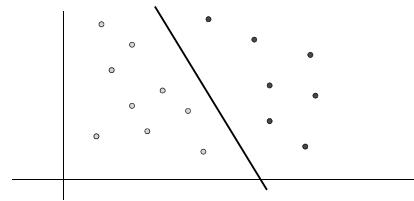


CS522 Advanced Database Systems
 Classification: Introduction to Support Vector Machine

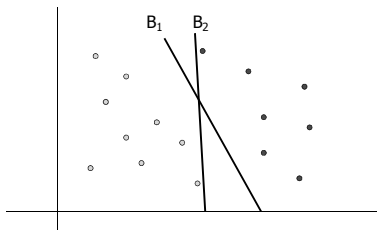
Chengyu Sun
 California State University, Los Angeles

Support Vector Machine (SVM)

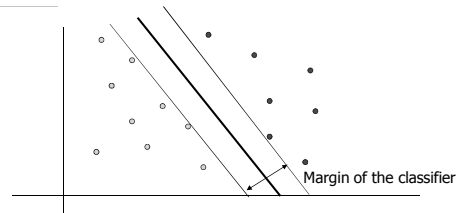


- Find a hyperplane (decision boundary) that will separate the data.

Which Boundary Is Better?



Maximum Margin Hyperplane

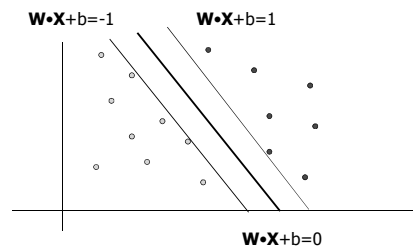


- Maximum margin hyperplane (MMH) minimizes the worst-case generalization error.

Linear SVM ...

- Binary classification
- Record: $\{x_1, x_2, \dots, x_n, y\}$
 - Attribute values: $\mathbf{X} = (x_1, x_2, \dots, x_n)$
 - Class label: $y \in \{1, -1\}$
- Decision boundary: $\mathbf{W} \cdot \mathbf{X} + b = 0$
- Classification
 - $y = 1$ if $\mathbf{W} \cdot \mathbf{X} + b > 0$
 - $y = -1$ if $\mathbf{W} \cdot \mathbf{X} + b < 0$

... Linear SVM



$$\text{Margin} = \frac{2}{\|\mathbf{W}\|} = \frac{2}{\sqrt{w_1^2 + w_2^2 + \dots + w_n^2}}$$

SVM Training Example

	X1	X2	X3	Y
R1	0	1	0	1
R2	1	1	0	-1
R3	0	0	1	1
R4	1	0	1	-1
R5	0	0	0	-1

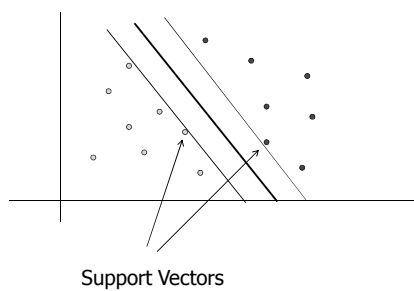
- R1: $w_2 + b \geq 1$ R4: $w_1 + w_3 + b \leq -1$
 R2: $w_1 + w_2 + b \leq -1$ R5: $b \leq -1$
 R3: $w_3 + b \geq 1$

Training Linear SVM

- Find \mathbf{W} that satisfies the inequalities and maximize the margin $2/||\mathbf{W}'||$
 - Constrained (convex) quadratic optimization problem
 - Solvable by numerical methods such as quadratic programming

See Chapter 5.5 of Introduction to Data Mining by Tan, Steinbach, and Kumar

Support Vectors

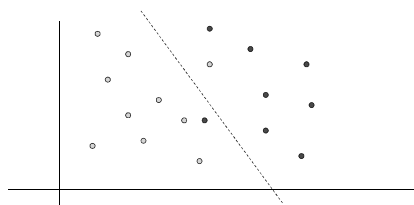


Decision Boundary of Linear SVM

$$\left(\sum_{i=1}^N \lambda_i y_i \mathbf{X}_i \cdot \mathbf{X}\right) + b = 0$$

- (\mathbf{X}_i, y_i) are training records that satisfy $y_i(\mathbf{W} \cdot \mathbf{X}_i + b) = 1$, i.e. *support vectors*

Linear SVM – Non-separable Case



Introduce a Slack Variable ξ

$$\begin{aligned} \mathbf{W} \cdot \mathbf{X}_i + b &\geq 1 && \text{if } y_i = 1 \\ \mathbf{W} \cdot \mathbf{X}_i + b &\leq -1 && \text{if } y_i = -1 \end{aligned}$$



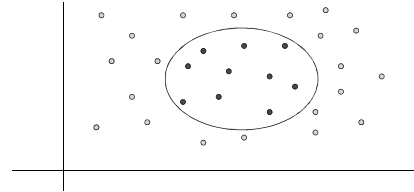
$$\begin{aligned} \mathbf{W} \cdot \mathbf{X}_i + b &\geq 1 - \xi_i && \text{if } y_i = 1 \\ \mathbf{W} \cdot \mathbf{X}_i + b &\leq -1 + \xi_i && \text{if } y_i = -1 \end{aligned}$$

Revise the Objective Function

$$f(\mathbf{W}) = \frac{\|\mathbf{W}\|^2}{2} + C \left(\sum_{i=1}^N \xi_i \right)^k$$

C and k are user specified parameters

Non-linear Decision Boundary



- ◆ Transform the data to another coordinate space so a linear boundary can be found

Transformation Example

Non-linear Decision Boundary in 2D space:

$$(x_1 - 1)^2 + (x_2 - 1)^2 - 1 = 0$$

$$\begin{array}{l} \downarrow \\ x'_1 = x_1 \\ x'_2 = x_2 \\ x'_3 = x_1^2 \\ x'_4 = x_2^2 \end{array}$$

Linear Decision Boundary in 4D space:

$$x'_3 - 2x'_1 + x'_4 + 2x'_2 + 1 = 0$$

Problems of Transformation

- ◆ We don't know the non-linear decision boundary (so we don't know how to do the transformation)
- ◆ Computation becomes more costly with more dimensions

Kernel Function to the Rescue

- ◆ Training records only appear in the optimization process in the form of dot product $\phi(\mathbf{X}_i) \cdot \phi(\mathbf{X}_j)$
 - ϕ is the transformation function
- ◆ Kernel function $K(\mathbf{X}_i, \mathbf{X}_j) = \phi(\mathbf{X}_i) \cdot \phi(\mathbf{X}_j)$
- ◆ So we can do the computation in the original space *without even knowing what the transformation function is*

Kernel Functions

Polynomial kernel of degree h : $K(\mathbf{X}_i, \mathbf{X}_j) = (\mathbf{X}_i \cdot \mathbf{X}_j + 1)^h$

Gaussian radial basis function kernel: $K(\mathbf{X}_i, \mathbf{X}_j) = e^{-\|\mathbf{X}_i - \mathbf{X}_j\|^2 / 2\sigma^2}$

Sigmoid kernel: $K(\mathbf{X}_i, \mathbf{X}_j) = \tanh(\kappa \mathbf{X}_i \cdot \mathbf{X}_j - \delta)$

Kernel Functions and SVM Classifiers

- ◆ Use of different kernel functions result in different classifiers
- ◆ There's no golden rule to determine which kernel function is better
- ◆ The accuracy difference by using different kernel functions is usually not significant in practice

LIBSVM

- ◆ LIBSVM: a Library for Support Vector Machines by *Chih-Chung Chang* and *Chih-Jen Lin*
 - <http://www.csie.ntu.edu.tw/~cjlin/libsvm/>

Multiclass Classification with Binary Classifier

- ◆ Train a number of binary classifiers, each solving a binary classification problem
- ◆ Combine the results to solve the multiclass classification problem

The One-Against-Rest (1-r) Approach

- ◆ For k classes $\{c_1, c_2, \dots, c_k\}$, train k binary classifiers M_i , each classifies $\{c_i, \text{not-}c_i\}$
 - A positive classification by M_i gives one vote to c_i
 - A negative classification by M_i gives one vote to every class other than c_i

1-r Example

- ◆ Three classes c_1, c_2 , and c_3
- ◆ Three classifiers M_1, M_2 , and M_3
- ◆ Classify record r ??

Case 1:	M_1	M_2	M_3	Case 2:	M_1	M_2	M_3
	c_1	not c_2	not c_3		c_1	not c_2	c_3

The One-Against-One (1-1) Approach

- ◆ For k classes $\{c_1, c_2, \dots, c_k\}$, train $k(k-1)/2$ binary classifiers, each classifies $\{c_i, c_j\}$

1-1 Example

- ◆ Three classes c_1 , c_2 , and c_3
- ◆ Three classifiers M_1 , M_2 , and M_3
- ◆ Classify record r ??

Case 1:

M_1	M_2	M_3
$\{c_1, c_2\}$	$\{c_1, c_3\}$	$\{c_2, c_3\}$
c_1	c_1	c_3

Case 2:

M_1	M_2	M_3
$\{c_1, c_2\}$	$\{c_1, c_3\}$	$\{c_2, c_3\}$
c_1	c_3	c_2

Error-Correcting Output Coding (ECOC)

- ◆ Encode each class label with a n -bit code word
- ◆ Train n binary classifiers, one for each bit
- ◆ The predicted class is the one whose codeword is the closest in Hamming distance to the classifiers' output

Error-Correcting Output Coding (ECOC) Example

Class	Codeword
c_1	1 1 1 1 1 1 1
c_2	0 0 0 0 1 1 1
c_3	0 0 1 1 0 0 1
c_4	0 1 0 1 0 1 0

- ◆ Suppose the classifiers' output: 0 1 1 1 1 1 1, what's the predicted class??

About ECOC

- ◆ If d is the minimum distance between any pair of code words, ECOC can correct up to $\lfloor (d-1)/2 \rfloor$ errors
- ◆ There are many algorithms in coding theory to generate n -bit code words with given Hamming distance
- ◆ For multiclass classification, column-wise separation is also important

Readings

- ◆ Textbook Chapter 6.7