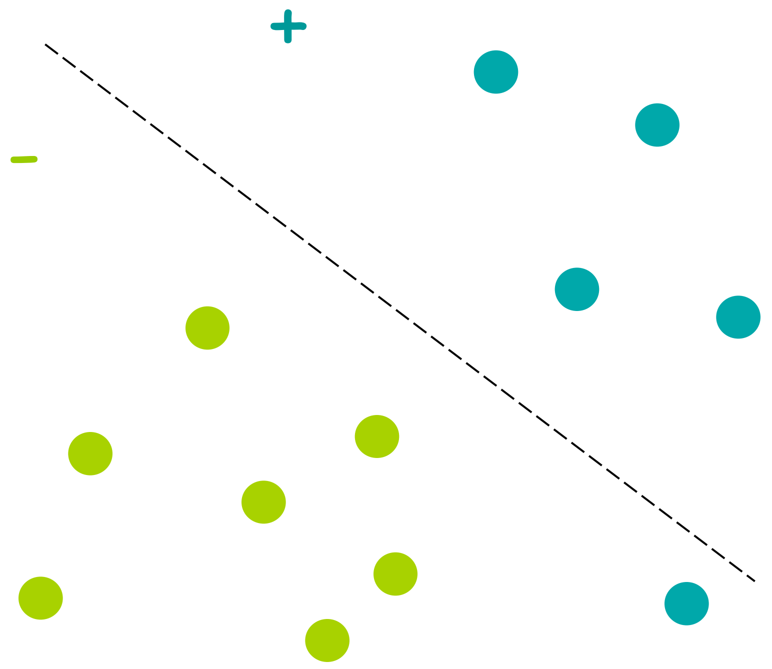
Optimization Methods in Machine Learning
Lecture 15

Optimization Methods for SVMs

- Stochastic gradient method
- Block-coordinate descent
- Active set method

Support Vector Machines



Classification SVM Problem

Given a training set of $(x_1, y_1), \ldots, (x_n, y_n)$,
$x_i \in \mathbf{R}^d$, $y \in \{+1, -1\}$

$$\begin{array}{ll} \min_{\xi, w} & \frac{1}{2} w^\top w + c \sum_{i=1}^{n} \xi_i \\ \text{s.t.} & y_i(w^\top x_i) \geq 1 - \xi_i, \quad i = 1, \ldots, n \\ & \xi_i \geq 0, \qquad i = 1, \ldots, n. \end{array}$$

Classification SVM Problem

Given a training set of $(x_1, y_1), \ldots, (x_n, y_n)$,
$x_i \in \mathbf{R}^d$, $y \in \{+1, -1\}$

$$\begin{aligned} \min_{\xi, w} \quad & \frac{1}{2} w^\top w + c \sum_{i=1}^{n} \xi_i \\ \text{s.t.} \quad & y_i(w^\top x_i) \geq 1 - \xi_i, \quad i = 1, \ldots, n \\ & \xi_i \geq 0, \qquad i = 1, \ldots, n. \end{aligned}$$

What happened to β?

$$w^\top x + \beta = (w, \beta)^\top (x, 1)$$

Stochastic gradient approach

Unconstrained formulation of the SVM problem

Given a training set $S = \{(x_1, y_1), \ldots, (x_n, y_n)\}$,
$x_i \in \mathbf{R}^d$, $y \in \{+1, -1\}$

$$\min_w f(w) = \frac{\lambda}{2}\|w\|^2 + \frac{1}{n}\sum_{i=1}^{n} \ell(w, (x, y))$$

where

$$\ell(w, (x, y)) = \max\{0, 1 - y_i(w^\top x_i)\}$$

Find $f(w) \leq f(w^*) + \epsilon$ - ϵ-optimal solution.

Subgradient step

Consider the training set S
and for a given w define $S^+ = \{(x,y) \in S\, y(w^\top x) < 1\}$

$$f(w) = \frac{\lambda}{2}\|w\|^2 + \frac{1}{|S|}\sum_{(x,y)\in S} \ell(w,(x,y))$$

an "app,subgradient of $f(w)$:

$$\partial_w f(w) = \lambda w_t - \frac{1}{|S|}\sum_{(x,y)\in S^+} yx$$

Compute a subgradient step of lenth η_t.

$$w_{t+\frac{1}{2}} = w_t - \eta_t \partial_w f(w)$$

It can be shown that at optimality $\|w\| \leq 1/\sqrt{\lambda}$,
hence we can project $w_{t+\frac{1}{2}}$ onto the ball to obtain w_{t+1}.

SVM problem using Huber loss function

Given a training set $S = \{(x_1, y_1), \ldots, (x_n, y_n)\}$,
$x_i \in \mathbf{R}^d$, $y \in \{+1, -1\}$

$$\min_w f(w) = \frac{\lambda}{2}\|w\|^2 + \frac{1}{n}\sum_{i=1}^{n} \phi_\mu(w, (x_i, y_i))$$

where

$$\phi_\mu(w, (x, y)) = \begin{cases} 0 & y(w^\top x) \geq 1 \\ \frac{(y(w^\top x) - 1)^2}{2\mu} & 1 - \mu < y_i(w^\top x) < 1 \\ 1 - y(w^\top x) - \frac{\mu}{2} & y(w^\top x) \leq 1 - \mu \end{cases}$$

Find $f(w) \leq f(w^*) + \epsilon$ - ϵ-optimal solution in $O(\frac{1}{\epsilon})$ iterations

Approximate subgradient step

Consider a subset of the training set $A_t \subseteq S$
and for a given w define $A_t^+ = \{(x, y) \in A^t : y(w^\top x) < 1\}$

$$f_t(w) = \frac{\lambda}{2}\|w\|^2 + \frac{1}{|A_t|}\sum_{(x,y)\in A_t} \ell(w, (x, y))$$

an "approximate" subgradient of $f(w)$:

$$\partial_w f_t(w) = \lambda w_t - \frac{1}{|A_t|}\sum_{(x,y)\in A_t^+} yx$$

Compute a subgradient step of lenth η_t.

$$w_{t+\frac{1}{2}} = w_t - \eta_t \partial_w f(w, A_t)$$

It can be shown that at optimality $\|w\| \leq 1/\sqrt{\lambda}$,
hence we can project $w_{t+\frac{1}{2}}$ onto the ball to obtain w_{t+1}.

Stochastic Gradient Method

Choose w_1, such that $\|w_1\| \leq \frac{1}{\sqrt{\lambda}}$.

For $t = 1, 2, \ldots, T$

- Choose $A_t \subset S$, where $|A_t| = k$.
- Set $A_t^+ = \{(x, y) \in A_t : y(w^\top x) < 1\}$.
- $\eta_t = \frac{1}{\lambda t}$
- $w_{t+\frac{1}{2}} = (1 - \eta_t \lambda) w_t + \frac{\eta_t}{k} \sum_{(x,y)\in A_t^+} yx$
- $w_{t+1} = \min\{1, \frac{1/\sqrt{\lambda}}{\|w_{t+\frac{1}{2}}\|}\} w_{t+\frac{1}{2}}$

Convergence in expectation

Find $E(f(\bar{w})) \leq f(w^*) + \epsilon$

ϵ-optimal solution in expectation,

where $\bar{w} = \frac{1}{t} \sum_{i=1}^{t} w_i$.

Why does this work?

- Each iteration of the algorithm takes $O(n_t s)$ operations, where s is the number of nonzeros attributes of each data point x_i and n_t is the size of A_t. There are no subproblems to solve.

- When $A_t = S$ and hence $n_t = n$, the algorithm takes at most $\tilde{O}(\frac{R^2}{\lambda\epsilon})$, iterations where $R = \max_i \|x_i\|$.

- When $|A_t| < n$, then we need an assumption that elements in A_t a drawn from S as i.i.d. samples.

- With probability $1-\delta$ the algorithm achieves ϵ-optimal solution in at most $\tilde{O}(\frac{R^2}{\delta\lambda\epsilon})$, iterations.

- This means that the probabilistic complexity of this method does not depend on the size of the training set at all!

Stochastic Approximation for Machine Learning

$$\min_{\mathbf{w}} L(\mathbf{w}) = \mathbb{E}[\underbrace{\ell}_{|\ell'|\leq 1}(\langle \mathbf{w}, \underbrace{\mathbf{x}}_{\|x\|_2 \leq X} \rangle, y)]$$

- Our previous approach was a mixed approach:
  - SAA: collect sample of size *m* and minimize empirical error (w/ norm constraint):

$$\min_{\|\mathbf{w}\|_2 \leq B} \hat{L}(\mathbf{w}) = \frac{1}{m}\sum_{i=1}^{m} \ell(\langle \mathbf{w}, \mathbf{x}_i \rangle, y_i)$$

  - Optimize this with SGD, i.e. applying SA to the *empirical objective*
    - At each SGD iteration, pick random (x,y) from empirical sample
  - SGD guarantee is on *empirical* suboptimality:

$$\hat{L}(\overline{\mathbf{w}}^{(k)}) \leq \hat{L}(\hat{\mathbf{w}}) + O\left(\sqrt{\frac{X^2B^2}{k}}\right)$$

  - To get guarantee on L(w$^{(k)}$), need to combined with uniform concentration:

$$\sup_{\|\mathbf{w}\| \leq B} \left|\hat{L}(\mathbf{w}) - L(\mathbf{w})\right| \leq O\left(\sqrt{\frac{X^2B^2}{m}}\right)$$

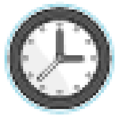
- Pure SA approach:
  - Optimize L(w) directly
  - Same SGD guarantee, but directly to the generalization error:

$$L(\overline{\mathbf{w}}^{(k)}) \leq L(\mathbf{w}^*) + O\left(\sqrt{\frac{X^2 \|\mathbf{w}^*\|_2^2}{k}}\right)$$

More Data ⇒ More Work?

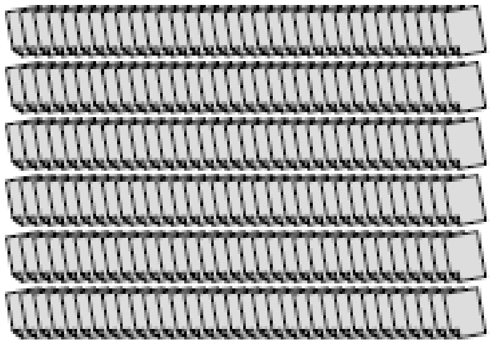
10k training examples

1 hour

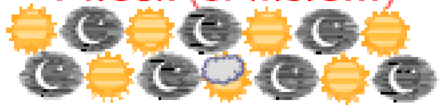


2.3% error *(when using the predictor)*

1M training examples



1 week (or more…)



2.29% error

Can always sample and get same runtime:

1 hour 2.3% error

Can we leverage the excess data to **reduce** runtime?

10 minutes 2.3% error

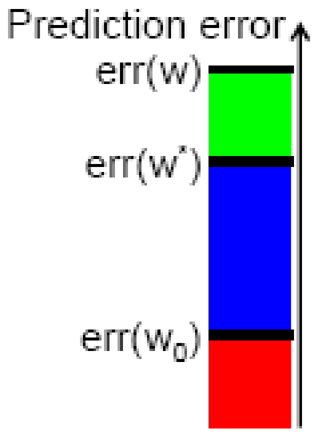
But I really care about that 0.01% gain

Study runtime increase as a function of target accuracy

My problem is so hard, I *have* to crunch 1M examples

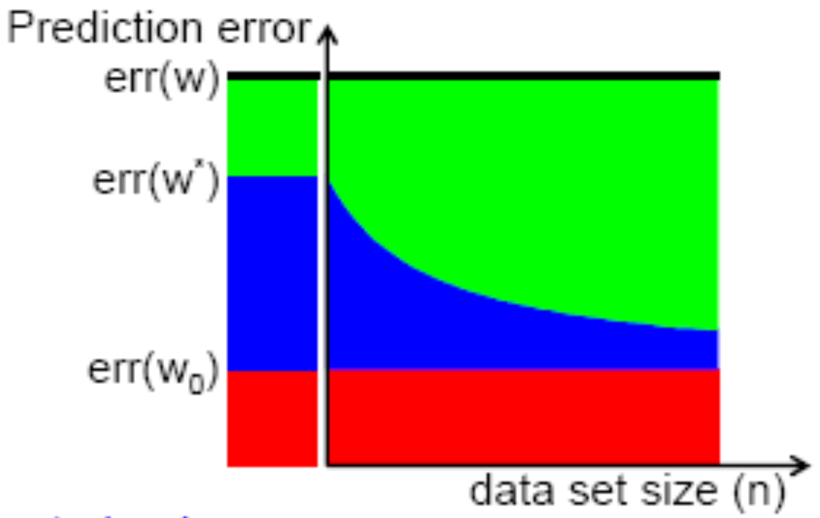
Study runtime increase as a function of problem difficulty (e.g. small margin)

Error Decomposition



- Approximation error:
  - Best error achievable by large-margin predictor
  - Error of population minimizer
    w_0 = argmin $E[f(w)]$ = argmin $\lambda|w|^2 + E_{x,y}[loss(\langle w,x \rangle;y)]$
- Estimation error:
  - Extra error due to replacing E[loss] with empirical loss
    w^* = arg min $f_n(w)$
- Optimization error:
  - Extra error due to only optimizing to within finite precision

The Double-Edged Sword



- When data set size increases:
  - **Estimation error** decreases
  - Can increase **optimization error**,
    i.e. optimize to within lesser accuracy ⇒ fewer iterations
  - But handling more data is expensive
    e.g. runtime of each iteration increases
- Stochastic Gradient Descent,
  e.g. PEGASOS (Primal Efficient Sub-Gradient Solver for SVMs)
  [Shalev-Shwartz Singer Srebro, ICML'07]
  - Fixed runtime per iteration
  - Runtime to get fixed accuracy does not increase with n

Optimization Problem

$$w^* = \sum_{i=1}^{n} \alpha_i y_i x_i, \quad 0 \le \alpha_i \le c$$

$$\begin{array}{lll} \min_{\alpha,\beta,\xi} & \frac{1}{2}\alpha^\top Q\alpha + c\sum_{i=1}^{n}\xi_i & \\ \text{s.t.} & -Q\alpha + y\beta + s_i - \xi_i = -1, & i = 1, \ldots, n \\ & s_i \ge 0, \xi \ge 0, 0 \le \alpha_i \le c, & i = 1, \ldots, n, \end{array}$$

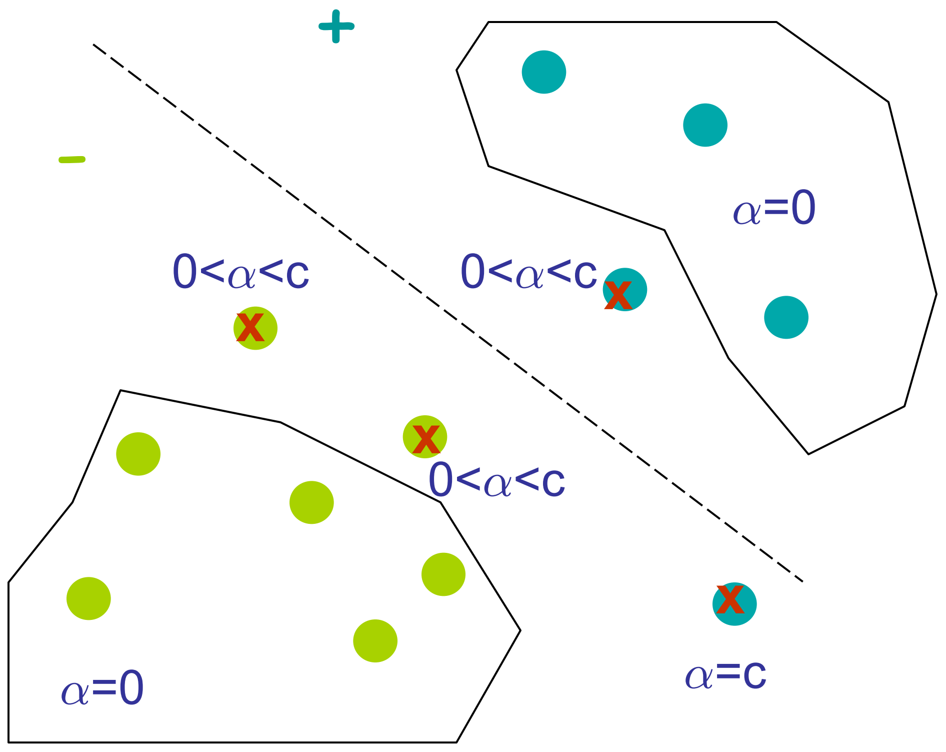
$$Q_{ij} = y_i y_j {x_i}^\top x_j \quad \text{or} \quad Q_{ij} = y_i y_j K(x_i, x_j)$$

Linear formulation

Kernel formulation

$$\begin{array}{ll} \min_{\alpha} & \frac{1}{2}\alpha^\top Q\alpha - e^\top\alpha \\ \text{s.t.} & y^\top\alpha = 0, \\ & 0 \le \alpha \le c, \end{array}$$

Support Vectors



Decomposition Methods

Dual Optimization Problem

$$\begin{aligned} \min_{\alpha} \quad & \frac{1}{2}\alpha^\top Q\alpha - e^\top \alpha \\ \text{s.t.} \quad & y^\top \alpha = 0, \\ & 0 \le \alpha \le c, \end{aligned}$$

Decomposition approach

Given any dual feasible solution, (α, β), we partition $I = \{1, \ldots, n\}$ into B and N:

- $\forall i \in B \ 0 < \alpha_i < c$.
- $\forall i \in N \ 0 \leq \alpha_i \leq c$.

$B \cup N = I$ and $B \cap N = \emptyset$.

Based on the partition (B, N) we define Q_{BB} $(Q_{BN}$, Q_{NB}, $Q_{NN})$ y_B (y_N) and α_B (α_N)

Active set method for convex QP

Solution of an LP is always at the vertex. In the case of QP it can be anywhere.

$$Q = \begin{bmatrix} Q_{BB} & {Q_{NB}}^\top \\ Q_{NB} & Q_{NN} \end{bmatrix}.$$

Idea: temporarily fix all α_N to their current values and solve the reduced problem in terms of α_B only.

$$\begin{array}{ll} \min_\alpha & \frac{1}{2}{\alpha_B}^\top Q_{BB}\alpha_B + {e_B}^\top \alpha_B + {\alpha_N}^\top Q_{NB}\alpha_B + \frac{1}{2}{\alpha_N}^\top Q_{NN}\alpha_N - {e_N}^\top \alpha_N \\ \text{s.t.} & {y_B}^\top \alpha_B = -{y_N}^\top \alpha_N, \\ & 0 \le \alpha_B \le c, \end{array}$$

Solve this "small" QP problem by any method

Decomposition Method

How to determine the next set B? Look for steepest descent direction of size |B|.

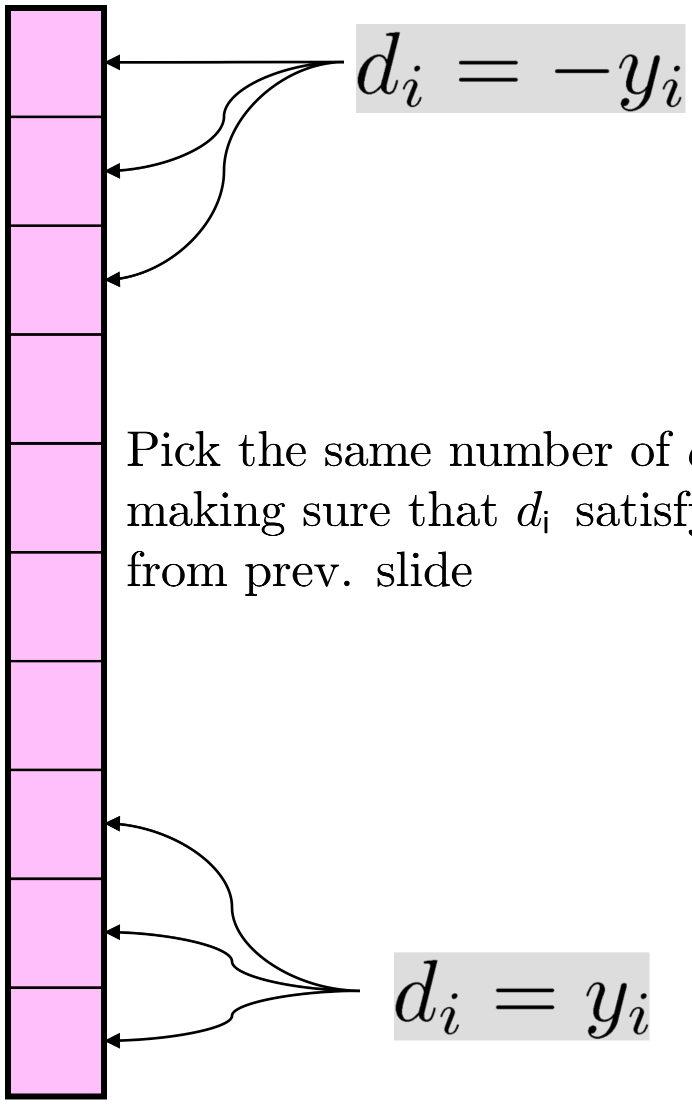
$$\min_d \nabla_\alpha f(\alpha)^\top d = \nabla_\alpha(\frac{1}{2}\alpha^\top Q\alpha - e^\top \alpha)d = (\alpha^\top Q - e)d$$

$$\begin{aligned}
\min\nolimits_{\mathrm{d}} \quad & (\alpha^\top Q - e)d \\
\text{s.t.} \quad & y^\top d = 0 \\
& -e \le d \le e \\
& d_i \le 0 \text{ if } \alpha_i = C \\
& d_i \ge 0 \text{ if } \alpha_i = 0 \\
& |\{i : d_i \neq 0\}| = |B|
\end{aligned}$$

Finding the new set B

Ordered vector

$$\nabla_\alpha f(\alpha) = Y(\alpha^\top Q - e)$$



Pick the same number of $d_i = y_i$ and $d_i =$ ¡ y_i making sure that d_i satisfy the conditions from prev. slide

Workload of a decomposition method

- $$\begin{aligned} \min_\alpha \quad & \frac{1}{2}{\alpha_B}^\top Q_{BB}\alpha_B + {e_B}^\top \alpha_B + {\alpha_N}^\top Q_{NB}\alpha_B \\ \text{s.t.} \quad & {y_B}^\top \alpha_B = -{y_N}^\top \alpha_N, \\ & 0 \le \alpha_B \le c, \end{aligned}$$

  If using an interior point method, empirical complexity is $O(n_B^3)$.

- Computing $\alpha^\top Q - e$ is almost equivalent to computing ${\alpha_B}^\top Q$ which is $O(n_B n)$.

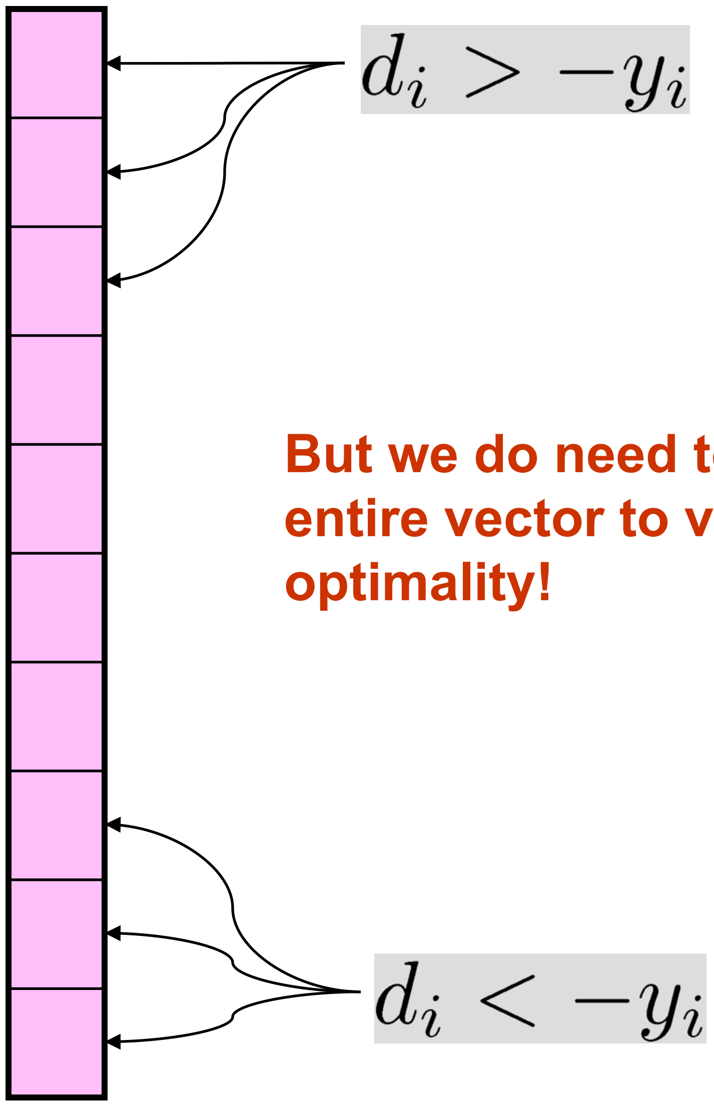
- The complexity of the second step can be reduced by "shrinking" - considering only "important" part of $\alpha^\top Q - e$ vector.

Reducing the cost of finding the new set B

Reduce the size of the vector

$$Y(\alpha^\top Q - e)$$

by ignoring the elements that are likely to be in the middle (for example because they were in the middle last 100 iterations)



But we do need to compute the entire vector to verify optimality!

Complexity

Per iteration:

- Need to solve $Q_{ss}p = r$ at each iteration, where Q_{ss} is $n_s \times n_s$, n_s number of active support vectos for ASMs, but can be any number (2 or more) for the DMs.
- In ASMs, by updating the Cholesky of Q_{ss} the work reduced to $O(n_s^2)$. For DMs have to solve each subproblem independently.
- Need to search for negative s and x_i, $O(n_s n)$ operations.
- By considering only a small number of "promising" candidates, the work is substantially reduced.

Bound on the number of iterations

- Active set method - finite to obtain the exact solution, but could be exponential.
- Decomposition methods - $O(n^2/\epsilon)$ - not polynomial.

Interior point methods Cplex, OOQP, OOPS, Mosek	$O(k^2n) \times O(n \log(1/\epsilon))$ $O(k^2n)$ in practice, very accurate solutions
Active set method SVM-QP, Cplex	Exponential in theory $O(n_s n^2)$ in practice, very accurate solutions
Decomposition methods SMO, SVM^{light}	O(n^2/ϵ), reasonably accurate solutions
Cutting plane methods SVM^{perf}	$O(Rns/\epsilon)$, no accurate solutions
Stochastic Gradient Pegasos	$O(Rs/\epsilon)$, probabilitsitc results, requires i.i.d samples, no accurate solutions

Optimality Conditions

$$\begin{aligned} \min_\alpha \quad & \frac{1}{2}\alpha^\top Q\alpha - e^\top\alpha \\ \text{s.t.} \quad & y^\top\alpha = 0, \\ & 0 \le \alpha \le c, \end{aligned}$$

KKT conditions

$$\begin{aligned} & \alpha_i s_i = 0, \quad i = 1, \dots, n, \\ & (c - \alpha_i)\xi_i = 0, \quad i = 1, \dots, n, \\ & y^\top\alpha = 0, \\ & -Q\alpha + y\beta + s - \xi = -e, \\ & 0 \le \alpha \le c, \ s \ge 0, \ \xi \ge 0. \end{aligned}$$

Active Set

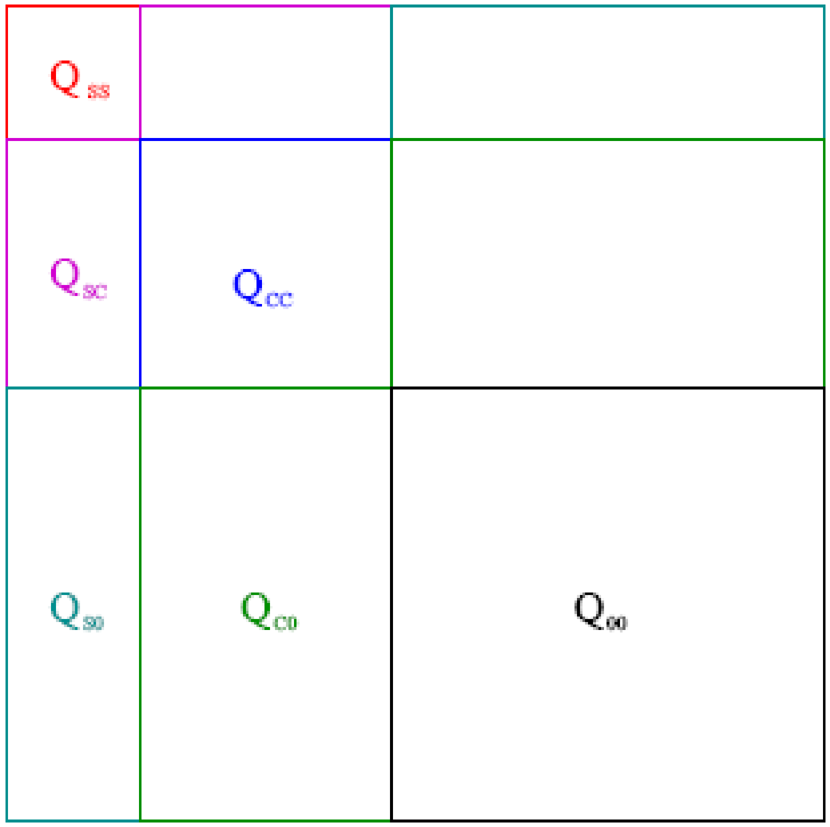
Given a dual basic feasible solution, (α, β, s, ξ), we partition $I = \{1, \ldots, n\}$ into $\mathbf{I_0}$, $\mathbf{I_c}$ and $\mathbf{I_s}$:

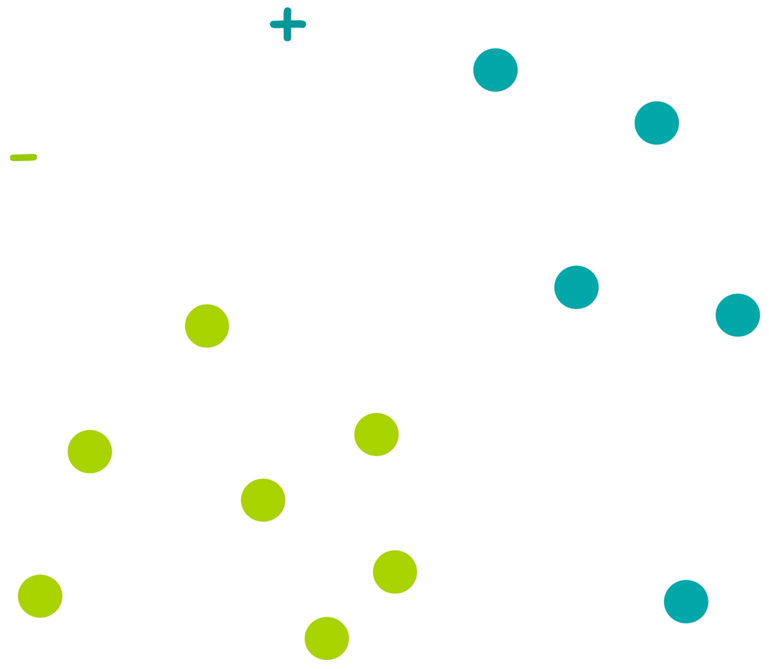
- $\forall i \in \mathbf{I_0}$ $\xi_i = 0$ and $\alpha_i = 0$, $(s_i \geq 0?)$
- $\forall i \in \mathbf{I_c}$ $s_i = 0$ and $\alpha_i = c$, $(\xi_i \geq 0?)$
- $\forall i \in \mathbf{I_s}$ $s_i = \xi_i = 0$ and $0 < \alpha_i < c$.

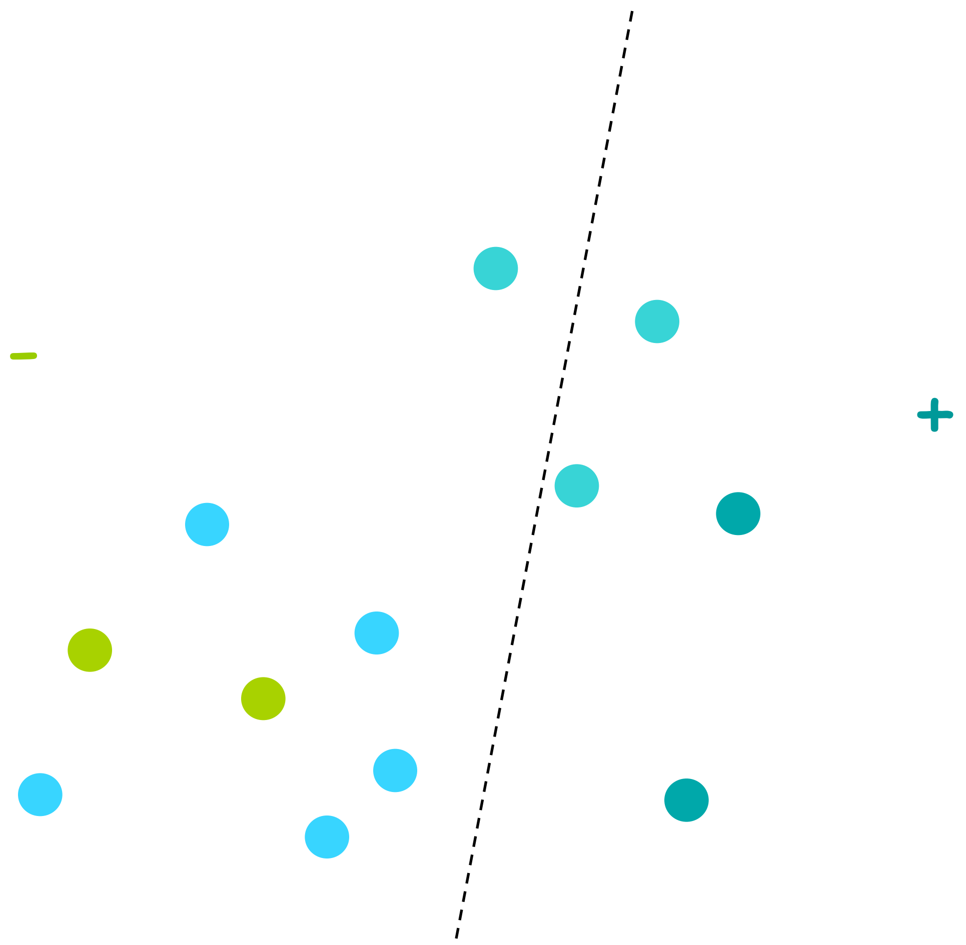
$\mathbf{I_0} \cup \mathbf{I_c} \cup \mathbf{I_s} = I$ and $\mathbf{I_0} \cap \mathbf{I_c} = \mathbf{I_c} \cap \mathbf{I_s} = \mathbf{I_0} \cap \mathbf{I_s} = \emptyset$.

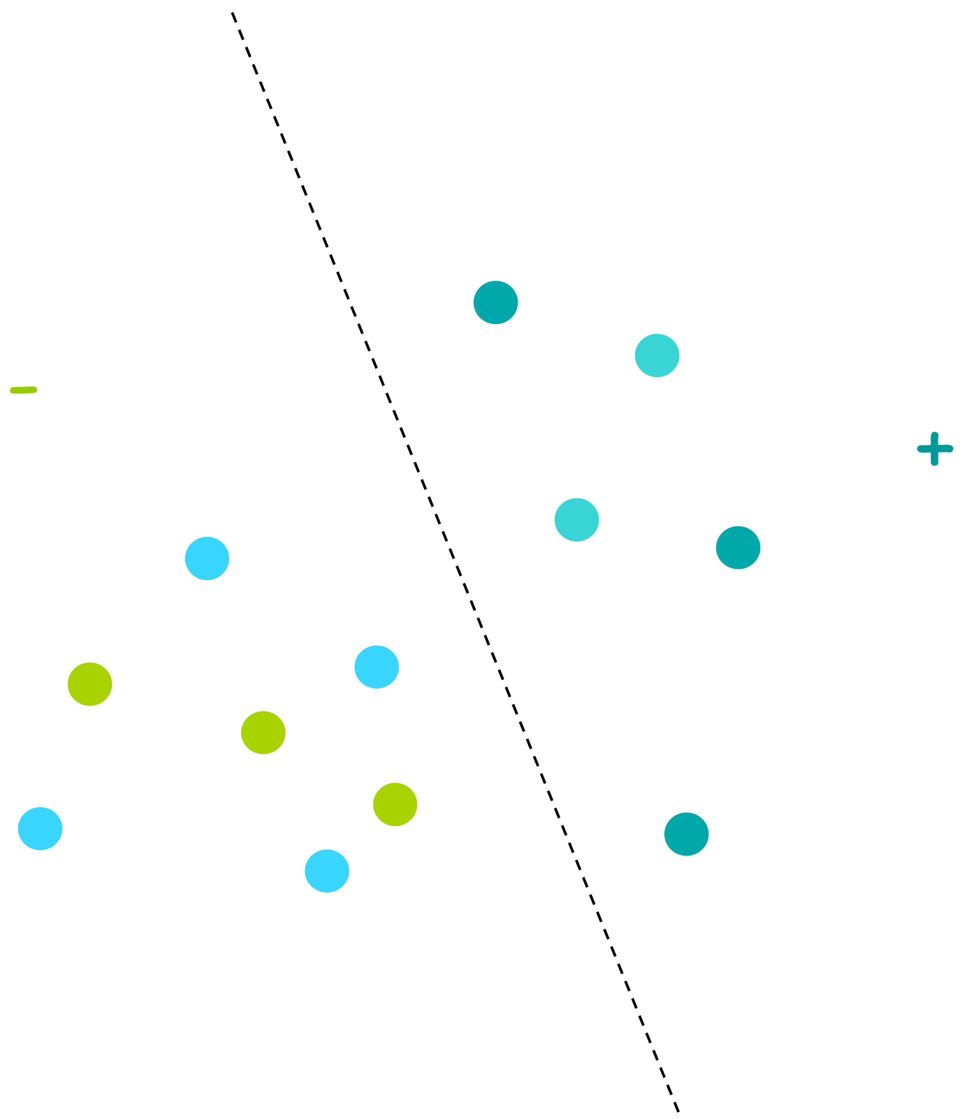
Based on the partition $(\mathbf{I_0}, \mathbf{I_c}, \mathbf{I_s})$ we define Q_{ss} (Q_{cs}, Q_{sc}, Q_{cc}, Q_{0s}, Q_{00}), y_s (y_c, y_0) and α_s (α_c, α_0)

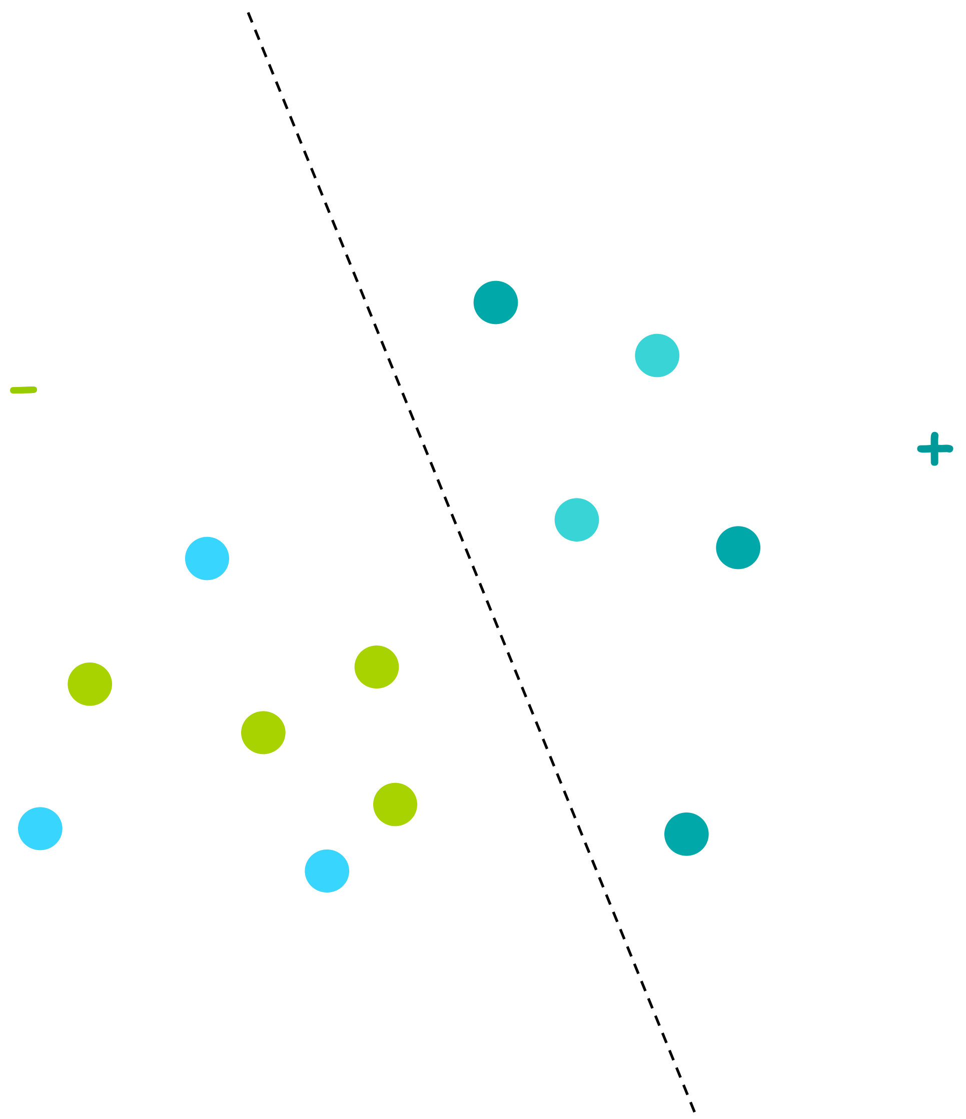
Partitioning of matrix Q











Active Set Method

Step 1

(i) Solve

$$\begin{array}{ll} \min_{\alpha_s} & \frac{1}{2}{\alpha_s}^\top Q_{ss}\alpha_s + ce^\top Q_{cs}\alpha_s - e^\top \alpha_s \\ \text{s.t.} & {y_s}^\top \alpha_s = -{y_c}^\top \alpha_c \end{array}$$

(ii) From the current iterate make a step toward the solution until for some $i \in \mathbf{I_s}$ $(\alpha_s)_i = 0$ or $(\alpha_s)_j = c$ or until solution is reached.

(iii) If for some $i \in \mathbf{I_s}$, $(\alpha_s)_i = 0$
Then update $I_s = I_s \backslash \{i\}$, $I_0 = I_0 \cup \{i\}$, and go to step (i).

(iv) If for some $i \in \mathbf{I_s}$, $(\alpha_s)_i = c$
then update $I_s = I_s \backslash \{i\}$, $I_c = I_c \cup \{i\}$, and go to step (i).

(v) If the optimum is reached in step (ii), proceed to **Step 2**.

Active Set Method

Step 2

(i) Compute s_0

$$s_0 = -Q_{0s}\alpha_s - y_0\beta + 1 - cQ_{0c}e$$

and ξ_c

$$\xi_c = Q_{cs}\alpha_s + y_c\beta - 1 + cQ_{cc}e$$

(ii) Find $i_0 = \text{argmin}_i\{s_i : i \in \mathbf{I_0}\}$.

Find $i_c = \text{argmin}_i\{\xi_i : i \in \mathbf{I_c}\}$.

(iii) If $s_{i_0} \geq 0$ and $\xi_{i_c} \geq 0$ then the current solution is optimal, **Exit**.

If $s_{i_0} \leq \xi_{i_c}$, then $I_s = I_s \cup \{i_0\}$ and $I_0 = I_0 \backslash \{i_0\}$.

Else, $I_s = I_s \cup \{i_c\}$ and $I_c = I_c \backslash \{i_c\}$.

Go to **Step 1**.

Active Set Method

Step 1

(i) Solve a system with matrix

$$\begin{bmatrix} Q_{ss} & y \\ y^\top & 0 \end{bmatrix}.$$

If factorization $Q_{ss} = G_s {G_s}^\top$ is available, then work is $\mathbf{O(n_s^2)}$.

(ii) Step toward solution. $\mathbf{O(n_s)}$

(iii) If for some $i \in \mathbf{I_s}$, $(\alpha_s)_i = 0$, then update $I_s = I_s \backslash \{i\}$, $I_0 = I_0 \cup \{i\}$, update G_s by removing a row. $\mathbf{O(n_s^2)}$

(iv) If for some $i \in \mathbf{I_s}$, $(\alpha_s)_i = c$ then update $I_s = I_s \backslash \{i\}$, $I_c = I_c \cup \{i\}$, update $e^\top Q_{cs}$ and G_s by removing a row. $\mathbf{O(n_s^2)} + \mathbf{O(n_c)}$

Active Set Method

Step 2

(i)

$$s_0 = -Q_{0s}\alpha_s - y_0\beta + 1 - cQ_{0c}e$$
$$\xi_c = Q_{cs}\alpha_s + y_c\beta - 1 + cQ_{cc}e$$
$$\mathbf{O(n_s n)}$$

(ii) Find $i_0 = \text{argmin}_i\{s_i : i \in \mathbf{I_0}\}$, $i_c = \text{argmin}_i\{\xi_i : i \in \mathbf{I_c}\}$. $\mathbf{O(n)}$

(iii) If $s_{i_0} \le \xi_{i_c}$, then $I_s = I_s \cup \{i_0\}$ and $I_0 = I_0 \backslash \{i_0\}$.
Update G_s by adding a row

Else, $I_s = I_s \cup \{i_c\}$ and $I_c = I_c \backslash \{i_c\}$.
Update $e^\top Q_{cs}$ and G_s by adding a row

$$\mathbf{O(n_s^2) + O(n_c)}$$

Complexity

Active set method:

- Need to solve $Q_{ss}p = r$ at each iteration, where Q_{ss} is completely dense, $k_s \times k_s$.
- By updating the Cholesky of Q_{ss} the work reduced to $O(k_s^2)$.
- Need to search for negative s and x_i, $O(k_s n)$ operations.
- By considering only a small number of "promising" candidates, the work is substantially reduced.