Support Vector Machines for Data Classification

Chih-Jen Lin

Department of Computer Science

National Taiwan University

Talk at Centrum voor Wiskunde en Informatica, February 9, 2004

Outline

- Support vector classification
- Example: engine misfire detection
- Discussion and conclusions

Data Classification

- Given training data in different classes (labels known)

  Predict test data (labels unknown)
- Examples
  - Handwritten digits recognition
  - Spam filtering
- Training and testing
- Methods:
  - Nearest Neighbor
  - Neural Networks
  - Decision Tree

- Support vector machines: a new method

  Becoming more and more popular

  We will discuss its current status

- A good classification method:
  - Avoid underfitting: small training error
  - Avoid overfitting: small testing error

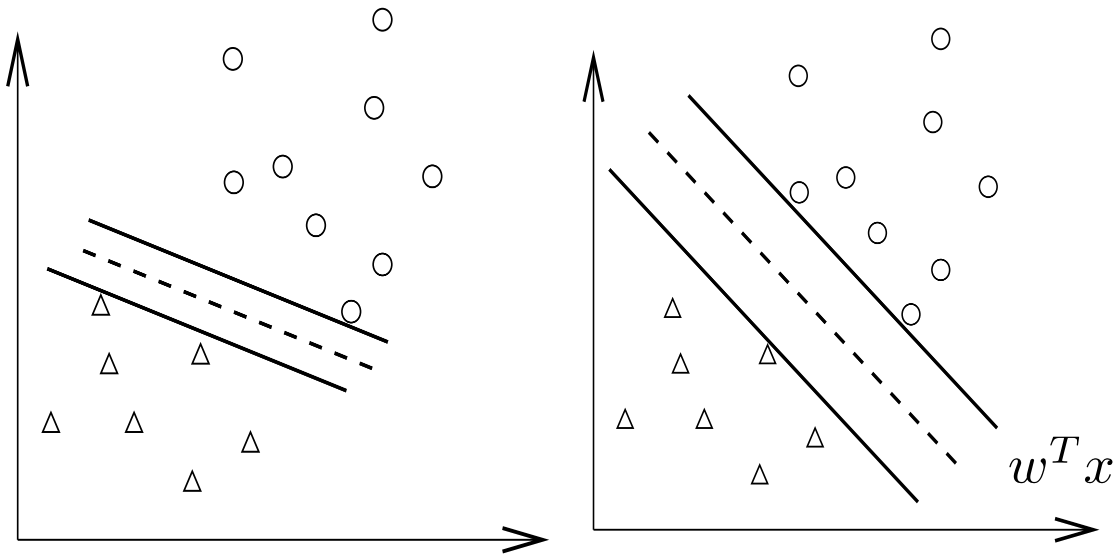
Support Vector Classification

- Training vectors : $x_i, i = 1, \ldots, l$
- Consider a simple case with two classes:

  Define a vector y

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

- A hyperplane which separates all data



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

- A separating hyperplane: $w^T x + b = 0$

$$(w^T x_i) + b > 0 \quad \text{if } y_i = 1$$
$$(w^T x_i) + b < 0 \quad \text{if } y_i = -1$$

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

  Variables: w and b : Need to know coefficients of a plane

  Many possible choices of w and b

- Select w, b with the maximal margin.

  Maximal distance between $w^T x + b = \pm 1$

  Vapnik's statistical learning theory.

$$
\begin{aligned}
(w^T x_i) + b &\geq 1 \quad \text{if } y_i = 1 \\
(w^T x_i) + b &\leq -1 \quad \text{if } y_i = -1
\end{aligned} \tag{1}
$$

- Distance between $w^T x + b = 1$ and -1:

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

- $\max 2/\|w\| \equiv \min w^T w/2$

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

Higher Dimensional Feature Spaces

- Earlier we tried to find a linear separating hyperplane

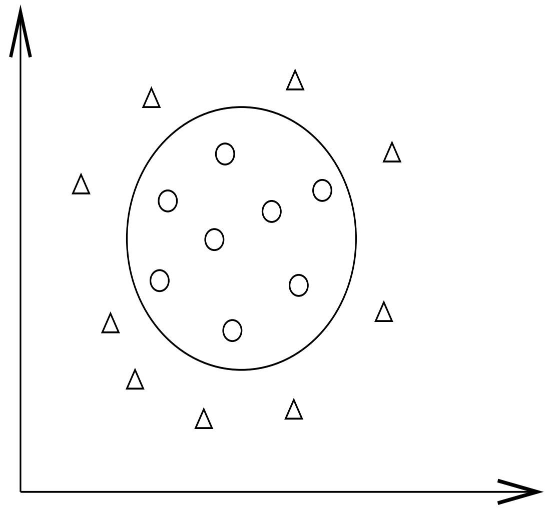
  Data may not be linear separable

- Non-separable case: allow training errors

$$
\begin{array}{ll}
\min_{w,b,\xi} & \frac{1}{2} w^T w + C \sum_{i=1}^{l} \xi_i \\
\text{subject to} & y_i((w^T x_i) + b) \geq 1 - \xi_i, \\
& \xi_i \geq 0, \ i = 1, \ldots, l
\end{array}
$$

- $\xi_i > 1$, x_i not on the correct side of the separating plane
- C: large penalty parameter, most ξ_i are zero

- Avoid underfitting; nonlinear separating hyperplane linear separable in other spaces ?



- Higher dimensional (maybe infinite) feature space

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

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

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

- A standard problem [Cortes and Vapnik, 1995]:

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

- C: adjust "training error" and "generalization"

Finding the Decision Function

- w: a vector in a high dimensional space
  $\Rightarrow$ maybe infinite variables
- The dual problem

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

  where $Q_{ij} = y_i y_j \phi(x_i)^T \phi(x_j)$ and $e = [1, \ldots, 1]^T$

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

- Primal and dual: optimization theory. Not trivial.
  Infinite dimensional programming.

- A finite problem:

  #variables = #training data

- $Q_{ij} = y_i y_j \phi(x_i)^T \phi(x_j)$ needs a closed form

  Efficient calculation of high dimensional inner products

  Kernel trick, $K(x_i, x_j) = \phi(x_i)^T \phi(x_j)$

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

$$\begin{aligned}\phi(x_i) \quad &= \quad (1, \sqrt{2}(x_i)_1, \sqrt{2}(x_i)_2, \sqrt{2}(x_i)_3, (x_i)_1^2, \\ & \quad\quad (x_i)_2^2, (x_i)_3^2, \sqrt{2}(x_i)_1(x_i)_2, \sqrt{2}(x_i)_1(x_i)_3, \sqrt{2}(x_i)_2(x_i)_3),\end{aligned}$$

  Then $\phi(x_i)^T \phi(x_j) = (1 + x_i^T x_j)^2$.

- Popular methods: $K(x_i, x_j) =$

$$e^{-\gamma\|x_i - x_j\|^2}, \text{ (Radial Basis Function)}$$

$$(x_i^T x_j/a + b)^d \text{ (Polynomial kernel)}$$

- Decision function:

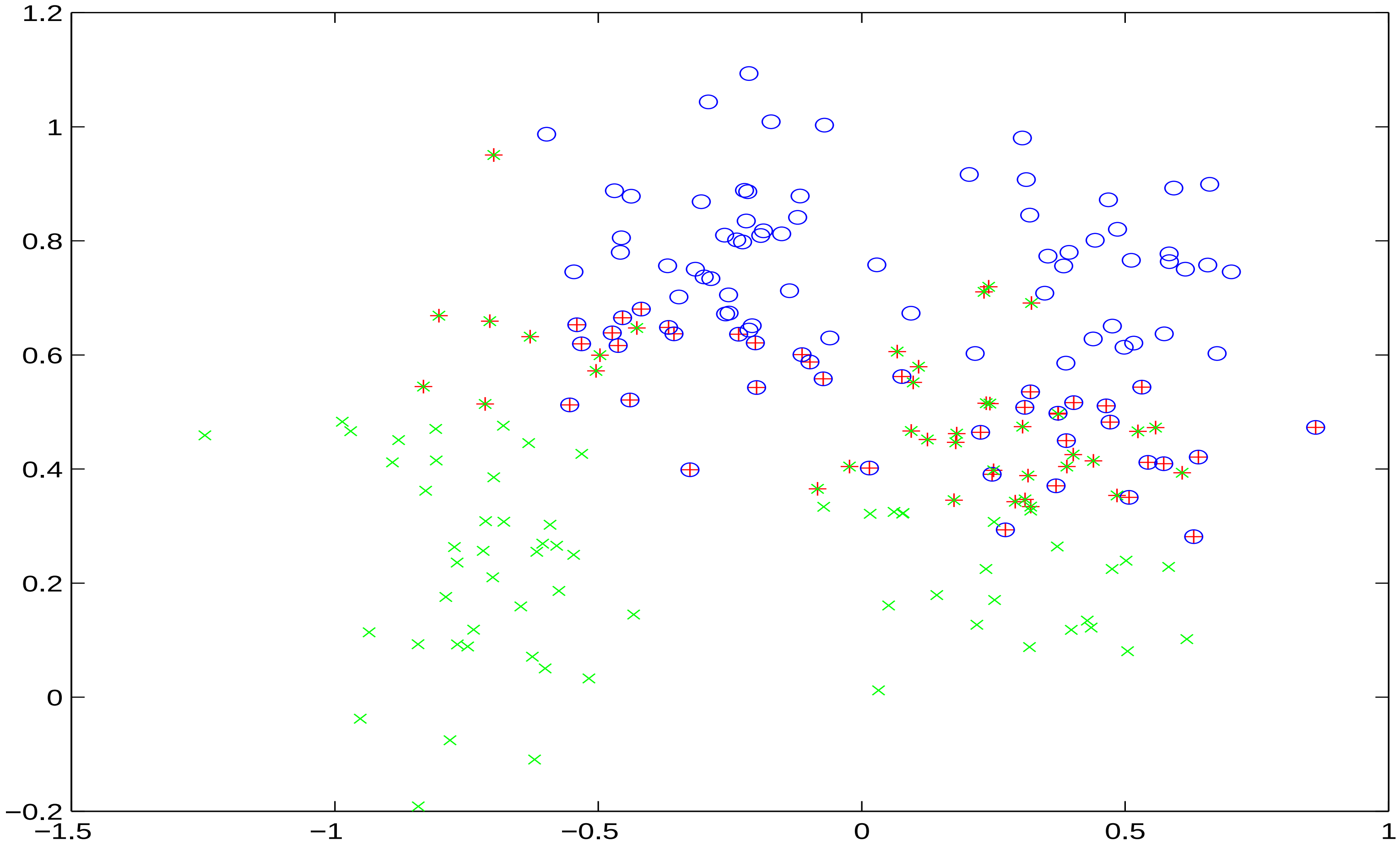
$$\begin{aligned} & w^T\phi(x) + b \\ = & \sum_{i=1}^{l} \alpha_i y_i \phi(x_i)^T \phi(x) + b \end{aligned}$$

No need to have w

- > 0: 1st class, < 0: 2nd class
- Only $\phi(x_i)$ of $\alpha_i > 0$ used

$\alpha_i > 0 \Rightarrow$ support vectors

Support Vectors: More Important Data



Example: Engine Misfire Detection

- First problem of IJCNN Challenge 2001, data from Ford
- Given time series length $T = 50,000$
- The kth data

$$x_1(k), x_2(k), x_3(k), x_4(k), x_5(k), y(k)$$

- Example:

```
0.000000 -0.999991 0.169769 0.000000 1.000000
0.000000 -0.659538 0.169769 0.000292 1.000000
0.000000 -0.660738 0.169128 -0.020372 1.000000
1.000000 -0.660307 0.169128 0.007305 1.000000
0.000000 -0.660159 0.169525 0.002519 1.000000
0.000000 -0.659091 0.169525 0.018198 1.000000
0.000000 -0.660532 0.169525 -0.024526 1.000000
0.000000 -0.659798 0.169525 0.012458 1.000000
```

- $y(k) = \pm 1$: output, affected only by $x_1(k), \ldots, x_4(k)$

- $x_5(k)$: not related to the output

  $x_5(k) = 1$, kth data considered for evaluating accuracy

  i.e., not used for testing; can still be used in training
- 50,000 training data, 100,000 testing data (in two sets)
- Past and future information may affect $y(k)$
- $x_1(k)$: periodically nine 0s, one 1, nine 0s, one 1, and so on.
- $x_4(k)$ more important

Background: Engine Misfire Detection

- Known after the competition
- Engine misfire: a substantial fraction of a cylinder's air-fuel mixture fails to ignite
- Frequent misfires: pollutants and costly replacement
- On-board detection:
  Engine crankshaft rational dynamics with a position sensor
- Training data: from some expensive experimental environment

Encoding Schemes

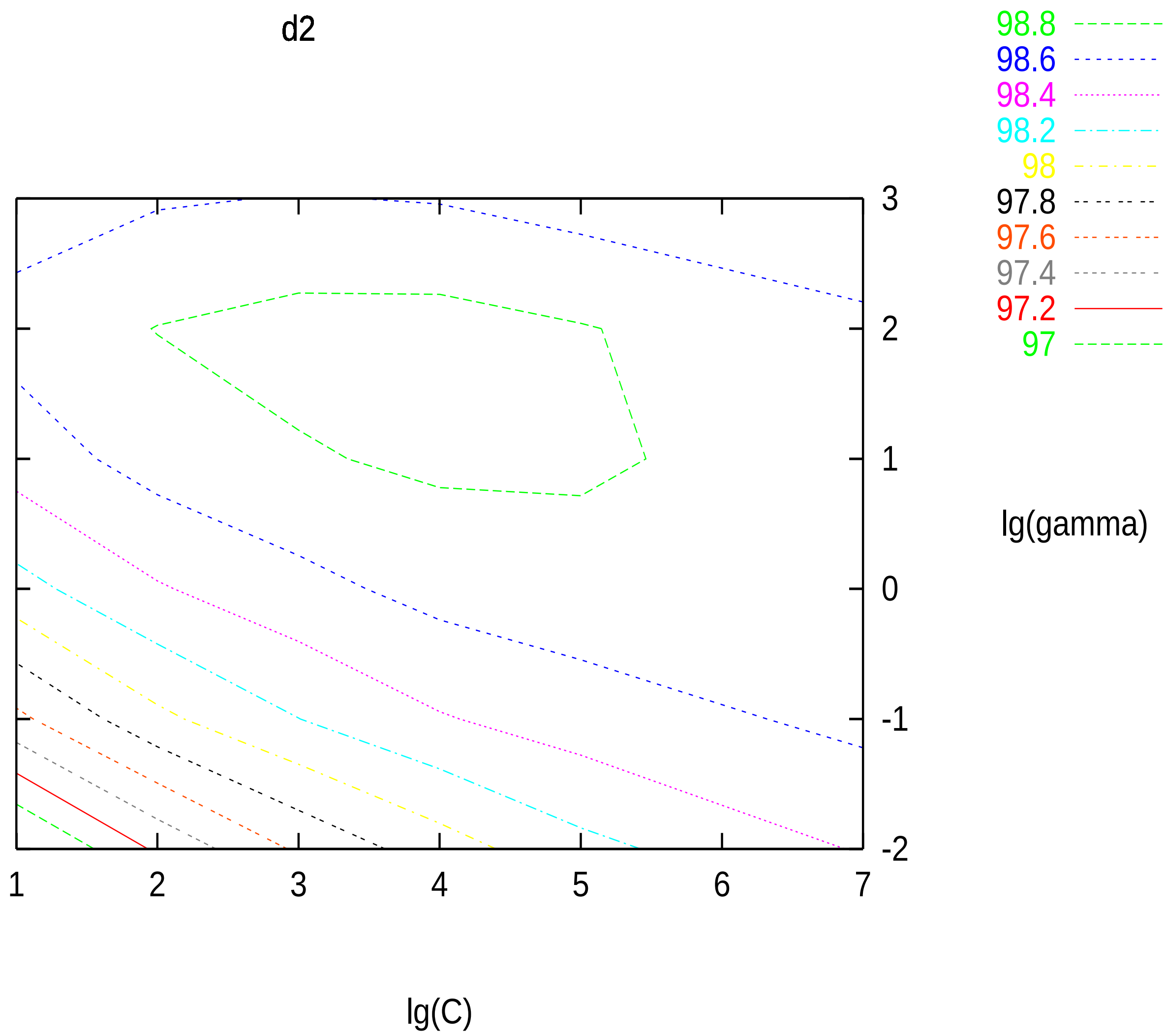
- For SVM: each data is a vector
- $x_1(k)$: periodically nine 0s, one 1, nine 0s, one 1, ...
  - 10 binary attributes
    $x_1(k-5), \ldots, x_1(k+4)$ for the kth data
  - $x_1(k)$: an integer in 1 to 10
  - Which one is better
  - We think 10 binaries better for SVM
- $x_4(k)$ more important
  Including $x_4(k-5), \ldots, x_4(k+4)$ for the kth data
- Each training data: 22 attributes

Training SVM

- Selecting parameters; generating a good model for prediction
- RBF kernel $K(x_i, x_j) = \phi(x_i)^T \phi(x_j) = e^{-\gamma \|x_i - x_j\|^2}$
- Two parameters: γ and C
- Five-fold cross validation on 50,000 data

  Data randomly separated to five groups.

  Each time four as training and one as testing
- Use $C = 2^4, \gamma = 2^2$ and train 50,000 data for the final model


d2
98.8
98.6
98.4
98.2
98
97.8
97.6
97.4
97.2
97
3
2
1
0
-1
-2
lg(gamma)
1
2
3
4
5
6
7
lg(C)

- Test set 1: 656 errors, Test set 2: 637 errors
- About 3000 support vectors of 50,000 training data

  A good case for SVM
- This is just the outline. There are other details.
- It is essential to do model selection.

A General Procedure

1. Conduct simple scaling on the data
2. Consider RBF kernel $K(x, y) = e^{-\gamma \|x-y\|^2}$
3. Use cross-validation to find the best parameter C and γ
4. Use the best C and γ to train the whole training set
5. Test

- Best C and γ by training $k - 1$ and the whole ?

  In theory, a minor difference

  No problem in practice
- If accuracy still not satisfactory, further techniques needed

A Software: LIBSVM

- A library for SVM (in both C++ and Java)

  http://www.csie.ntu.edu.tw/~cjlin/libsvm

  - Classification and regression
  - Scripts for procedures mentioned above
- Interfaces:
  - Matlab: developed at Ohio State University
  - R (and S-Plus): developed at Technische Universität Wien
  - Python: developed at HP Labs.
  - Perl: developed at Simon Fraser University
  - Ruby: developed at CWI

- Used in many integrated machine learning/data mining packages

Current Status of SVM

- In my opinion, after careful data pre-processing
  Appropriately use NN or SVM $\Rightarrow$ similar accuracy
- But, users may not use them properly
- The chance of SVM
  Easier for users to appropriately use it
- The ambition: replacing part of NN
  (i.e., replacing it on some applications)

Discussion and Conclusions

- SVM: a simple and effective classification method
- Applications: key to improve SVM
- All my research results can be found at
  `http://www.csie.ntu.edu.tw/~cjlin`