Maximum Margin Hyperplane

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

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

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

Training SVM
- | \( X_1 \) | \( X_2 \) | \( X_3 \) | \( Y \) |
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>R1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>R2</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>R3</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>R4</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>R5</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

R1: \( w_1 + b \geq 1 \)  
R2: \( w_1 + w_2 + b \leq -1 \)  
R3: \( w_1 + b \geq 1 \)  
R4: \( w_1 + w_3 + b \leq -1 \)  
R5: \( b \leq -1 \)
Objective Function –
Maximum Margin

\[ W \cdot X + b = 0 \]

\[ W \cdot X + b = 1 \]

\[ W \cdot X + b = -1 \]

Margin = \( \frac{2}{||W||} \)

Solving Linear SVM

- Find \( W \) that satisfies the inequalities and maximize the margin \( \frac{2}{||W||} \)
- 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

Issues To Be Addressed

- Complexity when the training set is large
- Linear Non-separable case
- Non-linear decision boundary

Support Vectors

Support Vectors

Decision Boundary of Linear SVM

\[ \left( \sum_{i=1}^{N} \lambda_i y_i X_i \cdot X \right) + b = 0 \]

\( (X_i, y_i) \) are training records that satisfy \( y_i (W \cdot X_i + b) = 1 \), i.e. support vectors

Linear SVM – Non-separable Case
Introduce a Slack Variable $\xi$

$\mathbf{W} \cdot \mathbf{X}_i + b \geq 1$ if $y_i = 1$
$\mathbf{W} \cdot \mathbf{X}_i + b \leq -1$ if $y_i = -1$

$\mathbf{W} \cdot \mathbf{X}_i + b \geq 1 - \xi_i$ if $y_i = 1$
$\mathbf{W} \cdot \mathbf{X}_i + b \leq -1 + \xi_i$ if $y_i = -1$

Revise the Objective Function

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

$c$ and $\kappa$ 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:

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

$$\begin{align*}
x_1' &= x_1 \\
x_2' &= x_2 \\
x_3' &= x_3 \\
x_4' &= x_4
\end{align*}$$

Linear Decision Boundary in 4D space:

$$x_1' - 2x_2' + x_3' + 2x_4' + 1 = 0$$

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 $\phi(X_i) \cdot \phi(X_j)$
- $\phi$ is the transformation function
- Kernel function $K(X_i, X_j) = \phi(X_i) \cdot \phi(X_j)$
- So we can do the computation in the original space without even knowing what the transformation function is
Kernel Functions

- Polynomial kernel of degree \( d \): 
  \[ K(X_i, X_j) = (X_i \cdot X_j + 1)^d \]

- Gaussian radial basis function kernel: 
  \[ K(X_i, X_j) = e^{-\gamma \|X_i - X_j\|^2} \]

- Sigmoid kernel: 
  \[ K(X_i, X_j) = \tanh(\alpha X_i \cdot X_j - \delta) \]

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

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 \( \{c_1, c_2, ..., c_k\} \), 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 \( r \)?

<table>
<thead>
<tr>
<th>Case 1:</th>
<th>Case 2:</th>
</tr>
</thead>
<tbody>
<tr>
<td>( M_1 )</td>
<td>( M_1 )</td>
</tr>
<tr>
<td>( c_1 )</td>
<td>( c_1 )</td>
</tr>
<tr>
<td>not ( c_2 )</td>
<td>not ( c_2 )</td>
</tr>
<tr>
<td>( M_2 )</td>
<td>( M_2 )</td>
</tr>
<tr>
<td>( c_2 )</td>
<td>( c_2 )</td>
</tr>
<tr>
<td>not ( c_1 )</td>
<td>not ( c_1 )</td>
</tr>
<tr>
<td>( M_3 )</td>
<td>( M_3 )</td>
</tr>
<tr>
<td>( c_3 )</td>
<td>( c_3 )</td>
</tr>
</tbody>
</table>
The One-Against-One (1-1) Approach

- For $k$ classes $\{c_1, c_2, ..., c_k\}$, 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:

<table>
<thead>
<tr>
<th>$M_1$</th>
<th>$M_2$</th>
<th>$M_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$c_1$</td>
<td>$c_1$</td>
<td>$c_3$</td>
</tr>
</tbody>
</table>

Case 2:

<table>
<thead>
<tr>
<th>$M_1$</th>
<th>$M_2$</th>
<th>$M_3$</th>
<th>$M_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$c_1$</td>
<td>$c_1$</td>
<td>$c_3$</td>
<td>$c_2$</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Class</th>
<th>Codeword</th>
</tr>
</thead>
<tbody>
<tr>
<td>$c_1$</td>
<td>1 1 1 1 1 1 1</td>
</tr>
<tr>
<td>$c_2$</td>
<td>0 0 0 0 1 1 1</td>
</tr>
<tr>
<td>$c_3$</td>
<td>0 0 1 1 0 0 1</td>
</tr>
<tr>
<td>$c_4$</td>
<td>0 1 0 1 0 1 0</td>
</tr>
</tbody>
</table>

- 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 9.3