Kernel Method and Support Vector Machines

Nguyen Duc Dung, Ph.D.
Institute of Information Technology, VAST

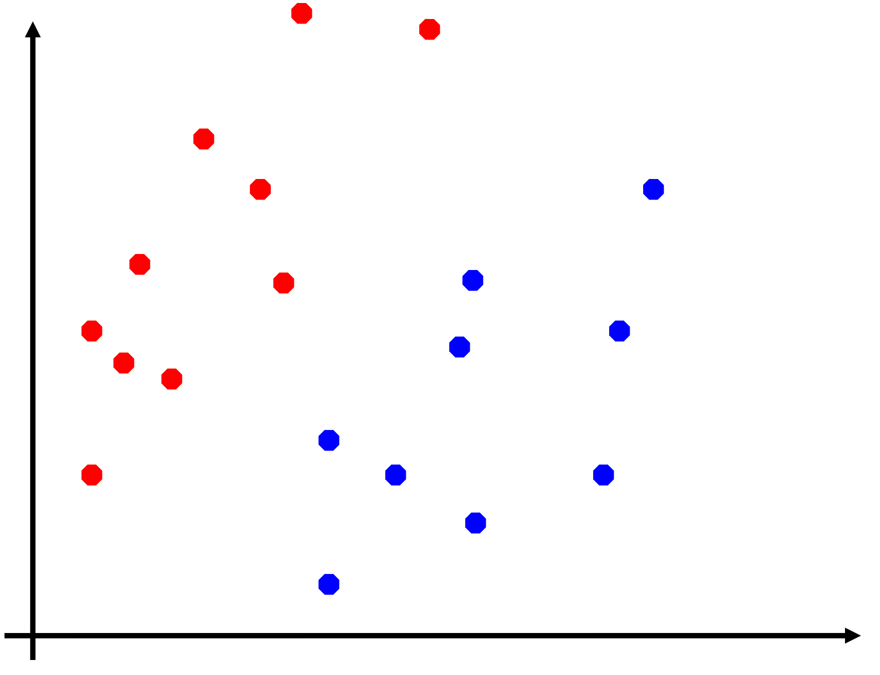
Outline

- **Reference**
  - Books, papers, slides, software
- **Support vector machines (SVMs)**
  - The maximum-margin hyper-plane
  - Kernel method
- **Implementation**
  - Approaches
  - Sequential minimal optimization (SMO)
- **Open problems**
- **SVMs Approximation**
  - Reduced set method

Reference

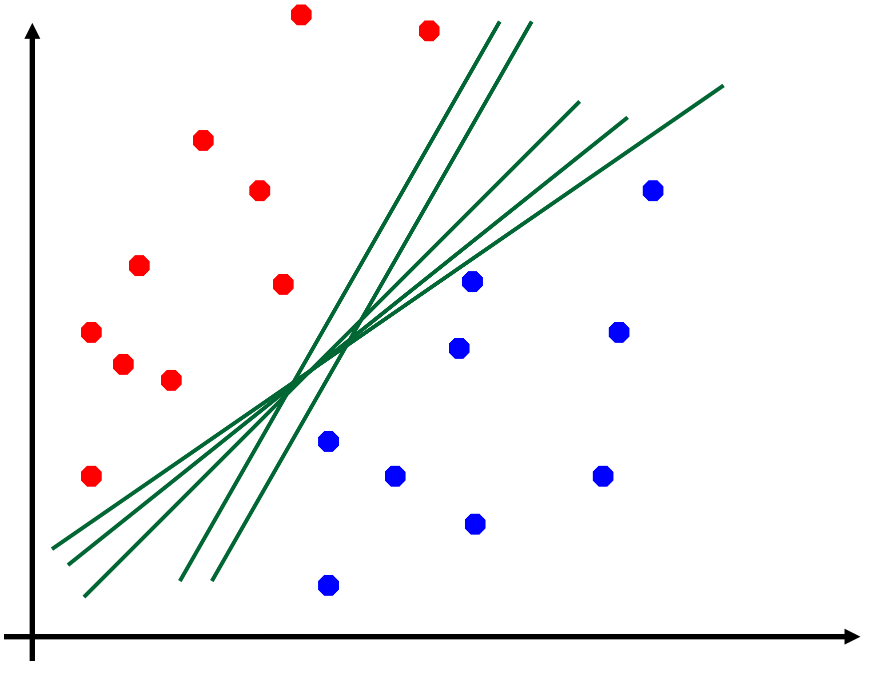
- **Book**
  - Cristianini, N., Shawe-Taylor, J., *An Introduction to Support Vector Machines*, Cambridge University Press, (2000). http://www.support-vector.net/index.html
  - Bernhard Schölkopf and Alex Smola. **Learning with Kernels**. MIT Press, Cambridge, MA, 2002.
- **Paper**
  - C. J. C. Burges. A Tutorial on Support Vector Machines for Pattern Recognition. *Knowledge Discovery and Data Mining*, 2(2), 1998.
- **Slide**
  - N. Cristianini. ICML'01 tutorial, 2001.
- **Software**
  - LibSVM (NTU), SVM^{light} (joachims.org)
- **Online resource**
  - http://www.kernel-machines.org/

Classification Problem



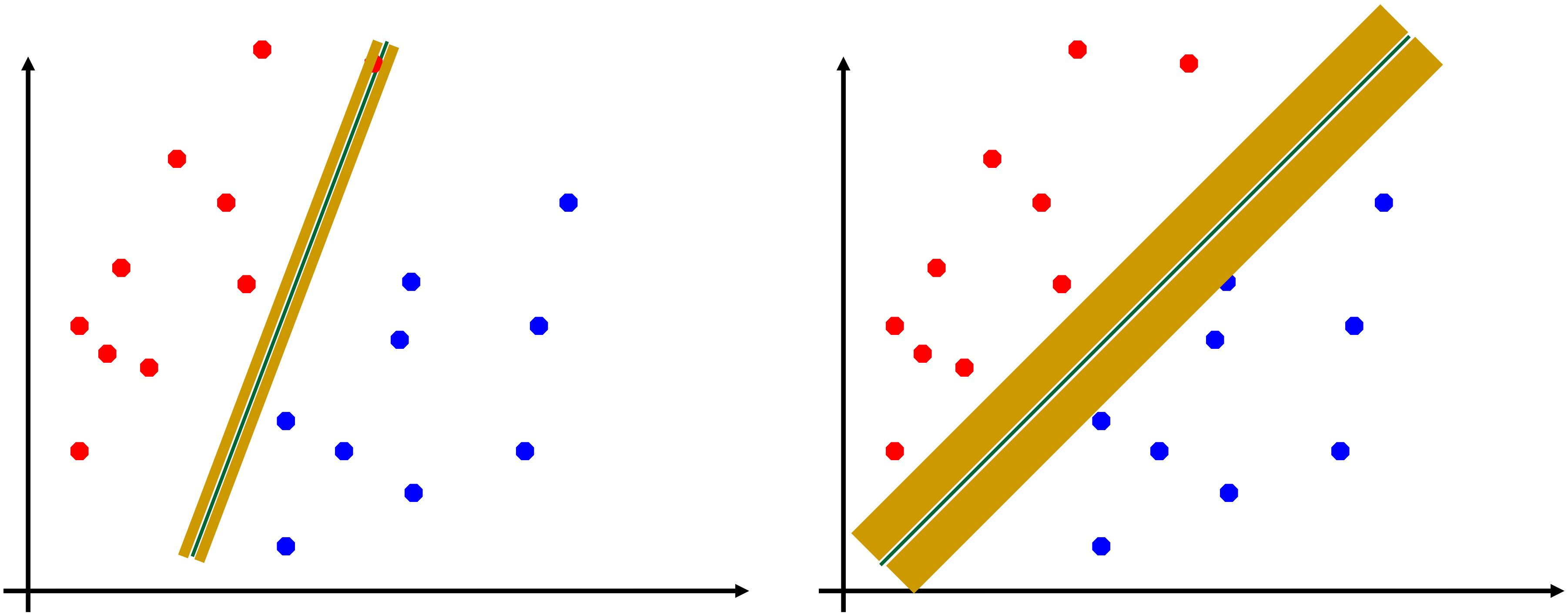
How would we classify this data set?

Linear Classifiers



There are many lines that can be linear classifiers.
Which one is the better classifier?

Margin of a Linear Classifier



- **Margin**: The **minimum distance** to the training data
- **SVM solution**: The linear classifier with **maximum margin**

Margin

of a Linear Function $f(x) = w.x + b$

- Functional margin

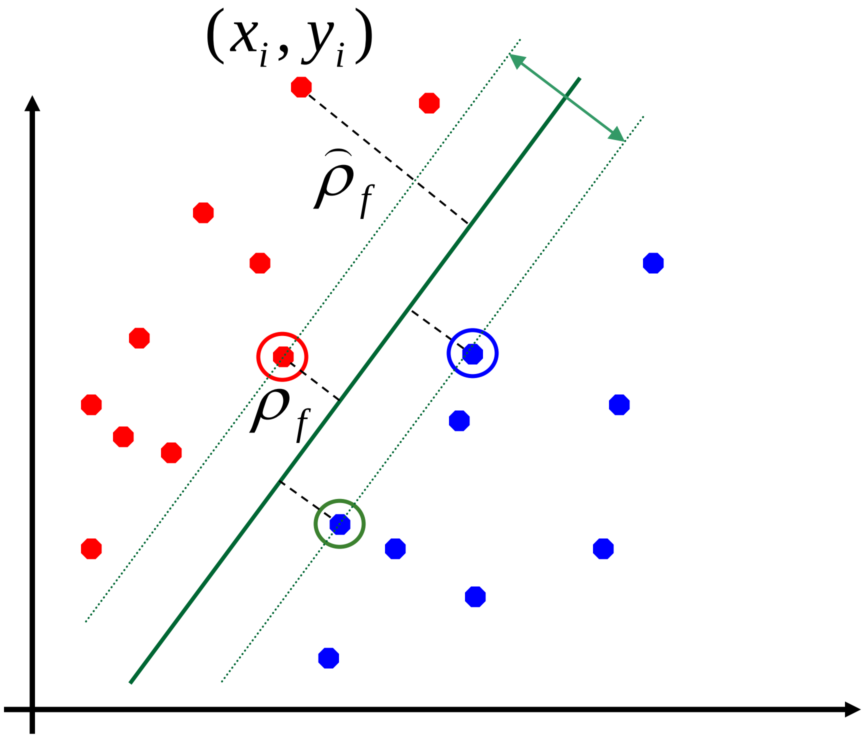
$$\hat{\rho}_f(\boldsymbol{x}_i, y_i) = y_i(\boldsymbol{w} \cdot \boldsymbol{x}_i + b)$$

- Geometric margin

$$\rho_f(\boldsymbol{x}_i, y_i) = \frac{\hat{\rho}_f(\boldsymbol{x}_i, y_i)}{\|\boldsymbol{w}\|}$$

- Margin $\rho_f = \min_{i=1\ldots l} \rho_f(\boldsymbol{x}_i, y_i)$

- SVM solution $\boxed{f^* = \arg\max_f \rho_f}$



A Bound on Expected Risk

of a Linear Classifier $f = sign(w.x)$

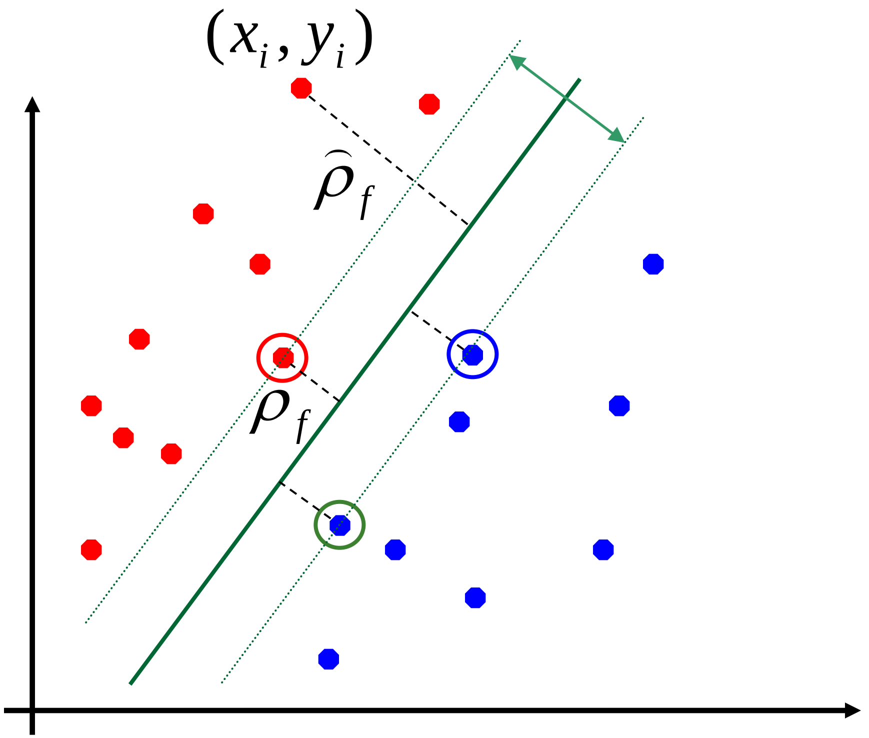
With a probability at least $(1 - \delta)$, $\delta \in (0,1)$

$$R[f] \leq R_{emp}[f] + \sqrt{\frac{c}{l}\left(\frac{R^2\Lambda^2}{\boxed{\rho_f^{\,2}}}\ln^2 l + \ln\frac{1}{\delta}\right)}$$

where R_{emp} is training error, l is training size, ρ_f is the margin, $\|\mathbf{w}\| \leq \Lambda$, $\|\mathbf{x}\| \leq R$, c is a constant

Larger margin, smaller bound

Finding the Maximum-Margin Classifier



$$f^* = \arg\max_f \rho_f$$

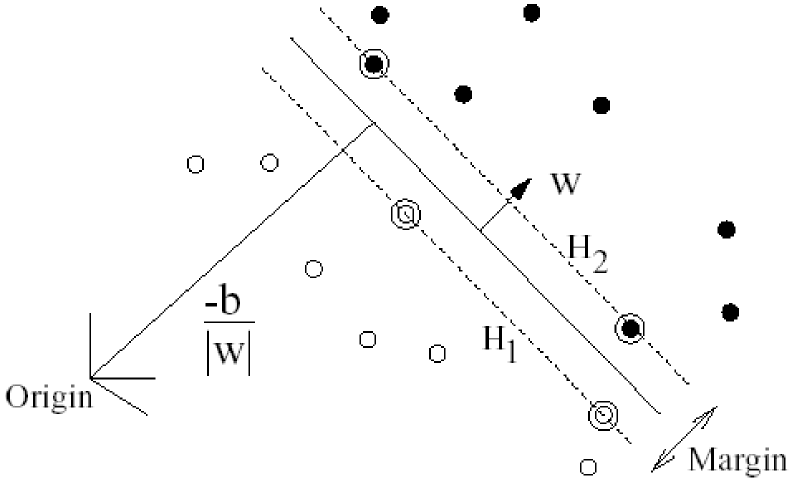
$$\textit{minimize} \quad \frac{1}{2}\|\mathbf{w}\|^2,$$

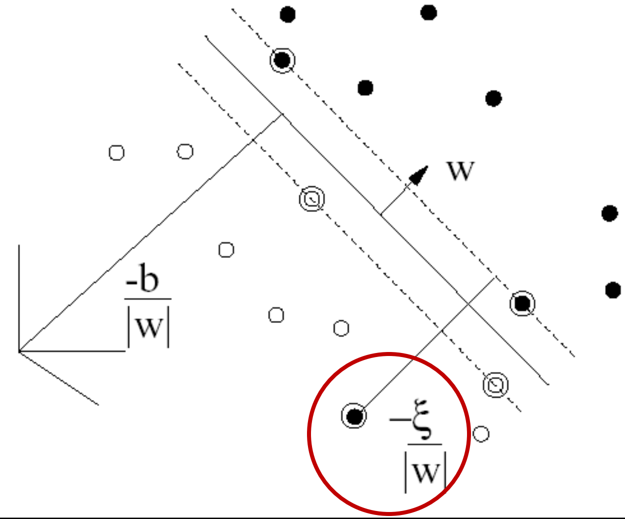
$$\textit{subject to} \quad y_i(w \cdot x_i + b) \geq 1, i = 1, ..., l$$

Minimize normal vector

Constrain functional margin ≥ 1

Soft and Hard Margin





$$\min_{w,b} \frac{1}{2} \| w \|^2$$
$$s.t.\; y_i (w.x_i + b) \geq 1, i = 1,\ldots,l$$

Hard (maximum) margin

$$\min_{w,b} \frac{1}{2} \| w \|^2 + C \sum_{i=1}^{n} \xi_i^{\,p}$$
$$s.t.\;\; y_i (wx_i + b) \geq 1 - \xi_i ,$$
$$\xi_i \geq 0, i = 1,\ldots,l$$

Soft (maximum) margin

Lagrangian Optimization

Definition 1 *Given an optimization problem with convex domain* $\Omega \subseteq \mathbb{R}^d$

$$\text{minimize} \quad f(w), w \in \Omega \tag{2.20}$$

$$\text{subject to} \quad g_i(w) \leq 0, i = 1, ..., k \tag{2.21}$$

$$h_i(w) = 0, i = 1, ..., m \tag{2.22}$$

The generalized Lagrangian function is defined as

$$L(w, \alpha, \beta) = f(w) + \sum_{i=1}^{k} \alpha_i g_i(w) + \sum_{i=1}^{m} \beta_i h_i(w) \tag{2.23}$$

Definition 2 *The Lagrangian dual problem of the primal problem is the following problem*

$$\text{maximize} \quad \theta(\alpha, \beta) \tag{2.24}$$

$$\text{subject to} \quad \alpha \geq 0 \tag{2.25}$$

where $\theta(\alpha, \beta) = \inf_{w \in \Omega} L(w, \alpha, \beta)$

Kuhn-Tucker Theorem

Theorem 3 *(Kuhn-Tucker) Given an optimization problem with convex domain* $\Omega \subseteq \mathbb{R}^d$

$$\begin{aligned} \textit{minimize} \quad & f(w), w \in \Omega && (2.27) \\ \textit{subject to} \quad & g_i(w) \leq 0, i = 1, ..., k && (2.28) \\ & h_i(w) = 0, i = 1, ..., m && (2.29) \end{aligned}$$

with $f \in C^1$ *convex and* g_i, h_i *affine, necessary and efficient conditions for a normal point* w^* *to be optimum are the existence of* α^* *and* β^* *such that*

$$\begin{aligned} \frac{\partial L(w^*, \alpha^*, \beta^*)}{\partial w} &= 0 && (2.30) \\ \frac{\partial L(w^*, \alpha^*, \beta^*)}{\partial \beta} &= 0 && (2.31) \\ \alpha_i^* g_i(w^*) &= 0, i = 1, ..., k && (2.32) \\ g_i(w^*) &\leq 0, i = 1, ..., k && (2.33) \\ \alpha_i &\geq 0, i = 1, ..., k && (2.34) \end{aligned}$$

Optimization

$$\min_{w,b} \frac{1}{2} \| w \|^2 + C \sum_{i=1}^{n} \xi_i^{\,p}$$

$$s.t.\ \ y_i(wx_i + b) \geq 1 - \xi_i,$$

$$\xi_i \geq 0, i = 1, \ldots, l$$

Primal problem

$$L(w, b, \xi, \alpha, \beta) = \frac{1}{2} w^2 + C \sum_{i=1}^{l} \xi_i - \sum_{i=1}^{l} \alpha_i (y_i (w \cdot x_i + b) - 1 + \xi_i) - \sum_{i=1}^{l} \beta_i \xi_i$$

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

$$\boxed{\mathbf{w} = \sum_{\alpha_i \neq 0} y_i \alpha_i x_i}$$

$$\frac{\partial L(w, \alpha, \beta)}{\partial \xi_i} = C - \alpha_i - \beta_i = 0$$

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

Dual problem

$$\min_{\alpha_i} \frac{1}{2} \sum_{i,j=1}^{l} y_i y_j \alpha_i \alpha_j \langle x_i, x_j \rangle - \sum_{i=1}^{l} \alpha_i$$

$$\text{s.t.:}\ \ 0 \leq \alpha_i \leq C, i = 1, \ldots, l,$$

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

(Linear) Support Vector Machines

- Training

$$\boxed{\begin{aligned}&\min_{\alpha_i} \frac{1}{2}\sum_{i,j=1}^{l} y_i y_j \alpha_i \alpha_j \langle x_i, x_j \rangle - \sum_{i=1}^{l} \alpha_i \\ &\text{s.t.:}\quad 0 \le \alpha_\mathrm{i} \le C, i = 1,\dots, l, \\ &\qquad \sum_{i=1}^{l} y_i \alpha_i = 0.\end{aligned}}$$

  - Quadratic optimization
  - l variables
  - l^2 coefficients

- Testing

$$\boxed{f(x) = w \cdot x + b}$$

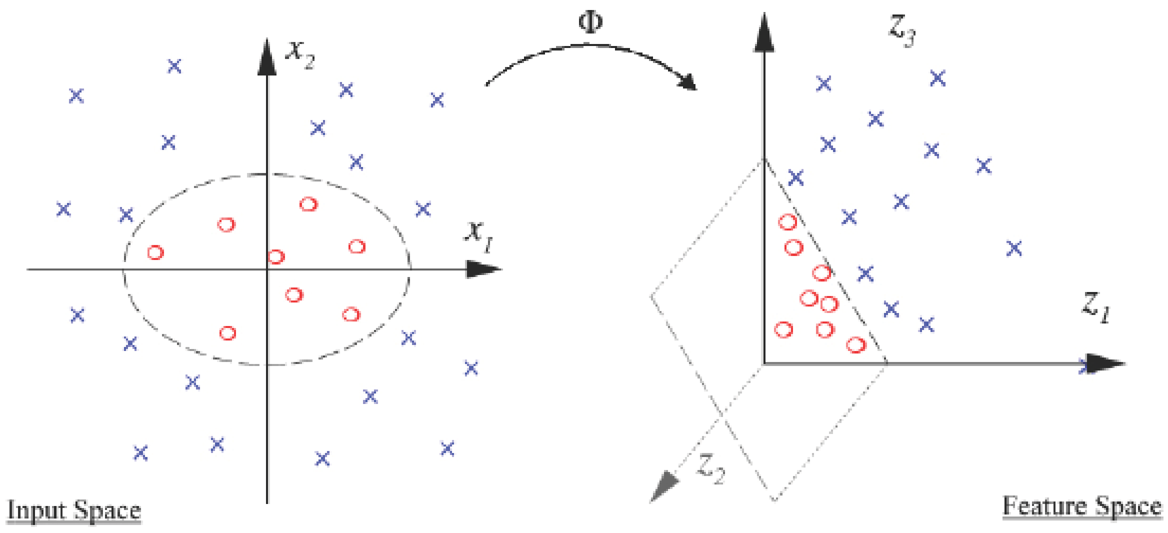
  - Norm of the hyperplane

$$\mathbf{w} = \sum_{\alpha_i \neq 0} y_i \alpha_i x_i$$

  - (x_i, α_i), $\alpha_i \# 0$ – ***support vector***

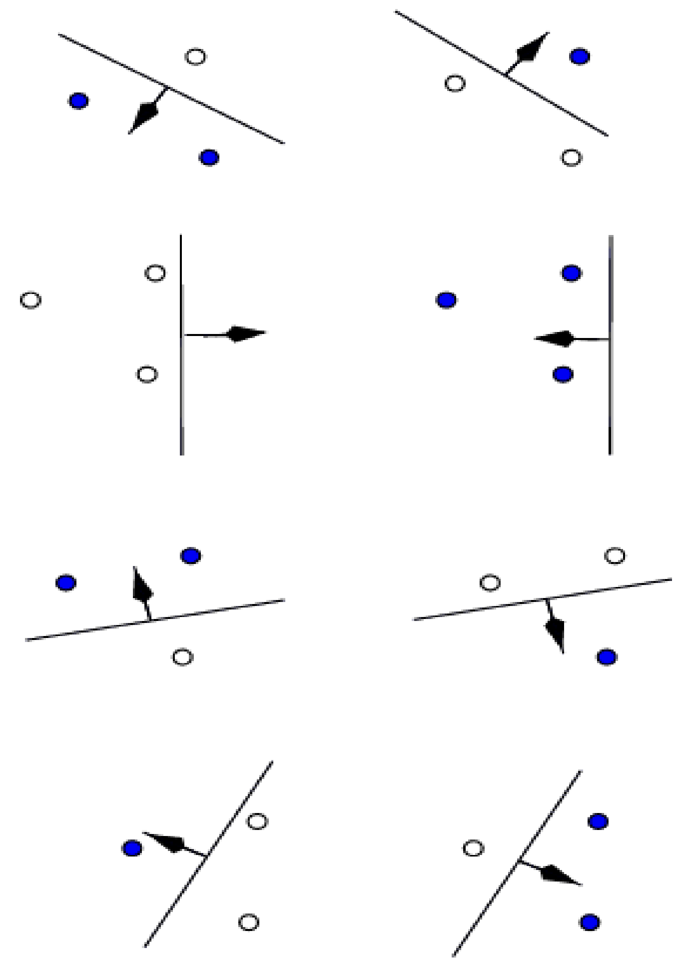
Kernel Method

- Problem
  - Most datasets are **linearly non-separable**
- Solution
  - Map input data into a ***higher dimensional*** feature space
  - Find the optimal hyperplane in feature space



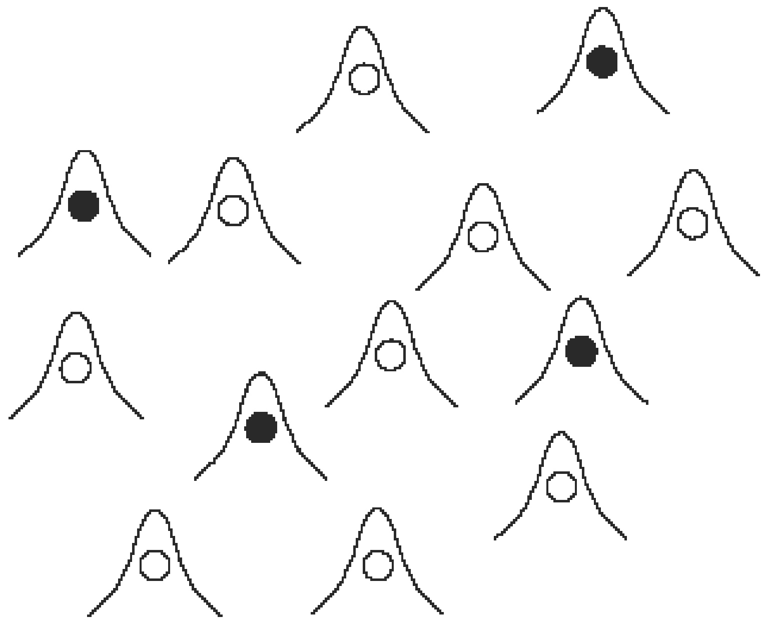
Hyperplane in Feature Space

- VC-dimension of a class of functions: the maximum number of points that can be shattered
- VC-dimension of linear functions in R^d is $d+1$
- Dimension of feature space is high



- **Linear functions** in feature space has high VC-dimension, or **high capacity**

VC Dimension: Example



Gaussian RBF SVMs of sufficiently small width can classify an arbitrary large number of training points correctly, and thus ***have infinite VC dimension***

Linear SVMs

- Training

$$\min_{\alpha_i} \frac{1}{2}\sum_{i,j=1}^{l} y_i y_j \alpha_i \alpha_j \boxed{\langle x_i, x_j \rangle} - \sum_{i=1}^{l} \alpha_i$$

$$\text{s.t.:}\quad 0 \le \alpha_i \le C, i = 1,\ldots,l,$$

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

  - Quadratic optimization
  - l variables
  - l^2 coefficients

- Testing

$$f(x) = sign\left(\sum_{\alpha_i \neq 0} y_i \alpha_i \boxed{\langle x, x_i \rangle} + b\right)$$

  - Norm of the hyperplane

$$w = \sum_{\alpha_i \neq 0} y_i \alpha_i x_i$$

  - (x_i, α_i), $\alpha_i \# 0$ – support vector

SVMs work with ***pairs*** of data (dot product), not sample

Non-linear SVMs

- **Kernel**: to calculate dot product between two vectors in feature space $\boxed{K(x,y) = \langle\Phi(x), \Phi(y)\rangle}$

- Training

$$\min_{\alpha_i} \frac{1}{2}\sum_{i,j=1}^{l} y_i y_j \alpha_i \alpha_j \boxed{K(x_i, x_j)} - \sum_{i=1}^{l}\alpha_i$$

$$\text{s.t.}: \quad 0 \le \alpha_i \le C, i = 1,\dots,l,$$

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

- Testing

$$f(x) = sign\left(\sum_{\alpha_i \neq 0} y_i \alpha_i \boxed{K(x, x_i)} + b\right)$$

Norm of the hyperplane

$$\Psi = \sum_{\alpha_i \neq 0} y_i \alpha_i \Phi(x_i)$$

The maximal margin algorithm works indirectly in feature space via kernel, or Φ is not known explicitly

Kernel

- Linear: $K(x,y) = \langle x.y \rangle$
- Gaussian: $K(x,y) = \exp(-\gamma\|x-y\|^2)$
  - Dimension of feature space: *infinite*
- Polynomial: $K(x,y) = \langle x.y \rangle^p$
  - Dimension of feature space: $\binom{d+p-1}{p}$, where d – input space dimension

Theorem 4 *(Mercer) To guarantee that a continuous symmetric function $K(u,v)$ in $L_2(C)$ has an expansion*

$$K(u,v) = \sum_{i=1}^{\infty} a_k z_k(u) z_k(v) \tag{2.53}$$

with positive coefficients $a_k > 0$ (i.e., $K(u,v)$ describes an inner product in some feature space), it is necessary and sufficient that the condition

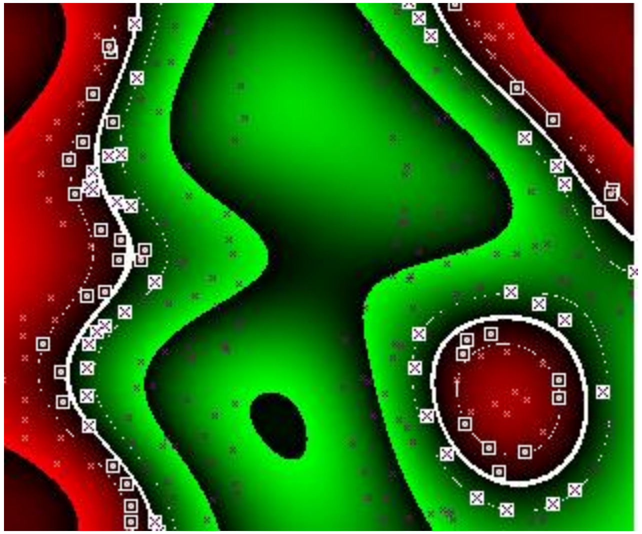
$$\int_C \int_C K(u,v) g(u) g(v) \, du \, dv \geq 0 \tag{2.54}$$

is valid for all $g \in L_2(C)$ (C being a compact subset of $\mathbb{R}^d$)

Support Vector Learning

- **Task**
  - Given a set of labeled data

    $T = \{(x_i, y_i)\}_{i=1,\dots,l} \subset R^d \times \{-1,+1\}$
  - Find the decision function



- **Training**

Time: O(l^3), Memory: O(l^2)

$$\min_{\alpha_i} \frac{1}{2} \sum_{i,j=1}^{l} y_i y_j \alpha_i \alpha_j K(x_i, x_j) - \sum_{i=1}^{l} \alpha_i$$

$$\text{s.t.: } 0 \le \alpha_i \le C, i = 1, \dots, l,$$

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

- **Testing**

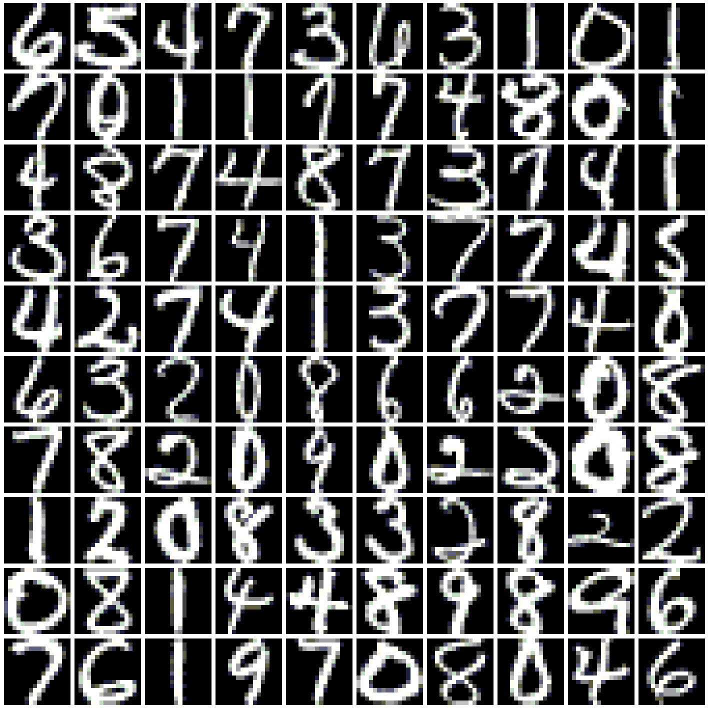
Time: O(Ns)

$$f(x) = sign\left(\sum_{\alpha_i \neq 0} y_i \alpha_i K(x, x_i) + b \right)$$

MNIST Data: SVM vs. Other

- Data
  - 60,000/10,000 training/testing
- Performance

Method	Testing error (%)
linear classifier (1-layer NN)	12.0
K-nearest-neighbors	5.0
40 PCA + quadratic classifier	3.3
SVM, Gaussian Kernel	*1.4*
2-layer NN, 300 hidden units, mean square error	4.7
Convolutional net LeNet-4	**1.1**



Hand written data

(Source: http://yann.lecun.com/)

SVM: Probability Output

- SVM solution

$$f(x) = \sum_{\alpha_i \neq 0} y_i \alpha_i K(x_i, x) + b$$

- Probability estimation

$$\boxed{p(y=+1 \mid x) \approx \frac{1}{1+e^{Af(x)+B}}}$$

- Maximum likelihood approach

$$(A, B) = \arg\min_{a,b} F(a,b) = -\sum_{i=1}^{l} \big(t_i \log(p_i) + (1-t_i)\log(1-p_i)\big)$$

where $p_i = p(y=+1 \mid x_i) \approx \dfrac{1}{1+e^{af(x)+b}}$,

$$t_i = \begin{cases} \dfrac{N_+ + 1}{N_+ + 2} & \text{if } y_i = +1, \\ \dfrac{1}{N_- + 1} & \text{if } y_i = -1 \end{cases}, i = 1, \ldots, l. (N_+ : \#\text{positive}, \; N_+ : \#\text{negative})$$

Outline

- **Reference**
  - Books, papers, slides, software
- **Support vector machines (SVMs)**
  - The maximum-margin hyperplane
  - Kernel method
- **Implementation**
  - Approaches
  - Sequential minimal optimization
- **Open problems**

SVM Training

Problem

$$\min_{\alpha_i} F(\boldsymbol{\alpha}) = \frac{1}{2}\sum_{i,j=1}^{l} y_i y_j \alpha_i \alpha_j K_{ij} - \sum_{i=1}^{l} \alpha_i$$

$$\text{s.t.}: \quad 0 \le \alpha_\text{i} \le C, i = 1, \ldots, l,$$

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

Quadratic programming (QP)

- Obj. function: quadratic w.r.t. $\boldsymbol{\alpha}$
- Number of variable: l
- Number of parameter: l^2
- Complexity
  - Time: $O(l^3)$ *or* $O(N_S^3 + N_S^2 l + N_S dl)$
  - Memory: ***$O(l^2)$***
- Constraint: box, linear

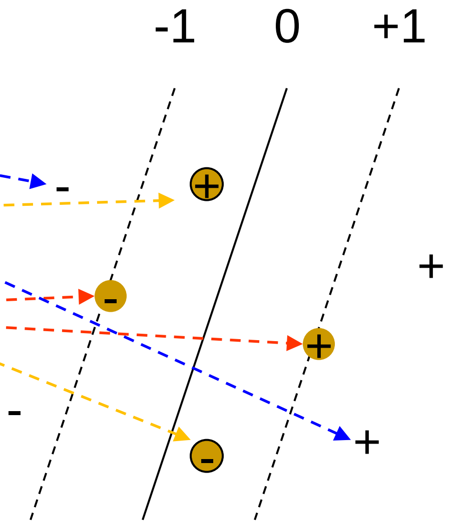
Approach

- **Gradient method**
  - Modified gradient projection (Bottou et al., 94)
- **Divide-and-conquer**
  - Decomposition alg. (e.g. Osuna et al., 97, Joachims, 99)
  - Sequential minimal optimization (SMO) (Plat, 99)
- **Parallelization**
  - Cascade SVM (Peter et al., 05)
  - Parallel mixture of SVM (Collobert et al., 02)
- **Approximation**
  - Online and active learning (e. g. Bordes et al., 05)
  - Core SVM (Tsang et al., 05, 07)
- **Combination of methods**

Optimality

The Karush-Kuhn-Tucker (KKT) conditions

$$\begin{cases} y_i f(x_i) > 1 & for \quad \alpha_i = 0, \\ y_i f(x_i) < 1 & for \quad \alpha_i = C, \\ y_i f(x_i) < 1 & for \quad 0 < \alpha_i < C, \end{cases}$$



where $f(x) = \sum_{i=1}^{l} y_i \alpha_i K(x, x_i) + b$

SMO Algorithm

- Initialize solution (zero)
- **While** (!*StoppingCondition*)
- Select two vector {*i*,*j*}
- Optimize on {*i*,*j*}
- **EndWhile**

SMO: Optimization

- Problem

$$\min_{\alpha_i} F(\boldsymbol{\alpha}) = \frac{1}{2}\sum_{i,j=1}^{l} y_i y_j \alpha_i \alpha_j K_{ij} - \sum_{i=1}^{l} \alpha_i$$

$$\text{s.t.:}\quad 0 \le \alpha_i \le C, i = 1,\dots,l,$$

$$\sum_{k=1}^{l} \alpha_k y_k = 0$$

$$\rightarrow \forall (i,j) : y_i \alpha_i + y_j \alpha_j = const$$

$$\rightarrow \alpha_j = y_j (const - y_i \alpha_i)$$

- Fixing all $\alpha_k, k \ne i, j$

$$F(\boldsymbol{\alpha}) = F(\alpha_i) = A\alpha_i^2 + B\alpha_i + C$$

- Updating scheme

(without the box constraint)

$$\boxed{\begin{aligned} \alpha_i^{new} &= \alpha_i^{old} + \frac{y_i\left(E_j^{old} - E_i^{old}\right)}{2\kappa_{ij}}, \\ \alpha_j^{new} &= \alpha_j^{old} + \frac{y_j\left(E_i^{old} - E_j^{old}\right)}{2\kappa_{ij}}. \end{aligned}}$$

$$E_i = \sum_{k=1}^{l} y_k \alpha_k K(x_k, x_i) - y_i, i = 1,\dots,l,$$

$$\kappa_{ij} = K_{ii} + K_{jj} - 2K_{ij}$$

Selection Heuristic and Stopping Condition

- Maximum violating pair

$$\begin{cases} i = \arg\max\left\{-E_k \mid k \in I_{up}\right\} \\ j = \arg\min\left\{-E_k \mid k \in I_{low}\right\} \end{cases}$$

- Maximum gain

$$\begin{cases} i = \arg\max\left\{-E_k \mid k \in I_{up}\right\} \\ j = \arg\max\left\{\left|\Delta F_{ik}\right| \mid k \in I_{low}, -E_k < -E_i\right\} \end{cases}$$

where $I_{up} = \{t \mid \alpha_t < C, y_t = +1 \text{ or } \alpha_t > 0, y_t = -1\}$

$I_{low} = \{t \mid \alpha_t < C, y_t = -1 \text{ or } \alpha_t > 0, y_t = +1\}$

- Stopping condition: $\boxed{\left|E_i - E_j\right| < \varepsilon(10^{-3})}$

Sequential Minimal Optimization

- Training problem

$$\min_{\alpha_i} \frac{1}{2}\sum_{i,j=1}^{l} y_i y_j \alpha_i \alpha_j K(x_i, x_j) - \sum_{i=1}^{l} \alpha_i$$

$$\text{s.t.:}\quad 0 \le \alpha_{\text{i}} \le C, i = 1,\ldots,l,$$

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

- Functional margin

$$E_i = \sum_{k=1}^{l} y_k \alpha_k K(x_k, x_i) - y_i$$

- Selection heuristic

$$i = \arg\max_k \{-E_k \mid k \in I_{up}(\alpha)\}$$

$$j = \arg\max_k \{|\Delta L_{ik}| \mid k \in I_{low}(\alpha), E_k < E_i\}$$

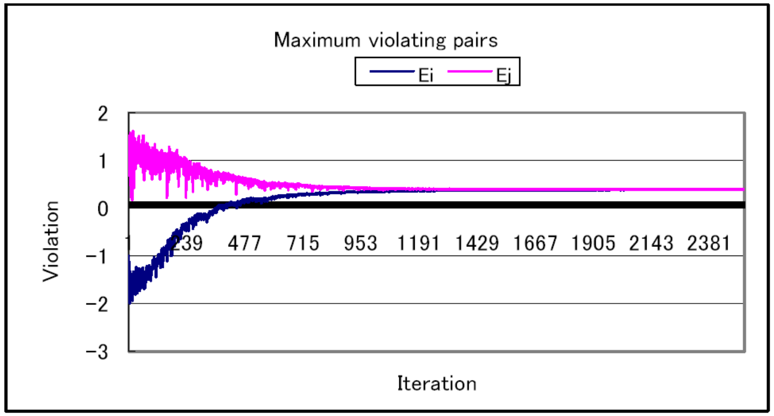
- Updating scheme

$$\alpha_i^{new} = \alpha_i^{old} + \frac{y_i\left(E_j^{old} - E_i^{old}\right)}{2\kappa_{ij}},$$

$$\alpha_j^{new} = \alpha_j^{old} + \frac{y_j\left(E_i^{old} - E_j^{old}\right)}{2\kappa_{ij}}.$$

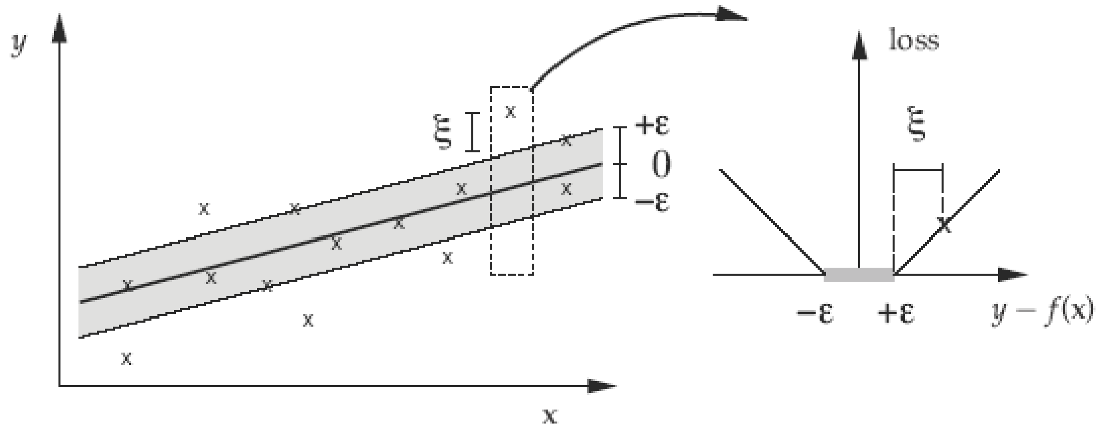
- Stopping condition

$$|E_i - E_j| < \varepsilon$$



Support Vector Regression (1)

- Training data $S = \{(x_i, y_i)\}_{i = 1,\ldots,l} \subset \boldsymbol{R}^N \times R$
- Linear regressor $y = f(x) = w \bullet x + b$
- ε-loss function



$$L^{\epsilon}((\boldsymbol{x}_i, y_i), f) = |y_i - f(\boldsymbol{x}_i)|_{\epsilon} = \max(0, |y_i - f(\boldsymbol{x}_i)| - \epsilon)$$

Support Vector Regression (2)

- Optimization: minimizing

$$\boxed{\frac{1}{2}\left\|\boldsymbol{w}\right\|_2^2 + C\sum_{i=1}^{l} L^{\epsilon}((\boldsymbol{x}_i, y_i), f)}$$

- Dual problem

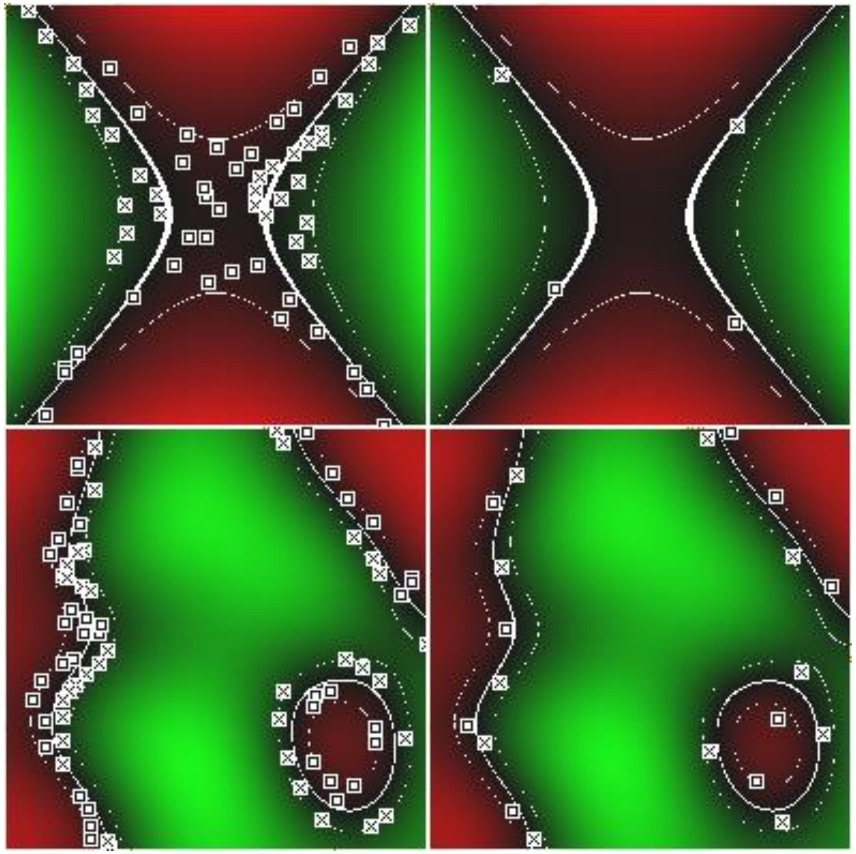
$$\boxed{\begin{array}{ll} maximize & \sum_{i=1}^{l}(\alpha_i^- - \alpha_i^+)y_i - \epsilon\sum_{i=1}^{l}(\alpha_i^- + \alpha_i^+) \\ & -\frac{1}{2}\sum_{i,j=1} l(\alpha_i^- - \alpha_i^+)(\alpha_j^- - \alpha_j^+)\boldsymbol{x_i}\cdot\boldsymbol{x_j}, \\ subject\ to & 0 \le \alpha_i^+, \alpha_i^- \le C, i = 1, \ldots, l \\ & \sum_{i=1}^{l}(\alpha_i^- - \alpha_i^+) = 0, i = 1, \ldots, l \end{array}}$$

Open Problems

- **Model selection**
  - Kernel type
  - Parameter setting
- **Speed and size**
  - Training: time $O(N_S^2 l)$, space $O(N_S l)$
  - Testing: $O(N_S)$
- **Multi-class application**
  - One-versus-rest
  - One-versus-one
- **Categorical data**

Open Problems

- **Model selection**
  - Kernel type
  - Parameter setting
- **Speed and size**
  - Training: time $O(N_S^2 l)$, space $O(N_S l)$
  - Testing: $O(N_S)$
- **Multi-class application**
  - One-versus-rest
  - One-versus-one
- **Categorical data**



Reduced Set Method

To replace original machine

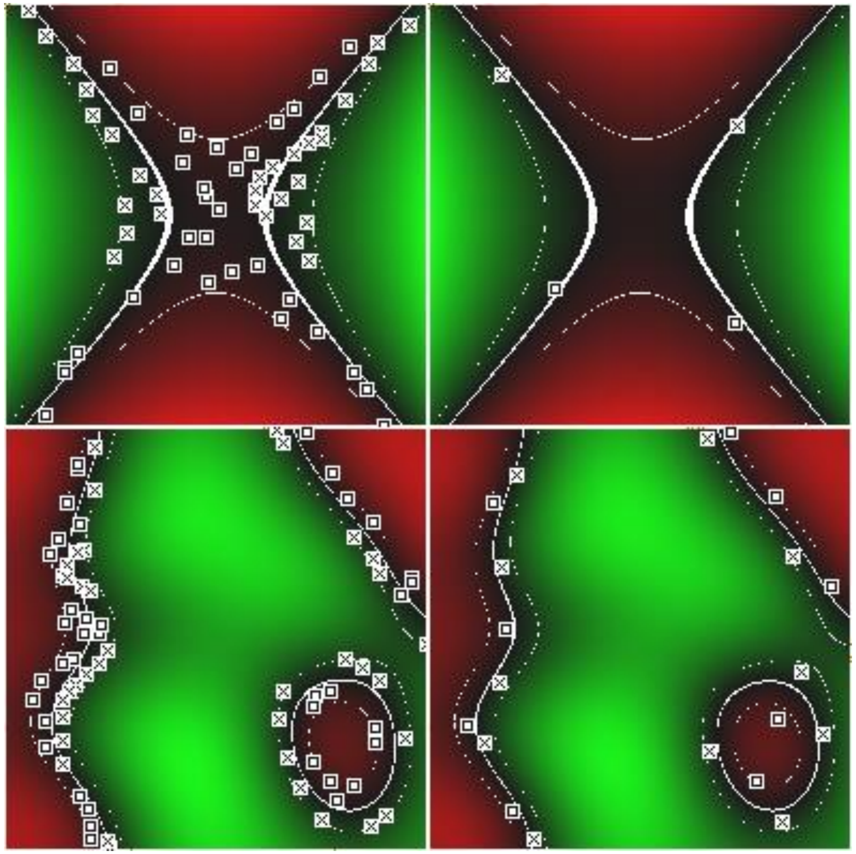
$$y = sign\left(\sum_{i=1}^{Ns} \alpha_i K(x_i, x) + b\right) \quad (1)$$

By a simplified machine

$$y' = sign\left(\sum_{j=1}^{Nz} \beta_j K(z_j, x) + b\right) \quad (2)$$

with $N_Z << N_S$,
(1) and (2) are similar

$\{(x_i,\alpha_i)\}_{i=1,\ldots,Ns}$ – original vectors
$\{(z_j,\beta_j)\}_{j=1,\ldots,Nz}$ – reduced vectors



Performance

- Dataset

Name	Dimension	# Class	# Training	#Testing
DNA	180	3	2,000	1,186
Satimage	36	6	4,435	2,000
Shuttle	9	7	43,500	14,500
USPS	256	10	7,291	2,007

- Result

Data	DNA		Satimage		Shuttle		USPS	
% of SV	#SV	Acc. (%)	#SV	Acc. (%)	#SV	Acc. (%)	#SV	Acc. (%)
100%	843	95.62	1215	89.75	4191	**99.03**	1670	94.77
50%	422	95.62	608	89.75	2096	**99.03**	835	94.77
10%	84	95.53	122	89.45	419	**99.03**	167	94.67
5%	42	95.19	61	89.25	210	**99.03**	84	93.92
1%	8	95.03	12	**78.00**	42	**99.04**	***45***	**89.59**

High reduction rate, no change in predictive accuracy

Practice

- **Download the following SVM tools**
  - LibSVM: LIBSVM -- A Library for Support Vector Machines (ntu.edu.tw)
- **Select one classification problem**
  - From UCI repository
  - From LibSVM data sets
- **Report on progress in selected problem**
  - State-of-the-art works
  - SVM: model selection, performance