Training algorithms for fuzzy support vector machines with noisy data

Presented by Josh Hoak

Chun-fu Lin$^1$    Sheng-de Wang$^1$

$^1$National Taiwan University

13 April 2010
Problem: SVMs are particularly susceptible to outliers. Fuzzy SVMs are a method to cope with outliers in SVMs.
A Motivating Example
Recap: Linear Support Vector Machine

- **Training Data:** \((y_1, x_1) \ldots (y_m, x_m) : x_i \in \mathbb{R}^n, y_i \in \{1, -1\}\)
- **Goal:** Draw a hyperplane separating data with two classes; that is, find a vector \(w\) and a translation \(b\) such that

\[
  w \cdot x + b = 0,
\]

where \(x\) is an example of from the training data, subject to the **constraint**:

\[
  y_i(w \cdot x_i + b) \geq 1, \quad \text{for } i = 1, \ldots, l
\]
Non-linear SVM

- **Non-linear Mapping:** Transform the data with some function and then try to separate. Let $\phi : \mathbb{R}^n \rightarrow \mathcal{F}$, where $\mathcal{F}$ indicates the feature space. Then,

$$z = \phi(x)$$

- **Error term:** Add an error term $\xi$ to the constraint:

$$y_i(w \cdot z_i + b) \geq 1 - \xi_i, \quad i = 1, \ldots, l$$
Non-linear SVMs continued

- Equivalently, we can minimize:

\[ \frac{1}{2} \mathbf{w} \cdot \mathbf{w} - C \sum_{i=1}^{l} \xi_i, \]

subject to

\[ y_i (\mathbf{w} \cdot \mathbf{z}_i - b) \geq 1 - \xi_i \]
Non-linear SVMs continued

**Kernel Trick**: Find a *kernel* function $K(\cdot, \cdot) : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathcal{F}$ such that:

$$K(x_i, x_j) = \phi(x_i) \cdot \phi(x_j) = z_i \cdot z_j$$

**Reformulation**: Optimal hyperplane is then reformulated as finding:

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

Decision function: $f_D(x) = \text{sign}(f_H(X))$
Fuzzy SVMs

- **Training Data**

\[(y_1, x_1, s_1), \ldots, (y_l, x_l, s_l), \quad \sigma \leq s_i \leq 1, \quad (\sigma > 0)\]

- **Fuzzy membership**: $s_i$ is viewed as our *confidence* that the corresponding point $x_i$ has class $y_i$
Fuzzy SVMs

▶ **Optimal Hyperplane:** The solution to minimizing:

\[
\frac{1}{2} \mathbf{w} \cdot \mathbf{w} + C \sum_{i=1}^{l} s_i \xi_i,
\]

constrained by

\[
y_i(\mathbf{w} \cdot \mathbf{z}_i + b) \geq 1 - \xi
\]

▶ Equivalently we can write:

\[
\frac{1}{2} \mathbf{w} \cdot \mathbf{w} + C \sum_{i=1}^{l} \psi(\xi_i)
\]
A more familiar formulation

Maximize:

\[
W(\alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \alpha_i \alpha_j y_i y_j K(x_i, x_j)
\]

Subject to:

\[
\sum_{i=1}^{l} y_i \alpha_i = 0,
\]

\[
0 \leq \alpha_i \leq s_i C, \quad i = 1, \ldots, l
\]
How do we find the confidence values?
Fuzzy SVMs: The error function

Note: We model the theoretical error $\psi(\xi_i)$ by the probability that a point is noise $- p_x(x_i)$. The error function becomes $\sum_{i=1}^{l} p_x(x_i)\xi_i$

One model:

$$p_x(x_i) = \begin{cases} 1, & \text{if } h(x_i) > h_c, \\ \sigma, & \text{if } h(x_i) < h_T, \\ \sigma + (1 - \sigma) \left( \frac{h(x_i) - h_T}{h_c - h_T} \right)^d, & \text{otherwise}. \end{cases}$$

New definitions: $h_c$ is the confidence factor, $h_T$ is the ‘trashy’ factor, $h(x)$ is a heuristic function.
Fuzzy SVMs: The error function

![Diagram showing the mapping between probability density function $p_x(x)$ and heuristic function $h(x)$.

Fig. 1. The mapping between the probability density function $p_x(x)$ and the heuristic function $h(x)$.}
Generating fuzzy memberships – $p_x(x)$

- We choose $\sigma > 0$ as a lower bound.
- Let’s suppose that our fuzzy membership (or outlier status) is based on only one feature $t$. Then, we have:

$$s_i = h(x_i) = f(t_i).$$
Let the maximum of these be $t_{\text{max}}$ and let $t_{\text{min}}$ be the minimum; when $t_i = t_{\text{min}}$, we want the output to be $\sigma$.

If we make $s_i$ be a linear function of $t$ then we get

$$s_i = f(t_i) = at_i + b$$

Solving the system of equations:

$$\sigma = a(t_{\text{min}}) + b$$
$$1 = a(t_{\text{max}}) + b$$
Generating fuzzy memberships (continued)

- Solving, we get:

\[ s_i = f(t_i) = \frac{1 - \sigma}{t_{\text{max}} - t_{\text{min}}} t_i + \frac{t_{\text{max}} \sigma - t_{\text{min}}}{t_{\text{max}} - t_{\text{min}}} \]

- If we wish to make the function polynomial, we get:

\[ s_i = f(t_i) = (1 - \sigma) \left( \frac{t_i - t_{\text{min}}}{t_{\text{max}} - t_{\text{min}}} \right)^2 + \sigma \]
Fuzzy SVM: The heuristic function

**Strategy 1:** Kernel-target alignment. Defined as

\[
A_{KT} = \frac{\sum_{i=1}^{l} f_{K}(x_i, y_i)}{l \sqrt{\sum_{i,j=1}^{l} K^2(x_i, x_j)}}
\]

where \( f_{K}(x_i, y_i) = \sum_{j=1}^{l} y_i y_j K(x_i, x_j) \)

Idea: Use \( f_{K}(x_i, y_i) \) as the heuristic function.
Fuzzy SVMs: The heuristic function—

**Strategy 2:** **k-NN.** Find the nearest neighbors of a data point \( x_i \) (of the same class). Assume that the data point with fewer nearest neighbors (of the same class) has higher probability of being noisy data. Let the heuristic function be:

\[
h(x_i) = n_i,
\]

where \( n_i \) is the number of nearest neighbors.
Overall Procedure

1. Use the original algorithm of SVMs to get the optimal kernel parameters and the regularization parameter $C$.

2. Fix the kernel parameters and the regularization parameter $C$ from (1), and then find the other parameters in FSVMs.
   2.1 Define the heuristic function.
   2.2 Use exhaustive search to find the $h_T$, $h_C$, $d$, and $\sigma$. 

Chun-fu Lin, Sheng-de Wang
Training algorithms for fuzzy support vector machines with noisy data
## Results

Table 1: Error rates for SVMs and FSVMs using KT and k-NN

<table>
<thead>
<tr>
<th></th>
<th>TR</th>
<th>SVMs</th>
<th>KT</th>
<th>k-NN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Banana</td>
<td>6.7</td>
<td>11.5 ± 0.7</td>
<td>*10.4 ± 0.5</td>
<td>11.4 ± 0.6</td>
</tr>
<tr>
<td>B. Cancer</td>
<td>18.3</td>
<td>26.0 ± 4.7</td>
<td>25.3 ± 4.4</td>
<td>*25.2 ± 4.1</td>
</tr>
<tr>
<td>Diabetes</td>
<td>19.4</td>
<td>23.5 ± 1.7</td>
<td>*23.3 ± 1.7</td>
<td>23.5 ± 1.7</td>
</tr>
<tr>
<td>German</td>
<td>16.2</td>
<td>23.6 ± 2.1</td>
<td>*23.3 ± 2.3</td>
<td>23.6 ± 2.1</td>
</tr>
<tr>
<td>Heart</td>
<td>12.8</td>
<td>16.0 ± 3.3</td>
<td>*14.2 ± 3.1</td>
<td>23.6 ± 2.1</td>
</tr>
<tr>
<td>Image</td>
<td>3.0</td>
<td>3.0 ± 0.6</td>
<td>*2.9 ± 0.7</td>
<td>-</td>
</tr>
<tr>
<td>Ringnorm</td>
<td>0.0</td>
<td>*1.7 ± 0.1</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>F. Solar</td>
<td>32.6</td>
<td>*32.4 ± 1.8</td>
<td>32.4 ± 1.8</td>
<td>32.4 ± 1.8</td>
</tr>
<tr>
<td>Splice</td>
<td>0.0</td>
<td>*10.9 ± 0.7</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Thyroid</td>
<td>0.4</td>
<td>4.8 ± 2.2</td>
<td>*4.9 ± 2.3</td>
<td>-</td>
</tr>
<tr>
<td>Titanic</td>
<td>19.6</td>
<td>22.4 ± 1.0</td>
<td>*22.3 ± 0.9</td>
<td>*22.3 ± 1.1</td>
</tr>
<tr>
<td>Twonorm</td>
<td>0.4</td>
<td>3.0 ± 0.2</td>
<td>*2.4 ± 0.1</td>
<td>2.9 ± 0.2</td>
</tr>
<tr>
<td>Waveform</td>
<td>3.5</td>
<td>*9.9 ± 0.4</td>
<td>9.9 ± 0.4</td>
<td>-</td>
</tr>
</tbody>
</table>

13 data sets from the UCI, DELVE and STAT-LOG
Multi-class SVM

**Ensemble Method**: Use the OAA (One Against All) method of multiple-classification. Given \( M \) classes, we construct \( M \) binary SVM classifiers that separate one class from the rest. The class associated with the classifier that outputs the highest value given an example is then chosen.

**Fuzzy Membership Function**: When training,

\[
Fuzz(x) = \begin{cases} 
1 & \text{if the output of the ensemble on } x \text{ is } 1, \\
 h & \text{if the output of ensemble on } x \text{ is } -1.
\end{cases}
\]
### Confusion Matrix

<table>
<thead>
<tr>
<th>Actual Class</th>
<th>Predicted Class</th>
<th>Earn</th>
<th>Acq</th>
<th>Money-fx</th>
<th>Grain</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Earn</td>
<td>Earn</td>
<td>679</td>
<td>3</td>
<td>3</td>
<td>7</td>
<td>692</td>
</tr>
<tr>
<td>Acq</td>
<td>24</td>
<td>425</td>
<td>2</td>
<td>2</td>
<td>453</td>
<td></td>
</tr>
<tr>
<td>Money-Fx</td>
<td>6</td>
<td>5</td>
<td>86</td>
<td>4</td>
<td>101</td>
<td></td>
</tr>
<tr>
<td>Grain</td>
<td>12</td>
<td>1</td>
<td>2</td>
<td>87</td>
<td>102</td>
<td></td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>721</strong></td>
<td><strong>434</strong></td>
<td><strong>93</strong></td>
<td><strong>100</strong></td>
<td><strong>1348</strong></td>
<td></td>
</tr>
</tbody>
</table>

Note: Overall Accuracy = 0.945

Data: Reuters Documents
## Results

<table>
<thead>
<tr>
<th>Classifier</th>
<th>4-fold Precision</th>
<th>Recall</th>
<th>F Measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>OAA-FSVM(1, 0.5)</td>
<td><strong>0.775</strong></td>
<td>0.624</td>
<td>0.672</td>
</tr>
<tr>
<td>OAA-FSVM(1, 0.6)</td>
<td>0.772</td>
<td><strong>0.630</strong></td>
<td><strong>0.675</strong></td>
</tr>
<tr>
<td>OAA-SVM</td>
<td>0.764</td>
<td>0.616</td>
<td>0.659</td>
</tr>
</tbody>
</table>

Chun-fu Lin, Sheng-de Wang

Training algorithms for fuzzy support vector machines with noisy data.
Thoughts

- Statistically significant results?
- Diminishing returns?
- Cost?
- How much do outliers affect the model?