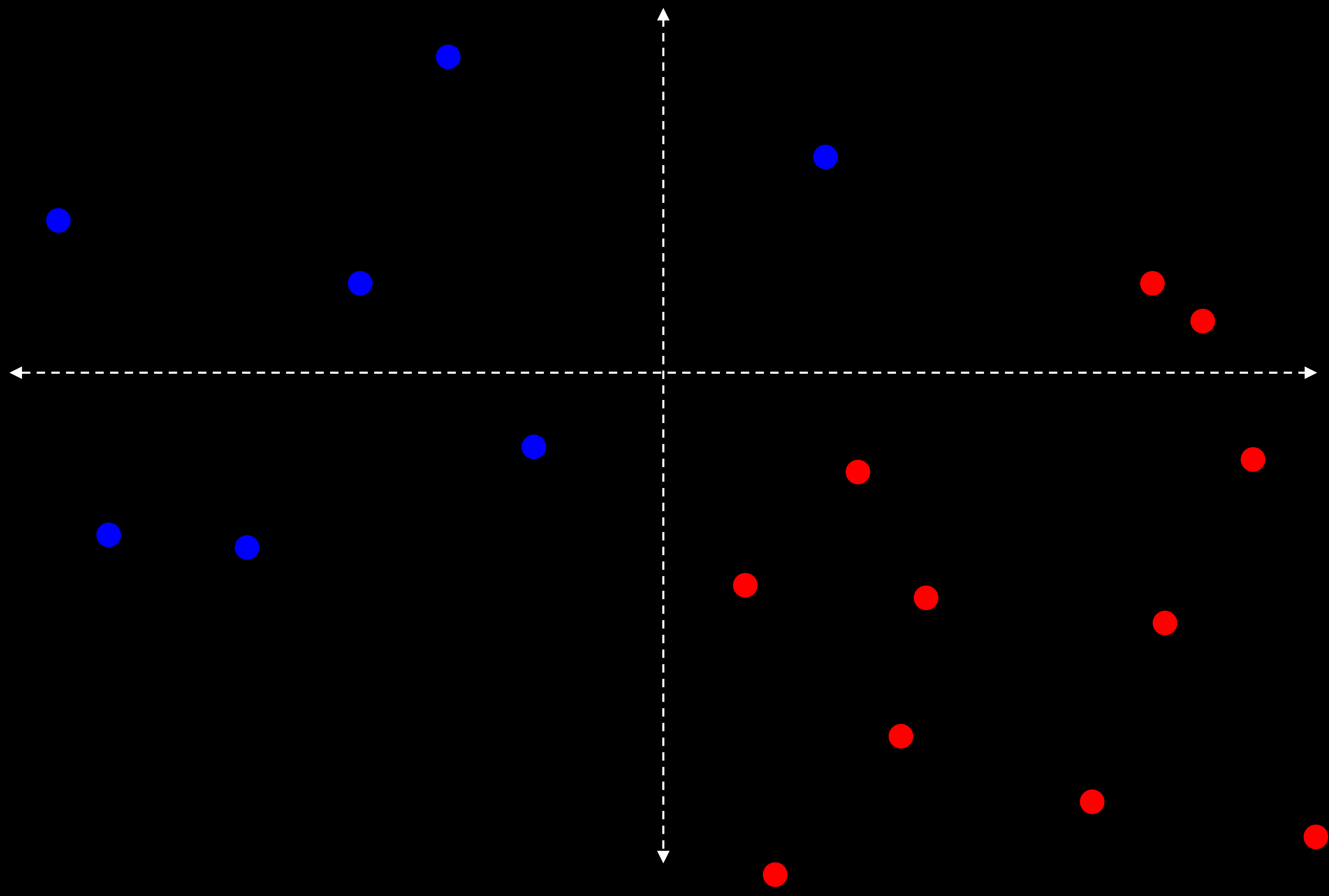
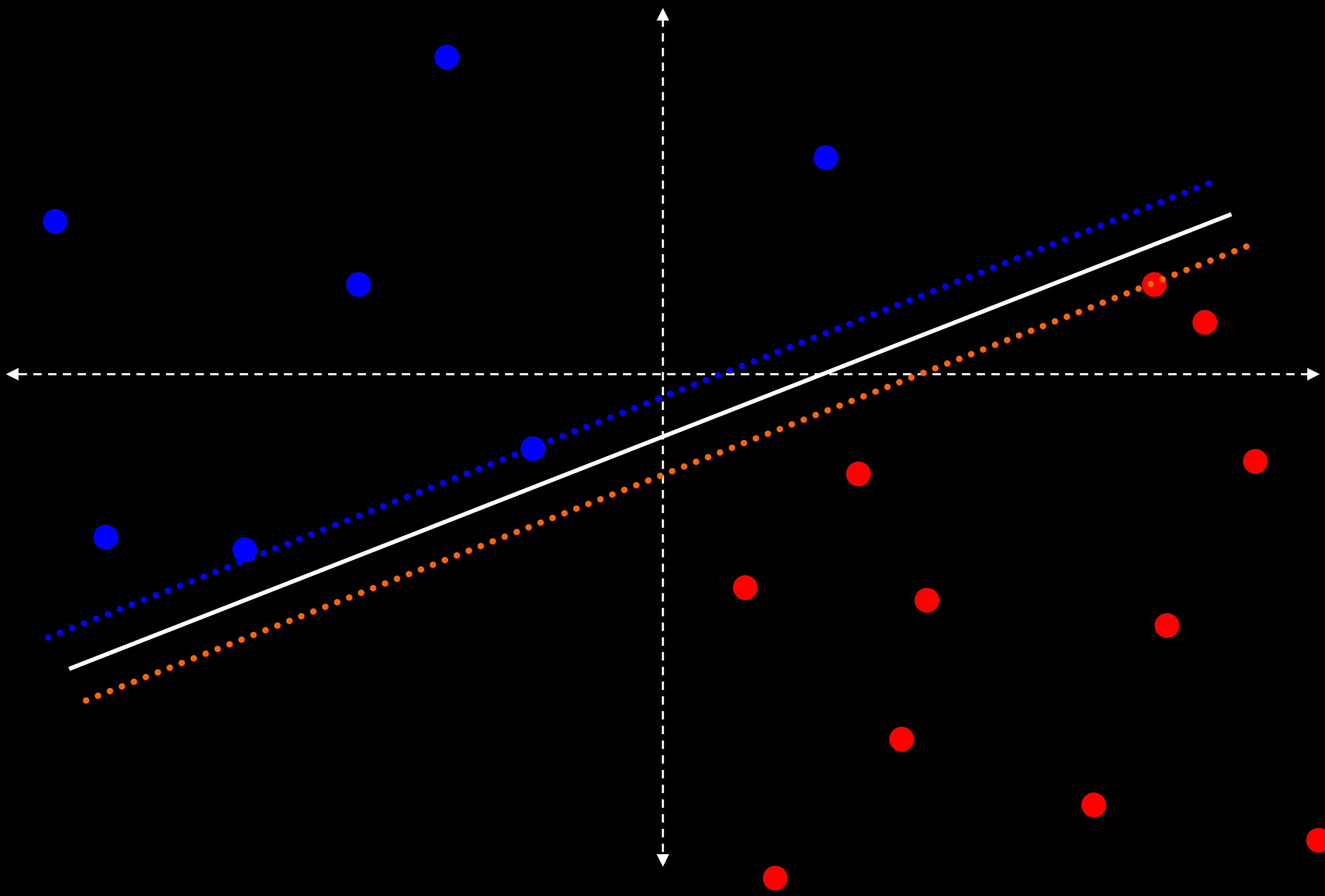
Support Vector Machines

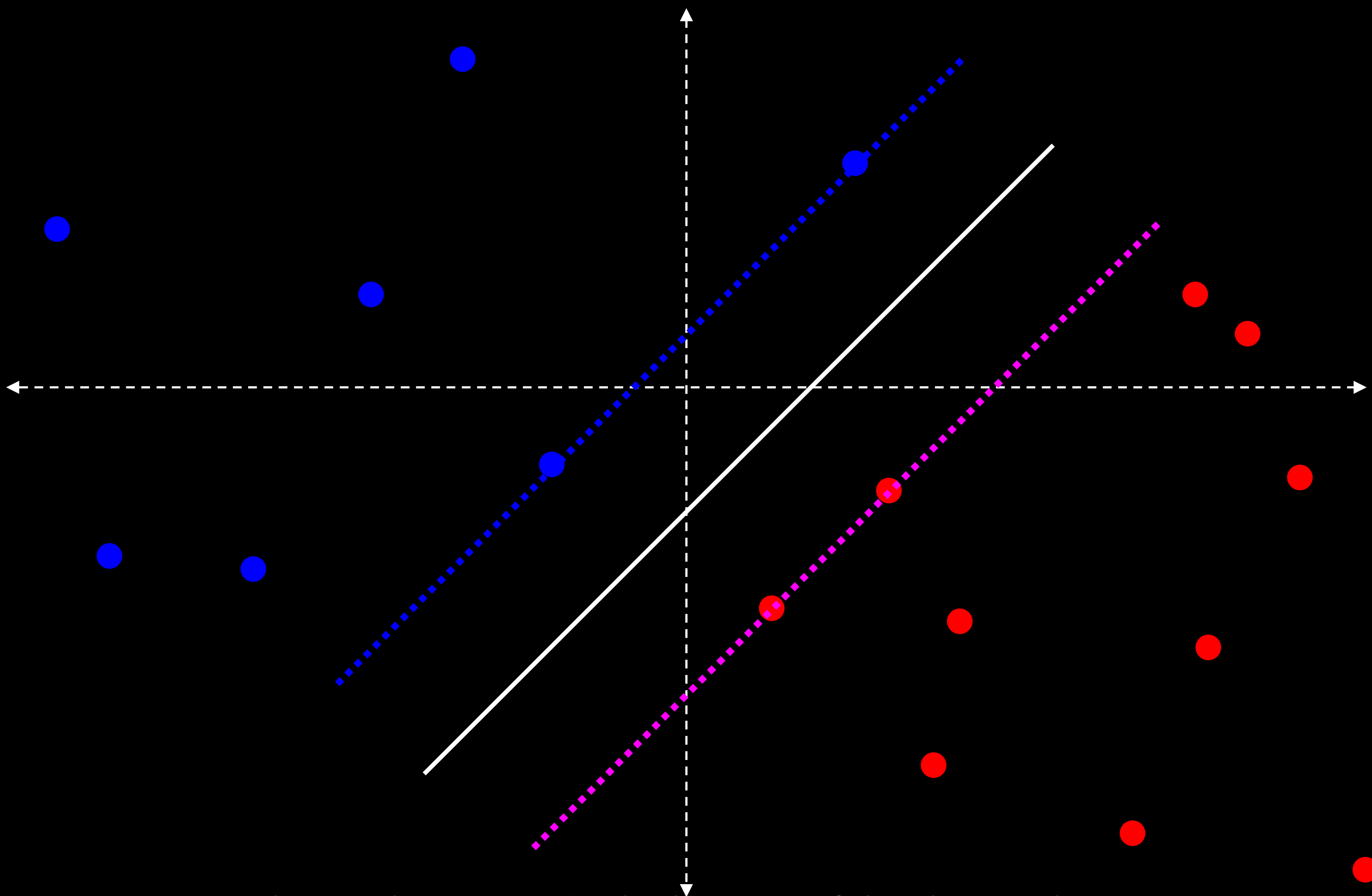
Binary Classification



A Separating Hyperplane

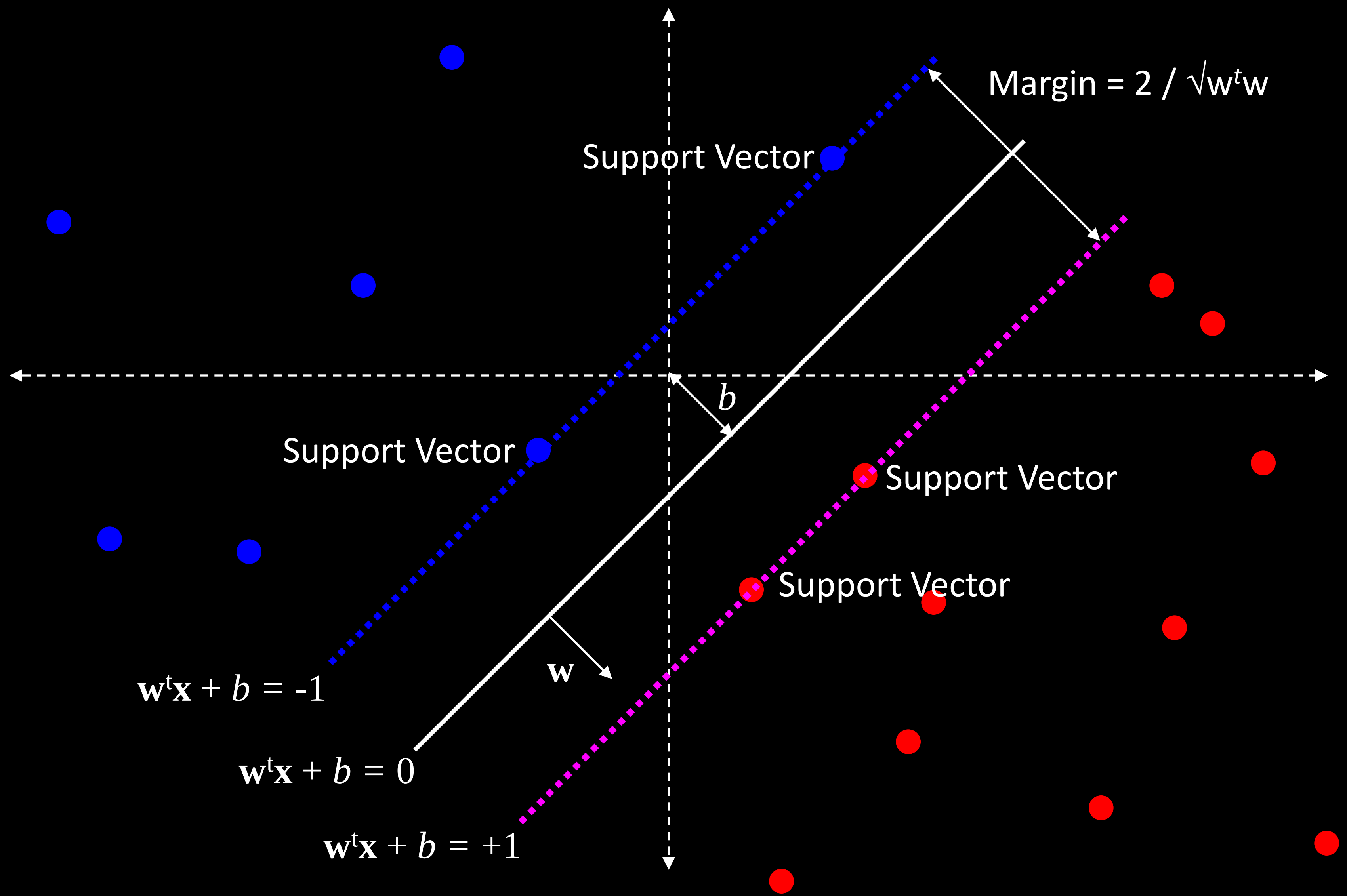


Maximum Margin Hyperplane



Geometric Intuition: Choose the perpendicular bisector of the shortest line segment joining the convex hulls of the two classes

SVM Notation



Hard Margin SVM Primal

- Maximize $2/|\mathbf{w}|$
  such that $\mathbf{w}^t\mathbf{x}_i + b \geq +1$ if $y_i = +1$
  $\mathbf{w}^t\mathbf{x}_i + b \leq -1$ if $y_i = -1$

- Difficult to optimize directly

- Convex Quadratic Program (QP) reformulation
- Minimize $\frac{1}{2}\mathbf{w}^t\mathbf{w}$
  such that $y_i(\mathbf{w}^t\mathbf{x}_i + b) \geq 1$

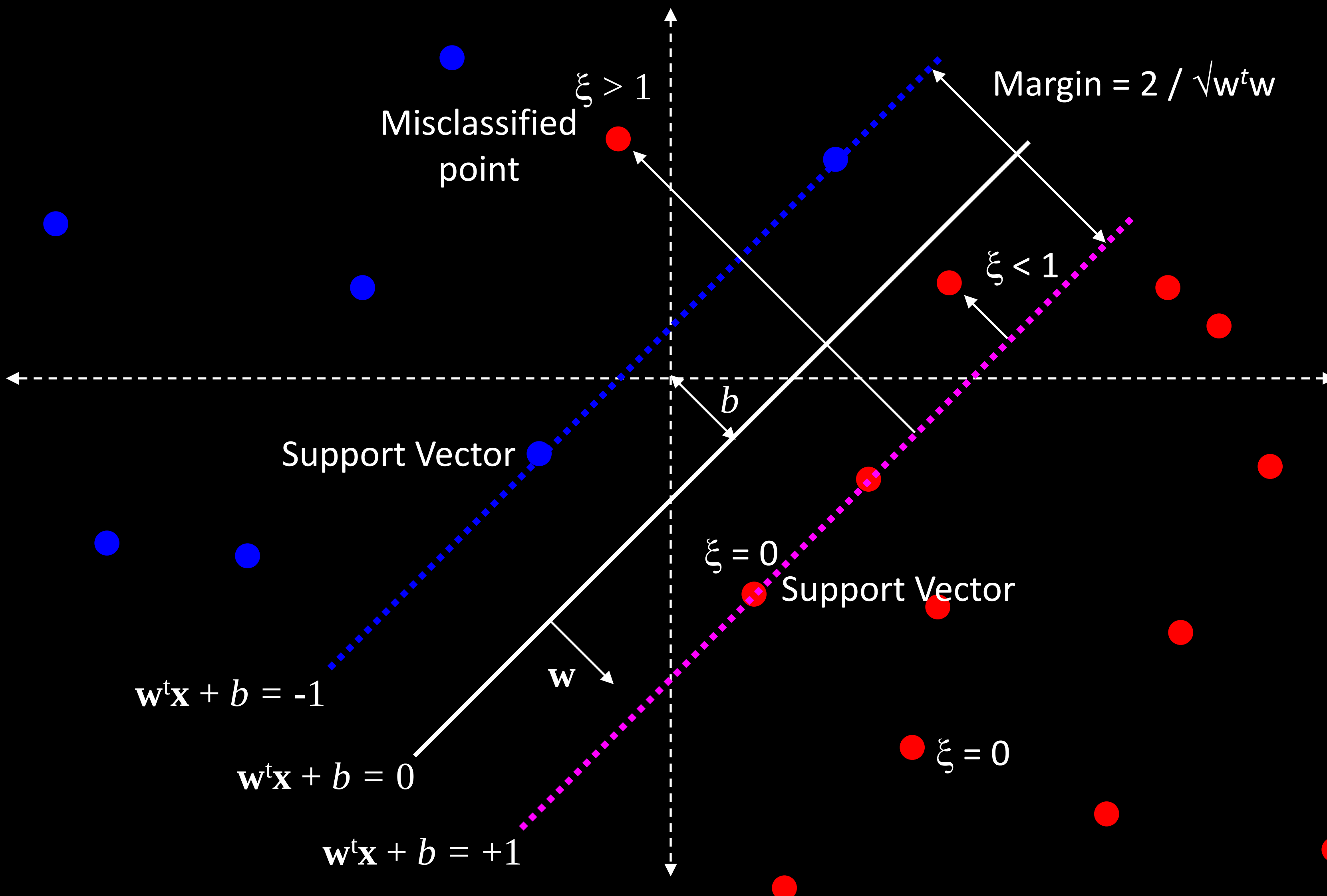
- Convex QPs can be easy to optimize

Linearly Inseparable Data

- Minimize $\frac{1}{2}\mathbf{w}^t\mathbf{w} + C\,\#(\text{Misclassified points})$
  such that $y_i(\mathbf{w}^t\mathbf{x}_i + b) \geq 1$ (for “good” points)

- The optimization problem is NP Hard in general
- Disastrous errors are penalized the same as near misses

Inseparable Data – Hinge Loss



The C-SVM Primal Formulation

- Minimize $\frac{1}{2}\mathbf{w}^t\mathbf{w} + C\sum_i \xi_i$

  such that $y_i(\mathbf{w}^t\mathbf{x}_i + b) \geq 1 - \xi_i$

  $\xi_i \geq 0$

- The optimization is a convex QP
- The globally optimal solution will be obtained
- Number of variables = $D + N + 1$
- Number of constraints = $2N$
- Solvers can train on 800K points in 47K (sparse) dimensions in less than 2 minutes on a standard PC

Fan *et al*., "LIBLINEAR" JMLR 08
Bordes *et al*., "LaRank" ICML 07

The *C*-SVM Dual Formulation

- Maximize $\quad \mathbf{1}^t\alpha - \frac{1}{2}\alpha^t\mathbf{YKY}\alpha$

  such that $\quad \mathbf{1}^t\mathbf{Y}\alpha = 0$

  $\quad \mathbf{0} \leq \alpha \leq \mathbf{C}$

- **K** is a kernel matrix such that $\mathbf{K}_{ij} = K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^t\mathbf{x}_j$
- α are the dual variables (Lagrange multipliers)
- Knowing α gives us **w** and *b*
- The dual is also a convex QP
  - Number of variables = N
  - Number of constraints = $2N + 1$

Fan *et al*., "LIBSVM" JMLR 05
Joachims, "SVMLight"

Duality

- Primal P = $\text{Min}_{\mathbf{x}}$ $f_0(\mathbf{x})$
  s. t. $f_i(\mathbf{x}) \leq 0 \quad 1 \leq i \leq N$
  $h_i(\mathbf{x}) = 0 \quad 1 \leq i \leq M$

- Lagrangian $L(\mathbf{x},\lambda,\mu) = f_0(\mathbf{x}) + \Sigma_i\, \lambda_i f_i(\mathbf{x}) + \Sigma_i\, \mu_i h_i(\mathbf{x})$

- Dual D = $\text{Max}_{\lambda,\mu}$ $\text{Min}_{\mathbf{x}}\, L(\mathbf{x},\lambda,\mu)$
  s. t. $\boldsymbol{\lambda} \geq \mathbf{0}$

Duality

- The Lagrange dual is always concave (even if the primal is not convex) and might be an easier problem to optimize

- Weak duality : $P \geq D$
  - Always holds

- Strong duality : $P = D$
  - Does not always hold
  - Usually holds for convex problems
  - Holds for the SVM QP

Karush-Kuhn-Tucker (KKT) Conditions

- If strong duality holds, then for $\mathbf{x}^*$, λ^* and μ^* to be optimal the following KKT conditions must necessarily hold

- Primal feasibility : $f_i(\mathbf{x}^*) \leq 0$ & $h_i(\mathbf{x}^*) = 0$ for $1 \leq i$
- Dual feasibility : $\lambda^* \geq 0$
- Stationarity : $\nabla_{\mathbf{x}} L(\mathbf{x}^*, \lambda^*, \mu^*) = 0$
- Complimentary slackness : $\lambda_i^* f_i(\mathbf{x}^*) = 0$

- If $\mathbf{x}^+$, λ^+ and μ^+ satisfy the KKT conditions for a convex problem then they are optimal

SVM – Duality

- Primal P $= \text{Min}_{\mathbf{w},\xi,b}$ $\quad \frac{1}{2}\mathbf{w}^t\mathbf{w} + \mathbf{C}^t\xi$

  s. t. $\quad \mathbf{Y}(\mathbf{X}^t\mathbf{w} + b\mathbf{1}) \geq \mathbf{1} - \xi$

  $\xi \geq \mathbf{0}$

- Lagrangian $L(\alpha,\beta, \mathbf{w},\xi,b) \quad = \frac{1}{2}\mathbf{w}^t\mathbf{w} + \mathbf{C}^t\xi \; - \beta^t\xi$

  $-\alpha^t[\mathbf{Y}(\mathbf{X}^t\mathbf{w} + b\mathbf{1}) - \mathbf{1} + \xi]$

- Dual D $= \text{Max}_{\alpha}$ $\quad \mathbf{1}^t\alpha - \frac{1}{2}\alpha^t\mathbf{YKY}\alpha$

  s. t. $\quad \mathbf{1}^t\mathbf{Y}\alpha = 0$

  $\mathbf{0} \leq \alpha \leq \mathbf{C}$

SVM – KKT Conditions

- Lagrangian $L(\alpha,\beta,\mathbf{w},\xi,b) = \frac{1}{2}\mathbf{w}^t\mathbf{w} + \mathbf{C}^t\xi - \beta^t\xi$
  $-\alpha^t[\mathbf{Y}(\mathbf{X}^t\mathbf{w} + b\mathbf{1}) - \mathbf{1} + \xi]$

- Stationarity conditions
  - $\nabla_{\mathbf{w}} L = 0 \Rightarrow \mathbf{w}^* = \mathbf{XY}\alpha^*$ (Representer Theorem)
  - $\nabla_{\xi} L = 0 \Rightarrow \mathbf{C} = \alpha^* + \beta^*$
  - $\nabla_b L = 0 \Rightarrow \alpha^{*t}\mathrm{Y}\mathbf{1} = \mathbf{0}$

- Complimentary Slackness conditions
  - $\alpha_i^* \, [\, y_i(\mathbf{x}_i^t\mathbf{w}^* + b^*) - \mathbf{1} + \xi_i^*] = 0$
  - $\beta_i^* \xi_i^* = 0$

Support Vector Classification

- Training data $(\mathbf{x}_i, y_i), i = 1, \ldots, l, \mathbf{x}_i \in R^n, y_i = \pm 1$

$$\min_{\mathbf{w}} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{l} \max(0, 1 - y_i\mathbf{w}^T\phi(\mathbf{x}_i))$$

- C: regularization parameter
- High dimensional (maybe infinite) feature space

$$\phi(\mathbf{x}) = [\phi_1(\mathbf{x}), \phi_2(\mathbf{x}), \ldots]^T.$$

- We omit the bias term b
- $\mathbf{w}$: may have infinite variables

Support Vector Classification (Cont'd)

- The dual problem (finite # variables)

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \quad f(\boldsymbol{\alpha}) = \ & \frac{1}{2}\boldsymbol{\alpha}^T Q \boldsymbol{\alpha} - \mathbf{e}^T \boldsymbol{\alpha} \\ \text{subject to} \quad & 0 \leq \alpha_i \leq C, i = 1, \ldots, l, \end{aligned}$$

  where $Q_{ij} = y_i y_j \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$ and $\mathbf{e} = [1, \ldots, 1]^T$
- At optimum

$$\boxed{\mathbf{w} = \sum_{i=1}^{l} \alpha_i y_i \phi(\mathbf{x}_i)}$$

- Kernel: $\boxed{K(\mathbf{x}_i, \mathbf{x}_j) \equiv \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)}$

Large Dense Quadratic Programming

- $Q_{ij} \neq 0$, Q : an l by l fully dense matrix

$$\begin{array}{ll} \min_{\boldsymbol{\alpha}} & f(\boldsymbol{\alpha}) = \frac{1}{2}\boldsymbol{\alpha}^T Q\boldsymbol{\alpha} - \mathbf{e}^T\boldsymbol{\alpha} \\ \text{subject to} & 0 \leq \alpha_i \leq C, i = 1, \ldots, l \end{array}$$

- 50,000 training points: 50,000 variables:
  $(50,000^2 \times 8/2)$ bytes $=$ 10GB RAM to store Q
- Traditional methods:
  Newton, Quasi Newton cannot be directly applied
- Now most use decomposition methods
  [Osuna et al., 1997, Joachims, 1998, Platt, 1998]

Decomposition Methods

- We consider a one-variable version
  Similar to coordinate descent methods
- Select the ith component for update:

$$\begin{aligned} \min_{d} \quad & \frac{1}{2}(\boldsymbol{\alpha} + d\mathbf{e}_i)^T Q(\boldsymbol{\alpha} + d\mathbf{e}_i) - \mathbf{e}^T(\boldsymbol{\alpha} + d\mathbf{e}_i) \\ \text{subject to} \quad & 0 \leq \alpha_i + d \leq C \end{aligned}$$

  where

$$\mathbf{e}_i \equiv [\underbrace{0 \ldots 0}_{i-1} \; 1 \; 0 \ldots 0]^T$$

- $\boldsymbol{\alpha}$: current solution; the ith component is changed

Avoid Memory Problems

- The new objective function

$$\frac{1}{2}Q_{ii}d^2 + (Q\boldsymbol{\alpha} - \mathbf{e})_i d + \text{ constant}$$

- To get $(Q\boldsymbol{\alpha} - \mathbf{e})_i$, only Q's ith row is needed

$$(Q\boldsymbol{\alpha} - \mathbf{e})_i = \sum_{j=1}^{l} Q_{ij}\alpha_j - 1$$

- Calculated when needed. Trade time for space
- Used by popular software (e.g., *SVM*^{light}, LIBSVM)
  They update 10 and 2 variables at a time

Decomposition Methods: Algorithm

- Optimal d:

$$-\frac{(Q\boldsymbol{\alpha} - \mathbf{e})_i}{Q_{ii}} = -\frac{\sum_{j=1}^{l} Q_{ij}\alpha_j - 1}{Q_{ii}}$$

- Consider lower/upper bounds: $[0, C]$
- Algorithm:

  While $\boldsymbol{\alpha}$ is not optimal

  1. Select the ith element for update
  2. $\alpha_i \leftarrow \min\left(\max\left(\alpha_i - \frac{\sum_{j=1}^{l} Q_{ij}\alpha_j - 1}{Q_{ii}}, 0\right), C\right)$

Select an Element for Update

Many ways

- Sequential (easiest)
- Permuting $1, \ldots, l$ every l steps
- Random
- Existing software check gradient information

$$\nabla_1 f(\boldsymbol{\alpha}), \ldots, \nabla_l f(\boldsymbol{\alpha})$$

But is $\nabla f(\boldsymbol{\alpha})$ available?

Select an Element for Update (Cont'd)

- We can easily maintain gradient

$$\nabla f(\boldsymbol{\alpha}) = Q\boldsymbol{\alpha} - \mathbf{e}$$

$$\nabla_s f(\boldsymbol{\alpha}) = (Q\boldsymbol{\alpha})_s - 1 = \sum_{j=1}^{l} Q_{sj}\alpha_j - 1$$

- Initial $\boldsymbol{\alpha} = \mathbf{0}$

$$\nabla f(\mathbf{0}) = -\mathbf{e}$$

- α_i updated to $\bar{\alpha}_i$

$$\nabla_s f(\boldsymbol{\alpha}) \leftarrow \nabla_s f(\boldsymbol{\alpha}) + \boxed{Q_{si}(\bar{\alpha}_i - \alpha_i)}, \quad \forall s$$

- $O(l)$ if Q_{si} $\forall s$ (ith column) are available

Select an Element for Update (Cont'd)

- No matter maintaining $\nabla f(\boldsymbol{\alpha})$ or not

  Q's ith row (column) always needed

$$\bar{\alpha}_i \leftarrow \min\left(\max\left(\alpha_i - \frac{\sum_{j=1}^{l} Q_{ij}\alpha_j - 1}{Q_{ii}}, 0\right), C\right)$$

  Q is symmetric

- Using $\nabla f(\boldsymbol{\alpha})$ to select i: faster convergence

  i.e., fewer iterations

Decomposition Methods: Using Gradient

The new procedure

- $\boldsymbol{\alpha} = \mathbf{0}, \nabla f(\boldsymbol{\alpha}) = -\mathbf{e}$
- While $\boldsymbol{\alpha}$ is not optimal
  1. Select the ith element using $\nabla f(\boldsymbol{\alpha})$
  2. $\bar{\alpha}_i \leftarrow \min\left(\max\left(\alpha_i - \frac{\sum_{j=1}^{l} Q_{ij}\alpha_j - 1}{Q_{ii}}, 0\right), C\right)$
  3. $\nabla_s f(\boldsymbol{\alpha}) \leftarrow \nabla_s f(\boldsymbol{\alpha}) + Q_{si}(\bar{\alpha}_i - \alpha_i), \forall s$

Cost per iteration

- $O(ln)$, l: # instances, n: # features
- Assume each $Q_{ij} = y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)$ takes $O(n)$

Linear SVM

- Primal without the bias term b

$$\min_{\mathbf{w}} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{l} \max\left(0, 1 - y_i\mathbf{w}^T\mathbf{x}_i\right)$$

- Dual

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \quad & f(\boldsymbol{\alpha}) = \frac{1}{2}\boldsymbol{\alpha}^T Q\boldsymbol{\alpha} - \mathbf{e}^T\boldsymbol{\alpha} \\ \text{subject to} \quad & 0 \le \alpha_i \le C, \forall i \end{aligned}$$

- $Q_{ij} = y_i y_j \mathbf{x}_i^T \mathbf{x}_j$

Revisit Decomposition Methods

- While $\boldsymbol{\alpha}$ is not optimal
  1. Select the ith element for update
  2. $\alpha_i \leftarrow \min\left(\max\left(\alpha_i - \frac{\sum_{j=1}^{l} Q_{ij}\alpha_j - 1}{Q_{ii}}, 0\right), C\right)$
- $O(ln)$ per iteration; n: # features, l: # data
- For linear SVM, define

$$\mathbf{w} \equiv \sum_{j=1}^{l} y_j \alpha_j \mathbf{x}_j \in R^n$$

- $O(n)$ per iteration

$$\sum_{j=1}^{l} Q_{ij}\alpha_j - 1 = \sum_{j=1}^{l} y_i y_j \mathbf{x}_i^T \mathbf{x}_j \alpha_j - 1 = y_i \mathbf{w}^T \mathbf{x}_i - 1$$

- All we need is to maintain $\mathbf{w}$. If

$$\bar{\alpha}_i \leftarrow \alpha_i$$

then $O(n)$ for

$$\mathbf{w} \leftarrow \mathbf{w} + (\bar{\alpha}_i - \alpha_i) y_i \mathbf{x}_i$$

- Initial $\mathbf{w}$

$$\boldsymbol{\alpha} = \mathbf{0} \quad \Rightarrow \quad \mathbf{w} = \mathbf{0}$$

- Give up maintaining $\nabla f(\boldsymbol{\alpha})$
- Select i for update

  Sequential, random, or

  Permuting $1, \ldots, l$ every l steps

Algorithms for Linear and Nonlinear SVM

Linear:

- While $\boldsymbol{\alpha}$ is not optimal
  1. Select the ith element for update
  2. $\bar{\alpha}_i \leftarrow \min\left(\max\left(\alpha_i - \frac{y_i \mathbf{w}^T \mathbf{x}_i - 1}{Q_{ii}}, 0\right), C\right)$
  3. $\mathbf{w} \leftarrow \mathbf{w} + (\bar{\alpha}_i - \alpha_i) y_i \mathbf{x}_i$

Nonlinear:

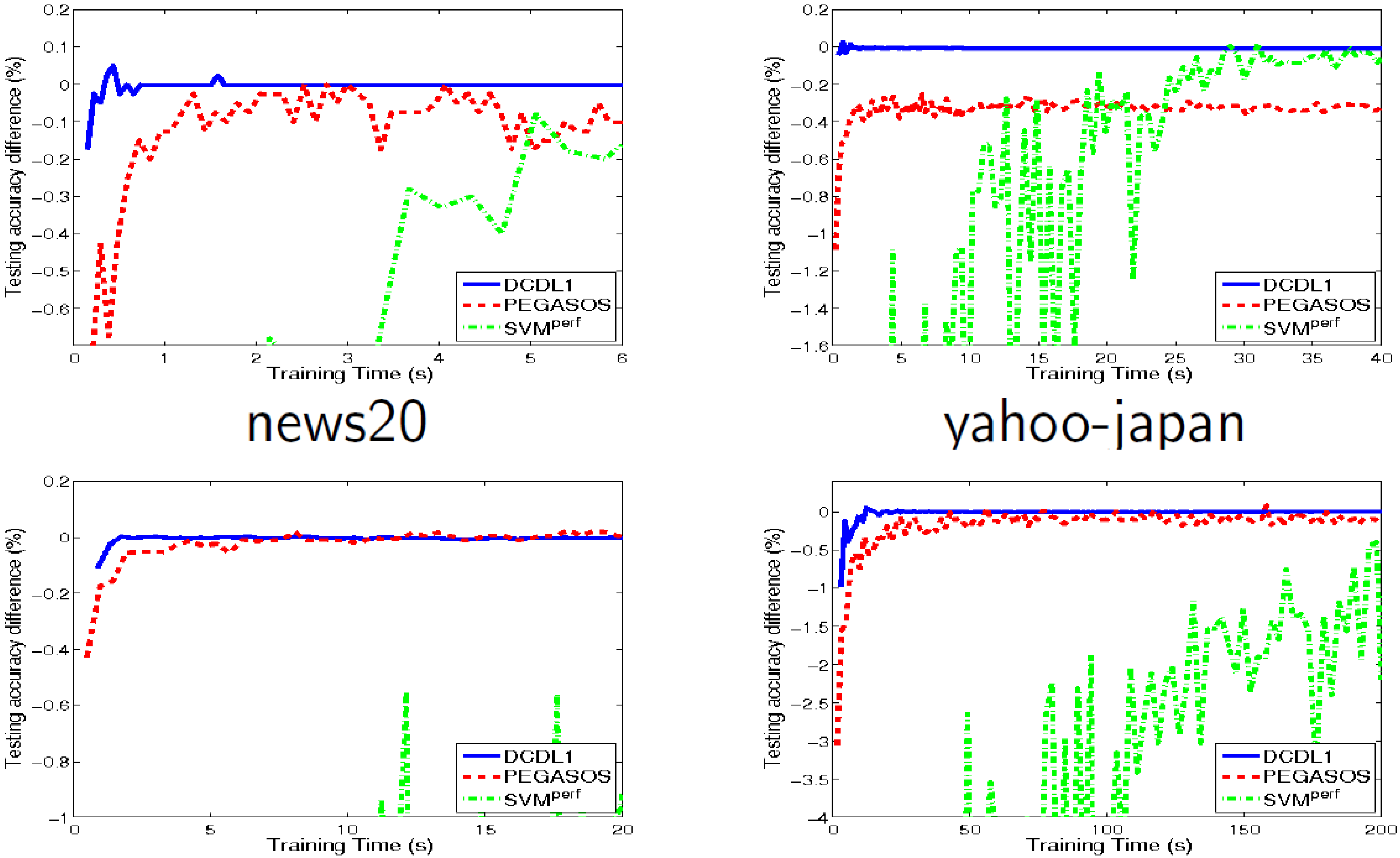
- While $\boldsymbol{\alpha}$ is not optimal
  1. Select the ith element using $\nabla f(\boldsymbol{\alpha})$
  2. $\bar{\alpha}_i \leftarrow \min\left(\max\left(\alpha_i - \frac{\sum_{j=1}^{l} Q_{ij}\alpha_j - 1}{Q_{ii}}, 0\right), C\right)$
  3. $\nabla_s f(\boldsymbol{\alpha}) \leftarrow \nabla_s f(\boldsymbol{\alpha}) + Q_{si}(\bar{\alpha}_i - \alpha_i), \forall s$

Analysis

- Decomposition method for nonlinear (also linear):
  $O(ln)$ per iteration (used in LIBSVM)
- New way for linear:
  $O(n)$ per iteration (used in LIBLINEAR)
- Faster if # iterations not l times more
- Experiments

Problem	l: # data	n: # features
news20	19,996	1,355,191
yahoo-japan	176,203	832,026
rcv1	677,399	47,236
yahoo-korea	460,554	3,052,939

Testing Accuracy versus Training Time



news20 yahoo-japan

rcv1 yahoo-korea