Machine Learning
Lecture 5: Implementing SVM

Konstantin Tretyakov, Phaedra Agius

March 05, 2008
Last Time: Linear Classifier

\[ f_{w, b}(x) = \text{sign}(w^T x + b) \]
Last Time: Maximal Margin Linear Classifier

- Distance of point $\mathbf{x}$ to hyperplane: $\frac{\mathbf{w}^T \mathbf{x} + b}{\|\mathbf{w}\|}$
- Margin of instance $(\mathbf{x}_i, y_i)$: $y_i \frac{\mathbf{w}^T \mathbf{x}_i + b}{\|\mathbf{w}\|}$
Last Time: Maximal Margin Linear Classifier

- Classifier margin: \( \min_i \left( y_i \frac{w^T x_i + b}{\|w\|} \right) \)
Last Time: Maximal Margin Linear Classifier

- Classifier margin: \( \min_i \left( y_i \frac{w^T x_i + b}{\|w\|} \right) \)
- We can safely rescale \( w \) and \( b \) without changing the solution.
Last Time: Maximal Margin Linear Classifier

- Classifier margin: \( \min_i \left( y_i \frac{w^T x_i + b}{\|w\|} \right) \)
- We can safely rescale \( w \) and \( b \) without changing the solution.
- Let’s fix the scale of \( w \) so that \( \min_i \left( y_i \left( w^T x_i + b \right) \right) = 1 \)
Last Time: Maximal Margin Linear Classifier

- Classifier margin: \( \min_i \left( y_i \frac{w^T x_i + b}{\|w\|} \right) \)
- We can safely rescale \( w \) and \( b \) without changing the solution.
- Let's fix the scale of \( w \) so that \( \min_i \left( y_i (w^T x_i + b) \right) = 1 \)
- Then maximizing classifier margin is:

\[
\max_{w,b} \left( \min_i \left( y_i \frac{w^T x_i + b}{\|w\|} \right) \right) = \max_{w,b} \left( \frac{\min_i(y_i(w^T x_i + b))}{\|w\|} \right) \\
= \max_{w,b} \left( \frac{1}{\|w\|} \right) \rightarrow \text{need to minimize } \|w\|
Last Time: Maximal Margin Linear Classifier

$$\min_w \frac{1}{2} \|w\|^2$$

s.t. \( \forall i \quad y_i(w^T x_i + b) \geq 1. \)
Last Time: Maximal Margin Linear Classifier

$$\min_w \frac{1}{2} \|w\|^2$$

s.t. $$\forall i \quad y_i(w^T x_i + b) \geq 1.$$ 

... or with slack variables:

$$\min_w \frac{1}{2} \|w\|^2 + C \sum_i \xi_i$$

s.t. $$\forall i \quad y_i(w^T x_i + b) \geq 1 - \xi_i,$$

$$\xi_i \geq 0.$$
Today: How to Minimize That?

• Primal form
  • The primal form is a quadratic programme with linear constraints with $m$ variables.
  • $\Rightarrow$ it is “easy”.
  • As an example, we’ll consider here something “simple”: constrained gradient descent.
Today: How to Minimize That?

- **Primal form**
  - The primal form is a *quadratic programme* with *linear constraints* with \( m \) variables.
  - \( \Rightarrow \) it is “easy”.
  - As an example, we’ll consider here something “simple”: constrained gradient descent.

- **Dual form (the SVM)**
  - Also a *quadratic programme* with *linear constraints* with \( n \) variables.
  - Has advantages: sparse solution, allows to use kernels.
  - Lots of specific optimizations.
Constrained Gradient Descent

Let's consider the separable case:

\[
\min_w \frac{1}{2} \|w\|^2
\]

s.t. \( \forall i \ y_i(w^T x_i + b) \geq 1. \)
Let’s consider the separable case:

\[
\min_w \frac{1}{2} \|w\|^2 \\
\text{s.t. } \forall i \quad y_i x_i^T w + y_i b \geq 1.
\]
Constrained Gradient Descent

Step of the descent:

\[- \frac{\partial f}{\partial w} = -w \quad - \frac{\partial f}{\partial b} = 0\]

Projection to constrained region: ?
There are Better Methods

Most mathematical packages have built-in quadratic optimization routines you can easily use. E.g. Scilab:

```matlab
X = [0 0; 0 1; 1 0; 1 1];
y = [1; 1; 1; -1];

dX = diag(y)*X;
fX = [dX y];

[w,lagr,f] = quapro(eye(3,3),zeros(3,1),-fX,-ones(y));
```
The original problem
\[
\min_w \frac{1}{2} \|w\|^2
\]
\[
\text{s.t. } \forall i \quad y_i(w^T x_i + b) \geq 1.
\]
is equivalent to
\[
\min_w \max_{\alpha} \left( \frac{1}{2} \|w\|^2 + \sum_i \alpha_i (1 - y_i(w^T x_i + b)) \right)
\]
\[
\alpha \geq 0
\]
Dual: Reminder

Thanks to convexity we can switch min and max to get the equivalent *dual* problem:

$$\max_{\alpha} \min_w \left( \frac{1}{2} \|w\|^2 + \sum_i \alpha_i (1 - y_i (w^T x_i + b)) \right)$$

$$\alpha \geq 0$$

The inner min is easy to solve, and we're only left with one max:

$$\max_{\alpha} \left( \sum_i \alpha_i - \frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y_i y_j x_i^T x_j \right)$$

$$\alpha^T y = 0 \quad \alpha \geq 0$$
Or, in other terms:

\[
\min_\alpha \left( \frac{1}{2} \alpha^T Q \alpha - \alpha^T 1 \right)
\]

\[\alpha^T y = 0 \quad \alpha \geq 0\]

For the version with slack variables there will also be the constraint

\[\alpha \leq C\]
Dual: Reminder

Or, in other terms:

$$\min_\alpha \left( \frac{1}{2} \alpha^T Q \alpha - \alpha^T 1 \right)$$

$$\alpha^T y = 0 \quad \alpha \geq 0$$

For the version with slack variables there will also be the constraint

$$\alpha \leq C$$

- This, again, is a quadratic programme, now with $n$ variables, but much simpler constraints.
- This problem is *sparse*. 
Dual: Problems

- The matrix $Q$ is $n \times n$ which can be difficult to keep in memory for large $n$.
- The algorithm itself can be slow for large $n$.
- We can exploit sparseness to optimize the optimization:
  - Chunking
  - Decomposition
Chunking

- The solution actually depends only on the support vectors.
The solution actually depends only on the support vectors.

Therefore:

- Start with a small working set of points (hence small $Q$).
  Find corresponding $\alpha$.
- Examine the margin for all other points. If it’s $\geq 1$ we’re done.
- Else, increase working set and retrain.
• The solution actually depends only on the support vectors.
• Therefore:
  • Start with a small working set of points (hence small $Q$). Find corresponding $\alpha$.
  • Examine the margin for all other points. If it’s $\geq 1$ we’re done.
  • Else, increase working set and retrain.
• The algorithm will definitely converge to the correct solution.
• However, if we’re unlucky, the working set may still grow too large to handle.
Decomposition

- Update only a subset of the variables at each step.

\[\begin{align*}
\alpha & = \begin{pmatrix} \alpha_B \\ \alpha_N \end{pmatrix} \\
y & = \begin{pmatrix} y_B \\ y_N \end{pmatrix} \\
Q & = \begin{pmatrix} Q_{BB} & Q_{BN} \\ Q_{NB} & Q_{NN} \end{pmatrix}
\end{align*}\]

The new problem:

\[
\begin{align*}
\min_{\alpha_B} & \quad \frac{1}{2} \alpha_B^T Q_{BB} \alpha_B - \alpha_B^T Q_{BN} \alpha_N \\
\text{s.t.} & \quad \alpha_B^T y_B + \alpha_N^T y_N = 0 \\
& \quad 0 \leq \alpha \leq C
\end{align*}
\]
Decomposition

• Update only a subset of the variables at each step.
• Let $N$ be the set of variables, the values of which we keep fixed, and $B$ be the set of variables to be updated.
• Arrange $\alpha$, $y$ and $Q$ properly:

\[
\alpha = \begin{bmatrix} \alpha_B \\ \alpha_N \end{bmatrix} \quad y = \begin{bmatrix} y_B \\ y_N \end{bmatrix} \quad Q = \begin{bmatrix} Q_{BB} & Q_{BN} \\ Q_{NB} & Q_{NN} \end{bmatrix}
\]

• The new problem:

\[
\min_{\alpha_B} \left( \frac{1}{2} \alpha_B^T Q_{BB} \alpha_B - \alpha_B^T (1 - Q_{BN} \alpha_N) \right)
\]

s.t.

\[
\alpha_B^T y_B + \alpha_N^T y_N = 0 \quad 0 \leq \alpha \leq C
\]
Decomposition: Selecting the Working Set

- Consider the derivative of the objective function:

\[
\frac{\partial}{\partial \alpha} \left( \frac{1}{2} \alpha^T Q \alpha - \alpha^T 1 \right) = Q \alpha - 1
\]

- Pick the components of the derivative with largest absolute values*.
Decomposition: Shrinking

- It becomes clear fairly early in the iterations, which instances turn out *not* to be support vectors, these can be thrown away.
- Similarly, it becomes fairly early clear, for which instances will the $\alpha$ end up at bound (i.e. $\alpha = C$). These $\alpha$ values can be fixed to $C$ and forgotten about too.
• Decomposition
• Shrinking
• Termination when all constraints satisfied to given precision
• LRU cache for kernel evaluations
• Rather reliable software. Current version: 6.01.
- Consider again the decomposition idea:

\[
\min_{\alpha_B} \left( \frac{1}{2} \alpha_B^T Q_{BB} \alpha_B - \alpha_B^T (1 - Q_{BN} \alpha_N) \right)
\]

s.t.

\[
\alpha_B^T y_B + \alpha_N^T y_N = 0 \quad 0 \leq \alpha \leq C
\]

- What if $|B| = 2$?
Sequential Minimal Optimization (SMO)

\[ \alpha_2 = C \]
\[ \alpha_1 = 0 \]
\[ \alpha_2 = 0 \]
\[ y_1 \neq y_2 \Rightarrow \alpha_1 - \alpha_2 = \gamma \]

\[ \alpha_2 = C \]
\[ \alpha_1 = C \]
\[ \alpha_1 = 0 \]
\[ \alpha_2 = 0 \]
\[ y_1 = y_2 \Rightarrow \alpha_1 + \alpha_2 = \gamma \]
### SVM\textsuperscript{light} vs SMO (1999)

<table>
<thead>
<tr>
<th>Examples</th>
<th>SVM\textsuperscript{light}</th>
<th>SMO</th>
<th>Chunking</th>
<th>Minimum</th>
<th>total SV</th>
<th>BSV</th>
</tr>
</thead>
<tbody>
<tr>
<td>2477</td>
<td>18.0</td>
<td>26.3</td>
<td>64.9</td>
<td>3.6</td>
<td>431</td>
<td>47</td>
</tr>
<tr>
<td>3470</td>
<td>28.2</td>
<td>44.1</td>
<td>110.4</td>
<td>7.8</td>
<td>571</td>
<td>69</td>
</tr>
<tr>
<td>4912</td>
<td>46.2</td>
<td>83.6</td>
<td>372.5</td>
<td>13.2</td>
<td>671</td>
<td>96</td>
</tr>
<tr>
<td>7366</td>
<td>102.0</td>
<td>156.7</td>
<td>545.4</td>
<td>27.0</td>
<td>878</td>
<td>138</td>
</tr>
<tr>
<td>9888</td>
<td>174.6</td>
<td>248.1</td>
<td>907.6</td>
<td>46.8</td>
<td>1075</td>
<td>187</td>
</tr>
<tr>
<td>17188</td>
<td>450.0</td>
<td>581.0</td>
<td>3317.9</td>
<td>123.6</td>
<td>1611</td>
<td>363</td>
</tr>
<tr>
<td>24692</td>
<td>843.0</td>
<td>1214.0</td>
<td>6659.7</td>
<td>222.6</td>
<td>1994</td>
<td>506</td>
</tr>
<tr>
<td>49749</td>
<td>2834.4</td>
<td>3863.5</td>
<td>23877.6</td>
<td>706.2</td>
<td>3069</td>
<td>948</td>
</tr>
<tr>
<td>Scaling</td>
<td>1.7</td>
<td>1.7</td>
<td>2.0</td>
<td>1.7</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### SMO vs $SVM^{light}$ (1999)

<table>
<thead>
<tr>
<th>Experiment</th>
<th>SMO Time (sec)</th>
<th>SVM$^{light}$ Time (sec)</th>
<th>Chunking Time (sec)</th>
<th>SMO Scaling Exponent</th>
<th>SVM$^{light}$ Scaling Exponent</th>
<th>Chunking Scaling Exponent</th>
</tr>
</thead>
<tbody>
<tr>
<td>AdultLin</td>
<td>13.7</td>
<td>217.9</td>
<td>20711.3</td>
<td>1.8</td>
<td>2.1</td>
<td>3.1</td>
</tr>
<tr>
<td>AdultLinD</td>
<td>21.9</td>
<td>n/a</td>
<td>21141.1</td>
<td>1.0</td>
<td>n/a</td>
<td>3.0</td>
</tr>
<tr>
<td>WebLin</td>
<td>339.9</td>
<td>3980.8</td>
<td>17164.7</td>
<td>1.6</td>
<td>2.2</td>
<td>2.5</td>
</tr>
<tr>
<td>WebLinD</td>
<td>4589.1</td>
<td>n/a</td>
<td>17332.8</td>
<td>1.5</td>
<td>n/a</td>
<td>2.5</td>
</tr>
<tr>
<td>AdultGaussK</td>
<td>442.4</td>
<td>284.7</td>
<td>11910.6</td>
<td>2.0</td>
<td>2.0</td>
<td>2.9</td>
</tr>
<tr>
<td>AdultGauss</td>
<td>523.3</td>
<td>737.5</td>
<td>n/a</td>
<td>2.0</td>
<td>n/a</td>
<td>2.0</td>
</tr>
<tr>
<td>AdultGaussKD</td>
<td>1433.0</td>
<td>n/a</td>
<td>14740.4</td>
<td>2.5</td>
<td>n/a</td>
<td>2.8</td>
</tr>
<tr>
<td>AdultGaussD</td>
<td>1810.2</td>
<td>n/a</td>
<td>n/a</td>
<td>2.0</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>WebGaussK</td>
<td>2477.9</td>
<td>2949.5</td>
<td>23877.6</td>
<td>1.6</td>
<td>2.0</td>
<td>2.0</td>
</tr>
<tr>
<td>WebGauss</td>
<td>2538.0</td>
<td>6923.5</td>
<td>n/a</td>
<td>1.6</td>
<td>1.8</td>
<td>n/a</td>
</tr>
<tr>
<td>WebGaussKD</td>
<td>23365.3</td>
<td>n/a</td>
<td>50371.9</td>
<td>2.6</td>
<td>n/a</td>
<td>2.0</td>
</tr>
<tr>
<td>WebGaussD</td>
<td>24758.0</td>
<td>n/a</td>
<td>n/a</td>
<td>1.6</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>MNIST</td>
<td>19387.9</td>
<td>38452.3</td>
<td>33109.0</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
</tr>
</tbody>
</table>
Questions?