University of Colorado **Boulder**

Department of Computer Science

CSCI 5622: Machine Learning

Chris Ketelsen

# Lecture 13: Support Vector Machines
# Soft-Margin SVM and the SMO Algorithm

# Binary Classification

**Given**: $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{m}$ training examples

$$\mathbf{x}_i \in \mathbb{R}^D \qquad y_i \in \{-1, +1\}$$

**Goal**: Given new data $\mathbf{x}$, predict its label $y$

**SVM**: Learn a linear decision rule of the form

$$\hat{y} = \begin{cases} +1 & \text{if } \mathbf{w}^T\mathbf{x} + b \geq 0 \\ -1 & \text{if } \mathbf{w}^T\mathbf{x} + b < 0 \end{cases}$$

# Our Roadmap

**Last Time**:

- Talk linear SVM when data is linearly separable

**This Time**:

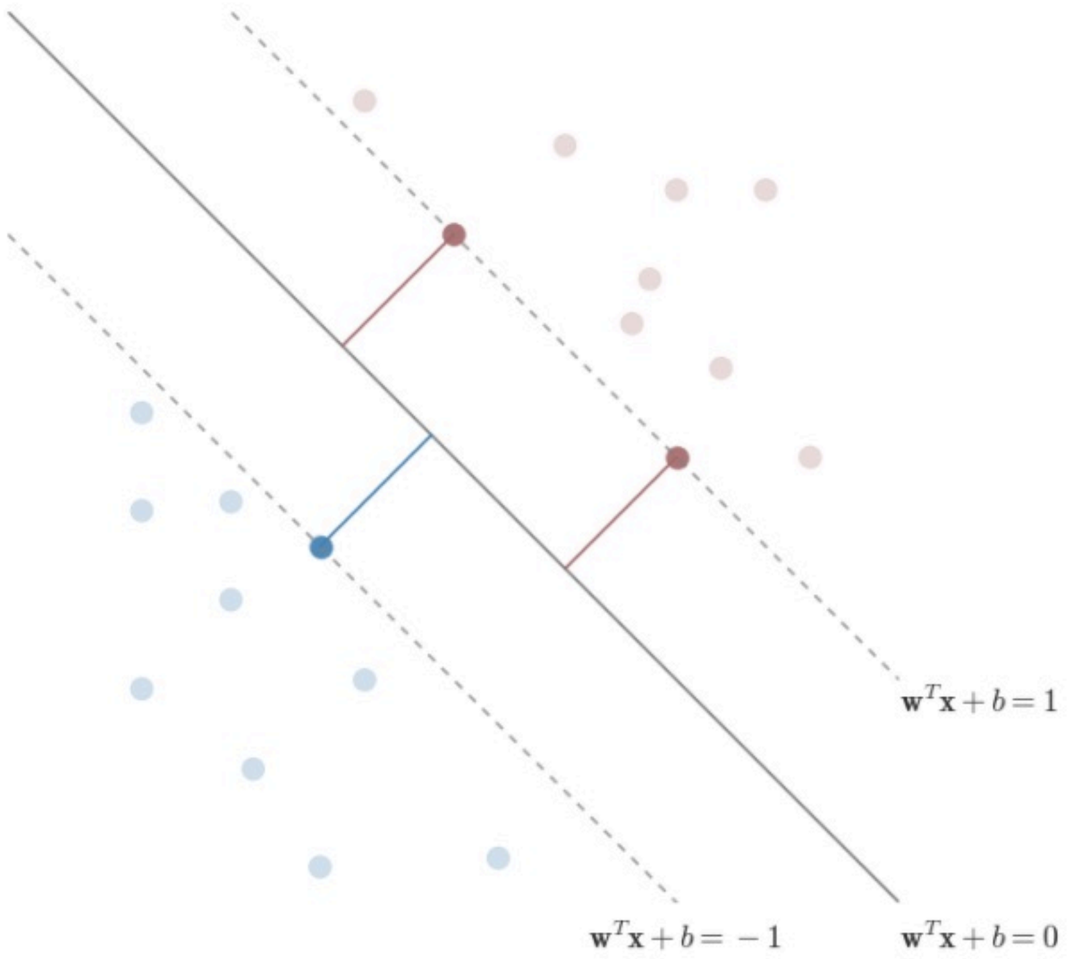
- Talk linear SVM when data is not linearly separable
- Look at efficient algorithm for finding weights

**Next Time**:

- Look at SVM as a nonlinear classifier
- Learn the Kernel Trick

# Last Time: The Hard-Margin SVM

Find weights $\mathbf{w}$ and bias $b$ that maximize **margin** $M$



# Last Time: The Hard-Margin SVM

Optimization Problem for Parameters:

$$\begin{aligned} \min_{\mathbf{w},b} \quad & \frac{1}{2}\|\mathbf{w}\|^2 \\ \text{s.t.} \quad & y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 \quad \text{for } i = 1, \ldots, m \end{aligned}$$

Define Lagrangian

$$L(\mathbf{w}, b, \alpha) = \frac{1}{2}\|\mathbf{w}\|^2 + \sum_{i=1}^{m} \alpha_i \left[1 - y_i(\mathbf{w}^T\mathbf{x}_i + b)\right], \text{ s.t. } \alpha_i \geq 0$$

# Last Time: The Hard-Margin SVM

Optimization Problem for Parameters:

$$\begin{aligned} \min_{\mathbf{w},b} \quad & \frac{1}{2}\|\mathbf{w}\|^2 \\ \text{s.t.} \quad & y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 \quad \text{for } i = 1, \ldots, m \end{aligned}$$

Define Lagrangian

$$L(\mathbf{w}, b, \alpha) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{m} \alpha_i \left[y_i(\mathbf{w}^T\mathbf{x}_i + b) - 1\right], \text{ s.t. } \alpha_i \geq 0$$

# Last Time: The Hard-Margin SVM

We could solve problem two different ways

$$\max_{\alpha} \min_{\mathbf{w},b} L(\mathbf{w}, b, \alpha) \quad \leq \quad \min_{\mathbf{w},b} \max_{\alpha} L(\mathbf{w}, b, \alpha)$$

Dual Primal

We decided to solve the dual problem

This lead to a bunch of conditions on parameters

# Last Time: The Hard-Margin SVM

**Primal and Dual Feasibility**:

$$y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1, \quad \alpha_i \geq 0$$

**From Maximizing w.r.t. w and $b$**:

$$\mathbf{w} = \sum_{i=1}^{m} \alpha_i y_i \mathbf{x}_i, \qquad \sum_{i=1}^{m} \alpha_i y_i = 0$$

**Complementary Slackness Condition**:

$$\alpha_i \left[ y_i(\mathbf{w}^T\mathbf{x}_i + b) - 1 \right] = 0$$

# Last Time: The Hard-Margin SVM

**Primal and Dual Feasibility**:

$$y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1, \quad \alpha_i \geq 0$$

**From Maximizing w.r.t. w and $b$**:

$$\mathbf{w} = \sum_{i=1}^{m} \alpha_i y_i \mathbf{x}_i, \quad \sum_{i=1}^{m} \alpha_i y_i = 0$$

**Complementary Slackness Condition**:

$$\alpha_i \left[ y_i(\mathbf{w}^T\mathbf{x}_i + b) - 1 \right] = 0$$

All together: **Karush–Kuhn–Tucker (KKT) conditions**

# Last Time: The Hard-Margin SVM

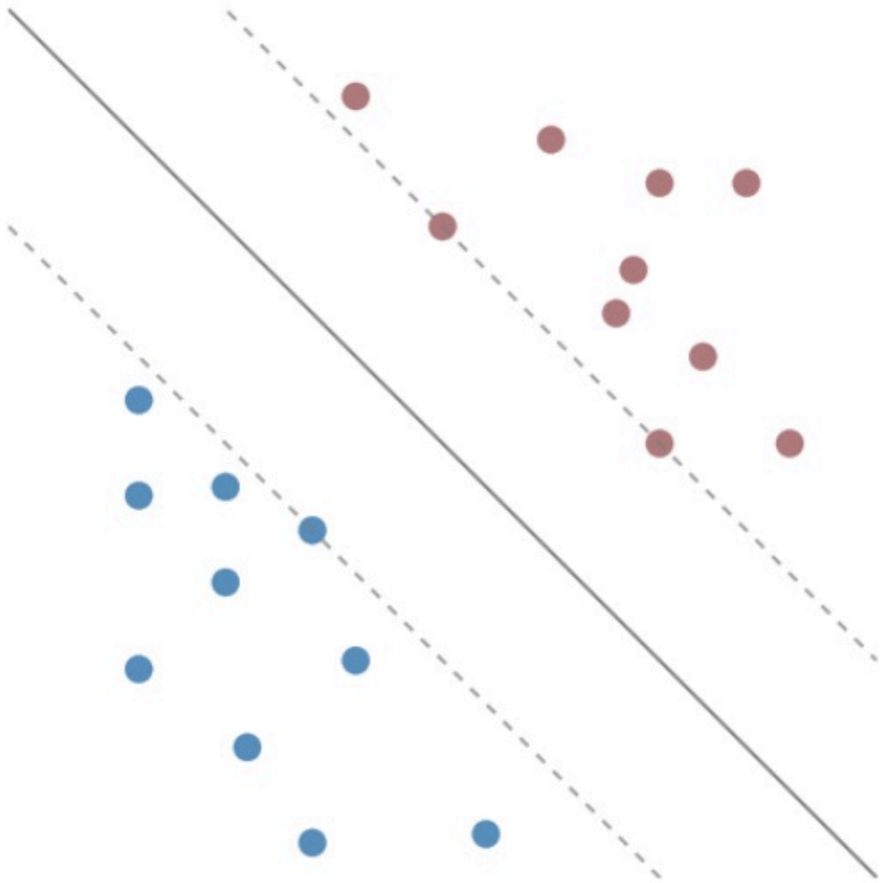
The dual problem became

$$\begin{aligned} \max_{\alpha} \quad & \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{x}_j^T \mathbf{x}_i \\ \text{s.t.} \quad & \alpha_i > 0 \text{ if } y_i(\mathbf{w}^T \mathbf{x}_i + b) = 1 \\ & \alpha_i = 0 \text{ if } y_i(\mathbf{w}^T \mathbf{x}_i + b) > 1 \\ & \sum_i \alpha_i y_i = 0 \end{aligned}$$

We noted that objective function only depends on $\mathbf{x}_j^T \mathbf{x}_i$

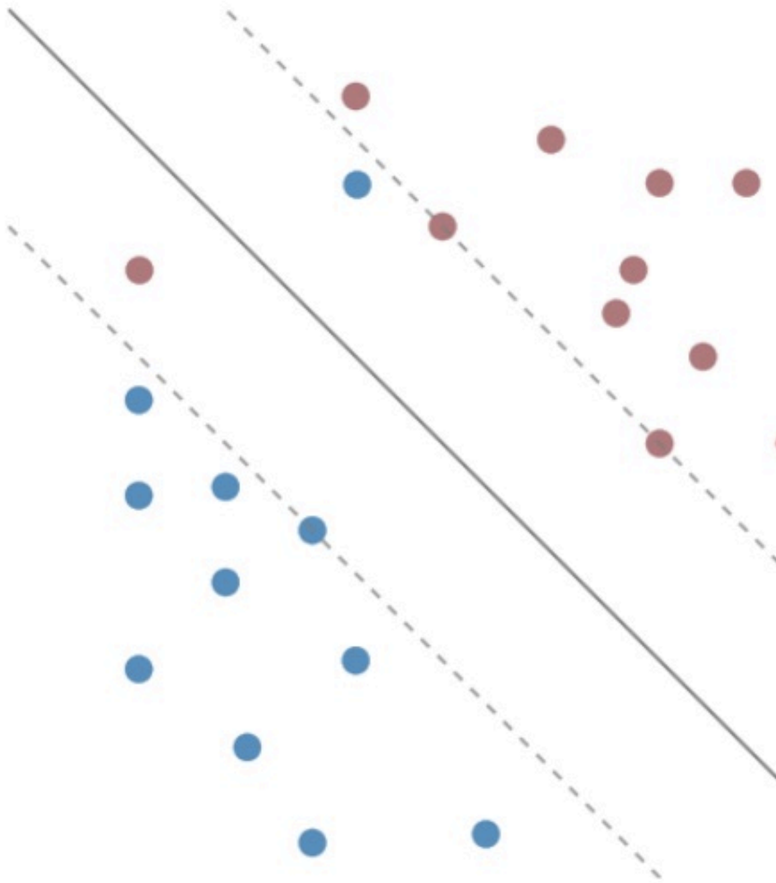
# Soft-Margin SVMs

OK, before we only considered the linearly separable case



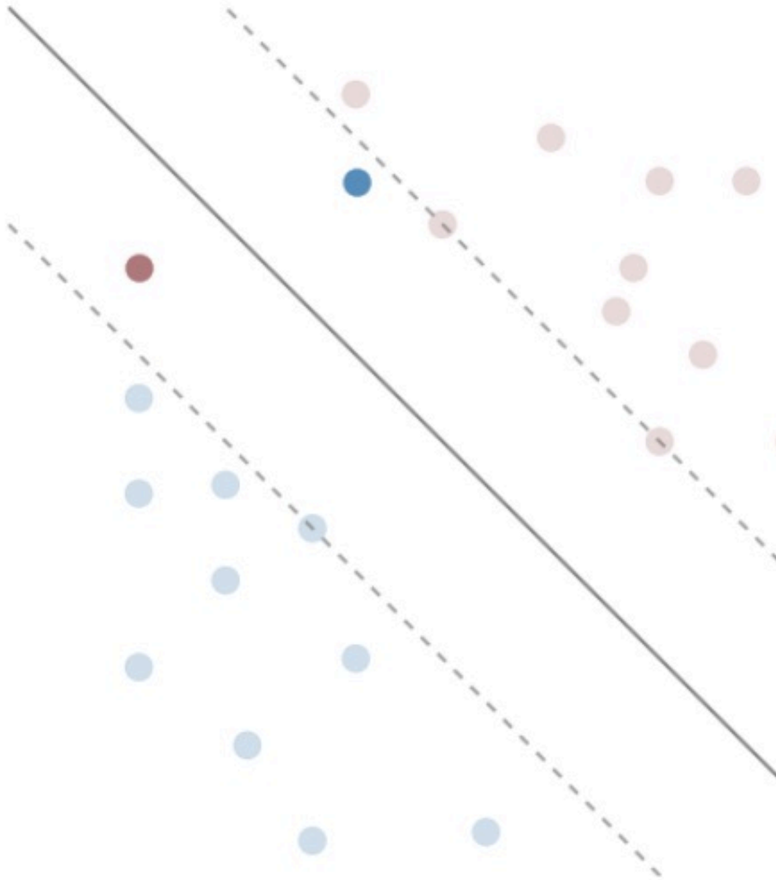
# Soft-Margin SVMs

Now we want to consider the non separable case



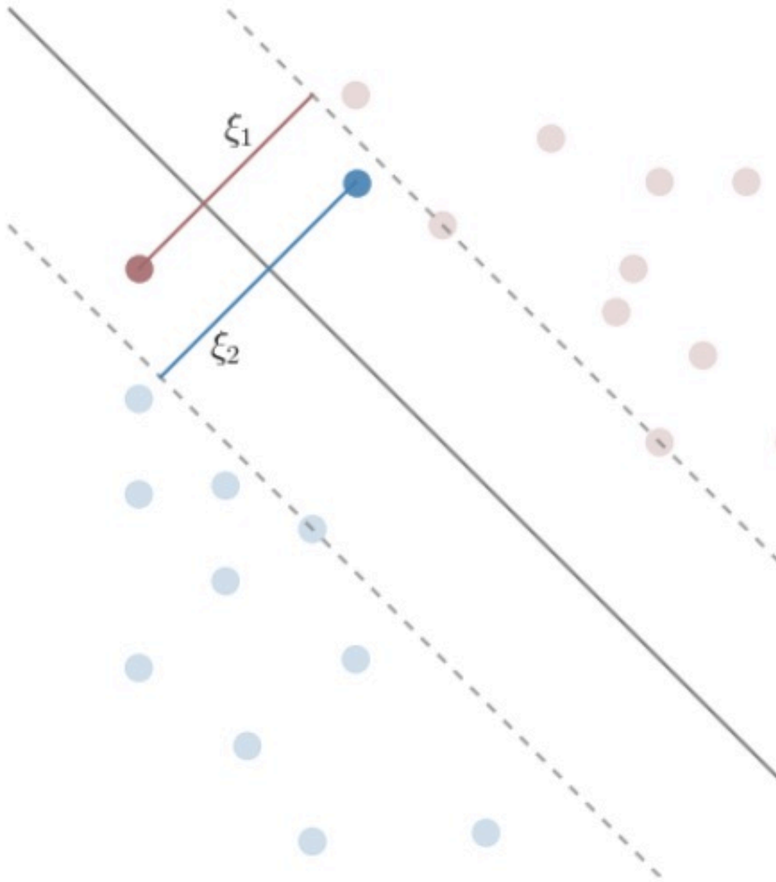
# Soft-Margin SVMs

Now we want to consider the non separable case



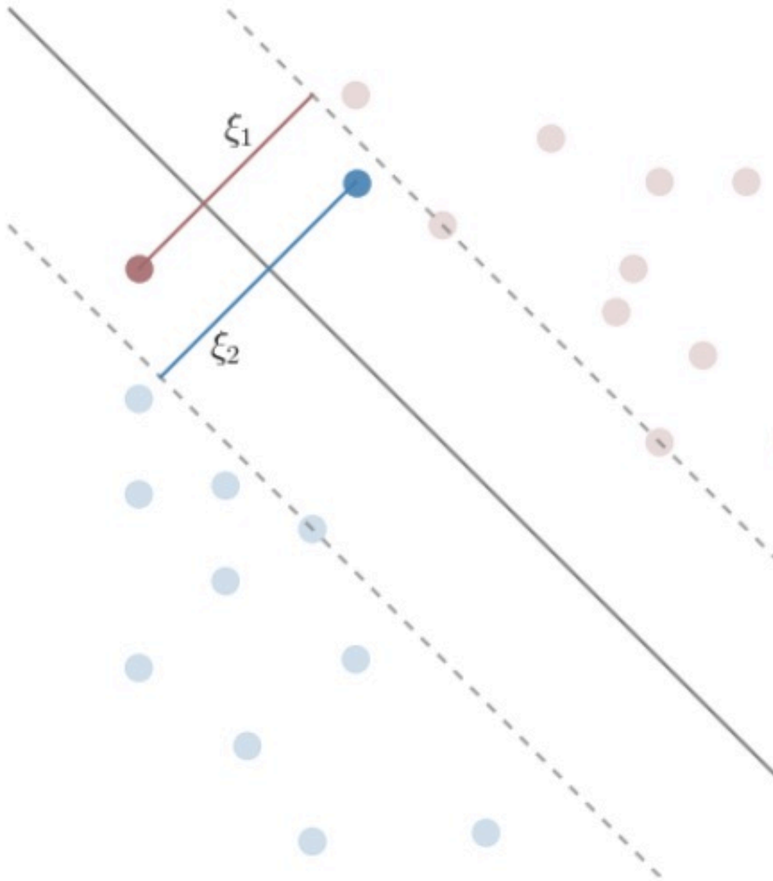
# Soft-Margin SVMs

Introduce nonnegative slack variable $\xi_i$ for each point in training set



# Soft-Margin SVMs

Allow some $\xi_i > 0$ but hope most are $\xi_i = 0$



# Soft-Margin SVMs

How does this change the mathematical lanscape?

# Soft-Margin SVMs

How does this change the mathematical lanscape?

**Primal Objective Function**:

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

# Soft-Margin SVMs

How does this change the mathematical lanscape?

**Primal Objective Function**:

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

$C$ is a tuning parameter that balances

- Maximizing the margin
- Classifying training examples correctly

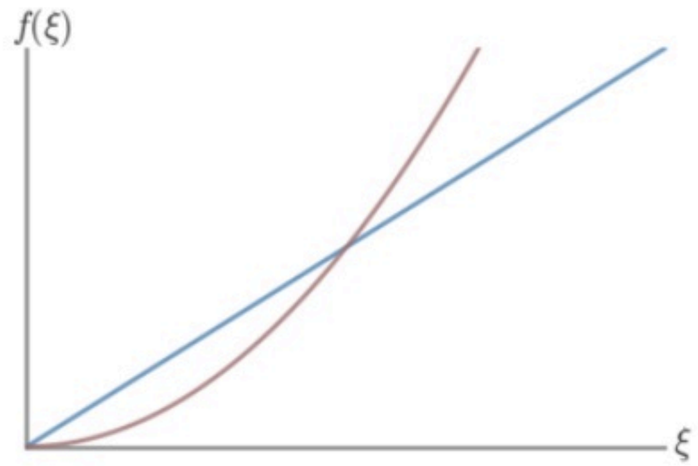
# Soft-Margin SVMs

How does this change the mathematical lanscape?

**Primal Objective Function**:

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

Exponent $p$ determines how bad *wrongness* of point scales



# Soft-Margin SVMs

How does this change the mathematical lanscape?

**Primal Objective Function**:

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

Exponent $p$ determines how bad *wrongness* of point scales

We'll choose $p = 1$ but larger values common as well

# Soft-Margin SVMs

How does this change the mathematical lanscape?

**Primal Constraints**:

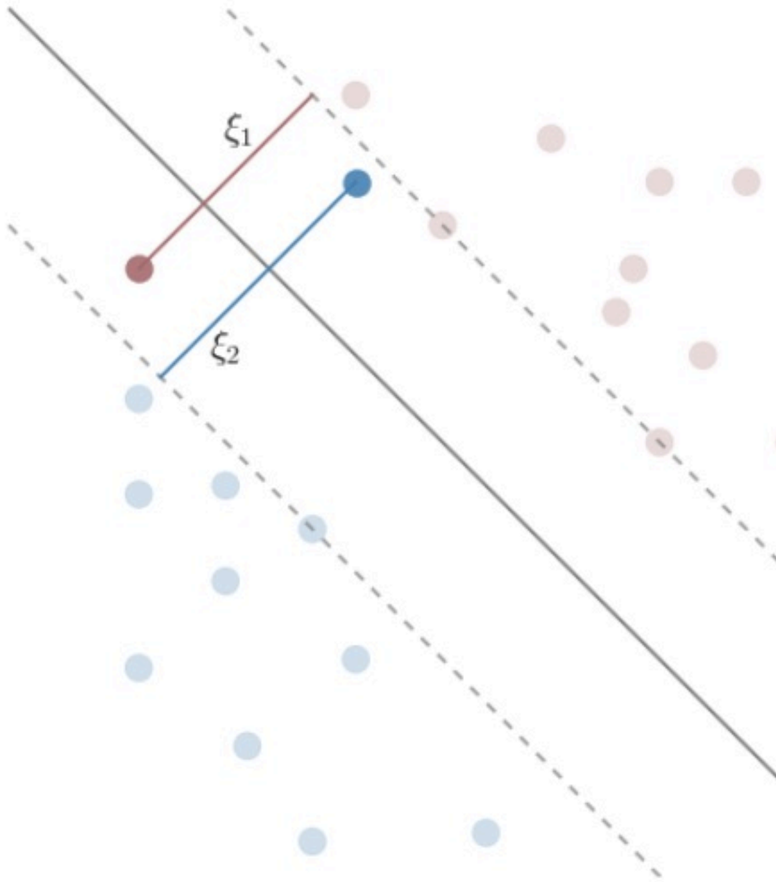
$$y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 \quad \text{becomes} \quad y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i$$

**Intuition**:

- $\xi_i = 0$ means at least one margin on correct side of DB
- $\xi_i = 1/2$ means one-half margin on correct side of DB
- $\xi_i = 2$ means one margin on *wrong* side of DB

# Soft-Margin SVMs

Introduce nonnegative slack variable $\xi_i$ for each point in training set



# Soft-Margin SVMs

How does this change the mathematical lanscape?

**New Lagrangian**:

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

$$-\sum_{i=1}^{m}\alpha_i\left[y_i(\mathbf{w}^T\mathbf{x}_i + b) - 1 + \xi_i\right] - \sum_{i=1}^{m}\beta_i\xi_i$$

For Dual Problem, minimize $L(\mathbf{w}, b, \xi, \alpha, \beta)$ w.r.t. $\mathbf{w}$, $b$, and $\xi$

$$\nabla_{\mathbf{w}}L = \mathbf{0}, \quad \frac{\partial L}{\partial b} = 0, \quad \nabla_{\xi}L = \mathbf{0}$$

# New KKT Conditions

**Primal and Dual Feasibility**:

$$y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0, \quad \alpha_i \geq 0, \quad \beta_i \geq 0$$

**From Maximizing w.r.t. w and $b$**:

$$\mathbf{w} = \sum_{i=1}^{m} \alpha_i y_i \mathbf{x}_i, \quad \sum_{i=1}^{m} \alpha_i y_i = 0, \quad \alpha_i + \beta_i = C$$

**Complementary Slackness Conditions**:

$$\alpha_i \left[ y_i(\mathbf{w}^T\mathbf{x}_i + b) - 1 + \xi_i \right] = 0, \quad \beta_i \xi_i = 0$$

# New Dual Problem

Miraculously, doesn't change the dual objective function

$$\max_{\alpha} \quad \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{x}_j^T \mathbf{x}_i$$

# New Dual Problem

Miraculously, doesn't change the dual objective function

$$\max_{\alpha} \quad \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{x}_j^T \mathbf{x}_i$$

But does modify one of the constraints. Now have

$$0 \leq \alpha_i \leq C, \quad \text{for } i = 1, \ldots, m$$

Easy proof from KKT conditions:

$$\alpha_i + \beta_i = C \quad \Rightarrow \quad \alpha_i = C - \beta_i$$

But $\beta_i \geq 0$ implies that $\alpha \leq C$. Together with old constraint gives

$$0 \leq \alpha_i \leq C$$

## Better Slackness Constraint

$$\alpha_i \left[ y_i(\mathbf{w}^T \mathbf{x}_i + b) - 1 + \xi_i \right] = 0$$

Now three possible cases for $\mathbf{x}_i$:

- $\mathbf{x}_i$ on correct side of support vector boundary
- $\mathbf{x}_i$ on support vector boundary
- $\mathbf{x}_i$ on wrong side of support vector boundary

## Better Slackness Constraint

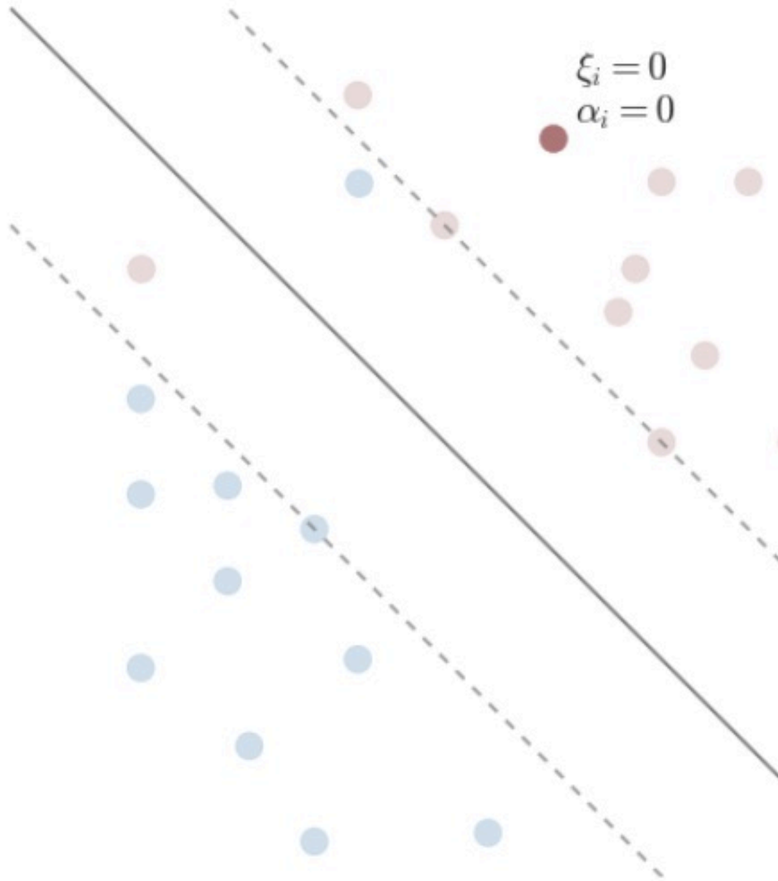
$$\alpha_i \left[ y_i(\mathbf{w}^T \mathbf{x}_i + b) - 1 + \xi_i \right] = 0$$

Now three possible cases for $\mathbf{x}_i$:

- $\mathbf{x}_i$ on correct side of support vector boundary

# Better Slackness Constraint

$\mathbf{x}_i$ on correct side of support vector boundary



# Better Slackness Constraint

$$\alpha_i \left[ y_i(\mathbf{w}^T \mathbf{x}_i + b) - 1 + \xi_i \right] = 0$$

Now three possible cases for $\mathbf{x}_i$:

- $\mathbf{x}_i$ on correct side of support vector boundary

$$\xi_i = 0 \quad \Rightarrow \quad \alpha_i \left[ y_i(\mathbf{w}^T \mathbf{x}_i + b) - 1 \right] = 0$$

$$\text{Inactive Constraint} \quad \Rightarrow \quad \alpha_i = 0 \quad \text{and} \quad y_i(\mathbf{w}^T \mathbf{x}_i + b) > 1$$

## Better Slackness Constraint

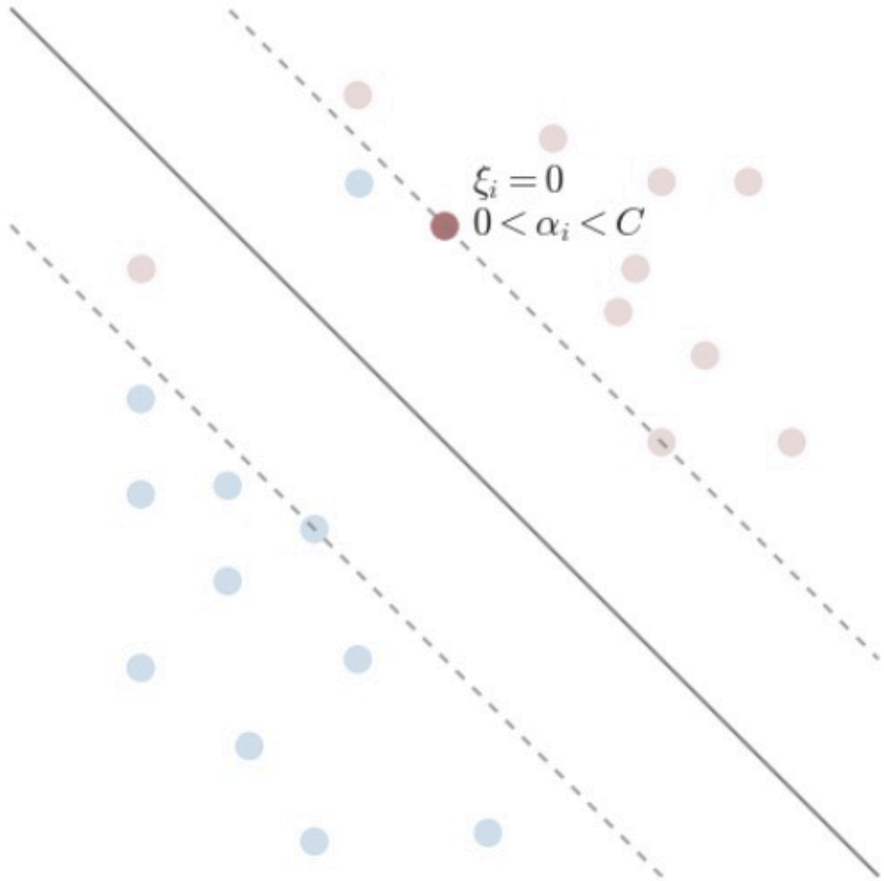
$$\alpha_i \left[ y_i(\mathbf{w}^T \mathbf{x}_i + b) - 1 + \xi_i \right] = 0$$

Now three possible cases for $\mathbf{x}_i$:

- $\mathbf{x}_i$ on support vector boundary

# Better Slackness Constraint

$\mathbf{x}_i$ on support vector boundary



# Better Slackness Constraint

$$\alpha_i \left[ y_i(\mathbf{w}^T \mathbf{x}_i + b) - 1 + \xi_i \right] = 0$$

Now three possible cases for $\mathbf{x}_i$:

- $\mathbf{x}_i$ on support vector boundary

$$\xi_i = 0 \quad \Rightarrow \quad \alpha_i \left[ y_i(\mathbf{w}^T \mathbf{x}_i + b) - 1 \right] = 0$$

$$\text{Active Constraint} \quad \Rightarrow \quad 0 < \alpha_i < C \quad \text{and} \quad y_i(\mathbf{w}^T \mathbf{x}_i + b) = 1$$

## Better Slackness Constraint

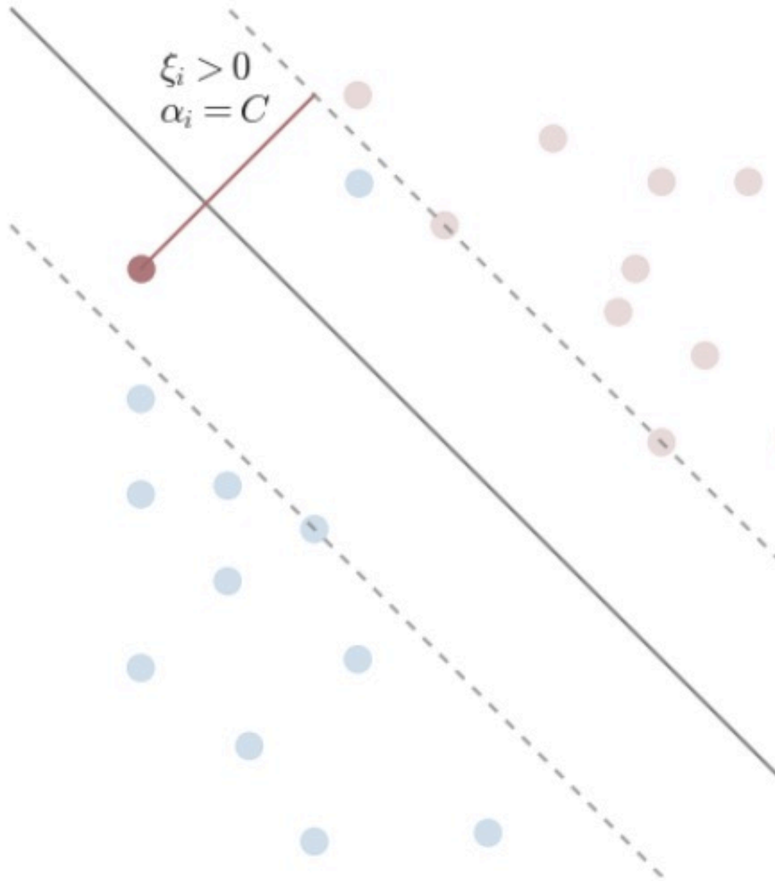
$$\alpha_i \left[ y_i(\mathbf{w}^T \mathbf{x}_i + b) - 1 + \xi_i \right] = 0$$

Now three possible cases for $\mathbf{x}_i$:

- $\mathbf{x}_i$ on wrong side of support vector boundary

# Soft-Margin SVMs

$\mathbf{x}_i$ on wrong side of support vector boundary



## Better Slackness Constraint

$$\alpha_i \left[ y_i(\mathbf{w}^T \mathbf{x}_i + b) - 1 + \xi_i \right] = 0$$

Now three possible cases for $\mathbf{x}_i$:

- $\mathbf{x}_i$ on wrong side of support vector boundary

$$\xi_i > 0 \quad \Rightarrow \quad \beta_i = 0 \quad \Rightarrow \quad \alpha_i = C$$

Plugging into complementary slackness condition:

$$y_i(\mathbf{w}^T \mathbf{x}_i + b) = 1 - \xi_i \leq 1$$

Final Form: $\alpha_i = C$ and $y_i(\mathbf{w}^T \mathbf{x}_i + b) \leq 1$

# Final Dual Problem

After all that work, we have

$$\begin{aligned} \max_{\alpha} \quad & \textstyle\sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{x}_j^T \mathbf{x}_i \\ \text{s.t.} \quad & 0 \le \alpha_i \le C \text{ for } \quad i = 1, \ldots, m \\ & \textstyle\sum_i \alpha_i y_i = 0 \end{aligned}$$

with simplified KKT conditions

$$\begin{aligned} \alpha_i = 0 \quad &\Rightarrow \quad y_i(\mathbf{w}^T \mathbf{x}_i + b) \ge 1 \\ \alpha_i = C \quad &\Rightarrow \quad y_i(\mathbf{w}^T \mathbf{x}_i + b) \le 1 \\ 0 < \alpha_i < C \quad &\Rightarrow \quad y_i(\mathbf{w}^T \mathbf{x}_i + b) = 1 \end{aligned}$$

# The Sequential Minimal Optimizer

**Trivia**:

- Invented by Jon Platt in 1998 at Microsoft Research
- Called *Minimal* due to solving small sub-problems

# Brief Interlude: Coordinate Ascent

Define $$L(\alpha_1, \ldots, \alpha_m) = \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{x}_j^T \mathbf{x}_i$$

**Idea**: Loop over each training example, change $\alpha_i$ to make $L(\alpha)$ as large as possible

In [ ]:
```
while not converged:
    for ii in [1,...,m]:
        alpha[ii] = argmax_ahat L(alpha[1], ..., ahat, ..., alpha[m])
```

Coordinate ascent works OK for lots of problems

Won't work for us though because of constraint $\sum_{i=1}^{m} \alpha_i y_i = 0$

Have to change more than one $\alpha_i$ at a time!

# The SMO Algorithm

Define $$L(\alpha_1, \ldots, \alpha_m) = \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{x}_j^T \mathbf{x}_i$$

**Idea**: If we can't change only one $\alpha_i$ then change **TWO** $\alpha_i$'s

**General Outline**:

In [ ]:
```
while not converged:
    for ii in [1,...,m]:
        select jj != ii at random (or via heuristic)
        update alpha[ii] and alpha[jj] to maximize L(alpha)
    if KKT conditions satisfied:
        exit
```

# The SMO Algorithm

Given an $(i, j)$-pair, how to we optimize $L(\alpha)$?

We know from the linear equality constraint that

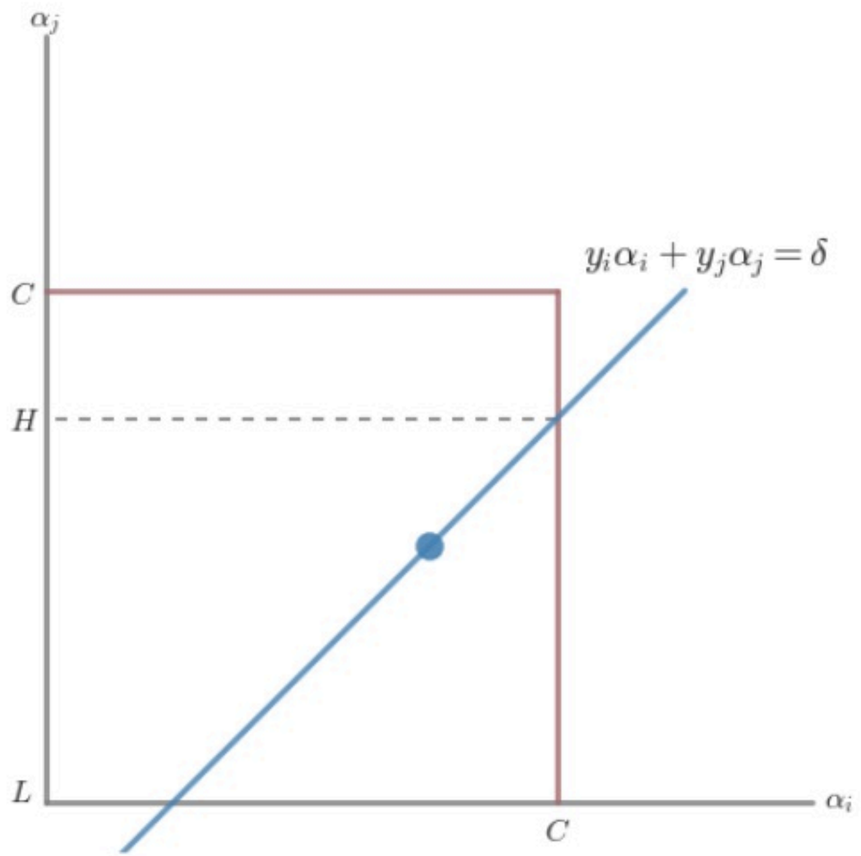
$$\sum_{i=1}^{m} \alpha_i y_i = 0 \quad \Rightarrow \quad \alpha_i y_i + \alpha_j y_j = -\sum_{k \neq i,j} \alpha_k y_k := \delta$$

Further, we know that both $\alpha$'s must satisfy $0 \leq \alpha \leq C$.

Need to choose $\alpha_i$ and $\alpha_j$ to satisfy inequality constraints, the equality constraint, and maximize $L(\alpha)$ as much as possible.

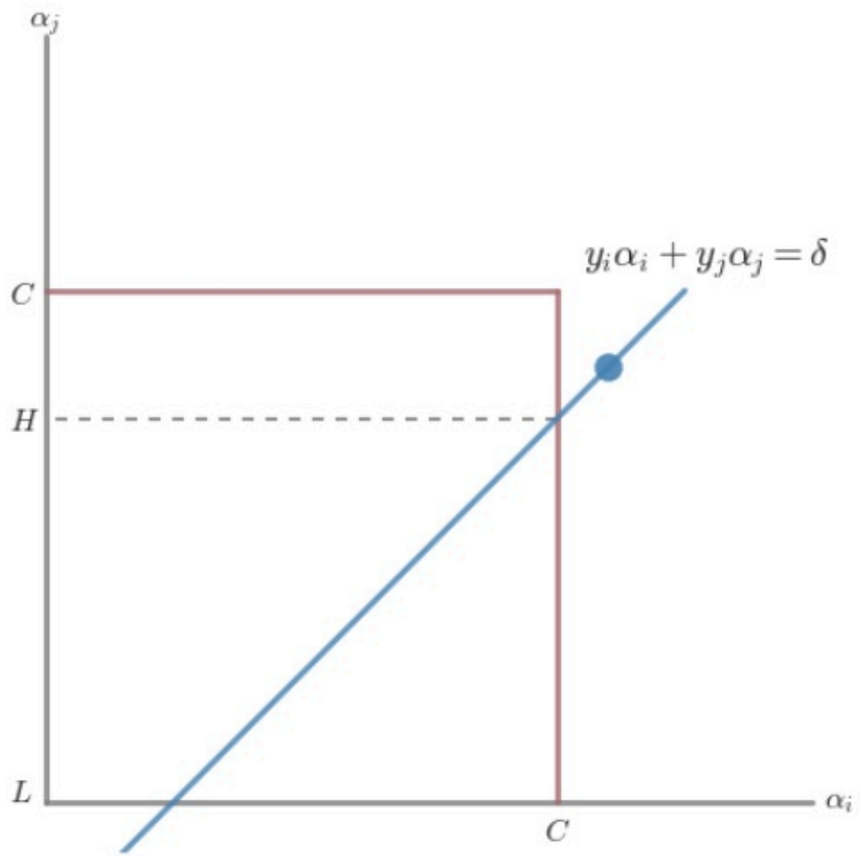
# The SMO Algorithm

Given an $(i, j)$-pair, how to we optimize $L(\alpha)$?



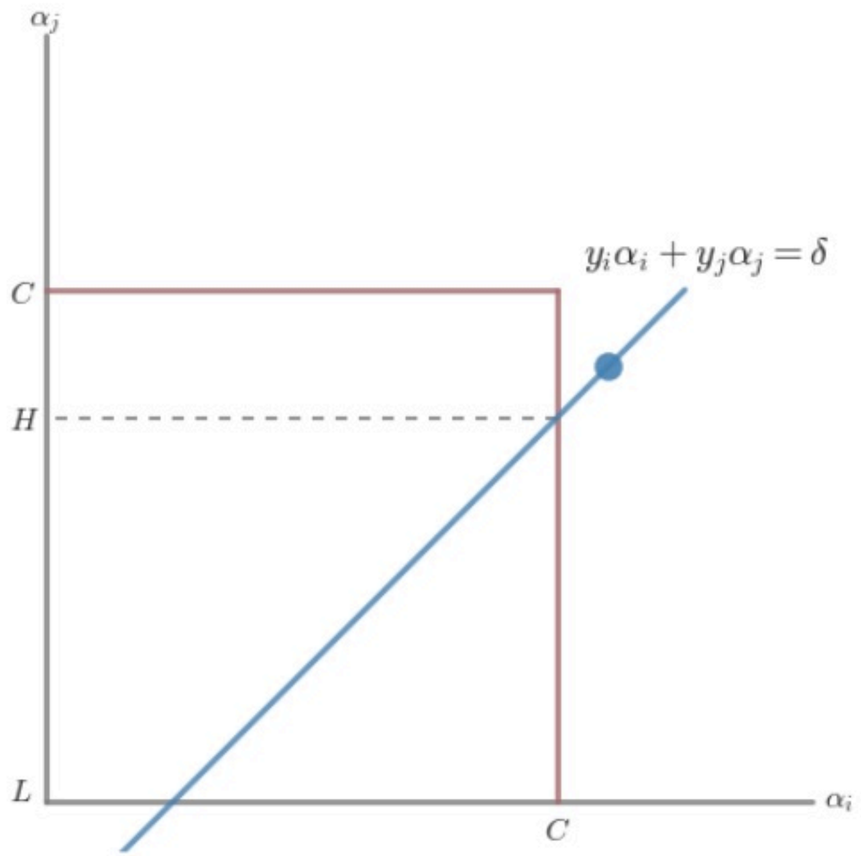
# The SMO Algorithm

Given an $(i, j)$-pair, how to we optimize $L(\alpha)$?



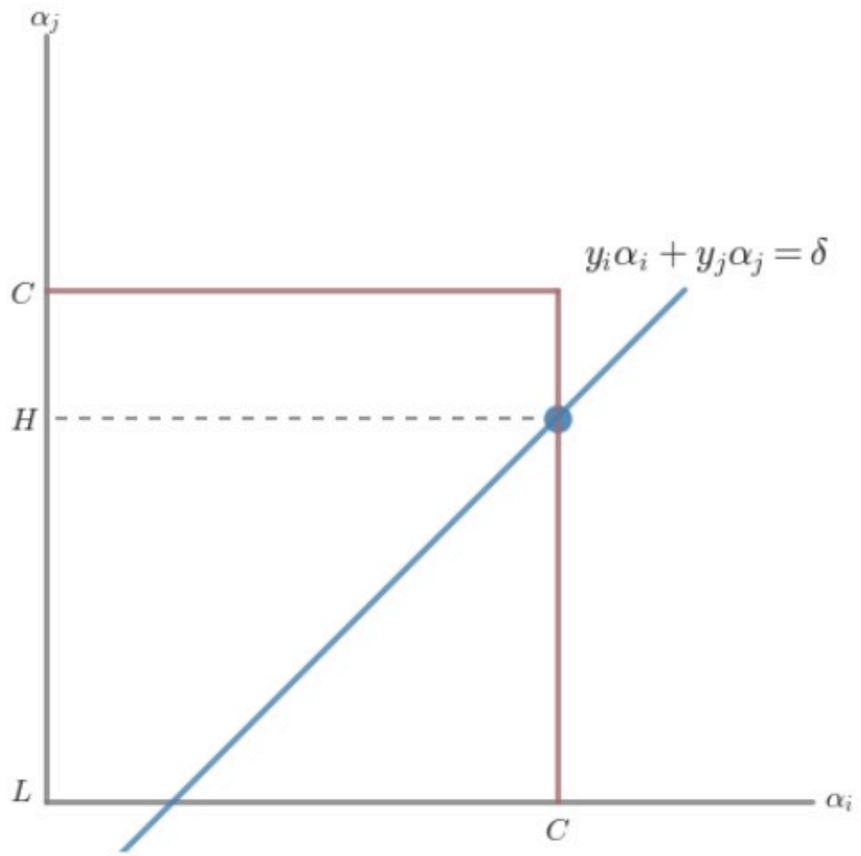
# The SMO Algorithm

What if updated value puts us outside the box?



# The SMO Algorithm

Clip It So It Fits!



## The SMO Algorithm

Values of $L$ and $H$ depend on $y_i$ and $y_j$ (b/c $\alpha_j = y_j\delta - s\alpha_i$)

If $y_i \neq y_j$: $\quad L = \max(0, \alpha_j - \alpha_i), \quad H = \min(C, C + \alpha_j - \alpha_i)$

If $y_i = y_j$: $\quad L = \max(0, \alpha_i + \alpha_j - C), \quad H = \min(C, \alpha_i + \alpha_j)$

$$\alpha_j = \begin{cases} H & \text{if } \alpha_j > H \\ \alpha_j & \text{if } L \leq \alpha_j \leq H \\ L & \text{if } \alpha_j < L \end{cases}$$

Then set $\quad \alpha_i = \alpha_i^{\text{old}} + s(\alpha_j^{\text{old}} - \alpha_j)$

## The SMO Algorithm

Values of $L$ and $H$ depend on $y_i$ and $y_j$ (b/c $\alpha_j = y_j\delta - s\alpha_i$)

If $y_i \neq y_j$: $\quad L = \max(0, \alpha_j - \alpha_i), \quad H = \min(C, C + \alpha_j - \alpha_i)$

If $y_i = y_j$: $\quad L = \max(0, \alpha_i + \alpha_j - C), \quad H = \min(C, \alpha_i + \alpha_j)$

$$\alpha_j = \begin{cases} H & \text{if } \alpha_j > H \\ \alpha_j & \text{if } L \leq \alpha_j \leq H \\ L & \text{if } \alpha_j < L \end{cases}$$

Then set $\quad \alpha_i = \alpha_i^{\text{old}} + s(\alpha_j^{\text{old}} - \alpha_j)$

**Note**: Similar update for bias $b$. See paper for details.

# The SMO Algorithm

**How do we start?**

Probably could guess ... Initialize $\alpha$'s and $b$ to zero

**When do we stop?**

When the KKT conditions are satisfied for each $i = 1, \ldots, m$

$$\begin{aligned} \alpha_i = 0 \quad &\Rightarrow \quad y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1 \\ \alpha_i = C \quad &\Rightarrow \quad y_i(\mathbf{w}^T \mathbf{x}_i + b) \leq 1 \\ 0 < \alpha_i < C \quad &\Rightarrow \quad y_i(\mathbf{w}^T \mathbf{x}_i + b) = 1 \end{aligned}$$

If $\alpha_i$ satisfies KKT skip $\alpha_i$ in the update

If you do a full pass without updating any $\alpha$'s, quit cuz you're done

# Coming Up

- The Kernel Trick for nonlinear classification

# In Class

# In Class

# In Class

# In Class

# In Class

# In Class

# In Class

# In Class