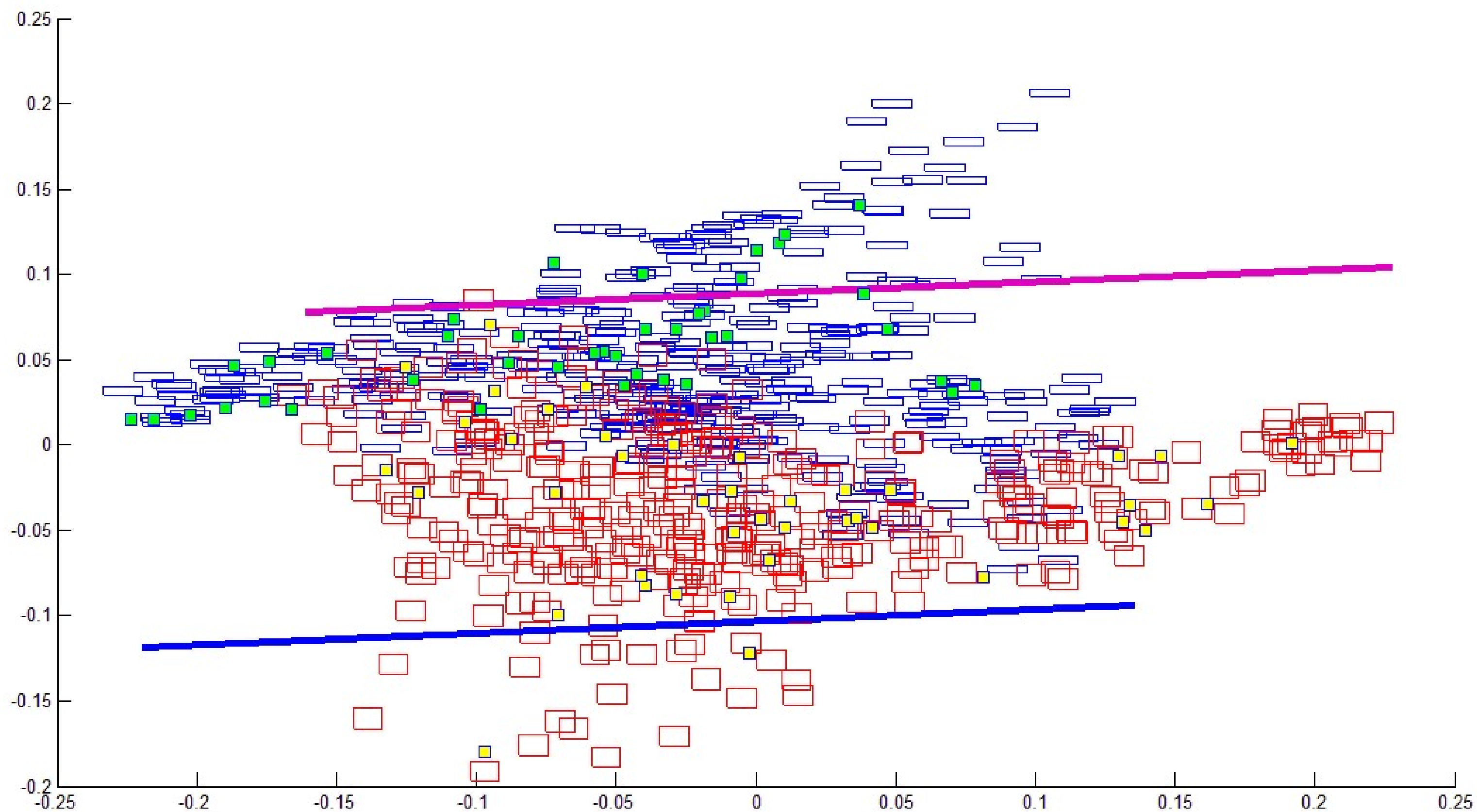
$$\text{subject to } y_i\left(w^T x_i + b\right) \geq 1 - \xi_i - d_i(1-\alpha)$$

$$\xi_i \geq 0, \quad i = 1, 2, \ldots, n$$

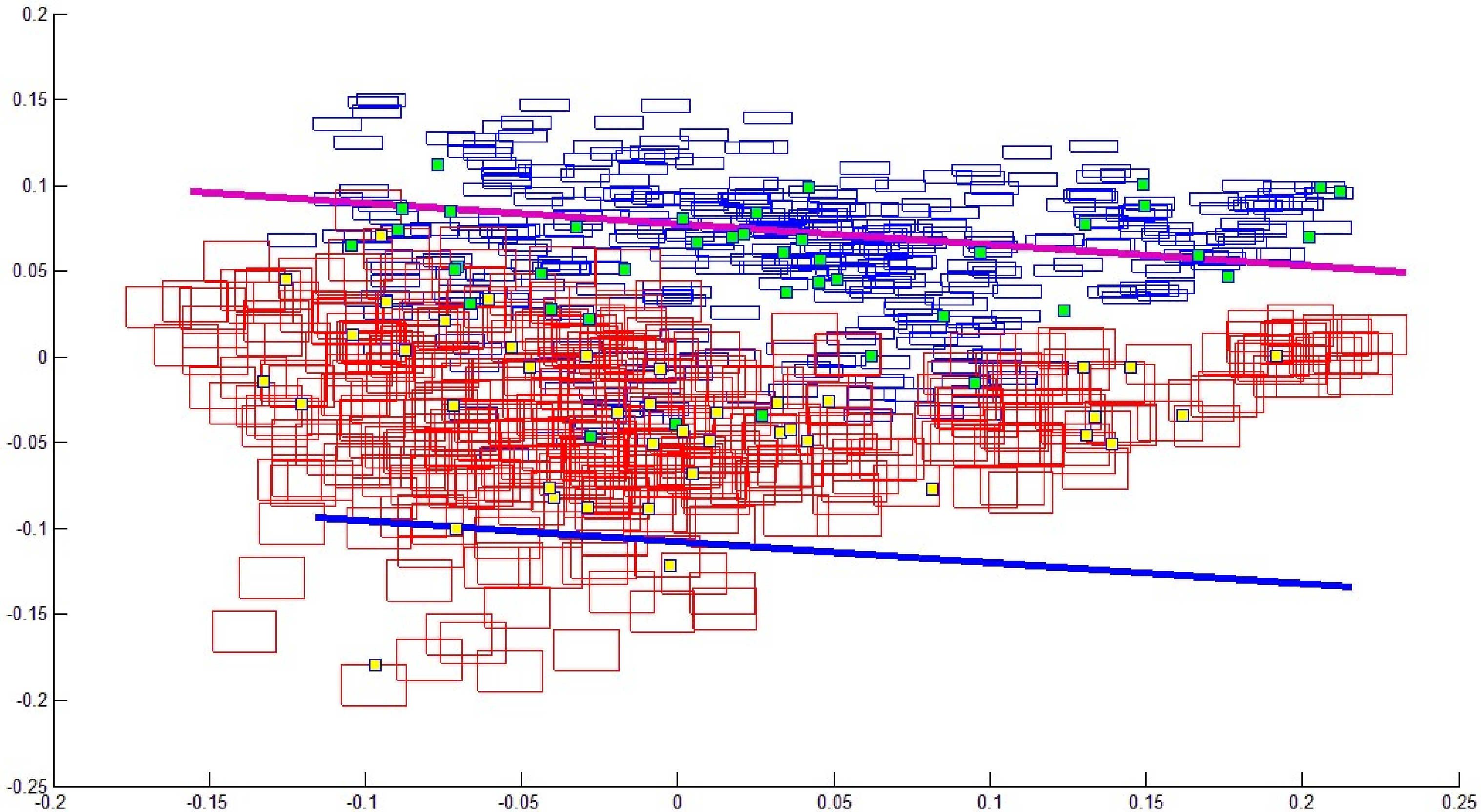


0.4
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35

Interval tsvm



Interval tsvm



Multi-hyperplane TSVM(MH-TSVM)

- First a clustring method take for all data.
- If in each class less than k% be from another class that is good
- Else increasing k%
- And keep clustring