Support Vector Machines
Algorithms

Laura Palagi[1]

[1]Department of Computer, Control, and Management Engineering Antonio Ruberti, Sapienza University of Rome

1st MINOA PhD school
Mixed-Integer Nonlinear Optimization meets Data Science
Ischia (Italy) - June 26, 2019

Recap: constrained formulations

Primal L_1-SVM

$$\begin{aligned} \min \quad & \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{l}\xi_i \\ & y^i\left[w^T x^i + b\right] \geq 1 - \xi_i \\ & \xi_i \geq 0 \end{aligned}$$

Dual L_1-SVM

$$\begin{aligned} \min_{\alpha \in \mathbb{R}^l} \quad & \frac{1}{2}\alpha^T K \alpha - e^T \alpha \\ \text{s.t.} \quad & \alpha^T y = 0 \\ & 0 \leq \alpha \leq C \end{aligned}$$

- decision function

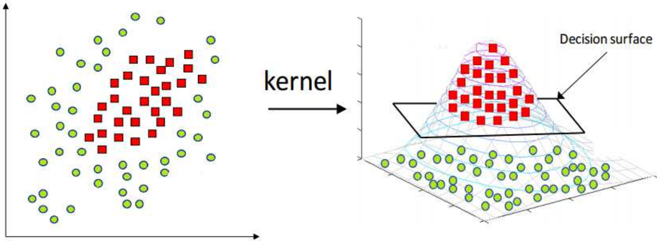
$$f(x) = \mathrm{sgn}\left({w^*}^T x + b^\star\right) = \mathrm{sgn}\left(\sum_{i=1}^{l} \alpha_i^\star y^i {x^T} x^i + b^\star\right).$$

- $K = \{y^i y^j {x^i}^T x^j\}$

Feature Map

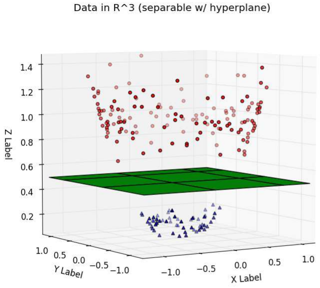
What happens if linear separation is not enough?

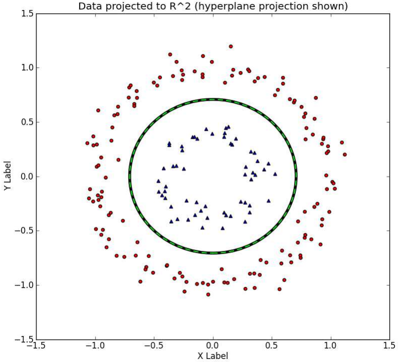
Idea: mapping the data of the input space onto a higher dimensional space called **feature space** and to define a linear classifier in this feature space.



Feature map

A linear separation surface in the feature space is a nonlinear separation surface in the input space





Nonlinear mapping

We map $x \to \Phi(x)$ into a possibly higher dimensional space

$$\phi(x) = [\phi_1(x), \phi_2(x), \dots]^T$$

Look to the primal

$$\begin{aligned} \min \quad & \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{l} \xi_i \\ & y^i\left[w^T \phi(x^i) + b\right] \geq 1 - \xi_i \\ & \xi_i \geq 0 \end{aligned}$$

we need to explicitly know the mapping ϕ.
The size of w is the size of $\phi(x)$, that may be infinite dimensional: how can I compute $\mathrm{sgn}(w^T\phi(x) + b)$?

$$\begin{aligned} \min_{\alpha \in \mathbb{R}^l} \quad & \frac{1}{2}\sum_i \sum_j y^i y^j \phi(x^i)^T \phi(x^j) \alpha_i \alpha_j - e^T \alpha \\ \text{s.t.} \quad & \alpha^T y = 0 \\ & 0 \leq \alpha \leq C \end{aligned}$$

SAPIENZA UNIVERSITÀ DI ROMA

Kernel Trick

Hint: the vectors $\phi(x)$ always appear within an inner product

1. in the dual objective function the elements of Q are of the form $y^i y^j \phi(x^i)^T \phi(x^j)$
2. in the decision function we have

$$f(x) = \mathrm{sgn}({w^*}^T x + b^*) = \mathrm{sgn}(\sum_{i=1}^{l} \alpha_i^* \phi(x^i)^T \phi(x) + b^*)$$

Use **kernel trick** to get back to a **finite** number of variables
It would be enough to have $\phi(x)^T \phi(y)$ in closed form

SAPIENZA
UNIVERSITÀ DI ROMA

Kernel function

Given a set $X \subseteq \Re^n$, a symmetric function

$$K : X \times X \to \Re$$

is a **kernel** if

$$K(x, y) = \phi(x)^T \phi(y) \qquad \forall x, y \in X, \tag{1}$$

where ϕ is an application $X \to \mathcal{H}$ and $\mathcal{H}$ is an Euclidean space

Let $K : X \times X \to \Re$ be a symmetric function. Then K is a kernel if and only if, for any choice of the vectors $x^1, \dots, x^\ell$ in X the Gram matrix

$$K = [K(x_i, x_j)]_{i,j=1,\dots,\ell}$$

is positive semidefinite.

Nonlinear SVM

Using the definition of kernel the dual training problem becomes

$$\begin{aligned} \min_{\alpha} \quad & \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} y^i y^j K(x^i, x^j)\alpha_i\alpha_j - \sum_{i=1}^{l}\alpha_i \\ s.t. \quad & \sum_{i=1}^{l}\alpha_i y^i = 0 \\ & 0 \le \alpha_i \le C \qquad i = 1, \dots, l. \end{aligned} \tag{2}$$

The decision function becomes

$$f(x) = \operatorname{sgn}\left(\sum_{i=1}^{l}\alpha_i^* K(x^i, x) + b^\star\right).$$

SAPIENZA UNIVERSITÀ DI ROMA

Examples of kernels

$x^i \in \Re^3$, $\phi(x^i) \in \Re^{10}$:

$$\phi(x^i) = [1, \sqrt{2}x_1^i, \sqrt{2}x_2^i, \sqrt{2}x_3^i, (x_1^i)^2, (x_2^i)^2, (x_3^i)^2, \sqrt{2}x_1^i x_2^i, \sqrt{2}x_1^i x_3^i, \sqrt{2}x_2^i x_3^i]^T$$

Then $\phi(x^i)^T \phi(x^j) = (1 + {x^i}^T x^j)^2$
Commonly used kernels:

Polynomial kernel $K(x, z) = (x^T z + 1)^p$ (p integer ≥ 1)

Gaussian kernel $K(x, z) = e^{-\|x-z\|^2/2\sigma^2}$ ($\sigma > 0$)

Hyperbolic kernel $K(x, z) = tanh(\beta x^T z + \gamma)$ (for suitable values of β and γ)

Look at new hyper parameters to be tuned !

SAPIENZA UNIVERSITÀ DI ROMA

Gaussian Kernel

$K(x, y)$ can be an inner product in **infinite** dimensional space.
Assume $x \in \mathbb{R}$ and $\gamma > 0$

$$
\begin{aligned}
& e^{-\gamma\|x_i-x_j\|^2} = e^{-\gamma(x_i-x_j)^2} = e^{-\gamma x_i^2+2\gamma x_i x_j-\gamma x_j^2} \\
& = e^{-\gamma x_i^2-\gamma x_j^2}\left(1+\frac{2\gamma x_i x_j}{1!}+\frac{(2\gamma x_i x_j)^2}{2!}+\frac{(2\gamma x_i x_j)^3}{3!}+\ldots\right) \\
& = e^{-\gamma x_i^2-\gamma x_j^2}\left(1 \cdot 1+\sqrt{\frac{2\gamma}{1!}}x_i \cdot \sqrt{\frac{2\gamma}{1!}}x_j+\sqrt{\frac{(2\gamma)^2}{2!}}x_i^2 \cdot \sqrt{\frac{(2\gamma)^2}{2!}}x_j^2\right. \\
& \left.+\sqrt{\frac{(2\gamma)^3}{3!}}x_i^3 \cdot \sqrt{\frac{(2\gamma)^3}{3!}}x_j^3+\ldots\right) = \phi(x^i)^T\phi(x^j)
\end{aligned}
$$

where

$$
\phi(x) = e^{-\gamma x^2}\left[1, \sqrt{\frac{2\gamma}{1!}}x, \sqrt{\frac{(2\gamma)^2}{2!}}x^2, \sqrt{\frac{(2\gamma)^3}{3!}}x^3, \ldots\right]^T
$$

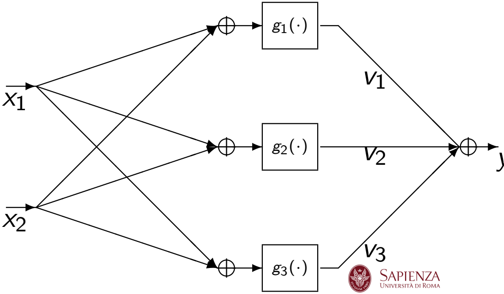
SVM and RBF networks

Gaussian kernel $K(x, z) = e^{-\|x-z\|^2/2\sigma^2}$ $(\sigma > 0)$.
The decision function is:

$$f_d(x) = \mathrm{sgn}\left(\sum_{i=1}^{l} \lambda_i^* y^i e^{-\|x-x^i\|^2/2\sigma^2}\right)$$

the output of a **shallow RBF network** where the number of neurons and centers are the SVs

$$g_i(x) = e^{-\|x-c_i\|^2/2\sigma^2}$$



SAPIENZA
UNIVERSITÀ DI ROMA

Training Problems

Training a SVM amounts to solve either the primal problem (huge number of constraints) or the dual (huge number of variables)

Primal L_1-(unbiased) SVM

$$\min \quad \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{l}\xi_i$$
$$y^i\left[w^T x^i + b\right] \geq 1 - \xi_i$$
$$\xi_i \geq 0$$

Dual L_1-(unbiased) SVM

$$\min_{\alpha\in\mathbb{R}^l} \quad \frac{1}{2}\alpha^T K\alpha - e^T\alpha$$
$$\text{s.t.} \quad \alpha^T y = 0$$
$$0 \leq \alpha \leq C$$

Two Loop optimization

- hyperparameters choice C & kernel's parameters (heuristic)
- parameter optimization w, b (primal) or α (dual) (exact)

Some example of joint selection with Gaussian Kernel involving MINLP [3].

Solving the dual

Consider the convex quadratic programming problem for SVM training in the case of classification problems:

$$\begin{aligned} \min_{\alpha} \quad f(\alpha) = \quad & \frac{1}{2}\alpha^T Q\alpha - e^T\alpha \\ s.t. \quad & y^T\alpha = 0 \\ & 0 \leq \alpha \leq C, \end{aligned} \tag{3}$$

where Q is a $l \times l$ symmetric and positive semidefinite matrix, $e \in \Re^l$ is the vector of ones, $y \in \{-1, 1\}^l$, and C is a positive scalar.
The Hessian matrix Q is dense, cannot be fully stored so that standard methods for quadratic programming cannot be used.

Optimality conditions

Thanks to the special structure of the constraints the KKT conditions can be written ia a very compact form

KKT conditions

A feasible point α^* is a global solution iff

$$\max_{i \in R(\alpha^\star)} \left\{ -\frac{(\nabla f(\alpha^\star))_i}{y_i} \right\} \leq \min_{j \in S(\alpha^\star)} \left\{ -\frac{(\nabla f(\alpha^\star))_j}{y_j} \right\}. \qquad (4)$$

$$R(\alpha) = \{i : \ (\alpha_i = 0, \& y_i = 1), (\alpha_i = C, \& y_i = -1), \ (0 < \alpha_i < C)\}$$

$$S(\alpha) = \{i : \ (\alpha_i = 0, \& y_i = -1), (\alpha_i = C, \& y_i = 1), \ (0 < \alpha_i < C)\},$$

It is equivalent to state that α^* is a global solution iff $\nexists$ a feasible and descent direction in α^*, i.e.

$$0 \leq \min_{d \text{ feasible in } \alpha^*} \nabla f(\alpha^*)^T d$$

SAPIENZA UNIVERSITÀ DI ROMA

From optimality conditions to sparse algorithms

Given a current estimate α^k (not KKT), a (conditional) gradient method takes a step along a d solving the LP

$$\begin{array}{ll} \min & \nabla f(\alpha^K)^T d \\ & d \text{ feasible in } \alpha^k \end{array}$$

The direction is NOT sparse: heavy update of ∇f and f

$$\begin{array}{ll} \min & \nabla f(\alpha^k)^T d \\ & d \text{ feasible in } \alpha^k \\ & d \text{ sparse} \end{array}$$

Decomposition methods

Choosing sparse d amounts changing only few components $i \in W^k \subset \{1, \dots, l\}$ of α

Decomposition Methods

The vector of variables α^k is partitioned into two subvectors $(\alpha^k_W, \alpha^k_{\overline{W}})$, where the **working set** $W \subset \{1, \dots, l\}$ identifies the variables to be updated and $\overline{W} = \{1, \dots, l\} \setminus W$.

Use the update

$$\alpha^{k+1} = \begin{cases} \alpha^*_W, \\ \alpha^k_{\overline{W}} \end{cases}$$

where

$$\begin{aligned} \alpha^*_W = \arg\min_{\alpha_W} \quad & f(\alpha_W, \alpha^k_{\overline{W}}) \\ & y_W^T \alpha_W = -y_{\overline{W}}^T \alpha^k_{\overline{W}} \\ & 0 \leq \alpha_W \leq C. \end{aligned}$$

Practical choices

$$\boxed{\text{Sparsity } \|d\|_0 = |W^k| = q \geq 2}$$

q must be **greater than or equal to** 2, due to the presence of the constraint $y^T\alpha = 0$

Saving in gradient update

$$\nabla f(\alpha^{k+1}) = \nabla f(\alpha^k) + Q\left(\alpha^{k+1} - \alpha^k\right) = \nabla f(\alpha^k) + \sum_{i \in W^k} Q_i(\alpha_i^{k+1} - \alpha_i^k)$$

Starting from the feasible $\alpha^0 = 0$ allow iterative update from $\nabla f(\alpha^0) = -e$

The full matrix Q is never used

SAPIENZA UNIVERSITÀ DI ROMA

Choice of the working set

Working set

The selection rule of W^k strongly affects convergence and speed of the algorithm

Manage a trade-off

- **Sequential Minimal Optimization** (SMO) algorithms, where $q = 2$;
- **General Decomposition Algorithms**, where $q > 2$ (around 10 in standard implementation SVM$^{\text{light}}$).

SMO-MVP

At each iteration k, in a SMO algorithm a quadratic subproblem of dimension 2 must be solved, and it is done **analitically** whch is equivalent to move along a feasible and descent directions having only two nonzero elements.

How do we find such sparse direction ?

From the violated KKT

$$\max_{i \in R(\alpha^k)} \left\{ -\frac{(\nabla f(\alpha^k))_i}{y_i} \right\} > \min_{j \in S(\alpha^k)} \left\{ -\frac{(\nabla f(\alpha^k))_j}{y_j} \right\}.$$

A **violating pair** $i \in R(\alpha^k)$, $j \in S(\alpha^k)$:

$$\left\{ -\frac{(\nabla f(\alpha^k))_i}{y_i} \right\} > \left\{ -\frac{(\nabla f(\alpha^k))_j}{y_j} \right\}$$

gives a descent direction.

Selection of a simple violating pairs is not sufficient to guarantee convergence.

Maximal Violating Pair

A convergent SMO algorithm can be defined using pairs of indices that most violates the optimality conditions.

A **maximal violating pair** $i \in I(\alpha)$, $j \in J(\alpha)$ with

$$I(\alpha) = \left\{ i : \; i \in \arg \max_{i \in R(\alpha)} \left\{ -\frac{(\nabla f(\alpha))_i}{y_i} \right\} \right\}$$

$$J(\alpha) = \left\{ j : \; j \in \arg \min_{j \in S(\alpha)} \left\{ -\frac{(\nabla f(\alpha))_j}{y_j} \right\} \right\}$$

corresponds to select a direction solving

$$\begin{array}{ll} \min & \nabla f(\alpha^k)^T d \\ & d \text{ feasible in } \alpha^k \\ & \|d\|_0 = 2 \end{array}$$

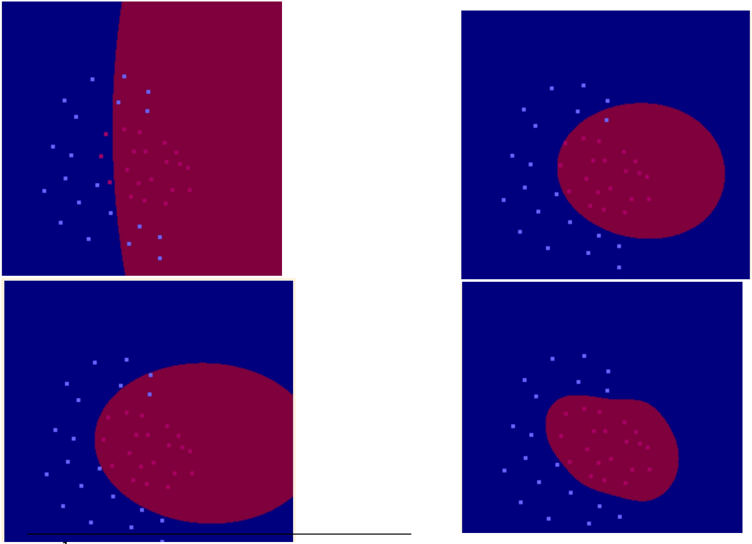
SMO-MVP

- **Inizialization.** Set $\alpha^0 = 0$ $\nabla f(\alpha^0) = -e$, $k = 0$.
- **While** $\Big($ the stopping criterion is not satisfied $\Big)$
  1. select $i \in I(\alpha^k)$, $j \in J(\alpha^k)$, and set $W = \{i, j\}$;
  2. compute analytically a solution $\alpha^* = \begin{pmatrix} \alpha_i^\star & \alpha_j^\star \end{pmatrix}^T$
  3. set $\alpha_{i,j}^{k+1} = \alpha_{i,j}^*$
  4. set $\nabla f(\alpha^{k+1}) = \nabla f(\alpha^k) + \sum_{i,j}(\alpha_h^{k+1} - \alpha_h^k)Q_h$;
  5. set $k = k + 1$.
- **end while**
- **Return** α^k

(Implemented in LIBSVM)

The two loops stage

Setting hyperparameters: C & γ: a toy example[1]



[1] Graphic Interface on https:www.csie.ntu.edu.tw̃cjlinlibsvm

Unbiased SVM $b = 0$

$$\min_{\lambda \in \mathbb{R}^l} \quad \frac{1}{2}\lambda^T K \lambda - e^T \lambda$$

$$\text{s.t.} \quad 0 \leq \lambda \leq C$$

The dual has only box constraints, and the cardinality of the working set can be set equal to 1 !

Coordinate descent

- select a component i holding all components $\alpha_j^{k+1} = \alpha_j^k$, $j \neq i$
- solve in closed form

$$\alpha_i^{k+1} = \min\left\{ C, \max\left\{ 0, \alpha_i^k - \frac{\nabla_i f(\alpha^k)}{Q_{ii}} \right\} \right\}$$

- easy trick for efficient gradient update for linear SVM (memorize intermediate $w = \sum \lambda_i^* y^i x^i$)
- Accuracy reached fast

Implemented in Liblinear

Primal algorithms

- Intuitively, kernel should give superior accuracy than linear. Roughly speaking, from the Taylor expansion of the Gaussian (RBF) kernel, linear SVM is a special case of RBF-kernel SVM
- Dual solution often not sparse (many support vectors)
- for some problems, accuracy by linear is as good as nonlinear, but training and testing are much faster
- Primal algorithms reach approximate solution faster [2]
- Lose the kernel. However the **representer theorem** which states that the optimal decision function can be written as a linear combination of kernel functions evaluated at the training samples allow to recover non linearities.

Cutting Plane Methods

Primal formulation with $b = 0$

$$\begin{array}{ll} \min\limits_{w,\xi} & \frac{1}{2}\|w\|^2 + \frac{C}{l}\sum_{i=1}^{l}\xi_i \\ \text{s.t.} & y^i\left[w^T x^i\right] - 1 + \xi_i \geq 0 \qquad i = 1,\dots,l \\ & \xi_i \geq 0 \qquad i = 1,\dots,l. \end{array}$$

Equivalent formulation: the Structural Classification SVM (SVM^{struct} [4])

$$\begin{array}{ll} \min\limits_{w,\xi} & \frac{1}{2}\|w\|^2 + C\xi \\ \text{s.t.} & \frac{1}{l}w^T\sum_{i=1}^{l}c_i y^i x^i \geq \frac{1}{l}\sum_{i=1}^{l}c_i - \xi. \ \forall \mathbf{c} \in \{0,1\}^l \\ & \xi \geq 0 \end{array}$$

It has an exponential number of constraints, BUT only one slack variable that is directly related to the infeasibility. If (w, ξ) satisfies all the constraints with precision ϵ, then the point $(w, \xi + \epsilon)$ is feasible.

Cutting Plane Algorithm

- **Inizialization.** $\mathcal{W} = \emptyset$.
- **Repeat**
  1. update (w, ξ) with the solution of

$$\begin{aligned} \min \quad & \frac{1}{2}\|w\|^2 + C\xi \\ \text{s.t.} \quad & \forall \mathbf{c} \in \mathcal{W} : \frac{1}{l} w^T \sum_{i=1}^{l} c_i y^i x^i \geq \frac{1}{l} \sum_{i=1}^{l} c_i - \xi \end{aligned} \tag{5}$$

  2. **for** $i = 1, \ldots, l$

$$c_i = \begin{cases} 1 & \text{if } y^i w^T x^i < 1 \\ 0 & \text{otherwise.} \end{cases}$$

  end for

  3. set $\mathcal{W} = \mathcal{W} \cup \{\mathbf{c}\}$.
- **Until** $\Big($ accuracy reached $\Big)$
- **Return** (w, ξ)

Unconstrained Formulations

Different unconstrained formulation of the primal problem can be defined:

$$\min_{w,b} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{l} \max\{0, 1 - y^i(w^T x^i + b)\} \qquad L_1\text{-SVM.}$$

$$\min_{w,b} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{l} \max^2\{0, 1 - y^i(w^T x^i + b)\} \qquad L_2\text{-SVM}$$

Another possibility is to replace the constraints $y^i(w^T x^i + b) \geq 1 - \xi^i$, by the equality constraints $y^i(w^T x^i + b) = 1 - \xi^i$. This leads to a regularized linear least squares problem

$$\min_{w,b} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{l} (y^i(w^T x^i + b) - 1)^2. \qquad \text{LS-SVM}$$

Unconstrained Formulations

The general unconstrained formulation takes the form

$$\min_{w,b} R(w,b) + C\sum_{i=1}^{l} L(w,b;x^i,y^i), \qquad (6)$$

where $R(w,b)$ is the **regularization term** and $L(w,b;x^i,y^i)$ is the **loss function** associated with the observation (x^i,y^i).
For nonlinear SVM the **representer theorem** is used, that amounts to set $w = \sum_{i=1}^{l} \beta_i \phi(x^i)$. As an example, the optimization problem corresponding to L_2-SVM is

$$\min_{\beta,b} \frac{1}{2}\beta^T K\beta + C\sum_{i=1}^{l} \max{}^2\{0, 1 - y^i\beta^T K_i\},$$

where K is the kernel matrix associated to the mapping ϕ and K_i is the $i-$th column.

Unconstrained Methods

Primal method the non smooth formulation L_1-SVM $(b = 0)$

$$\min_{w \in \mathbb{R}^n} \frac{\lambda}{2}\|w\|^2 + \sum_{i=1}^{l} \max\left\{0, 1 - y^i w^T x^i\right\}$$

$$v^k(i) = \partial_w \left(\max\left\{0, 1 - y^i {w^k}^T x^i\right\}\right) = \begin{cases} 0, & \text{if } 1 - y^i {w^k}^T x^i \leq 0 \\ -y^i x^i, & \text{otherwise.} \end{cases}$$

Pegasos is a stochastic sub-gradient method [6]

Stochastic Subgradient for L_1-SVM

Stochastic Subgradient
Set $w^1 = 0$

- **For** $k = 1, 2, \dots$
- Pick $i \in \{1 \dots, l\}$ uniformly at random
- Set $\partial_w f(w^k) = \lambda w^k + v^k(i)$
- Update

$$w^{k+1} = w^k - \frac{1}{k\lambda} \partial_w f(w^k)$$

- **Until** (stopping criterion)
- Outout w^k

Conclusion

Many others algorithms (Interior point, second order semismooth etc)[5, 1]

Optimization is very useful for machine learning

Machine learning knowledge must be exploited in designing effective optimization algorithms and software

SAPIENZA
UNIVERSITÀ DI ROMA

References (incomplete !)

E. Carrizosa and D. R. Morales.
Supervised classification and mathematical optimization.
Computers & Operations Research, 40(1):150–165, 2013.

O. Chapelle.
Training a support vector machine in the primal.
Neural computation, 19(5):1155–1178, 2007.

M. Fischetti.
Fast training of support vector machines with gaussian kernel.
Discrete Optimization, 22:183–194, 2016.

T. Joachims, T. Finley, and C.-N. J. Yu.
Cutting-plane training of structural svms.
Machine Learning, 77(1):27–59, 2009.

V. Piccialli and M. Sciandrone.
Nonlinear optimization and support vector machines.
4OR, 16(2):111–149, 2018.

S. Shalev-Shwartz, Y. Singer, N. Srebro, and A. Cotter.
Pegasos: Primal estimated sub-gradient solver for svm.
Mathematical programming, 127(1):3–30, 2011.