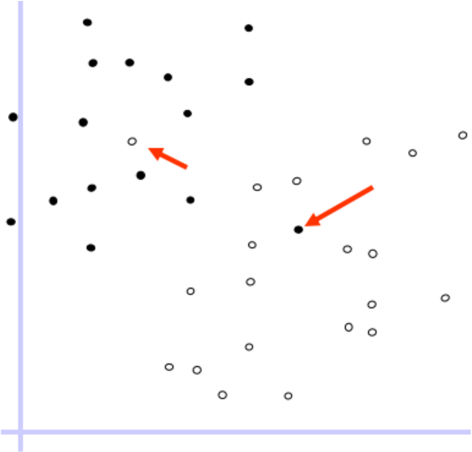
Department of Computer Science
UNIVERSITY OF COLORADO **BOULDER**

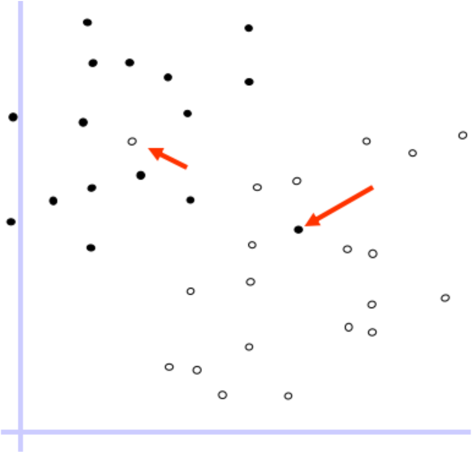
Slack SVMs

Jordan Boyd-Graber
University of Colorado Boulder
LECTURE 8A

Can SVMs Work Here?

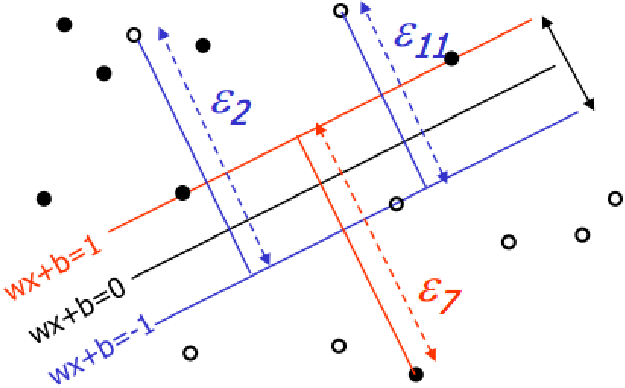


Can SVMs Work Here?



$$y_i(w \cdot x_i + b) \geq 1 \quad (1)$$

Trick: Allow for a few bad apples



New objective function

$$\min_{w,b,\xi} \frac{1}{2}||w||^2 + C \sum_{i=1} {\xi_i}^p \tag{2}$$

subject to $y_i(w \cdot x_i + b) \geq 1 - \xi_i \wedge \xi_i \geq 0, i \in [1, m]$

New objective function

$$\min_{w,b,\xi} \frac{1}{2}||w||^2 + C \sum_{i=1} {\xi_i}^p \tag{2}$$

subject to $y_i(w \cdot x_i + b) \geq 1 - \xi_i \wedge \xi_i \geq 0, i \in [1, m]$

- Standard margin

New objective function

$$\min_{w,b,\xi} \frac{1}{2}||w||^2 + C \sum_{i=1} {\xi_i}^p \tag{2}$$

subject to $y_i(w \cdot x_i + b) \geq 1 - \xi_i \wedge \xi_i \geq 0, i \in [1, m]$

- Standard margin
- How wrong a point is (slack variables)

New objective function

$$\min_{w,b,\xi} \frac{1}{2}||w||^2 + C \sum_{i=1} {\xi_i}^p \quad (2)$$

subject to $y_i(w \cdot x_i + b) \geq 1 - \xi_i \wedge \xi_i \geq 0, i \in [1, m]$

- Standard margin
- How wrong a point is (slack variables)
- Tradeoff between margin and slack variables

New objective function

$$\min_{w,b,\xi} \frac{1}{2}||w||^2 + C \sum_{i=1} {\xi_i}^p \quad (2)$$

subject to $y_i(w \cdot x_i + b) \geq 1 - \xi_i \wedge \xi_i \geq 0, i \in [1, m]$

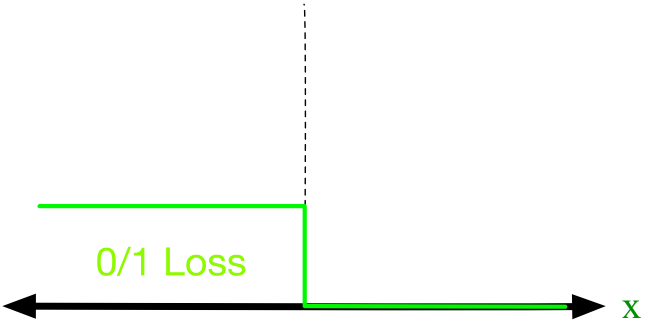
- Standard margin
- How wrong a point is (slack variables)
- Tradeoff between margin and slack variables
- How bad wrongness scales

Aside: Loss Functions

- Losses measure how bad a mistake is
- Important for slack as well

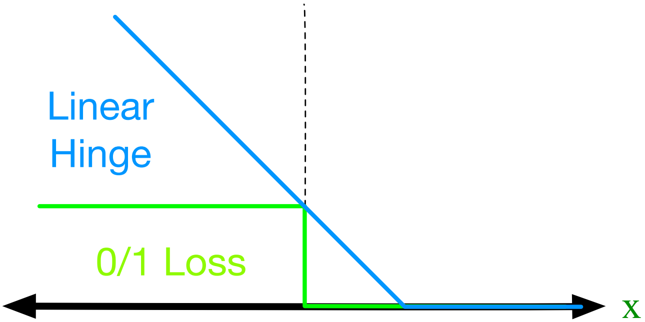
Aside: Loss Functions

- Losses measure how bad a mistake is
- Important for slack as well



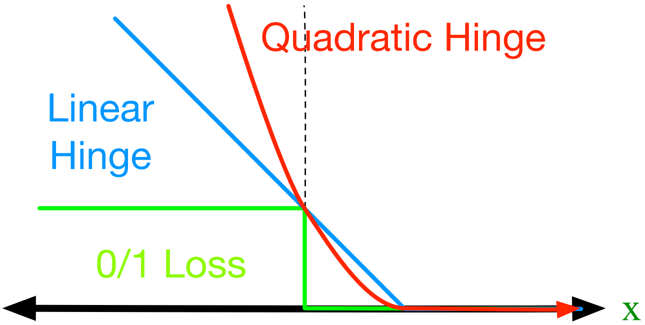
Aside: Loss Functions

- Losses measure how bad a mistake is
- Important for slack as well



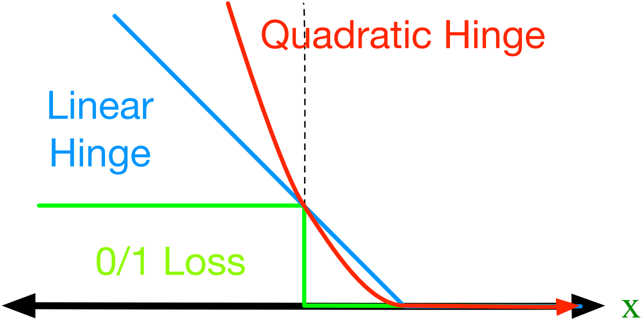
Aside: Loss Functions

- Losses measure how bad a mistake is
- Important for slack as well



Aside: Loss Functions

- Losses measure how bad a mistake is
- Important for slack as well



We'll focus on linear hinge loss

Optimizing Constrained Functions

Theorem: Lagrange Multiplier Method

Given functions $f(x_1, \ldots x_n)$ and $g(x_1, \ldots x_n)$, the critical points of f restricted to the set $g = 0$ are solutions to equations:

$$\begin{aligned} \frac{\partial f}{\partial x_i}(x_1, \ldots x_n) =& \lambda \frac{\partial g}{\partial x_i}(x_1, \ldots x_n) \quad \forall i \\ g(x_1, \ldots x_n) =& 0 \end{aligned}$$

This is $n + 1$ equations in the $n + 1$ variables $x_1, \ldots x_n, \lambda$.

Lagrange Example

Maximize $f(x, y) = \sqrt{xy}$ subject to the constraint $20x + 10y = 200$.

- Compute derivatives

Lagrange Example

Maximize $f(x, y) = \sqrt{xy}$ subject to the constraint $20x + 10y = 200$.

- Compute derivatives

$$\frac{\partial f}{\partial x} = \frac{1}{2}\sqrt{\frac{y}{x}} \quad \frac{\partial g}{\partial x} = 20$$
$$\frac{\partial f}{\partial y} = \frac{1}{2}\sqrt{\frac{x}{y}} \quad \frac{\partial g}{\partial y} = 10$$

Lagrange Example

Maximize $f(x, y) = \sqrt{xy}$ subject to the constraint $20x + 10y = 200$.

- Compute derivatives

$$\frac{\partial f}{\partial x} = \frac{1}{2}\sqrt{\frac{y}{x}} \qquad \frac{\partial g}{\partial x} = 20$$
$$\frac{\partial f}{\partial y} = \frac{1}{2}\sqrt{\frac{x}{y}} \qquad \frac{\partial g}{\partial y} = 10$$

- Create new systems of equations

Lagrange Example

Maximize $f(x, y) = \sqrt{xy}$ subject to the constraint $20x + 10y = 200$.

- Compute derivatives

$$\frac{\partial f}{\partial x} = \frac{1}{2}\sqrt{\frac{y}{x}} \quad \frac{\partial g}{\partial x} = 20$$
$$\frac{\partial f}{\partial y} = \frac{1}{2}\sqrt{\frac{x}{y}} \quad \frac{\partial g}{\partial y} = 10$$

- Create new systems of equations

$$\frac{1}{2}\sqrt{\frac{y}{x}} = 20\lambda$$
$$\frac{1}{2}\sqrt{\frac{x}{y}} = 10\lambda$$
$$20x + 10y = 200$$

Lagrange Example

- Dividing the first equation by the second gives us

$$\frac{y}{x} = 2 \tag{3}$$

- which means $y = 2x$, plugging this into the constraint equation gives:

$$\begin{aligned} 20x + 20(2x) &= 200 \\ x = 5 &\Rightarrow y = 10 \end{aligned}$$

New Lagrangian

$$\mathscr{L}(\vec{w}, b, \vec{\xi}, \vec{\alpha}, \vec{\beta}) = \frac{1}{2}||w||^2 + C\sum_{i=1}^{m}\xi_i \tag{4}$$

$$- \sum_{i=1}^{m}\alpha_i\left[y_i(w \cdot x_i + b) - 1 + \xi_i\right] \tag{5}$$

$$- \sum_{i=1}^{m}\beta_i\xi_i \tag{6}$$

New Lagrangian

$$\mathscr{L}(\vec{w}, b, \vec{\xi}, \vec{\alpha}, \vec{\beta}) = \frac{1}{2}||w||^2 + C \sum_{i=1}^{m} \xi_i \tag{4}$$

$$- \sum_{i=1}^{m} \alpha_i \left[y_i(w \cdot x_i + b) - 1 + \xi_i \right] \tag{5}$$

$$- \sum_{i=1}^{m} \beta_i \xi_i \tag{6}$$

Taking the gradients ($\nabla_w \mathscr{L}, \nabla_b \mathscr{L}, \nabla_{\xi_i}$) and solving for zero gives us

$$\sum_{i=1}^{m} \alpha_i y_i = 0 \quad (7) \qquad \vec{w} = \sum_{i=1}^{m} \alpha_i y_i x_i \quad (8) \qquad \alpha_i + \beta_i = C \quad (9)$$

New Lagrangian

$$\mathcal{L}(\vec{w}, b, \vec{\xi}, \vec{\alpha}, \vec{\beta}) = \frac{1}{2}||w||^2 + C\sum_{i=1}^{m} \xi_i \tag{4}$$

$$- \sum_{i=1}^{m} \alpha_i \left[y_i(w \cdot x_i + b) - 1 + \xi_i \right] \tag{5}$$

$$- \sum_{i=1}^{m} \beta_i \xi_i \tag{6}$$

Taking the gradients ($\nabla_w \mathcal{L}, \nabla_b \mathcal{L}, \nabla_{\xi_i}$) and solving for zero gives us

$$\sum_{i=1}^{m} \alpha_i y_i = 0 \quad (7) \qquad \vec{w} = \sum_{i=1}^{m} \alpha_i y_i x_i \quad (8) \qquad \alpha_i + \beta_i = C \quad (9)$$

New Lagrangian

$$\mathscr{L}(\vec{w}, b, \vec{\xi}, \vec{\alpha}, \vec{\beta}) = \frac{1}{2}||w||^2 + C\sum_{i=1}^{m}\xi_i \quad (4)$$

$$- \sum_{i=1}^{m}\alpha_i\left[y_i(w \cdot x_i + b) - 1 + \xi_i\right] \quad (5)$$

$$- \sum_{i=1}^{m}\beta_i\xi_i \quad (6)$$

Taking the gradients ($\nabla_w\mathscr{L}, \nabla_b\mathscr{L}, \nabla_{\xi_i}$) and solving for zero gives us

$$\sum_{i=1}^{m}\alpha_i y_i = 0 \quad (7) \qquad \vec{w} = \sum_{i=1}^{m}\alpha_i y_i x_i \quad (8) \qquad \alpha_i + \beta_i = C \quad (9)$$

New Lagrangian

$$\mathscr{L}(\vec{w}, b, \vec{\xi}, \vec{\alpha}, \vec{\beta}) = \frac{1}{2}||w||^2 + C\sum_{i=1}^{m}\xi_i \tag{4}$$

$$- \sum_{i=1}^{m}\alpha_i\left[y_i(w \cdot x_i + b) - 1 + \xi_i\right] \tag{5}$$

$$- \sum_{i=1}^{m}\beta_i\xi_i \tag{6}$$

Taking the gradients ($\nabla_w \mathscr{L}, \nabla_b \mathscr{L}, \nabla_{\xi_i}$) and solving for zero gives us

$$\sum_{i=1}^{m}\alpha_i y_i = 0 \quad (7) \qquad \vec{w} = \sum_{i=1}^{m}\alpha_i y_i x_i \quad (8) \qquad \alpha_i + \beta_i = C \quad (9)$$

Simplifying dual objective

$$\sum_{i=1}^{m} \alpha_i y_i = 0 \qquad \vec{w} = \sum_{i=1}^{m} \alpha_i y_i x_i \qquad \alpha_i + \beta_i = C$$

Simplifying dual objective

$$\sum_{i=1}^{m} \alpha_i y_i = 0 \qquad \vec{w} = \sum_{i=1}^{m} \alpha_i y_i x_i \qquad \alpha_i + \beta_i = C$$

$$\mathscr{L} = \frac{1}{2} \|\vec{w}_i\| - \sum_{i}^{m} \alpha_i y_i \vec{w} \cdot \vec{x}_i - \sum_{i}^{m} \alpha_i y_i b - \sum_{i=1}^{m} \beta_i \xi_i \tag{10}$$

Simplifying dual objective

$$\sum_{i=1}^{m} \alpha_i y_i = 0 \qquad \vec{w} = \sum_{i=1}^{m} \alpha_i y_i x_i \qquad \alpha_i + \beta_i = C$$

$$\mathscr{L} = \frac{1}{2} \left\| \sum_{i=1}^{m} \alpha_i y_i \vec{x}_i \right\| - \sum_{i}^{m} \sum_{j}^{m} \alpha_i \alpha_j y_i y_j (\vec{x}_j \cdot \vec{x}_i) - \sum_{i}^{m} \alpha_i y_i b - \sum_{i=1}^{m} \beta_i \xi_i \tag{10}$$

Simplifying dual objective

$$\sum_{i=1}^{m} \alpha_i y_i = 0 \qquad \vec{w} = \sum_{i=1}^{m} \alpha_i y_i x_i \qquad \alpha_i + \beta_i = C$$

$$\mathscr{L} = \frac{1}{2} \left\| \sum_{i=1}^{m} \alpha_i y_i \vec{x}_i \right\| - \sum_{i}^{m} \sum_{j}^{m} \alpha_i \alpha_j y_i y_j (\vec{x}_j \cdot \vec{x}_i) - \sum_{i}^{m} \alpha_i y_i b - \sum_{i=1}^{m} \beta_i \xi_i \tag{10}$$

Simplifying dual objective

$$\sum_{i=1}^{m} \alpha_i y_i = 0 \qquad \vec{w} = \sum_{i=1}^{m} \alpha_i y_i x_i \qquad \alpha_i + \beta_i = C$$

$$\mathscr{L} = \frac{1}{2} \left\| \sum_{i=1}^{m} \alpha_i y_i \vec{x}_i \right\| - \sum_{i}^{m} \sum_{j}^{m} \alpha_i \alpha_j y_i y_j (\vec{x}_j \cdot \vec{x}_i) - \sum_{i}^{m} \alpha_i y_i b + \sum_{i}^{m} \alpha_i \tag{10}$$

Simplifying dual objective

$$\sum_{i=1}^{m} \alpha_i y_i = 0 \qquad \vec{w} = \sum_{i=1}^{m} \alpha_i y_i x_i \qquad \alpha_i + \beta_i = C$$

$$\mathscr{L} = \frac{1}{2}\left\| \sum_{i=1}^{m} \alpha_i y_i \vec{x}_i \right\| - \sum_{i}^{m} \sum_{j}^{m} \alpha_i \alpha_j y_i y_j (\vec{x}_j \cdot \vec{x}_i) - \sum_{i}^{m} \alpha_i y_i b + \sum_{i}^{m} \alpha_i \tag{10}$$

Simplifying dual objective

$$\sum_{i=1}^{m} \alpha_i y_i = 0 \qquad \vec{w} = \sum_{i=1}^{m} \alpha_i y_i x_i \qquad \alpha_i + \beta_i = C$$

$$\mathscr{L} = \frac{1}{2} \left\| \sum_{i=1}^{m} \alpha_i y_i \vec{x}_i \right\| - \sum_{i}^{m} \sum_{j}^{m} \alpha_i \alpha_j y_i y_j (\vec{x}_j \cdot \vec{x}_i) - \sum_{i}^{m} \alpha_i y_i b + \sum_{i}^{m} \alpha_i \tag{10}$$

Simplifying dual objective

$$\sum_{i=1}^{m} \alpha_i y_i = 0 \qquad \vec{w} = \sum_{i=1}^{m} \alpha_i y_i x_i \qquad \alpha_i + \beta_i = C$$

$$\mathscr{L} = \frac{1}{2} \left\| \sum_{i=1}^{m} \alpha_i y_i \vec{x}_i \right\| - \sum_{i}^{m} \sum_{j}^{m} \alpha_i \alpha_j y_i y_j (\vec{x}_j \cdot \vec{x}_i) + \sum_{i}^{m} \alpha_i \tag{10}$$

First two terms are the same!

Simplifying dual objective

$$\sum_{i=1}^{m} \alpha_i y_i = 0 \qquad \vec{w} = \sum_{i=1}^{m} \alpha_i y_i x_i \qquad \alpha_i + \beta_i = C$$

$$\mathscr{L} = -\frac{1}{2} \sum_{i}^{m} \sum_{j}^{m} \alpha_i \alpha_j y_i y_j (\vec{x}_j \cdot \vec{x}_i) + \sum_{i}^{m} \alpha_i \tag{10}$$

Just like separable case, except that we add the constraint that $\alpha_i \leq C$!

Wrapup

- Adding slack variables don't break the SVM problem
- Very popular algorithm
  - SVMLight (many options)
  - Libsvm / Liblinear (very fast)
  - Weka (friendly)
  - pyml (Python focused, from Colorado)
- Next up: simple algorithm for finding SVMs

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).

$$\mathscr{L}(w, b, \alpha) = \frac{1}{2}||w||^2 - \sum_{i=1}^{m} \alpha_i \left[y_i(w \cdot x_i + b) - 1\right] \qquad (11)$$

Lagrange Multipliers

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

$$\mathscr{L}(w, b, \alpha) = \frac{1}{2}||w||^2 - \sum_{i=1}^{m} \alpha_i \left[y_i(w \cdot x_i + b) - 1 \right] \tag{11}$$

If $\alpha \neq 0$, then $y_i(w \cdot x_i + b) = \pm 1$.

Solving Lagrangian

Weights

$$\vec{w} = \sum_{i=1}^{m} \alpha_i y_i \vec{x}_i \tag{12}$$

Solving Lagrangian

Weights

$$\vec{w} = \sum_{i=1}^{m} \alpha_i y_i \vec{x}_i \tag{12}$$

Bias

$$0 = \sum_{i=1}^{m} \alpha_i y_i \tag{13}$$

Solving Lagrangian

Weights

$$\vec{w} = \sum_{i=1}^{m} \alpha_i y_i \vec{x}_i \tag{12}$$

Bias

$$0 = \sum_{i=1}^{m} \alpha_i y_i \tag{13}$$

Support Vector-ness

$$\alpha_i = 0 \vee y_i(w \cdot x_i + b) = 1 \tag{14}$$

Reparameterize in terms of α

$$\max_{\vec{\alpha}} \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{i=1}^{m} \alpha_i \alpha_j y_i y_j (\vec{x}_i \cdot \vec{x}_j) \tag{15}$$

Outline for SVM Optimization (SMO)

1. Select two examples i, j
2. Get a learning rate η
3. Update α_j
4. Update α_i

Plan

Dual Objective

Algorithm Big Picture

The Algorithm

Recap

Contrast with SG

- There's a learning rate η that depends on the data
- Use the error of an example to derive update
- You update multiple α at once

Contrast with SG

- There's a learning rate η that depends on the data
- Use the error of an example to derive update
- You update multiple α 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?

$$\alpha_i = 0 \Rightarrow \quad y_i(w \cdot x_i + b) \geq 1 \tag{16}$$

$$\alpha_i = C \Rightarrow \quad y_i(w \cdot x_i + b) \leq 1 \tag{17}$$

$$0 < \alpha_i < C \Rightarrow \quad y_i(w \cdot x_i + b) = 1 \tag{18}$$

(Karush-Kuhn-Tucker Conditions)

More details

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

$$\alpha_i = 0 \Rightarrow \quad y_i(w \cdot x_i + b) \geq 1 \tag{16}$$

$$\alpha_i = C \Rightarrow \quad y_i(w \cdot x_i + b) \leq 1 \tag{17}$$

$$0 < \alpha_i < C \Rightarrow \quad y_i(w \cdot x_i + b) = 1 \tag{18}$$

(Karush-Kuhn-Tucker Conditions)

- Keep checking (to some tolerance)

Plan

Dual Objective

Algorithm Big Picture

The Algorithm

Recap

Step 1: Select i and j

- Iterate over $i = \{1, \dots m\}$
- Repeat until KKT conditions are met
- Choose j randomly from $m - 1$ other options
- You can do better (particularly for large datasets)

Step 2: Optimize α_j

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

$y_i \neq y_j$

$$L = \max(0, \alpha_j - \alpha_i) \tag{19}$$

$$H = \min(C, C + \alpha_j - \alpha_i) \tag{20}$$

$y_i = y_j$

$$L = \max(0, \alpha_i + \alpha_j - C) \tag{21}$$

$$H = \min(C, \alpha_j + \alpha_i) \tag{22}$$

Step 2: Optimize α_j

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

$y_i \neq y_j$

$$L = \max(0, \alpha_j - \alpha_i) \quad (19)$$
$$H = \min(C, C + \alpha_j - \alpha_i) \quad (20)$$

$y_i = y_j$

$$L = \max(0, \alpha_i + \alpha_j - C) \quad (21)$$
$$H = \min(C, \alpha_j + \alpha_i) \quad (22)$$

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

Step 2: Optimize α_j

Compute errors for i and j

$$E_k \equiv f(x_k) - y_k \tag{23}$$

Step 2: Optimize α_j

Compute errors for i and j

$$E_k \equiv f(x_k) - y_k \tag{23}$$

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

$$\eta = 2x_i \cdot x_j - x_i \cdot x_i - x_j \cdot x_j \tag{24}$$

Step 2: Optimize α_j

Compute errors for i and j

$$E_k \equiv f(x_k) - y_k \tag{23}$$

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

$$\eta = 2x_i \cdot x_j - x_i \cdot x_i - x_j \cdot x_j \tag{24}$$

for new value for α_j

$$\alpha_j^* = \alpha_j^{(old)} - \frac{y_j(E_i - E_j)}{\eta} \tag{25}$$

Step 2: Optimize α_j

Compute errors for i and j

$$E_k \equiv f(x_k) - y_k \tag{23}$$

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

$$\eta = 2x_i \cdot x_j - x_i \cdot x_i - x_j \cdot x_j \tag{24}$$

for new value for α_j

$$\alpha_j^* = \alpha_j^{(old)} - \frac{y_j(E_i - E_j)}{\eta} \tag{25}$$

Similar to stochastic gradient, but with additional error term.

Step 2: Optimize α_j

Compute errors for i and j

$$E_k \equiv f(x_k) - y_k \tag{23}$$

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

$$\eta = 2x_i \cdot x_j - x_i \cdot x_i - x_j \cdot x_j \tag{24}$$

for new value for α_j

$$\alpha_j^* = \alpha_j^{(old)} - \frac{y_j(E_i - E_j)}{\eta} \tag{25}$$

Similar to stochastic gradient, but with additional error term. If α_j^* is outside $[L, H]$, clip it so that it is within the range.

Step 3: Optimize α_i

Set α_i:

$$\alpha_i^* = \alpha_i^{(old)} + y_i y_j \left(\alpha_j^{(old)} - \alpha_j \right) \tag{26}$$

Step 3: Optimize α_i

Set α_i:

$$\alpha_i^* = \alpha_i^{(old)} + y_i y_j \left(\alpha_j^{(old)} - \alpha_j \right) \tag{26}$$

This balances out the move that we made for α_j.

Step 4: Optimize the threshold b

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

- If $0 < \alpha_i < C$

$$b = b_1 = b - E_i - y_i(\alpha_i^* - \alpha_i^{(old)})x_i \cdot x_i - y_j(\alpha_j^* - \alpha_j^{(old)})x_i \cdot x_j \quad (27)$$

Step 4: Optimize the threshold b

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

- If $0 < \alpha_i < C$

$$b = b_1 = b - E_i - y_i(\alpha_i^* - \alpha_i^{(old)})x_i \cdot x_i - y_j(\alpha_j^* - \alpha_j^{(old)})x_i \cdot x_j \quad (27)$$

- If $0 < \alpha_j < C$

$$b = b_2 = b - E_j - y_i(\alpha_i^* - \alpha_i^{(old)})x_i \cdot x_j - y_j(\alpha_j^* - \alpha_j^{(old)})x_j \cdot x_j \quad (28)$$

Step 4: Optimize the threshold b

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

- If $0 < \alpha_i < C$

$$b = b_1 = b - E_i - y_i(\alpha_i^* - \alpha_i^{(old)})x_i \cdot x_i - y_j(\alpha_j^* - \alpha_j^{(old)})x_i \cdot x_j \quad (27)$$

- If $0 < \alpha_j < C$

$$b = b_2 = b - E_j - y_i(\alpha_i^* - \alpha_i^{(old)})x_i \cdot x_j - y_j(\alpha_j^* - \alpha_j^{(old)})x_j \cdot x_j \quad (28)$$

- If both α_i and α_j are at the bounds, then anything between b_1 and b_2 works, so we set

$$b = \frac{b_1 + b_2}{2} \quad (29)$$

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?

Iterations / Details

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

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 α's without changing anything**

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!