Advanced Topics in Machine Learning

1. Learning SVMs

Lars Schmidt-Thieme

Information Systems and Machine Learning Lab (ISMLL)
University of Hildesheim, Germany

Outline

Plan for the Lecture

Roughly three chapters planned:

1. Learning SVMs (and other classifiers)
2. Factorization Methods
3. Structured Prediction

Outline

Ridge Regression

$$\begin{aligned} \text{minimize } \mathrm{RSS}_\lambda(\hat\beta) :=& \mathrm{RSS}(\hat\beta) + \lambda \sum_{j=1}^{p} \hat\beta_j^2 \\ =& \sum_{i=1}^{n} (y_i - \hat y_i)^2 + \lambda \sum_{j=1}^{p} \hat\beta_j^2, \quad \hat y_i := \beta_0 + \langle \hat\beta, x_i \rangle \\ =& \text{L2 loss} \qquad + \lambda \text{ L2 reg.,} \quad \text{linear model} \end{aligned}$$

with $\lambda \geq 0$ (complexity/regularization parameter).

Logistic Regression

maximize

$$L^{\text{cond}}_{\mathcal{D}}(\hat\beta) = \prod_{i=1}^{n} p(Y = y_i \,|\, X = x_i; \hat\beta)$$
$$= \prod_{i=1}^{n} p(Y = 1 \,|\, X = x_i; \hat\beta)^{y_i}(1 - p(Y = 1 \,|\, X = x_i; \hat\beta))^{1-y_i}$$

with

$$p(Y = 1 \,|\, X) = \text{logistic}(\langle X, \beta\rangle) + \epsilon = \frac{e^{\sum_{i=1}^{n} \beta_i X_i}}{1 + e^{\sum_{i=1}^{n} \beta_i X_i}} + \epsilon$$

resulting to: maximize

$$\log L^{\text{cond}}_{\mathcal{D}}(\hat\beta) = \sum_{i=1}^{n} y_i \langle x_i, \hat\beta\rangle - \log(1 + e^{\langle x_i, \hat\beta\rangle})$$

Linear Support Vector Classification

$$\begin{aligned} \text{minimize } f(\beta, \xi) :=& \frac{1}{2}||\beta||^2 + \gamma \sum_{i=1}^{n} \xi_i \qquad \text{[LSVM]} \\ \text{w.r.t. } y_i(\beta_0 + \langle \beta, x_i \rangle) \geq& 1 - \xi_i, \quad i = 1, \ldots, n \\ \xi \geq& 0 \end{aligned}$$

for given $\gamma \geq 0$ (complexity/regularization parameter).
or equivalently

$$\begin{aligned} &\text{minimize } \frac{1}{2}||\beta||^2 + \gamma \sum_{i=1}^{n} [1 - y_i \hat{y}_i]_+, \quad \hat{y}_i := \beta_0 + \langle \beta, x_i \rangle \\ &= \text{L2 regular.} + \gamma \text{ hinge loss,} \qquad \text{linear model} \end{aligned}$$

Problem: hinge loss is not differentiable.

Support Vector Classification, Dual and Non-linear

Dual formulation (linear kernel):

$$\text{maximize} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \langle x_i, x_j \rangle$$

$$\text{w.r.t.} \sum_{i=1}^{n} \alpha_i y_i = 0$$

$$\alpha_i \geq 0, \quad \alpha_i \leq \gamma$$

Dual formulation (non-linear kernel k):

$$\text{maximize} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j k(x_i, x_j)$$

$$\text{w.r.t.} \sum_{i=1}^{n} \alpha_i y_i = 0$$

$$\alpha_i \geq 0, \quad \alpha_i \leq \gamma$$

Linear Support Vector Regression

$$\text{minimize} \sum_{i=1}^{n} [|y_i - \hat{y}_i| - \epsilon]_+ + \frac{\lambda}{2} ||\hat{\beta}||^2, \quad \hat{y}_i := \hat{\beta}_0 + \langle \hat{\beta}, x_i \rangle$$

$$= \epsilon\text{-insensitive loss} + \lambda \text{ L2 reg.}, \text{linear model}$$

Problem: ϵ-insensitive loss is not differentiable.

Support Vector Regression, Dual and Non-Linear

$$\min \epsilon \sum_{i=1}^{n} (\alpha_i^* - \alpha_i) - \sum_{i=1}^{n} y_i(\alpha_i^* - \alpha_i) + \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} (\alpha_i^* - \alpha_i)(\alpha_j^* - \alpha_j)\langle x_i, x_j \rangle$$

$$\text{s.t. } \sum_{i=1}^{n} (\alpha_i^* - \alpha_i) = 0$$

$$\alpha_i^* \alpha_i = 0$$

$$\alpha_i \geq 0, \quad \alpha_i^* \leq \frac{1}{\lambda}$$

$\langle x_i, x_j \rangle$ can be replaced by a non-linear kernel $k(x_i, x_j)$.

L1 regularization

For all learning problems, instead of L2 regularization

$$\mathrm{reg}_{L2}(\beta) := \frac{1}{2}||\beta||^2 = \frac{1}{2}\sum_{i=1}^{n}\beta_i^2$$

one also can use L1 regularization

$$\mathrm{reg}_{L1}(\beta) := ||\beta||_1 = \sum_{i=1}^{n}|\beta_i|$$

(or any other function penalizing large parameters).

Problem: the objective function will be non-differentiable.

Problems to address in first chapter

- How to learn non-linear SVMs efficiently ?
- How to learn linear SVMs in the primal efficiently?
  I.e., how to deal with non-differentiable objective functions in convex optimization?
- How to carry over learning algorithms for linear SVMs to non-linear SVMs?
- How to take advantage from sparse data ?
- How to choose the right regularization (L1, L2, ...) ?
- How to find good hyperparameters (λ, γ, ...) ?

Outline

Derivation of Dual Problem

Primal formulation:

$$\text{minimize } f(\beta, \beta_0, \xi) := \frac{1}{2}||\beta||^2 + \gamma\langle \mathbb{e}, \xi \rangle$$

$$\text{w.r.t.} \quad \begin{array}{lll} g(\beta, \beta_0, \xi) & := \mathbb{e} - \xi - y \odot (\beta_0 \mathbb{e} + X\beta) & \leq 0 \\ \tilde{g}(\xi) & := -\xi & \leq 0 \end{array}$$

Lagrange function:

$$\begin{aligned} L_f(\beta, \beta_0, \xi, \alpha, \tilde{\alpha}) :=& f(\beta, \beta_0, \xi) + \langle \alpha, g(\beta, \beta_0, \xi) \rangle + \langle \tilde{\alpha}, \tilde{g}(\xi) \rangle \\ =& \frac{1}{2}||\beta||^2 + \gamma\langle \mathbb{e}, \xi \rangle + \langle \alpha, \mathbb{e} - \xi - y \odot (\beta_0 \mathbb{e} + X\beta) \rangle + \langle \tilde{\alpha}, -\xi \rangle \\ =& \frac{1}{2}||\beta||^2 + \langle \alpha, \mathbb{e} \rangle - \beta_0 \langle \alpha, y \rangle - \langle \alpha, y \odot X\beta \rangle + \langle \gamma \mathbb{e} - \alpha - \tilde{\alpha}, \xi \rangle \end{aligned}$$

Note: $\mathbb{e} := (1, 1, \ldots, 1)^T$ and $a \odot b := (a_i \cdot b_i)_i = \text{diag}(a) b$ elementwise multiplication.

Derivation of Dual Problem

Lagrange function: $L_f(\beta, \beta_0, \xi, \alpha, \tilde{\alpha})$

$$= \frac{1}{2}||\beta||^2 + \langle \alpha, \mathbb{e} \rangle - \beta_0 \langle \alpha, y \rangle - \langle \alpha, y \odot X\beta \rangle + \langle \gamma \mathbb{e} - \alpha - \tilde{\alpha}, \xi \rangle$$

$$\frac{\partial L_f}{\partial \beta} = \beta^T - \alpha^T (y \odot X) \overset{!}{=} 0 \quad \Leftrightarrow \beta = X^T (y \odot \alpha) \qquad (I)$$

$$\frac{\partial L_f}{\partial \beta_0} = -\langle \alpha, y \rangle \overset{!}{=} 0 \quad \Leftrightarrow \langle y, \alpha \rangle = 0 \qquad (II)$$

$$\frac{\partial L_f}{\partial \xi} = \gamma \mathbb{e} - \alpha - \tilde{\alpha} \overset{!}{=} 0 \quad \Leftrightarrow \gamma \mathbb{e} - \alpha - \tilde{\alpha} = 0 \qquad (III)$$

$$\bar{f}(\alpha, \tilde{\alpha}) := \inf_{\beta, \beta_0, \xi} L_f(\beta, \beta_0, \xi, \alpha, \tilde{\alpha})$$

$$= \frac{1}{2}||X^T (y \odot \alpha)||^2 + \langle \alpha, \mathbb{e} \rangle - \langle \alpha, y \odot XX^T (y \odot \alpha) \rangle$$

$$= -\frac{1}{2} \alpha^T (XX^T \odot yy^T) \alpha + \langle \alpha, \mathbb{e} \rangle$$

Derivation of Dual Problem

$$
\begin{aligned}
\bar{f}(\alpha) &:= -\frac{1}{2}\alpha^T(XX^T \odot yy^T)\alpha + \langle \alpha, \mathbb{e} \rangle \\
\text{w.r.t. } \alpha &\geq 0 \\
\alpha &\leq \gamma && \text{(due to } \mathit{III}\text{)} \\
\langle y, \alpha \rangle &= 0 && (\mathit{II})
\end{aligned}
$$

Optimality Criterion I

Lemma (KKT for SVM)

A feasible point α, i.e., $\alpha \in [0, \gamma]$ with $\langle y, \alpha \rangle = 0$, is optimal, if and only if

$$y_i \hat{y}(x_i) \begin{cases} \geq 1 & \text{, for } \alpha_i = 0 \\ = 1 & \text{, for } 0 < \alpha_i < \gamma \\ \leq 1 & \text{, for } \alpha_i = \gamma \end{cases}$$

Proof. Choose the other parameters as follows:

$$\begin{aligned} \tilde{\alpha} &:= \gamma \mathbb{e} - \alpha \\ \beta &:= X^T (y \odot \alpha) \\ \beta_0 &:= y_i - \langle \beta, x_i \rangle, \quad \text{for any } i : 0 < \alpha_i < \gamma \\ \xi &:= [\mathbb{e} - y \odot (\beta_0 \mathbb{e} + X\beta)]_+ \end{aligned}$$

and show that the conditions above are equivalent to KKT:

Optimality Criterion II

KKT:

$$\begin{aligned}
g(x) \leq 0 \ (i) &\quad \Leftrightarrow 1 - \xi_i - y_i \hat{y}(x_i) \leq 0, \quad \xi_i \geq 0 \\
h(x) = 0 \ (ii) &\quad \text{n/a (no equality constraints)} \\
\lambda \geq 0 \ (iii) &\quad \Leftrightarrow \alpha_i \geq 0, \quad \tilde{\alpha}_i \geq 0 \\
\lambda_i g_i(x) = 0 \ (iv) &\quad \Leftrightarrow \alpha_i(1 - \xi_i - y_i \hat{y}(x_i)) = 0, \quad \tilde{\alpha}_i \xi_i = 0 \\
\frac{\partial f(x)}{\partial x} + \lambda^T \frac{\partial g(x)}{\partial x} + \nu^T \frac{\partial h(x)}{\partial x} = 0 \ (v) &\quad \Leftrightarrow \langle y, \alpha \rangle = 0, \ \text{choice of } \beta, \tilde{\alpha} \ (I - III)
\end{aligned}$$

"$\Rightarrow$": For $\alpha_i < \gamma$: $\rightsquigarrow \tilde{\alpha} > 0 \overset{\text{KKT (iv)}}{\rightsquigarrow} \xi_i = 0 \overset{\text{KKT (i)}}{\rightsquigarrow} y_i \hat{y}(x_i) \geq 1 - \xi_i = 1$

For $\alpha_i > 0$: $\overset{\text{KKT (iv)}}{\rightsquigarrow} y_i \hat{y}(x_i) = 1 - \xi_i \leq 1$

For $0 < \alpha_i < \gamma$ equality must hold.

Optimality Criterion III

"$\Leftarrow$": For $\alpha_i = 0$ KKT (iv a) holds trivially.
For $\alpha_i > 0$: $\rightsquigarrow y_i \hat{y}(x_i) \leq 1 \rightsquigarrow 1 - \xi_i - y_i \hat{y}(x_i) = 0$ (KKT (iv a)) with $\xi_i \geq 0$.

For $\alpha_i = \gamma$: $\rightsquigarrow \tilde{\alpha}_i = 0 \rightsquigarrow$ KKT (iv b)
For $\alpha_i < \gamma$: $\rightsquigarrow y_i \hat{y}(x_i) \geq 1 \rightsquigarrow \xi_i = 0 \rightsquigarrow$ KKT (iv b)

KKT (i) holds due to choice of ξ.

From Dual to Primal Parameters

Lemma

For linear SVMs primal parameters β can be computed from dual parameters α:

$$\begin{aligned}\beta =& X^T(y \odot \alpha)\\ \beta_0 =& y_i - \langle \beta, x_i \rangle, \quad \textit{for any } i : 0 < \alpha_i < \gamma\end{aligned}$$

Proof.

The formula for β is (I) above.

For $\alpha_i < \gamma$:

$$\begin{aligned} y_i \hat{y}(x_i) = y_i(\langle \beta, x_i \rangle + \beta_0) = 1 \\ \beta_0 = y_i - \langle \beta, x_i \rangle \end{aligned}$$

$\square$

Note: For nonlinear SVMs $\beta := \sum_{i=1}^n \alpha_i y_i \Phi(x_i)$ with features Φ.

Optimality Criterion (Variant)

Lemma (KKT for SVM (Keerthi et al. 2001))

A feasible point α, i.e., $\alpha \in [0, \gamma]$ with $\langle y, \alpha \rangle = 0$, is optimal, if and only if

$$\max_{i \in I_0} y_i - \sum_j \alpha_j y_j k(x_j, x_i) \leq \beta_0 \leq \min_{i \in I_1} y_i - \sum_j \alpha_j y_j k(x_j, x_i)$$

$$\text{with } I_0 := \{i \mid \alpha_i > 0, y_i = -1\} \cup \{i \mid \alpha_i < \gamma, y_i = +1\},$$
$$I_1 := \{i \mid \alpha_i > 0, y_i = +1\} \cup \{i \mid \alpha_i < \gamma, y_i = -1\}$$

Proof. For $\alpha_i > 0$:

$$y_i \hat{y}(x_i) = y_i (\sum_j \alpha_j y_j k(x_j, x_i) + \beta_0) \leq 1$$

$$\rightsquigarrow \beta_0 \leq y_1 - \sum_j \alpha_j y_j k(x_j, x_i), \quad \text{for } y_i = +1$$

$$\rightsquigarrow \beta_0 \geq y_1 - \sum_j \alpha_j y_j k(x_j, x_i), \quad \text{for } y_i = -1$$

Note: $I_0 \cap I_1 = \{i \mid 0 < \alpha_i < \gamma\}$.

Outline

Optimization Problem

$$
\begin{aligned}
\text{maximize } \bar{f}(\alpha) :=& \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j k(x_i, x_j) \\
\text{w.r.t. } \sum_{i=1}^{n} \alpha_i y_i =& 0 \\
\alpha_i \geq& 0, \quad \alpha_i \leq \gamma
\end{aligned}
$$

for given $\gamma \geq 0$ and kernel k.

Requires

- $n \times n$ kernel matrix $Q := (y_i y_j k(x_i, x_j))_{i,j=1,\ldots,n}$.
- $2n + 1$ constraints.

Usually we hope for **sparse solutions**, i.e., only a fraction of data points turn out to be support vectors, and thus $\alpha_i = 0$ for many i.

Chunking

Idea:

- initially select a random set of candidate support vectors
- iteratively
  - train a model on the candidate support vectors only and
  - select a new set of candidate support vectors
    - retaining the support vectors of the current model
    - plus those data points with largest prediction error for the current model.

Typically the candidate set is growing from iteration to iteration.

Problems:

- still requires to store the kernel matrix of all support vectors, so if the solution is not sparse, the kernel matrix may be too large.

Decomposition Methods

Idea:

- initially select a random set of active support vectors
- initialize $\alpha_i := 0$ for non-active support vectors
- iteratively
  - train a model for the residuum of the non-active support vectors on the active support vectors only and
  - select a new set of active support vectors,
    e.g., those data points with largest prediction error for the current model.

To train a model on the residuum of the non-active support vectors is the same as to train a model for the full model, keeping the α_i's for non-active support vectors fixed.

Decomposition Methods: Sequential Minimal Optimization

Idea (Platt 1999):

- Use only 2 active support vectors.

Advantages:

- minimization step can be done analytically
  - fast
  - easy to implement / does not require a QP solver
- no need to store a kernel matrix

SMO: Analytic Minimization Step

Lemma (SMO)

The maximum of the objective function is assumed for

$$\begin{aligned}
\tilde{\alpha}_2 &:= \alpha_2^{old} + y_2 \frac{(\hat{y}_1 - y_1) - (\hat{y}_2 - y_2)}{k(x_1, x_1) + k(x_2, x_2) - 2k(x_1, x_2)} \\
\alpha_2 &:= [\tilde{\alpha}_2]_U^V \\
\alpha_1 &:= \alpha_1^{old} + y_1 y_2 (\alpha_2^{old} - \alpha_2)
\end{aligned}$$

where

$$U := \begin{cases} [\alpha_2^{old} - \alpha_1^{old}]_+, & \textit{if } y_1 \neq y_2 \\ [\alpha_1^{old} + \alpha_2^{old} - \gamma]_+, & \textit{else} \end{cases}$$

$$V := \begin{cases} \gamma - [\alpha_1^{old} - \alpha_2^{old}]_+, & \textit{if } y_1 \neq y_2 \\ \min(\gamma, \alpha_1^{old} + \alpha_2^{old}), & \textit{else} \end{cases}$$

and $[x]_a^b := \min(\max(a, x), b)$.

SMO: Analytic Minimization Step

Proof.

$0 \leq \alpha_i \leq \gamma$ and due to $\sum_{i=1}^n \alpha_i y_i = 0$:

$$\begin{aligned}\alpha_1 y_1 + \alpha_2 y_2 &= \alpha_1^{\text{old}} y_1 + \alpha_2^{\text{old}} y_2 \\ \alpha_1 &= \alpha_1^{\text{old}} y_1 y_1 + (\alpha_2^{\text{old}} - \alpha_2) y_1 y_2\end{aligned}$$

so for $y_1 y_2 = -1$:

$$\begin{aligned}\alpha_1^{\text{old}} - (\alpha_2^{\text{old}} - \alpha_2) &\geq 0 & &\leadsto \alpha_2 \geq \alpha_2^{\text{old}} - \alpha_1^{\text{old}} \\ \alpha_1^{\text{old}} - (\alpha_2^{\text{old}} - \alpha_2) &\leq \gamma & &\leadsto \alpha_2 \leq \gamma - (\alpha_1^{\text{old}} - \alpha_2^{\text{old}})\end{aligned}$$

and for $y_1 y_2 = 1$:

$$\begin{aligned}\alpha_1^{\text{old}} + (\alpha_2^{\text{old}} - \alpha_2) &\geq 0 & &\leadsto \alpha_2 \leq \alpha_1^{\text{old}} + \alpha_2^{\text{old}} \\ \alpha_1^{\text{old}} + (\alpha_2^{\text{old}} - \alpha_2) &\leq \gamma & &\leadsto \alpha_2 \geq \alpha_1^{\text{old}} + \alpha_2^{\text{old}} - \gamma\end{aligned}$$

SMO: Analytic Minimization Step

The objective function for variables α_1 and α_2 only is

$$\alpha_1 + \alpha_2 - \frac{1}{2}k_{1,1}\alpha_1^2 - \frac{1}{2}k_{2,2}\alpha_2^2 - k_{1,2}y_1y_2\alpha_1\alpha_2 - y_1v_1\alpha_1 - y_2v_2\alpha_2 + \text{const.}$$

$$\text{with } k_{i,j} := k(x_i, x_j)$$

$$v_i := \sum_{j=3}^{n} y_j\alpha_j k_{i,j} = \hat{y}(x_i) - y_1\alpha_1 k_{i,1} - y_2\alpha_2 k_{i,2} - \beta_0$$

along the constraint $\alpha_1 + s\alpha_2 = c$ (with $s := y_1y_2$ and some constant c):

$$c - s\alpha_2 + \alpha_2 - \frac{1}{2}k_{1,1}(c - s\alpha_2)^2 - \frac{1}{2}k_{2,2}\alpha_2^2 - k_{1,2}s(c - s\alpha_2)\alpha_2$$
$$- y_1v_1(c - s\alpha_2) - y_2v_2\alpha_2 + \text{const.}$$

SMO: Analytic Minimization Step

$$c - s\alpha_2 + \alpha_2 - \frac{1}{2}k_{1,1}(c - s\alpha_2)^2 - \frac{1}{2}k_{2,2}\alpha_2^2 - k_{1,2}s(c - s\alpha_2)\alpha_2 \\ - y_1 v_1 (c - s\alpha_2) - y_2 v_2 \alpha_2 + \text{const.}$$

The maximum satisfies

$$\begin{aligned}\frac{\partial(\ldots)}{\partial\alpha_2} = -s + 1 + k_{1,1}(c - s\alpha_2)s - k_{2,2}\alpha_2 - k_{1,2}sc + 2k_{1,2}ss\alpha_2 \\ + y_1 v_1 s - y_2 v_2 = 0 \\ \alpha_2(k_{1,1} + k_{2,2} - 2k_{1,2}) &= 1 - s - cs(k_{1,1} - k_{1,2}) + y_2(v_1 - v_2) \\ &= y_2(y_2 - y_1 - cy_1(k_{1,1} - k_{1,2}) + v_1 - v_2)\end{aligned}$$

SMO: Analytic Minimization Step

$$\tilde\alpha_2(k_{1,1}+k_{2,2}-2k_{1,2}) = y_2(y_2 - y_1 - cy_1(k_{1,1}-k_{1,2}) + v_1 - v_2)$$

$$\begin{aligned}
\tilde\alpha_2(k_{1,1}+k_{2,2}-2k_{1,2})y_2\\
=&y_2 - y_1 - cy_1(k_{1,1}-k_{1,2})\\
&+\hat y_1 - y_1\alpha_1 k_{1,1} - y_2\alpha_2 k_{1,2} - \beta_0\\
&-\hat y_2 + y_2\alpha_1 k_{2,1} + y_2\alpha_2 k_{2,2} + \beta_0\\
=&(\hat y_1 - y_1) - (\hat y_2 - y_2) + (\alpha_1 + s\alpha_2)y_1(k_{1,1}-k_{1,2})\\
&-y_1\alpha_1 k_{1,1} - y_2\alpha_2 k_{1,2} + y_1\alpha_1 k_{2,1} + y_2\alpha_2 k_{2,2}\\
=&(\hat y_1 - y_1) - (\hat y_2 - y_2) + \alpha_1 y_1(k_{1,1}-k_{1,2}) + \alpha_2 y_2(k_{1,1}-k_{1,2})\\
&-y_1\alpha_1 k_{1,1} - y_2\alpha_2 k_{1,2} + y_1\alpha_1 k_{2,1} + y_2\alpha_2 k_{2,2}\\
=&(\hat y_1 - y_1) - (\hat y_2 - y_2) + \alpha_2 y_2(k_{1,1}+k_{2,2}-2k_{1,2})
\end{aligned}$$

$$\tilde\alpha_2 = \alpha_2 + y_2\frac{(\hat y_1 - y_1) - (\hat y_2 - y_2)}{k_{1,1}+k_{2,2}-2k_{1,2}}$$

SMO: Choice of Active Support Vectors

Original SMO (Platt 1999): choose (i,j) with

- i with $0 < \alpha_i < \gamma$ (if possible, otherwise any),
- j with $0 < \alpha_j < \gamma$ and maximal $|(\hat{y}_j - y_j) - (\hat{y}_i - y_i)|$
  (as approximation of the unclipped step length)
  (if possible, otherwise any with $0 < \alpha_j < \gamma$, otherwise any)

Worst violating pair (Keerthi et al. 2001): choose (i,j) with

$$i := \arg\max_{k \in I_0} y_k - \sum_l \alpha_l y_l k(x_l, x_k) \qquad = \arg\max_{k \in I_0} y_k \frac{\partial \bar{f}}{\partial \alpha_k}$$

$$j := \arg\min_{k \in I_1} y_k - \sum_l \alpha_l y_l k(x_l, x_k) \qquad = \arg\min_{k \in I_1} y_k \frac{\partial \bar{f}}{\partial \alpha_k}$$

SMO: Choice of Active Support Vectors

Using Second Order Information (R.-E. Fan, P.-H. Chen, and C.-J. Lin 2005): choose (i, j) with

$$i := \arg\max_{k \in I_0} y_k \frac{\partial \bar{f}}{\partial \alpha_k}$$

$$j := \arg\min_{k \in I_1} \frac{(-y_i \frac{\partial \bar{f}}{\partial \alpha_i} + y_k \frac{\partial \bar{f}}{\partial \alpha_k})^2}{k(x_i, x_i) + k(x_k, x_k) - 2k(x_i, x_k)}$$

SMO: Maintain β_0

$$\begin{aligned}\beta_0^{\text{new}} :=&\beta_0^{\text{old}} + y_k - \hat{y}^{\text{old}}(x_k)\\ &- (\alpha_i - \alpha_i^{\text{old}})y_i k(x_i, x_k) - (\alpha_j - \alpha_j^{\text{old}})y_j k(x_j, x_k), \qquad \text{if } 0 < \alpha_k < \gamma\end{aligned}$$

for $k \in \{i, j\}$.
This choice of β_0 enforces

$$\begin{aligned}0 \overset{!}{=}&y_k - \hat{y}^{\text{new}}(x_i) = y_k - (\sum_l \alpha_l^{\text{new}} y_l k(y_l, y_k) + \beta_0^{\text{new}})\\ =&y_k - (\sum_l \alpha_l^{\text{old}} y_l k(y_l, y_k) + \beta_0^{\text{old}} - \beta_0^{\text{old}}\\ &+ (\alpha_i - \alpha_i^{\text{old}})y_i k(x_i, x_k) + (\alpha_j - \alpha_j^{\text{old}})y_j k(x_j, x_k) + \beta_0^{\text{new}})\\ \beta_0^{\text{new}} =&\beta_0^{\text{old}} + y_k - \hat{y}^{\text{old}}(x_i)\\ &- (\alpha_i - \alpha_i^{\text{old}})y_i k(x_i, x_k) - (\alpha_j - \alpha_j^{\text{old}})y_j k(x_j, x_k)\end{aligned}$$

If neither i nor j is at bounds, one can choose any β_0^{new} in between, e.g., the mean.

SMO: Maintain $y_k - \hat{y}(x_k)$

$$\begin{aligned}
y_k - \hat{y}^{\text{new}}(x_k) =& y_k - \sum_l \alpha_l^{\text{new}} y_l k(y_l, y_k) + \beta_0^{\text{new}} \\
=& y_k - (\sum_l \alpha_l^{\text{old}} y_l k(y_l, y_k) + \beta_0^{\text{old}} \\
& + (\alpha_i - \alpha_i^{\text{old}}) y_i k(x_i, x_k) + (\alpha_j - \alpha_j^{\text{old}}) y_j k(x_j, x_k) + \beta_0^{\text{new}} - \beta_0^{\text{old}}) \\
=& y_k - \hat{y}^{\text{old}}(x_k) \\
& - (\alpha_i - \alpha_i^{\text{old}}) y_i k(x_i, x_k) - (\alpha_j - \alpha_j^{\text{old}}) y_j k(x_j, x_k) - \beta_0^{\text{new}} + \beta_0^{\text{old}}
\end{aligned}$$

$y_k - \hat{y}^{\text{new}}(x_k)$ is maintained for all k not at bounds.

SMO: Maintain β

For a linear SVM, β can be maintained:

$$\begin{aligned}\beta^{\text{new}} &= \sum_k \alpha_k^{\text{new}} y_k x_k \\ &= \sum_k \alpha_k^{\text{old}} y_k x_k + (\alpha_i - \alpha_i^{\text{old}}) y_i x_i + (\alpha_j - \alpha_j^{\text{old}}) y_j x_j \\ &= \beta^{\text{old}} + (\alpha_i - \alpha_i^{\text{old}}) y_i x_i + (\alpha_j - \alpha_j^{\text{old}}) y_j x_j\end{aligned}$$

β can be used to compute $\hat{y}(x_k)$ in the primal
(esp. for k not at bounds for which $\hat{y}(x_k)$ is not maintained).

SMO: Caching Kernel Values

Kernel values $k(x_k, x_k)$ can be pre-computed once as they are used in the denominator of the formula for $\alpha_i^{\text{new}}, \alpha_j^{\text{new}}$.

More general, kernel rows $k(x_i, \cdot)$ could be cached (e.g., least-recently-used strategy; Joachims 1999).

Outline

Complete Gradient

$$\text{maximize } \bar{f}(\alpha) := \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j k(x_i, x_j)$$

$$\text{w.r.t. } \sum_{i=1}^{n} \alpha_i y_i = 0$$

$$\alpha_i \geq 0, \quad \alpha_i \leq \gamma$$

for given $\gamma \geq 0$ and kernel k.

If solved via gradient descent,

$$\frac{\partial \bar{f}}{\partial \alpha_i} = 1 - \sum_{j=1}^{n} \alpha_j y_i y_j k(x_i, x_j)$$

computing a single gradient requires the full kernel matrix.

Partial Gradient

To avoid the use of the full kernel matrix, one could employ partial gradient descent.

- move along the partial gradient for coordinates $I \subseteq \{1, \ldots, n\}$:

$$(\frac{\partial \bar{f}}{\partial \alpha} \odot \delta_I)_i := \begin{cases} \frac{\partial \bar{f}}{\partial \alpha_i} = 1 - \sum_{j=1}^n \alpha_j y_i y_j k(x_i, x_j), & i \in I \\ 0, & \text{else} \end{cases}$$

- to preserve the sum constraint $\sum_i y_i \alpha_i = 0$, the smallest possible I has size 2, $I := \{i, j\}$, and the update is

$$\alpha_i := \alpha_i + y_i \Delta\alpha, \quad \alpha_j := \alpha_j - y_j \Delta\alpha, \quad \Delta\alpha > 0$$

$$\text{with } y_i \Delta\alpha \in \begin{cases} [-\min(\alpha_i, \alpha_j), \gamma - \max(\alpha_i, \alpha_j)] & , \text{if } y_i y_j = -1 \\ [-\min(\alpha_i, \gamma - \alpha_j), \min(\gamma - \alpha_i, \alpha_j)] & , \text{if } y_i y_j = +1 \end{cases}$$

Note: $(\delta_I)_i := \begin{cases} 1, & \text{if } i \in I \\ 0, & \text{else} \end{cases}$.

Steepest Descent Direction

- The coordinates i, j with steepest descent direction are

$$
\begin{aligned}
&\arg\max \{y_i \frac{\partial \bar{f}}{\partial \alpha_i} - y_j \frac{\partial \bar{f}}{\partial \alpha_j} \mid i \in I^+, j \in I^-\} \\
= &(\arg\max_{i \in I^+} y_i \frac{\partial \bar{f}}{\partial \alpha_i}, \arg\min_{j \in I^-} y_j \frac{\partial \bar{f}}{\partial \alpha_j}) \\
&\text{with } I^+ := \{i \mid \exists \Delta\alpha > 0 : \alpha_i + y_i \Delta\alpha \in [0, \gamma]\} \\
&\qquad I^- := \{i \mid \exists \Delta\alpha > 0 : \alpha_i - y_i \Delta\alpha \in [0, \gamma]\}
\end{aligned}
$$

Note:
$$
\begin{aligned}
I^+ &= \{i \mid y_i = +1 \wedge \alpha_i < \gamma \vee y_i = -1 \wedge \alpha_i > 0\}, \\
I^- &= \{i \mid y_i = -1 \wedge \alpha_i < \gamma \vee y_i = +1 \wedge \alpha_i > 0\}
\end{aligned}
$$

Optimal Step Length

- The optimal step length can be determined analytically:

$$\Delta\alpha = \frac{(y_i - \hat{y}(x_i)) - (y_j - \hat{y}(x_j))}{k(x_i, x_i) + k(x_j, x_j) - 2k(x_i, x_j)}$$
$$= \frac{F_i - F_j}{k(x_i, x_i) + k(x_j, x_j) - 2k(x_i, x_j)}$$
$$\text{with } F_k := y_k \frac{\partial \bar{f}}{\partial \alpha_k} = y_k - \sum_{l=1}^{n} \alpha_l y_l K(x_k, x_l)$$

(SMO lemma; still needs to be clipped).

Note: $F_k = y_k - \hat{y}(x_k) + \hat{\beta}_0$.

Maintained Quantities & Initialization

- For efficiency, maintain

$$\begin{aligned} F_k :=& y_k \frac{\partial \bar{f}}{\partial \alpha_k} = y_k - \sum_{l=1}^{n} \alpha_l y_l K(x_k, x_l) \\ =& F_k - \Delta\alpha_i y_i K(x_k, x_i) - \Delta\alpha_j y_j K(x_k, x_j) \\ =& F_k - \Delta\alpha(K(x_k, x_i) - K(x_k, x_j)) \end{aligned}$$

- Initially,

$$\begin{aligned} \alpha_i =& 0 \\ F_i =& y_i \\ i =& \text{any with } y_i = +1 \\ j =& \text{any with } y_j = -1 \end{aligned}$$

SMO Algorithm

```
(1)  learn-svm-smo(training predictors $x$, training targets $y$,
(2)                complexity $\gamma$, kernel $K$, accuracy $\epsilon$) :
(3)  $\hat\alpha_i := 0 \quad \forall i$
(4)  $F_i := y_i \quad \forall i$
(5)  choose $i, j$ with $y_i = +1, y_j = -1$
(6)  do
(7)      $\Delta\alpha := \frac{F_i - F_j}{K(x_i,x_i) + K(x_j,x_j) - 2K(x_i,x_j)}$
(8)      $\Delta\alpha := \begin{cases} y_i[y_i\Delta\alpha]^{\gamma-\max(\hat\alpha_i,\hat\alpha_j)}_{-\min(\hat\alpha_i,\hat\alpha_j)} & \text{, if } y_i y_j = -1 \\ y_i[y_i\Delta\alpha]^{\min(\gamma-\hat\alpha_i,\hat\alpha_j)}_{-\min(\hat\alpha_i,\gamma-\hat\alpha_j)} & \text{, if } y_i y_j = +1 \end{cases}$
(9)      $\hat\alpha_i := \hat\alpha_i + y_i\Delta\alpha$
(10)     $\hat\alpha_j := \hat\alpha_j - y_j\Delta\alpha$
(11)     $F_k := F_k - \Delta\alpha(K(x_k, x_i) - K(x_k, x_j)) \quad \forall k$
(12)     $i := \text{argmax}\,\{\, F_k \mid y_k = +1 \wedge \hat\alpha_k < \gamma \vee y_k = -1 \wedge \hat\alpha_k > 0 \,\}$
(13)     $j := \text{argmin}\,\{\, F_k \mid y_k = -1 \wedge \hat\alpha_k < \gamma \vee y_k = +1 \wedge \hat\alpha_k > 0 \,\}$
(14) while $F_i - F_j > \epsilon$
(15) return $\hat\alpha$
```

Note: improved version from Keerthi et al. 2001, not the original from Platt 1999.

Partial Newton Algorithm

In gradient descent, descent direction and step length are chosen sequentially, evtl. leading to non-optimal steps.

Both coordinates cannot be chosen simultaneously without accessing the whole kernel matrix.

But the second coordinate could be chosen s.t. the resulting increase in the objective function is maximal:

$$\begin{aligned} i &= \arg\max_{i \in I^+} \; y_i \frac{\partial \bar{f}}{\partial \alpha_i} \\ j(i) &= \arg\max_{j \in I^-} \Delta\alpha(i,j)\left(y_i \frac{\partial \bar{f}}{\partial \alpha_i} - y_j \frac{\partial \bar{f}}{\partial \alpha_j}\right) \\ &= \arg\max_{j \in I^-} \Delta\alpha(i,j)(F_i - F_j) \\ &= \arg\max_{j \in I^-} \frac{(F_i - F_j)^2}{K(x_i, x_i) + K(x_j, x_j) - 2K(x_i, x_j)} \end{aligned}$$

Partial Newton Algorithm (2/2)

Alternatively, the same update can be derived as an approximation to a partial Newton algorithm (i.e., a Newton algorithm on subproblems of just 2 coordinates; P. Chen, R. Fan, and C. Lin 2006).

Nested Decomposition

The decomposition principle could be used in a nested way:

- chose a subset $I \subseteq \{1, \ldots, n\}$ of active coordinates
- chose a subsubset $I^{(2)} \subseteq I$ of active coordinates (e.g., SMO).

Advantage: the inner optimization does not have to maintain the quantities for all coordinates (e.g., SMO has only to maintain F_i for $i \in I$).

Disadvantage: after completing the inner optimization, optimality has to be checked (what usually means that non-maintained quantities now have to be recomputed).

Shrinking (Joachims 1999):

- if $\alpha_i = \gamma$ and $y_i \hat{y}(x_i) < 1$ for some iterations, drop coordinate i.
- if $\alpha_i = 0$ and $y_i \hat{y}(x_i) > 1$ for some iterations, drop coordinate i.

Outline

Dual Problem without Intercept

$$\begin{aligned}\text{minimize } f(\beta,\xi) :=& \frac{1}{2}||\beta||^2 + \gamma\sum_{i=1}^{n}\xi_i\\ \text{w.r.t. } y_i\langle\beta,x_i\rangle \geq& 1-\xi_i, \quad i=1,\ldots,n\\ \xi \geq& 0\end{aligned}$$

for given $\gamma \geq 0$.

Dual formulation (non-linear kernel K):

$$\begin{aligned}\text{maximize } \bar{f}(\alpha) :=& \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{n}\alpha_i\alpha_j y_i y_j K(x_i,x_j)\\ \text{w.r.t. } \alpha_i \geq& 0, \quad \alpha_i \leq \gamma\end{aligned}$$

Dual Coordinate Descent (Hsieh et al. 2008)

- Do not use an intercept β_0.
  (but add a constant primal pseudo-attribute).
- Optimize one α_i at a time
  (optimal value can be computed analytically)
- Do not prioritize α_i's,
  but select all sequentially (in random order)
- Do not maintain gradients,
  but maintain $\hat{\beta}$ (for linear kernels).

Optimal Step Length

Lemma

$\bar{f}(\alpha_i; (\alpha_j)_{j \neq i})$ *assumes its maximum at*

$$\alpha_i := \alpha_i^{old} + \frac{1 - y_i \hat{y}(x_i)}{K(x_i, x_i)}$$

Proof.

$$\begin{aligned}\frac{\partial \bar{f}}{\partial \alpha_i}(\alpha_i^{\text{old}} + \Delta\alpha) &= 1 - y_i \sum_j \alpha_j y_j K(x_j, x_i) - \Delta\alpha y_i y_i K(x_i, x_i) \\ &= 1 - y_i \hat{y}(x_i) - \Delta\alpha K(x_i, x_i) \overset{!}{=} 0\end{aligned}$$

□

Costs for Computing One Partial Gradient

step	from scratch (dual)	from scratch (primal)	from maintained gradient
compute one partial gradient	$O(nm_{\text{nz}})$	$O(m_{\text{nz}})$	$O(1)$
maintain β	—	$O(m_{\text{nz}})$	—
maintain gradient	—	—	$O(nm_{\text{nz}})$
total	$O(nm_{\text{nz}})$	$O(m_{\text{nz}})$	$O(nm_{\text{nz}})$

$$n = \text{number of samples},$$
$$m = \text{number of (primal) attributes},$$
$$m_{\text{nz}} = \text{average number of nonzero attributes}$$

Costs for Computing One Partial Gradient

- Maintaining Gradients is useful when prioritizing coordinates to update (from scratch: $O(n^2 m_{\text{nz}})$).
- Computing gradients from scratch (primal) is only possible for linear SVMs.
- $\rightsquigarrow$ for nonlinear SVMs with prioritized coordinate selection: maintaining gradients saves considerable costs.
- $\rightsquigarrow$ for linear SVMs:
  maintaining gradients does not pay off.

Dual Coordinate Descent Algorithm

```
(1)  learn-linear-svm-coord-descent(training predictors $x$, training targets $y$,
(2)                                 complexity $\gamma$, accuracy $\epsilon$) :
(3)  $\hat{\alpha}_i := 0 \quad \forall i$
(4)  $\hat{\beta}_i := 0 \quad \forall i$
(5)  do
(6)      for $i := 1 \dots n$ in random order do
(7)          $\Delta\alpha_i := [\frac{1 - y_i \hat{\beta}^T x_i}{x_i^T x_i}]^{\gamma - \alpha_i}_{-\alpha_i}$
(8)          $\hat{\alpha}_i := \hat{\alpha}_i + \Delta\alpha_i$
(9)          $\hat{\beta} := \hat{\beta} + \Delta\alpha_i y_i x_i$
(10)     od
(11) while $\exists i : |\Delta\alpha_i| > \epsilon$
(12) return $(\hat{\alpha}, \hat{\beta})$
```

References

Chen, P.H., R.E. Fan, and C.J. Lin (2006): *A study on SMO-type decomposition methods for support vector machines*. In: *IEEE Transactions on Neural Networks* 17.4, pp. 893–908.

Fan, Rong-En, Pai-Hsuen Chen, and Chih-Jen Lin (Dec. 2005): *Working Set Selection Using Second Order Information for Training Support Vector Machines*. In: *J. Mach. Learn. Res.* 6, pp. 1889–1918.

Hsieh, C. J et al. (2008): *A dual coordinate descent method for large-scale linear SVM*. In: *Proceedings of the 25th international conference on Machine learning*, pp. 408–415.

Joachims, Thorsten (1999): *Making large-scale support vector machine learning practical*. In: ACM ID: 299104. Cambridge, MA, USA: MIT Press, pp. 169–184.

— (2006): *Training linear SVMs in linear time*. In: *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*. KDD '06. ACM ID: 1150429. Philadelphia, PA, USA: ACM, pp. 217–226.

Keerthi, S. S. et al. (Mar. 2001): *Improvements to Platt's SMO Algorithm for SVM Classifier Design*. In: *Neural Comput.* 13 (3), pp. 637–649.

Platt, John C. (1999): *Fast training of support vector machines using sequential minimal optimization*. In: *Advances in kernel methods*. Ed. by Bernhard Schölkopf, Christopher J. C. Burges, and Alexander J. Smola. Cambridge, MA, USA: MIT Press, pp. 185–208.

Elementwise Multiplication (Hadamard Product)

Elementwise multiplication is defined as follows:

$$\begin{aligned}
a \odot b &:= (\operatorname{diag}(a)\operatorname{diag}(b))_{i,i} = (a_i b_i)_{i=1,\dots,n}, && a, b \in \mathbb{R}^n\\
A \odot B &:= \qquad\qquad\qquad\qquad\quad = (A_{i,j} B_{i,j})_{i=1,\dots,n, j=1,\dots,m}, && A, B \in \mathbb{R}^{n,m}\\
a \odot B &:= \operatorname{diag}(a) B \qquad\quad\;\; = (a_i B_{i,j})_{i=1,\dots,n, j=1,\dots,m}, && a \in \mathbb{R}^n, B \in \mathbb{R}^{n,m}\\
A \odot b^T &:= A \operatorname{diag}(b) \qquad\quad\; = (A_{i,j} b_j)_{i=1,\dots,n, j=1,\dots,m}, && A \in \mathbb{R}^{n,m}, b \in \mathbb{R}^m
\end{aligned}$$

Commutativity:

$$\begin{aligned}
a \odot b &= b \odot a\\
A \odot B &= B \odot A\\
a \odot B &= (B^T \odot a^T)^T
\end{aligned}$$

But be careful:

$$a \odot B \neq B \odot a, \qquad \text{esp. as the latter is not defined}$$

Note: $(\operatorname{diag}(a))_{i,j} := \delta_{i=j} a_i$ denotes the diagonal matrix with diagonal a.

Elementwise Multiplication (Hadamard Product)

Associativity:

$$
\begin{aligned}
(a \odot b) \odot c =& a \odot (b \odot c)\\
(A \odot B) \odot C =& A \odot (B \odot C)\\
(a \odot B) \odot C =& a \odot (B \odot C)
\end{aligned}
$$

Elementwise Multiplication (Hadamard Product)

Associativity with multiplication (vector/matrix/vector):

$$
\begin{aligned}
a \odot (Bc) &= (a \odot B)c \\
(a^T B) \odot c^T &= a^T (B \odot c^T) \\
b \odot A \odot c^T &= A \odot (bc^T)
\end{aligned}
$$

But not in general (matrix/vector/vector; matrix/matrix/matrix):

$$
\begin{aligned}
A(b \odot c) &\neq (Ab) \odot c \\
A(B \odot C) &\neq (AB) \odot C
\end{aligned}
$$

Matrix/vector/vector is more complicated:

$$A(b \odot c) = (A \odot b^T)c$$