Solving SVMs (SMO Algorithms)

Jordan Boyd-Graber
University of Colorado Boulder
LECTURE 9A
Plan

Dual Objective

Algorithm Big Picture

The Algorithm

Recap
Lagrange Multipliers

Introduce Lagrange variables $\alpha_i \geq 0$, $i \in [1, m]$ for each of the $m$ constraints (one for each data point).

$$\mathcal{L}(w, b, \alpha) = \frac{1}{2}||w||^2 - \sum_{i=1}^{m} \alpha_i [y_i(w \cdot x_i + b) - 1] \quad (1)$$
Solving Lagrangian

Weights

$$\vec{w} = \sum_{i=1}^{m} \alpha_i y_i \vec{x}_i$$ (2)
## Solving Lagrangian

**Weights**

\[ \tilde{\mathbf{w}} = \sum_{i=1}^{m} \alpha_i y_i \tilde{x}_i \]  

(2)

**Bias**

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

(3)
Dual Objective

Solving Lagrangian

Weights

\[ \vec{w} = \sum_{i=1}^{m} \alpha_i y_i \vec{x}_i \]  

(2)

Bias

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

(3)

Support Vector-ness

\[ \alpha_i = 0 \vee y_i (w \cdot x_i + b) \leq 1 \]  

(4)
Reparameterize in terms of $\alpha$

$$\max_{\vec{\alpha}} \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{i=1}^{m} \alpha_i \alpha_j y_i y_j (\vec{x}_i \cdot \vec{x}_j)$$

(5)
Strawman: Coordinate Descent

- Why not optimize one coordinate $\alpha_i$ at a time?
Strawman: Coordinate Descent

• Why not optimize one coordinate $\alpha_i$ at a time?
• Constraints!
• So we’ll just minimize pairs $(\alpha_i, \alpha_j)$ at a time
Outline for SVM Optimization (SMO)

1. Select two examples $i, j$
2. Get a learning rate $\eta$
3. Update $\alpha_j$
4. Update $\alpha_i$
Plan

Dual Objective

Algorithm Big Picture

The Algorithm

Recap
Contrast with SG

- There’s a learning rate $\eta$ that depends on the data
- Use the error of an example to derive update
- You update multiple $\alpha$ at once
Contrast with SG

- There’s a learning rate $\eta$ that depends on the data
- Use the error of an example to derive update
- You update multiple $\alpha$ at once: if one goes up, the other should go down because $\sum y_i \alpha_i = 0$
More details

- We enforce every $\alpha_i < C$ (slackness)
- How do we know we’ve converged?
More details

- We enforce every $\alpha_i < C$ (slackness)
- How do we know we’ve converged?

$$\alpha_i = 0 \Rightarrow y_i(w \cdot x_i + b) \geq 1$$  \hspace{1cm} (6)

$$\alpha_i = C \Rightarrow y_i(w \cdot x_i + b) \leq 1$$  \hspace{1cm} (7)

$$0 < \alpha_i < C \Rightarrow y_i(w \cdot x_i + b) = 1$$  \hspace{1cm} (8)

(Karush-Kuhn-Tucker Conditions)
More details

- We enforce every $\alpha_i < C$ (slackness)
- How do we know we've converged?

\[
\alpha_i = 0 \implies y_i(w \cdot x_i + b) \geq 1 \tag{6}
\]
\[
\alpha_i = C \implies y_i(w \cdot x_i + b) \leq 1 \tag{7}
\]
\[
0 < \alpha_i < C \implies y_i(w \cdot x_i + b) = 1 \tag{8}
\]

(Karush-Kuhn-Tucker Conditions)

- Keep checking (to some tolerance)
Plan

Dual Objective

Algorithm Big Picture

The Algorithm

Recap
Step 1: Select $i$ and $j$

- Find some $i \in \{1, \ldots, m\}$ that violates KKT
- Choose $j$ randomly from $m - 1$ other options
- You can do better (particularly for large datasets)
- Repeat until KKT conditions are met
Step 2: Optimize $\alpha_j$

1. Compute upper ($H$) and lower ($L$) bounds that ensure $0 < \alpha_j \leq C$.

<table>
<thead>
<tr>
<th>$y_i \neq y_j$</th>
<th>$y_i = y_j$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$L = \max(0, \alpha_j - \alpha_i)$ (9)</td>
<td>$L = \max(0, \alpha_i + \alpha_j - C)$ (11)</td>
</tr>
<tr>
<td>$H = \min(C, C + \alpha_j - \alpha_i)$ (10)</td>
<td>$H = \min(C, \alpha_j + \alpha_i)$ (12)</td>
</tr>
</tbody>
</table>
1. Compute upper ($H$) and lower ($L$) bounds that ensure $0 < \alpha_j \leq C$.

$$y_i \neq y_j$$

- $$L = \max(0, \alpha_j - \alpha_i)$$ \hspace{1cm} (9)
- $$H = \min(C, C + \alpha_j - \alpha_i)$$ \hspace{1cm} (10)

This is because the update for $\alpha_i$ is based on $y_i y_j$ (sign matters)

$$y_i = y_j$$

- $$L = \max(0, \alpha_i + \alpha_j - C)$$ \hspace{1cm} (11)
- $$H = \min(C, \alpha_j + \alpha_i)$$ \hspace{1cm} (12)
Step 2: Optimize $\alpha_j$

Compute errors for $i$ and $j$

$$E_k \equiv f(x_k) - y_k$$  (13)
The Algorithm

Step 2: Optimize $\alpha_j$

Compute errors for $i$ and $j$

$$E_k \equiv f(x_k) - y_k$$  \hspace{1cm} (13)

and the learning rate (more similar, higher step size)

$$\eta = 2x_i \cdot x_j - x_i \cdot x_i - x_j \cdot x_j$$  \hspace{1cm} (14)
Step 2: Optimize $\alpha_j$

Compute errors for $i$ and $j$

$$E_k \equiv f(x_k) - y_k$$  \hspace{1cm} (13)

and the learning rate (more similar, higher step size)

$$\eta = 2x_i \cdot x_j - x_i \cdot x_i - x_j \cdot x_j$$  \hspace{1cm} (14)

for new value for $\alpha_j$

$$\alpha_j^* = \alpha_j^{(old)} - \frac{y_j(E_i - E_j)}{\eta}$$  \hspace{1cm} (15)

Similar to stochastic gradient, but with additional error term.
Step 2: Optimize $\alpha_j$

Compute errors for $i$ and $j$

$$E_k \equiv f(x_k) - y_k$$  \hspace{1cm} (13)

and the learning rate (more similar, higher step size)

$$\eta = 2x_i \cdot x_j - x_i \cdot x_i - x_j \cdot x_j$$  \hspace{1cm} (14)

for new value for $\alpha_j$

$$\alpha_j^* = \alpha_j^{(old)} - \frac{y_j (E_i - E_j)}{\eta}$$  \hspace{1cm} (15)

What if $x_i = x_j$?
Step 3: Optimize $\alpha_i$

Set $\alpha_i$:

$$\alpha_i^* = \alpha_i^{(old)} + y_i y_j \left( \alpha_j^{(old)} - \alpha_j \right)$$

(16)
Step 3: Optimize $\alpha_i$

Set $\alpha_i$:

$$\alpha_i^{*} = \alpha_i^{(old)} + y_i y_j \left( \alpha_j^{(old)} - \alpha_j \right)$$

(16)

This balances out the move that we made for $\alpha_j$. 
Step 4: Optimize the threshold $b$

We need the KKT conditions to be satisfied for these two examples.

- If $0 < \alpha_i < C$ (support vector)

$$b = b_1 = b - E_i - y_i(\alpha_i^* - \alpha_i^{(\text{old})})x_i \cdot x_i - y_j(\alpha_j^* - \alpha_j^{(\text{old})})x_j \cdot x_j$$ (17)
Step 4: Optimize the threshold $b$

We need the KKT conditions to be satisfied for these two examples.

- If $0 < \alpha_i < C$ (support vector)

$$
b = b_1 = b - E_i - y_i(\alpha_i^* - \alpha_i^{old})x_i \cdot x_i - y_j(\alpha_j^* - \alpha_j^{old})x_j \cdot x_j (17)$$

- If $0 < \alpha_j < C$ (support vector)

$$
b = b_2 = b - E_j - y_i(\alpha_i^* - \alpha_i^{old})x_i \cdot x_j - y_j(\alpha_j^* - \alpha_j^{old})x_j \cdot x_j (18)$$
Step 4: Optimize the threshold $b$

We need the KKT conditions to be satisfied for these two examples.

- If $0 < \alpha_i < C$ (support vector)
  \[
  b = b_1 = b - E_i - y_i (\alpha_i^* - \alpha_i^{(\text{old})}) x_i \cdot x_i - y_j (\alpha_j^* - \alpha_j^{(\text{old})}) x_i \cdot x_j \quad (17)
  \]

- If $0 < \alpha_j < C$ (support vector)
  \[
  b = b_2 = b - E_j - y_i (\alpha_i^* - \alpha_i^{(\text{old})}) x_i \cdot x_j - y_j (\alpha_j^* - \alpha_j^{(\text{old})}) x_j \cdot x_j \quad (18)
  \]

- If both $\alpha_i$ and $\alpha_j$ are at the bounds (well away from margin), then anything between $b_1$ and $b_2$ works, so we set
  \[
  b = \frac{b_1 + b_2}{2} \quad (19)
  \]
Iterations / Details

- What if $i$ doesn’t violate the KKT conditions?
- What if $\eta \geq 0$?
- When do we stop?
Iterations / Details

• What if $i$ doesn’t violate the KKT conditions? **Skip it!**
• What if $\eta \geq 0$?
• When do we stop?
The Algorithm

Iterations / Details

- What if $i$ doesn’t violate the KKT conditions? **Skip it!**
- What if $\eta \geq 0$? **Skip it!**
- When do we stop?
The Algorithm

Iterations / Details

• What if \( i \) doesn’t violate the KKT conditions? **Skip it!**
• What if \( \eta \geq 0? \) **Skip it!**
• When do we stop? **Until we go through \( \alpha \)'s without changing anything**
Recap

Plan

Dual Objective

Algorithm Big Picture

The Algorithm

Recap
Recap

- SMO: Optimize objective function for two data points
- Convex problem: Will converge
- Relatively fast
- Gives good performance
- Next HW!