Outline

- Basic concepts
- SVM primal/dual problems
- Training linear and nonlinear SVMs
- Parameter/kernel selection and practical issues
- Multi-class classification
- Discussion and conclusions

Outline

- Basic concepts
- SVM primal/dual problems
- Training linear and nonlinear SVMs
- Parameter/kernel selection and practical issues
- Multi-class classification
- Discussion and conclusions

Why SVM and Kernel Methods

- SVM: in many cases competitive with existing classification methods

  Relatively easy to use
- Kernel techniques: many extensions

  Regression, density estimation, kernel PCA, etc.

Support Vector Classification

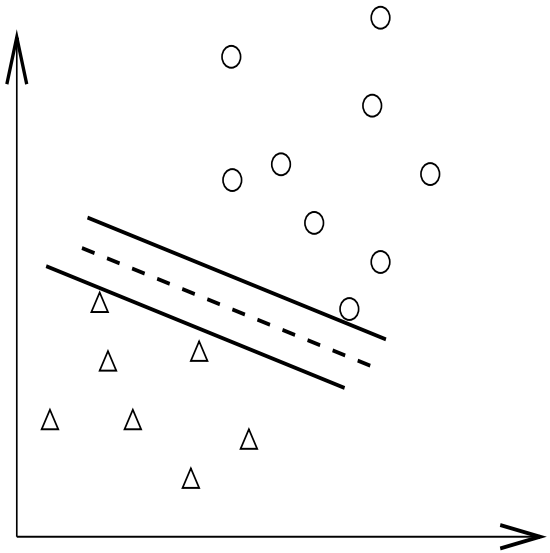
- Training vectors : $\mathbf{x}_i, i = 1, \ldots, l$
- Feature vectors. For example,

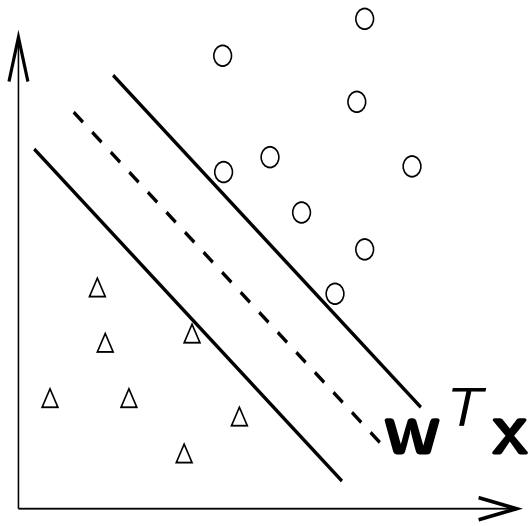
  A patient = [height, weight, ...]
- Consider a simple case with two classes:

  Define an indicator vector $\mathbf{y}$

$$y_i = \begin{cases} 1 & \text{if } \mathbf{x}_i \text{ in class 1} \\ -1 & \text{if } \mathbf{x}_i \text{ in class 2,} \end{cases}$$

- A hyperplane which separates all data





$$\mathbf{w}^T\mathbf{x} + b = \begin{bmatrix} +1 \\ 0 \\ -1 \end{bmatrix}$$

- A separating hyperplane: $\mathbf{w}^T\mathbf{x} + b = 0$

$$\begin{aligned} (\mathbf{w}^T\mathbf{x}_i) + b > 0 \quad & \text{if } y_i = 1 \\ (\mathbf{w}^T\mathbf{x}_i) + b < 0 \quad & \text{if } y_i = -1 \end{aligned}$$

- Decision function $f(\mathbf{x}) = \text{sgn}(\mathbf{w}^T\mathbf{x} + b)$, $\mathbf{x}$: test data

Many possible choices of $\mathbf{w}$ and b

Maximal Margin

- Distance between $\mathbf{w}^T\mathbf{x} + b = 1$ and -1:

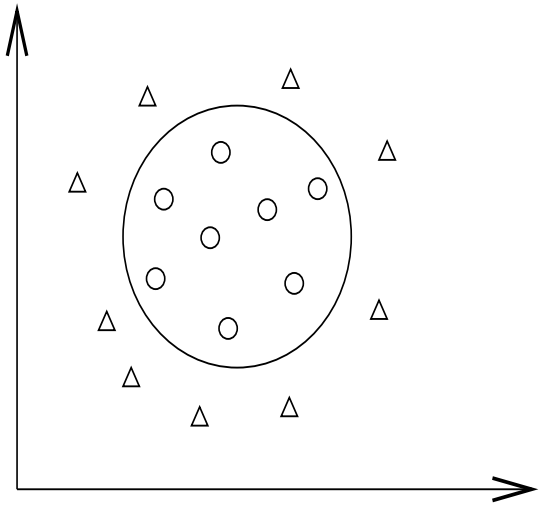
$$2/\|\mathbf{w}\| = 2/\sqrt{\mathbf{w}^T\mathbf{w}}$$

- A quadratic programming problem [Boser et al., 1992]

$$\begin{aligned} \min_{\mathbf{w},b} \quad & \frac{1}{2}\mathbf{w}^T\mathbf{w} \\ \text{subject to} \quad & y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1, \\ & i = 1, \ldots, l. \end{aligned}$$

Data May Not Be Linearly Separable

- An example:



- Allow training errors
- Higher dimensional (maybe infinite) feature space

$$\phi(\mathbf{x}) = (\phi_1(\mathbf{x}), \phi_2(\mathbf{x}), \ldots).$$

- Standard SVM [Cortes and Vapnik, 1995]

$$\begin{aligned} \min_{\mathbf{w},b,\boldsymbol{\xi}} \quad & \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{l}\xi_i \\ \text{subject to} \quad & y_i(\mathbf{w}^T\phi(\mathbf{x}_i) + b) \geq 1 - \xi_i, \\ & \xi_i \geq 0,\ i = 1, \ldots, l. \end{aligned}$$

- Example: $\mathbf{x} \in R^3, \phi(\mathbf{x}) \in R^{10}$

$$\begin{aligned} \phi(\mathbf{x}) \ = \ & (1, \sqrt{2}x_1, \sqrt{2}x_2, \sqrt{2}x_3, x_1^2, \\ & x_2^2, x_3^2, \sqrt{2}x_1x_2, \sqrt{2}x_1x_3, \sqrt{2}x_2x_3) \end{aligned}$$

Finding the Decision Function

- $\mathbf{w}$: maybe infinite variables
- The dual problem

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \quad & \frac{1}{2}\boldsymbol{\alpha}^T Q \boldsymbol{\alpha} - \mathbf{e}^T \boldsymbol{\alpha} \\ \text{subject to} \quad & 0 \le \alpha_i \le C, i = 1, \ldots, l \\ & \mathbf{y}^T \boldsymbol{\alpha} = 0, \end{aligned}$$

  where $Q_{ij} = y_i y_j \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$ and $\mathbf{e} = [1, \ldots, 1]^T$
- At optimum

$$\mathbf{w} = \sum_{i=1}^{l} \alpha_i y_i \phi(\mathbf{x}_i)$$

- A finite problem: #variables = #training data

Kernel Tricks

- $Q_{ij} = y_i y_j \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$ needs a closed form
- Example: $\mathbf{x}_i \in R^3, \phi(\mathbf{x}_i) \in R^{10}$

$$\phi(\mathbf{x}_i) = (1, \sqrt{2}(x_i)_1, \sqrt{2}(x_i)_2, \sqrt{2}(x_i)_3, (x_i)_1^2,$$
$$(x_i)_2^2, (x_i)_3^2, \sqrt{2}(x_i)_1(x_i)_2, \sqrt{2}(x_i)_1(x_i)_3, \sqrt{2}(x_i)_2(x_i)_3)$$

Then $\phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j) = (1 + \mathbf{x}_i^T \mathbf{x}_j)^2$.

- Kernel: $K(\mathbf{x}, \mathbf{y}) = \phi(\mathbf{x})^T \phi(\mathbf{y})$; common kernels:

$$e^{-\gamma \|\mathbf{x}_i - \mathbf{x}_j\|^2}, \text{ (Radial Basis Function)}$$
$$(\mathbf{x}_i^T \mathbf{x}_j / a + b)^d \text{ (Polynomial kernel)}$$

Can be inner product in infinite dimensional space

Assume $x \in R^1$ and $\gamma > 0$.

$$e^{-\gamma\|x_i - x_j\|^2} = e^{-\gamma(x_i - x_j)^2} = e^{-\gamma x_i^2 + 2\gamma x_i x_j - \gamma x_j^2}$$

$$= e^{-\gamma x_i^2 - \gamma x_j^2}\left(1 + \frac{2\gamma x_i x_j}{1!} + \frac{(2\gamma x_i x_j)^2}{2!} + \frac{(2\gamma x_i x_j)^3}{3!} + \cdots\right)$$

$$= e^{-\gamma x_i^2 - \gamma x_j^2}\left(1 \cdot 1 + \sqrt{\frac{2\gamma}{1!}} x_i \cdot \sqrt{\frac{2\gamma}{1!}} x_j + \sqrt{\frac{(2\gamma)^2}{2!}} x_i^2 \cdot \sqrt{\frac{(2\gamma)^2}{2!}} x_j^2\right.$$

$$\left. + \sqrt{\frac{(2\gamma)^3}{3!}} x_i^3 \cdot \sqrt{\frac{(2\gamma)^3}{3!}} x_j^3 + \cdots\right) = \phi(x_i)^T \phi(x_j),$$

where

$$\phi(x) = e^{-\gamma x^2}\left[1, \sqrt{\frac{2\gamma}{1!}} x, \sqrt{\frac{(2\gamma)^2}{2!}} x^2, \sqrt{\frac{(2\gamma)^3}{3!}} x^3, \cdots\right]^T.$$

More about Kernels

- How do we know kernels help to separate data?
- In R^l, any l independent vectors

  $\Rightarrow$ linearly separable

$$\begin{bmatrix} (\mathbf{x}^1)^T \\ \vdots \\ (\mathbf{x}^l)^T \end{bmatrix} \mathbf{w} = \begin{bmatrix} +\mathbf{e} \\ -\mathbf{e} \end{bmatrix}$$

- If K positive definite $\Rightarrow$ data linearly separable

  $K = LL^T$.

  Transforming training points to independent vectors in R^l

- So what kind of kernel should I use?
- What kind of functions are valid kernels?
- How to decide kernel parameters?
- Will be discussed later

Decision function

- At optimum

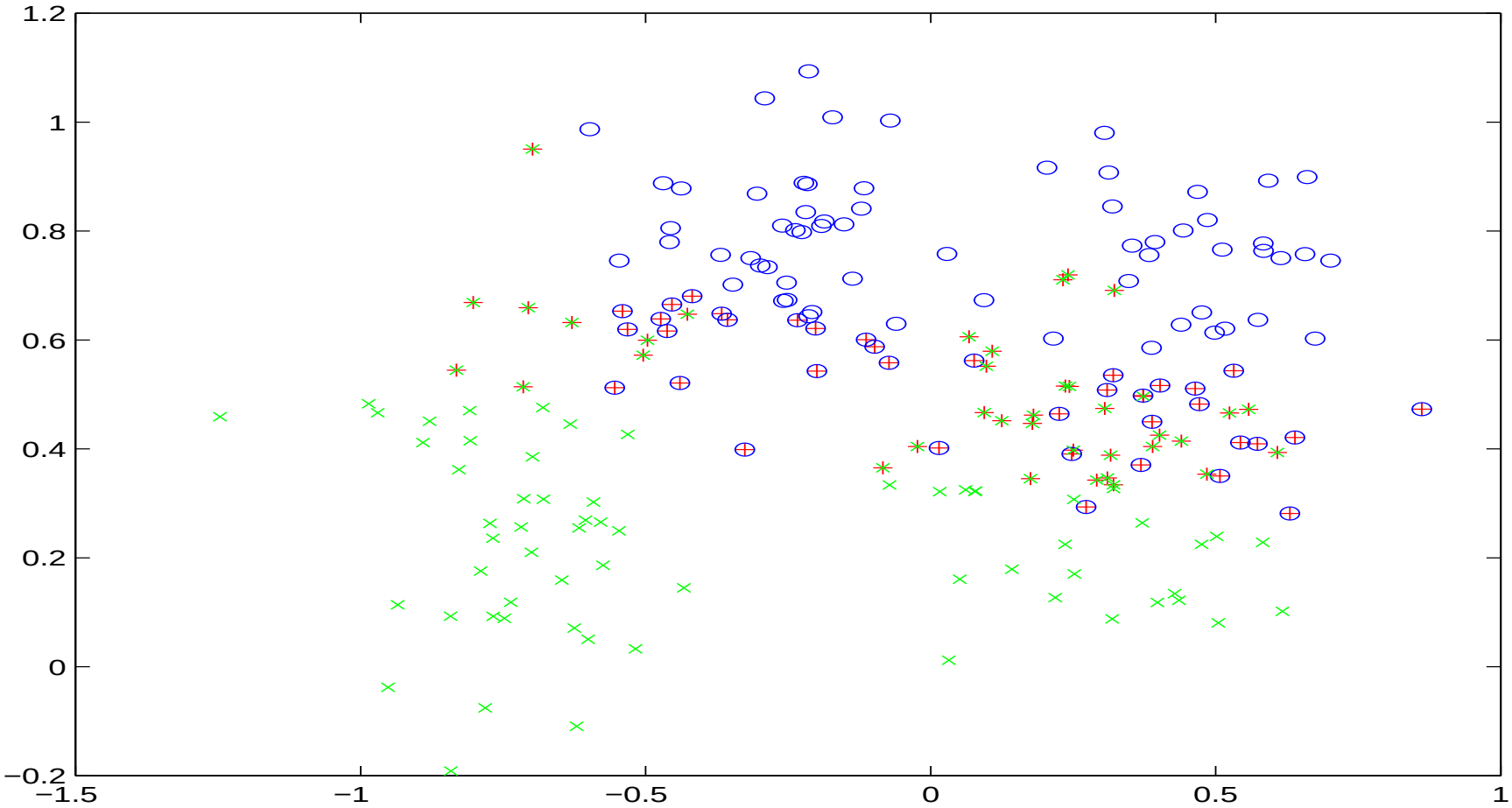
$$\mathbf{w} = \sum_{i=1}^{l} \alpha_i y_i \phi(\mathbf{x}_i)$$

- Decision function

$$\begin{aligned} & \mathbf{w}^T \phi(\mathbf{x}) + b \\ = & \sum_{i=1}^{l} \alpha_i y_i \phi(\mathbf{x}_i)^T \phi(\mathbf{x}) + b \\ = & \sum_{i=1}^{l} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b \end{aligned}$$

- Only $\phi(\mathbf{x}_i)$ of $\alpha_i > 0$ used $\Rightarrow$ support vectors

Support Vectors: More Important Data



- So we have roughly shown basic ideas of SVM
- A 3-D demonstration
  www.csie.ntu.edu.tw/~cjlin/libsvmtools/svmtoy3d
- Further references, for example,
  [Cristianini and Shawe-Taylor, 2000,
  Schölkopf and Smola, 2002]
- Also see discussion on kernel machines blackboard
  www.kernel-machines.org/phpbb/

Outline

- Basic concepts
- SVM primal/dual problems
- Training linear and nonlinear SVMs
- Parameter/kernel selection and practical issues
- Multi-class classification
- Discussion and conclusions

Deriving the Dual

- Consider the problem without ξ_i

$$\begin{aligned} \min_{\mathbf{w},b} \quad & \frac{1}{2}\mathbf{w}^T\mathbf{w} \\ \text{subject to} \quad & y_i(\mathbf{w}^T\phi(\mathbf{x}_i)+b) \geq 1, i = 1,\ldots,l. \end{aligned}$$

- Its dual

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \quad & \frac{1}{2}\boldsymbol{\alpha}^T Q\boldsymbol{\alpha} - \mathbf{e}^T\boldsymbol{\alpha} \\ \text{subject to} \quad & 0 \leq \alpha_i, \qquad i = 1,\ldots,l, \\ & \mathbf{y}^T\boldsymbol{\alpha} = 0. \end{aligned}$$

Lagrangian Dual

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

where

$$L(\mathbf{w}, b, \alpha) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{l} \alpha_i \left(y_i(\mathbf{w}^T \phi(\mathbf{x}_i) + b) - 1 \right)$$

Strong duality (be careful about this)

$$\min \ \text{Primal} = \max_{\alpha \geq 0} \left(\min_{\mathbf{w}, b} L(\mathbf{w}, b, \alpha) \right)$$

- Simplify the dual. When $\boldsymbol{\alpha}$ is fixed,

$$\min_{\mathbf{w},b} L(\mathbf{w}, b, \boldsymbol{\alpha}) =$$

$$\begin{cases} -\infty & \text{if } \sum_{i=1}^{l} \alpha_i y_i \neq 0 \\ \min_{\mathbf{w}} \frac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_{i=1}^{l} \alpha_i [y_i(\mathbf{w}^T\phi(\mathbf{x}_i) - 1] & \text{if } \sum_{i=1}^{l} \alpha_i y_i = 0 \end{cases}$$

- If $\sum_{i=1}^{l} \alpha_i y_i \neq 0$, decrease

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

in $L(\mathbf{w}, b, \boldsymbol{\alpha})$ to $-\infty$

- If $\sum_{i=1}^{l} \alpha_i y_i = 0$, optimum of the strictly convex $\frac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_{i=1}^{l} \alpha_i[y_i(\mathbf{w}^T\phi(\mathbf{x}_i) - 1]$ happens when

$$\frac{\partial}{\partial \mathbf{w}} L(\mathbf{w}, b, \boldsymbol{\alpha}) = 0.$$

- Thus,

$$\mathbf{w} = \sum_{i=1}^{l} \alpha_i y_i \phi(\mathbf{x}_i).$$

- Note that

$$\begin{aligned}\mathbf{w}^T\mathbf{w} &= \left(\sum_{i=1}^{l}\alpha_i y_i\phi(\mathbf{x}_i)\right)^T\left(\sum_{j=1}^{l}\alpha_j y_j\phi(\mathbf{x}_j)\right)\\ &= \sum_{i,j}\alpha_i\alpha_j y_i y_j\phi(\mathbf{x}_i)^T\phi(\mathbf{x}_j)\end{aligned}$$

- The dual is

$$\max_{\boldsymbol{\alpha}\geq 0}\begin{cases}\sum_{i=1}^{l}\alpha_i - \frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j\phi(\mathbf{x}_i)^T\phi(\mathbf{x}_j) & \text{if } \sum_{i=1}^{l}\alpha_i y_i = 0,\\ -\infty & \text{if } \sum_{i=1}^{l}\alpha_i y_i \neq 0.\end{cases}$$

- Lagrangian dual: $\max_{\boldsymbol{\alpha}\geq 0}\left(\min_{\mathbf{w},b} L(\mathbf{w}, b, \boldsymbol{\alpha})\right)$
- $-\infty$ definitely not maximum of the dual
  Dual optimal solution not happen when

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

.

- Dual simplified to

$$\max_{\boldsymbol{\alpha}\in R^l} \quad \sum_{i=1}^{l} \alpha_i - \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} \alpha_i\alpha_j y_i y_j \phi(\mathbf{x}_i)^T\phi(\mathbf{x}_j)$$

$$\text{subject to} \quad \mathbf{y}^T\boldsymbol{\alpha} = 0,$$

$$\alpha_i \geq 0, i = 1, \ldots, l.$$

More about Dual Problems

- After SVM is popular

  Quite a few people think that for any optimization problem

  $\Rightarrow$ Lagrangian dual exists and strong duality holds
- Wrong! We usually need

  Convex programming; Constraint qualification
- We have them

  SVM primal is convex; Linear constraints

- Our problems may be infinite dimensional
- Can still use Lagrangian duality

  See a rigorous discussion in [Lin, 2001]

Outline

- Basic concepts
- SVM primal/dual problems
- Training linear and nonlinear SVMs
- Parameter/kernel selection and practical issues
- Multi-class classification
- Discussion and conclusions

Training Nonlinear SVMs

- If using kernels, we solve the dual

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \quad & \frac{1}{2}\boldsymbol{\alpha}^T Q \boldsymbol{\alpha} - \mathbf{e}^T \boldsymbol{\alpha} \\ \text{subject to} \quad & 0 \le \alpha_i \le C, i = 1, \ldots, l \\ & \mathbf{y}^T \boldsymbol{\alpha} = 0 \end{aligned}$$

- Large dense quadratic programming
- $Q_{ij} \neq 0$, Q : an l by l fully dense matrix
- 30,000 training points: 30,000 variables:
  $(30,000^2 \times 8/2)$ bytes $=$ 3GB RAM to store Q:
- Traditional methods:
  Newton, Quasi Newton cannot be directly applied

Decomposition Methods

- Working on some variables each time (e.g., [Osuna et al., 1997, Joachims, 1998, Platt, 1998])
- Similar to coordinate-wise minimization
- Working set B, $N = \{1, \ldots, l\} \backslash B$ fixed
- Sub-problem at each iteration:

$$\min_{\boldsymbol{\alpha}_B} \quad \frac{1}{2} \begin{bmatrix} \boldsymbol{\alpha}_B^T & (\boldsymbol{\alpha}_N^k)^T \end{bmatrix} \begin{bmatrix} Q_{BB} & Q_{BN} \\ Q_{NB} & Q_{NN} \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha}_B \\ \boldsymbol{\alpha}_N^k \end{bmatrix} - \begin{bmatrix} \mathbf{e}_B^T & (\mathbf{e}_N^k)^T \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha}_B \\ \boldsymbol{\alpha}_N^k \end{bmatrix}$$

$$\text{subject to} \quad 0 \le \alpha_t \le C, t \in B, \ \mathbf{y}_B^T \boldsymbol{\alpha}_B = -\mathbf{y}_N^T \boldsymbol{\alpha}_N^k$$

Avoid Memory Problems

- The new objective function

$$\frac{1}{2}\boldsymbol{\alpha}_B^T Q_{BB}\boldsymbol{\alpha}_B + (-\mathbf{e}_B + Q_{BN}\boldsymbol{\alpha}_N^k)^T\boldsymbol{\alpha}_B + \text{ constant}$$

- B columns of Q needed
- Calculated when used

  Trade time for space

SVM doesn't Scale Up

Yes, if you use kernels

- Training millions of data is time consuming
- But other nonlinear methods face the same problem

  e.g., kernel logistic regression

Two possibilities

1. Linear SVMs: in some situations, can solve much larger problems
2. Approximation

Training Linear SVMs

- Linear kernel:

$$\min_{\mathbf{w},b,\boldsymbol{\xi}} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{l}\xi_i$$

$$\text{subject to} \quad y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i, \qquad \xi_i \geq 0.$$

- At optimum:

$$\xi_i = \max\big(0, 1 - y_i(\mathbf{w}^T\mathbf{x}_i + b)\big)$$

- Remaining variables: $\mathbf{w}, b$

$$\min_{\mathbf{w},b} \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{l}\max\left(0, 1 - y_i(\mathbf{w}^T\mathbf{x}_i + b)\right)$$

- #variables = #features + 1
- If #features small, easier to solve

- Traditional optimization methods can be applied
- Training time similar to methods such as logistic regression
- What if #features and #instances both large?
  Very challenging
- Some language/document problems are of this type

Multi-class Classification

- k classes
- One-against-the rest: Train k binary SVMs:

$$\begin{array}{lll} \text{1st class} & \text{vs.} & (2-k)\text{th class} \\ \text{2nd class} & \text{vs.} & (1, 3-k)\text{th class} \\ & \vdots & \end{array}$$

- k decision functions

$$(\mathbf{w}^1)^T\phi(\mathbf{x}) + b_1$$
$$\vdots$$
$$(\mathbf{w}^k)^T\phi(\mathbf{x}) + b_k$$

- Prediction:

$$\arg\max_j \ (\mathbf{w}^j)^T \phi(\mathbf{x}) + b_j$$

- Reason: If the 1st class, then we should have

$$\begin{aligned}
(\mathbf{w}^1)^T \phi(\mathbf{x}) + b_1 &\geq +1 \\
(\mathbf{w}^2)^T \phi(\mathbf{x}) + b_2 &\leq -1 \\
&\vdots \\
(\mathbf{w}^k)^T \phi(\mathbf{x}) + b_k &\leq -1
\end{aligned}$$

Multi-class Classification (Cont'd)

- One-against-one: train $k(k-1)/2$ binary SVMs
  $(1,2),(1,3),\ldots,(1,k),(2,3),(2,4),\ldots,(k-1,k)$
- If 4 classes $\Rightarrow$ 6 binary SVMs

$y_i = 1$	$y_i = -1$	Decision functions
class 1	class 2	$f^{12}(\mathbf{x}) = (\mathbf{w}^{12})^T\mathbf{x} + b^{12}$
class 1	class 3	$f^{13}(\mathbf{x}) = (\mathbf{w}^{13})^T\mathbf{x} + b^{13}$
class 1	class 4	$f^{14}(\mathbf{x}) = (\mathbf{w}^{14})^T\mathbf{x} + b^{14}$
class 2	class 3	$f^{23}(\mathbf{x}) = (\mathbf{w}^{23})^T\mathbf{x} + b^{23}$
class 2	class 4	$f^{24}(\mathbf{x}) = (\mathbf{w}^{24})^T\mathbf{x} + b^{24}$
class 3	class 4	$f^{34}(\mathbf{x}) = (\mathbf{w}^{34})^T\mathbf{x} + b^{34}$

- For a testing data, predicting all binary SVMs

Classes		winner
1	2	1
1	3	1
1	4	1
2	3	2
2	4	4
3	4	3

- Select the one with the largest vote

class	1	2	3	4
# votes	3	1	1	1

- May use decision values as well

More Complicated Forms

- For example,
  [Vapnik, 1998, Weston and Watkins, 1999]:

$$\min_{\mathbf{w},b,\xi} \quad \frac{1}{2}\sum_{m=1}^{k}\mathbf{w}_m^T\mathbf{w}_m + C\sum_{i=1}^{l}\sum_{m\neq y_i}\xi_i^m$$

$$\mathbf{w}_{y_i}^T\phi(\mathbf{x}_i) + b_{y_i} \geq \mathbf{w}_m^T\phi(\mathbf{x}_i) + b_m + 2 - \xi_i^m,$$

$$\xi_i^m \geq 0, i = 1,\ldots,l,\ m \in \{1,\ldots,k\}\backslash y_i.$$

  y_i: class of $\mathbf{x}_i$

- kl constraints
- Dual: kl variables; very large

- There are many other methods
- A comparison in [Hsu and Lin, 2002]
- Accuracy similar for many problems

  But 1-against-1 fastest for training

Why 1vs1 Faster in Training

- 1 vs. 1

  $k(k-1)/2$ problems, each $2l/k$ data on average
- 1 vs. all

  k problems, each l data
- If solving the optimization problem:

  polynomial of the size with degree d
- Their complexities

$$\frac{k(k-1)}{2} O\left(\left(\frac{2l}{k}\right)^d\right) \quad \text{vs.} \quad kO(l^d)$$

References I


Bakır, G. H., Bottou, L., and Weston, J. (2005).
Breaking svm complexity with cross-training.
In Saul, L. K., Weiss, Y., and Bottou, L., editors, *Advances in Neural Information Processing Systems 17*, pages 81–88. MIT Press, Cambridge, MA.


Boser, B., Guyon, I., and Vapnik, V. (1992).
A training algorithm for optimal margin classifiers.
In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, pages 144–152. ACM Press.


Chu, W., Keerthi, S., and Ong, C. (2003).
Bayesian trigonometric support vector classifier.
Neural Computation, 15(9):2227–2254.


Cortes, C., Haffner, P., and Mohri, M. (2003).
Positive definite rational kernels.
In *Proceedings of the 16th Annual Conference on Learning Theory*, pages 41–56.


Cortes, C. and Vapnik, V. (1995).
Support-vector network.
Machine Learning, 20:273–297.

References II


Cristianini, N. and Shawe-Taylor, J. (2000).
An Introduction to Support Vector Machines.
Cambridge University Press, Cambridge, UK.


Hsu, C.-W. and Lin, C.-J. (2002).
A comparison of methods for multi-class support vector machines.
IEEE Transactions on Neural Networks, 13(2):415–425.


Joachims, T. (1998).
Making large-scale SVM learning practical.
In Schölkopf, B., Burges, C. J. C., and Smola, A. J., editors, *Advances in Kernel Methods - Support Vector Learning*, Cambridge, MA. MIT Press.


Kao, W.-C., Chung, K.-M., Sun, C.-L., and Lin, C.-J. (2004).
Decomposition methods for linear support vector machines.
Neural Computation, 16(8):1689–1704.


Keerthi, S. S., Chapelle, O., and DeCoste, D. (2006).
Building support vector machines with reduced classifier complexity.
Journal of Machine Learning Research, 7:1493–1515.

References III


Keerthi, S. S. and Lin, C.-J. (2003).
Asymptotic behaviors of support vector machines with Gaussian kernel.
Neural Computation, 15(7):1667–1689.


Lanckriet, G., Cristianini, N., Bartlett, P., El Ghaoui, L., and Jordan, M. (2004).
Learning the Kernel Matrix with Semidefinite Programming.
Journal of Machine Learning Research, 5:27–72.


Lee, Y.-J. and Mangasarian, O. L. (2001).
RSVM: Reduced support vector machines.
In *Proceedings of the First SIAM International Conference on Data Mining*.


Lin, C.-J. (2001).
Formulations of support vector machines: a note from an optimization point of view.
Neural Computation, 13(2):307–317.


Lin, C.-J. (2002).
A formal analysis of stopping criteria of decomposition methods for support vector machines.
IEEE Transactions on Neural Networks, 13(5):1045–1052.

References IV


Osuna, E., Freund, R., and Girosi, F. (1997).
Training support vector machines: An application to face detection.
In *Proceedings of CVPR'97*, pages 130–136, New York, NY. IEEE.


Platt, J. C. (1998).
Fast training of support vector machines using sequential minimal optimization.
In Schölkopf, B., Burges, C. J. C., and Smola, A. J., editors, *Advances in Kernel Methods - Support Vector Learning*, Cambridge, MA. MIT Press.


Schölkopf, B. and Smola, A. J. (2002).
Learning with kernels.
MIT Press.


Syed, N. A., Liu, H., and Sung, K. K. (1999).
Incremental learning with support vector machines.
In *Workshop on Support Vector Machines, IJCAI99*.


Vapnik, V. (1998).
Statistical Learning Theory.
Wiley, New York, NY.

References V

Weston, J. and Watkins, C. (1999).
Multi-class support vector machines.
In Verleysen, M., editor, *Proceedings of ESANN99*, Brussels. D. Facto Press.

Yu, H., Yang, J., and Han, J. (2003).
Classifying large data sets using svms with hierarchical clusters.
In *KDD '03: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 306–315, New York, NY, USA. ACM Press.