

Classification in High Dimension

Tristan Mary-Huard
maryhuar@agroparistech.fr

Génétique Quantitative et Évolution - Le Moulon INRA/Univ. Paris
Sud/CNRS/AgroParisTech
AgroParisTech/INRA UMR 518



Overview

1 Introduction

- Basics in optimization
- Basics in classification

2 Logistic regression

- Classical logistic regression
- Regularized logistic regression

3 Support Vector Machines

- Linear SVM
- Kernel SVM

4 Theoretical guarantees

Prerequisites

"You know nothing, John Snow."

V.Vapnik

~~V.Koltchinskii~~

Traditional wildling saying

Overview

1 Introduction

- Basics in optimization
- Basics in classification

2 Logistic regression

- Classical logistic regression
- Regularized logistic regression

3 Support Vector Machines

- Linear SVM
- Kernel SVM

4 Theoretical guarantees

Basics in optimization

I - Theoretical aspects

An Introduction to Optimization [CZ13]

Convex Optimization [BV04]

(a.k.a. the convex surrogate of the Bible)

Standard optimization problem

Standard problem

$$\min_{x \in \Omega} f(x)$$

with $f : \mathbb{R}^p \rightarrow \mathbb{R}$ differentiable, and $\Omega \subset \mathbb{R}^p$.

Definition

x^* is a local minimizer iff

$$\exists \varepsilon / \forall x \in B(x^*, \varepsilon) \cap \Omega, f(x) \geq f(x^*)$$

x^* is a global minimizer iff

$$\forall x \in \Omega, f(x) \geq f(x^*)$$

First order necessary condition

Admissible direction

$d \in \mathbb{R}^p$ is admissible at point x if

$$\exists \alpha_0 > 0 / \forall \alpha \in [0, \alpha_0], x + \alpha d \in \Omega.$$

The directional derivative w.r.t. d is defined as

$$\frac{\partial f(x)}{\partial d} = \lim_{\alpha \rightarrow 0} \frac{f(x + \alpha d) - f(x)}{\alpha} = d^T \nabla f(x)$$

Theorem (1st order necessary condition)

If f is C^1 and x^ is a local minimizer of f over Ω . Then for all d admissible at point x^* ,*

$$d^T \nabla f(x^*) \geq 0$$

Note : If x^* is an interior point of Ω , then NC $\Rightarrow \nabla f(x^*) = 0$.

Convex optimization problems

Convex set, convex function

Ω is convex if $\forall (x, y, \lambda) \in \Omega^2 \times [0, 1]$,

$$\lambda x + (1 - \lambda)y \in \Omega$$

f is convex if $\forall (x, y, \lambda) \in \mathbb{R}^p \times \mathbb{R}^p \times [0, 1]$,

$$f(\lambda x + (1 - \lambda)y) \leq \lambda f(x) + (1 - \lambda)f(y)$$

Proposition

- If f is convex, any local minimizer is a global minimizer.
- If f is convex and differentiable,

$$f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle, \quad \forall x, y$$

Convex Optimization problems

Standard problem

$$\min_{x \in \Omega} f(x)$$

with $f: \mathbb{R}^p \rightarrow \mathbb{R}$ differentiable, and $\Omega \in \mathbb{R}^p$.

Theorem

Assume f is convex and differentiable, and Ω is convex. Then $x^* \in \Omega$ is a global minimizer iff

$$\langle \nabla f(x^*), y - x^* \rangle \geq 0, \quad \forall y$$

Primal optimization problem

Consider problem

$$\begin{aligned} & \text{minimize} && f(x) \\ & \text{subject to} && g_i(x) \leq 0, \quad \forall i = 1, \dots, m \end{aligned}$$

New objective function :

$$\begin{aligned} f(x) + \sum_{i=1}^m \max_{\lambda_i \geq 0} \lambda_i g_i(x) &= \max_{\lambda \geq 0} \left\{ f(x) + \sum_{i=1}^m \lambda_i g_i(x) \right\} \\ &= \max_{\lambda \geq 0} L(x, \lambda) \end{aligned}$$

$\lambda_1, \dots, \lambda_m$: Lagrange multipliers,

$L(\cdot, \cdot)$: Lagrange function.

The initial optimization problem becomes

$$\min_x \max_{\lambda \geq 0} L(x, \lambda) \quad (\mathcal{P})$$

Dual optimization problem

Alternatively, consider problem

$$\max_{\lambda \geq 0} \min_x L(x, \lambda) \quad (\mathcal{D})$$

(\mathcal{D}) is the *dual* problem associated with *primal* problem (\mathcal{P}) .

Note $G(\cdot)$ the dual function

$$G(\lambda) = \min_x L(x, \lambda)$$

Proposition

For all $\lambda \geq 0$, one has

$$G(\lambda) \leq p^*$$

where $p^* = f(x^*)$

Duality gap

Definition

Note $d^* = \max_{\lambda \geq 0} G(\lambda)$ the solution of (\mathcal{D}) . Then

$$p^* - d^* \geq 0$$

is called the duality gap.

If $p^* - d^* = 0$, then we say that **strong duality holds**.

Questions

- How does strong duality help ?
- When does strong duality hold ?

Complementary slackness conditions

Proposition

If strong duality holds, then

$$\lambda_i^* g_i(x^*) = 0, \quad \forall i = 1, \dots, m$$

where $\lambda^* = \arg \max_{\lambda \geq 0} G(\lambda)$.

Also note that x^* is the minimizer of $L(x, \lambda^*)$, therefore

$$\nabla L(x^*, \lambda^*) = 0$$

Karush Kuhn Tucker conditions

Proposition

If strong duality holds, the optimal Lagrange multiplier vector λ^* and the optimal solution x^* of (\mathcal{P}) satisfy

$$g_i(x^*) \leq 0, \quad \forall i = 1, \dots, m \quad (\text{primal feasibility})$$

$$\lambda_i^* \geq 0, \quad \forall i = 1, \dots, m \quad (\text{dual feasibility})$$

$$\lambda_i^* g_i(x^*) = 0, \quad \forall i = 1, \dots, m \quad (\text{compl. slackness})$$

$$\nabla L(x^*, \lambda^*) = \nabla f(x^*) + \sum_{i=1}^m \lambda_i^* \nabla g_i(x^*) = 0, \quad (\text{first order condition})$$

Strong duality does not hold in general, but holds under mild conditions for convex optimization problems...

Slater's constraint qualification

Proposition

Consider problem (\mathcal{P}) where f, g_1, \dots, g_m are convex functions. Then strong duality holds if there exists a strictly feasible point, satisfying

$$g_i(x) < 0, \quad \forall i = 1, \dots, m$$

Proof : Technical !

See [BV04]

Proposition

Assume (\mathcal{P}) is convex. Then if (λ^*, x^*) satisfy the KKT conditions, strong duality holds and (λ^*, x^*) is optimal.

So far...

Convex + differentiability

If f, g_1, \dots, g_m are differentiable and convex, then the KKT conditions are necessary and sufficient for optimality.

Potential use

- ★ Solve analytically the KKT conditions,
- ★ Guidelines for the development of efficient algorithms,
- ★ Solve the dual rather than the primal when easier !

Limitation

Some objective functions (hinge loss) and/or constraints (L_1 norm) are convex but not differentiable...

Subdifferential and subgradients

Recall that for a convex, differentiable function f

$$\forall x, y \quad f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle$$

Definition

Let $f: \mathbb{R}^p \rightarrow \mathbb{R}$. ω_x is a subgradient of f at point x if

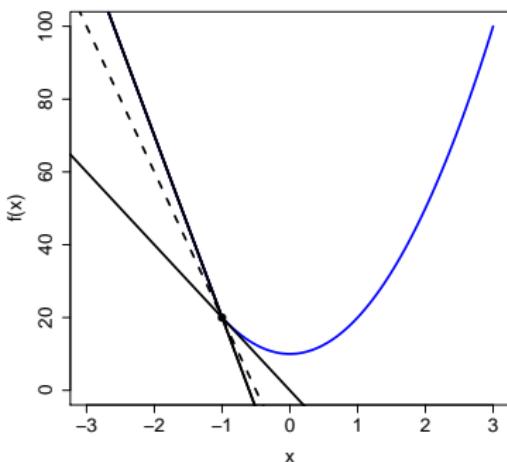
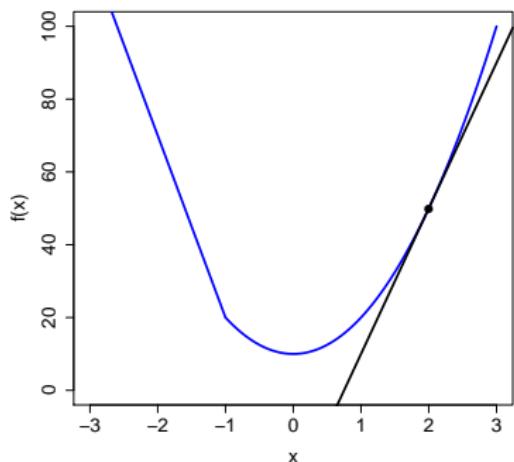
$$\forall x, y \quad f(y) \geq f(x) + \langle \omega_x, y - x \rangle$$

The set

$$\partial f(x) = \{ \omega \mid \forall y \quad f(y) \geq f(x) + \langle \omega, y - x \rangle \}$$

is called the subdifferential of f at point x

A graphical illustration



At $x = 2$ the function is differentiable

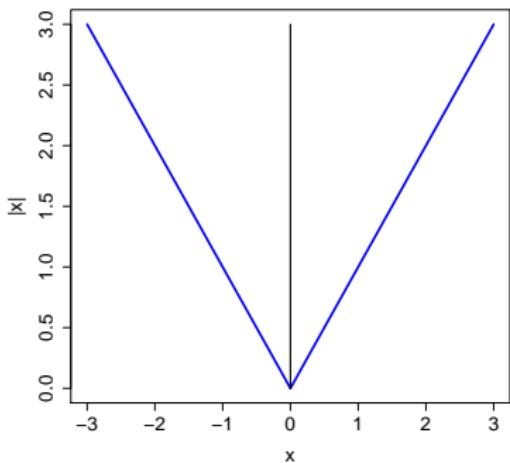
⇒ a unique tangent hyperplane

At $x = -1$ the function is not differentiable

⇒ many "lower" hyperplanes !

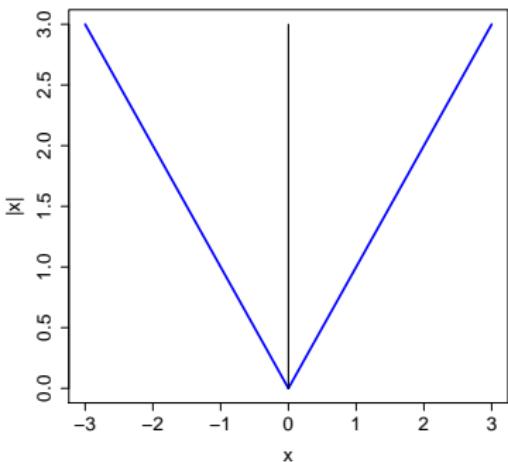
Subdifferential for the L_1 norm

$$\partial|x| =$$



Subdifferential for the L_1 norm

$$\partial|x| = \begin{cases} \text{sign}[x] & \text{if } x \neq 0, \\ [-1, 1] & \text{if } x = 0. \end{cases}$$



$$\partial||x||_1 = \{\omega \in \mathbb{R}^p / \omega_j = \text{sign}[x_j] \text{ if } x_j \neq 0, \omega_j \in [-1, 1] \text{ if } x_j = 0\}$$

Subdifferential and subgradients

Subdifferential and convexity

- ★ f is convex $\Rightarrow \partial f(x)$ is non-empty, $\forall x$,
- ★ f is convex and differentiable at $x \Rightarrow \partial f(x) = \{\nabla f(x)\}$.

Proof : See [Gir14]

Theorem

Assume f is convex and non-differentiable. Then

$$x^* = \arg \min_x f(x) \Leftrightarrow 0 \in \partial f(x^*)$$

KKT conditions revisited

Consider

$$\text{minimize } f(x)$$

$$\text{subject to } g_i(x) \leq 0, \forall i = 1, \dots, m$$

where f, g_1, \dots, g_m are convex but not differentiable everywhere.

Proposition

If strong duality holds, then necessary and sufficient conditions for primal and dual optimality of (λ^*, x^*) are

$$g_i(x^*) \leq 0, \quad \forall i = 1, \dots, m \quad (\text{primal feasibility})$$

$$\lambda_i^* \geq 0, \quad \forall i = 1, \dots, m \quad (\text{dual feasibility})$$

$$\lambda_i^* g_i(x^*) = 0, \quad \forall i = 1, \dots, m \quad (\text{compl. slackness})$$

$$0 \in \partial L(x^*, \lambda^*) = \partial f(x^*) + \sum_{i=1}^m \lambda_i^* \partial g_i(x^*), \quad (\text{first order condition})$$

Proof : Follows the same line as for the differentiable case.

Basics in optimization

II - Algorithm(s)

From theory to practice

Back to the unconstrained optimization problem

$$\min_x f(x)$$

If f is differentiable, then $\forall \alpha, d, \|d\|_2 = 1$:

$$\begin{aligned} f(x + \alpha d) &= f(x) + \alpha \nabla f(x)^T d + o(\alpha) \\ \Rightarrow |f(x + \alpha d) - f(x)| &\approx \alpha |\nabla f(x)^T d| \\ &\leq \alpha \|\nabla f(x)\|_2 \end{aligned}$$

Best direction : $-\frac{\nabla f(x)}{\|\nabla f(x)\|_2}$!

Gradient descent algorithm

Iterative procedure

$$\text{for } t = 1, \dots, T \quad x^{(t+1)} = x^{(t)} - \alpha_t \nabla f(x^{(t)})$$

$\alpha_t > 0$: step size parameter

Main difficulty : choice of $(\alpha_t)_t$.

- ★ constant stepsize,
- ★ decreasing stepsize,
- ★ "best" stepsize (a.k.a. steepest descent).

Both the convergence rate *and* the complexity depend on $(\alpha_t)_t$.

Example : Steepest gradient descent

Algorithm

Input x_0, ε

while $\left\| \nabla f(x^{(t)}) \right\| \geq \varepsilon$,

$$x^{(t+1)} = x^{(t)} - \alpha_t \nabla f(x^{(t)})$$

$$\text{where } \alpha_t = \arg \min_{\alpha > 0} f\left(x^{(t)} - \alpha \nabla f(x^{(t)})\right) \quad (1)$$

end

Properties

(i) $f(x^{(t+1)}) \leq f(x^{(t)})$ (descent property),

(ii) $\langle \nabla f(x^{(t+1)}), \nabla f(x^{(t)}) \rangle > 0$ (orthogonal directions),

(iii) If f is C^1 and strictly convex, then $(x^{(t)})$ converges to x^* .

Proof of (iii) : Technical ! See [CZ13].

Limitations

★ Solving (1) may be non-trivial

★ May be slow (see Accelerations, e.g. [N⁺07])

Alternative formulation of the gradient descent

Initial formulation

$$\text{At step } t+1, \quad x^{(t+1)} = x^{(t)} - \alpha_t \nabla f(x^{(t)})$$

Recasted as

$$x^{(t+1)} = \arg \min_x \left\{ f(x^{(t)}) + \langle \nabla f(x^{(t)}), x - x^{(t)} \rangle + \frac{1}{2\alpha_t} \|x - x^{(t)}\|_2^2 \right\}$$

Interpretation

- ★ $f(x^{(t)}) + \langle \nabla f(x^{(t)}), x - x^{(t)} \rangle$: linearization of f around $x^{(t)}$,
- ★ $\|x - x^{(t)}\|_2^2$: requires $x^{(t+1)}$ to be "not to far" from $x^{(t)}$,
- ★ α_t : rules the tradeoff.

Proximal gradient descent

$$\min_x f(x) + h(x)$$

f convex and differentiable (e.g. L_2 loss),
 h convex but non differentiable (e.g. L_1 norm).

Linearize the differentiable part to obtain :

$$x^{(t+1)} = \arg \min_x \left\{ f(x^{(t)}) + \langle \nabla f(x^{(t)}), x - x^{(t)} \rangle + h(x) + \frac{1}{2\alpha_t} \|x - x^{(t)}\|_2^2 \right\}$$

Proximal operator

$$\text{prox}_h(\theta) = \arg \min_z \left\{ \frac{1}{2} \|\theta - z\|_2^2 + h(z) \right\}$$

In practice

- 1/ Compute the classical gradient step $x^{(t)} - \alpha_t \nabla f(x^{(t)})$,
- 2/ project according to the proximal operator

$$x^{(t+1)} = \text{prox}_{\alpha_t h} \left(x^{(t)} - \alpha_t \nabla f(x^{(t)}) \right)$$

Application I : projected gradient descent

If minimization is subject to constraint $x \in \Omega \subsetneq \mathbb{R}^p$:

$$\begin{aligned} x^{(t+1)} &= \arg \min_{x \in \Omega} \left\{ f(x^{(t)}) + \langle \nabla f(x^{(t)}), x - x^{(t)} \rangle + \frac{1}{2\alpha_t} \|x - x^{(t)}\|_2^2 \right\} \\ &= \arg \min_x \left\{ f(x^{(t)}) + \langle \nabla f(x^{(t)}), x - x^{(t)} \rangle + \frac{1}{2\alpha_t} \|x - x^{(t)}\|_2^2 + I_\Omega(x) \right\} \end{aligned}$$

where $I_\Omega(x) = \begin{cases} 0 & \text{if } x \in \Omega, \\ +\infty & \text{otherwise.} \end{cases}$

In practice

- 1/ Compute the classical gradient step $x^{(t+1)} = x^{(t)} - \alpha_t \nabla f(x^{(t)})$,
- 2/ Project on Ω

$$x_{pr}^{(t+1)} = \Pi_\Omega(x^{(t+1)}).$$

Fast if projection can be easily computed...

Application II : projected gradient descent for lasso regression

$$x^{(t+1)} = \arg \min_x \left\{ f(x^{(t)}) + \langle \nabla f(x^{(t)}), x - x^{(t)} \rangle + \frac{1}{2\alpha_t} \|x - x^{(t)}\|_2^2 + \lambda \|x\|_1 \right\}$$

In practice

- 1/ Compute the classical gradient step $x^{(t+1)} = x^{(t)} - \alpha_t \nabla f(x^{(t)})$,
- 2/ Apply soft-thresholding to $x^{(t+1)}$

$$x_{pr,j}^{(t+1)} = \text{sign} \left[x^{(t+1)_j} \right] \times \left| \left| x^{(t+1)_j} \right| - \alpha_t \lambda \right|_+$$

Fast, easy, and amenable to parallelization.

Beyond first order algorithms

$$\min_x f(x)$$

f convex and twice differentiable

Newton algorithm

★ Consider 2nd order Taylor expansion of f :

$$\begin{aligned} f(y) &= f(x) + \langle \nabla f(x), y - x \rangle + \frac{1}{2}(y - x)^T H_f(x)(y - x) + o\left(\|y - x\|_2^2\right) \\ &= Q_x(y) + o\left(\|y - x\|_2^2\right) \end{aligned}$$

★ At step $t + 1$, use $Q_{x^{(t)}}$ as a proxy for f ...

$$x^{(t+1)} = \arg \min_x Q_{x^{(t)}}(x)$$

★ ... and get the (closed form) solution :

$$x^{(t+1)} = x^{(t)} - H_f(x^{(t)})^{-1} \nabla f(x^{(t)})$$

Take home message

Theoretical aspects

- ★ Mostly interested in convex problems,
- ★ Characterization of the solution(s),
- ★ Guidelines to derive efficient algorithms.

Gradient descent

- ★ Simple but quite versatile,
- ★ Can be generalized in many ways,
- ★ More suited to deal with large problems than Newton method (more on this latter).

Non-addressed points

- ★ Complexity of the different algorithms
- ★ Rates of convergence
- ★ Convexity vs strong convexity, smoothness, etc.

Basics in classification

Elements of Statistical Learning [FHT01]

A Probabilistic Theory of Pattern Recognition [DGL13]

Supervised classification

Goal

Predict the unknown label Y of an observation X .

- $Y \in \mathcal{Y}$ where $\mathcal{Y} = \{0, 1\}$ or $\mathcal{Y} = \{-1, 1\}$ (binary classif.),
- $X \in \mathcal{X} (= \mathbb{R}^p)$.

Supervision

$\mathbb{P}_{X,Y}$ is unknown.

Training set : $\mathcal{D}_n = (X_1, Y_1), \dots, (X_n, Y_n)$, where $(X_i, Y_i) \stackrel{i.i.d.}{\hookrightarrow} \mathbb{P}_{X,Y}$.

Classifier

One aims at building

$$\begin{aligned}\hat{h}: \mathcal{X} &\rightarrow \mathcal{Y} \\ X &\rightarrow \hat{Y}\end{aligned}$$

Some examples

Cancer prediction

Predict cancer grade (from 1 to 3) based on CNV.

★ $X_i = (X_{i1}, \dots, X_{ip})$, where

X_{ij} = Nb of copies of chrom. segment j in ind. i .

★ $\mathcal{X} = \mathbb{R}^p$

★ $\mathcal{Y} = \{1, 2, 3\}$

Credit scoring

Predict loan reimbursement based on social/economics/health measurements.

★ $X_i = (X_{i1}, \dots, X_{i3})$, where

X_{i1} = gross salary of ind. i ,

$X_{i2} \in \{1, \dots, K\}$ = socio-professional category of ind. i ,

$X_{i3} = 1$ if ind. i already has an ongoing loan, 0 otherwise.

★ $\mathcal{X} = \mathbb{R} \times \{1, \dots, K\} \times \{0, 1\}$

★ $\mathcal{Y} = \{"safe", "risky"\}$

Classification algorithms

Any strategy

$$\begin{aligned}\mathcal{A} : \bigcup_{n \geq 1} \{\mathcal{X} \times \mathcal{Y}\}^n &\rightarrow \mathcal{Y}^{\mathcal{X}} \\ \mathcal{D}_n &\mapsto \hat{h}\end{aligned}$$

defines a classification algorithm.

A few examples

- Discriminant analysis
- k NN
- Logistic regression
- Neural networks
- SVM
- CART & Random forest
- Boosting/bagging
- ...

Performance assessment

Quality of a classifier

$$L(\hat{h}) = \mathbb{P}(\hat{h}(X) \neq Y \mid \mathcal{D}_n) = \mathbb{E}[\ell_{HL}(Y, \hat{h}(X)) \mid \mathcal{D}_n]$$

where $\ell_{HL}(Y, \hat{h}(X)) = I_{\{\hat{h}(X) \neq Y\}}$ (case $\{0, 1\}$),

$$\ell_{HL}(Y, \hat{h}(X)) = I_{\{Y\hat{h}(X) < 0\}}$$
 (case $\{-1, 1\}$).

ℓ_{HL} : hard loss.

Empirical error rate

$$L_n(\hat{h}) = \frac{1}{n} \sum_{i=1}^n \ell_{HL}(\hat{h}(X_i), Y_i)$$

Bayes classifier

Assume - \mathbb{P}_X has a density w.r.t. Lebesgue measure,
- $\eta(x) = \mathbb{P}(Y = 1 | X = x)$ is defined everywhere,
and define

$$h_B(x) = \begin{cases} 1 & \text{if } \eta(x) > 0.5 \\ 0 & \text{if } \eta(x) < 0.5 \\ \mathcal{B}(0.5) & \text{otherwise.} \end{cases}$$

Proposition

$$h_B = \arg \min_h L(h)$$

Some notations

In the following, we will consider classifiers of the form

$$h_f(x) = I_{\{f(x) > 0\}} \quad \text{or} \quad h_f(x) = \text{sign}[f(x)]$$

Example 1 : Bayes classifier

$$h_B(x) = I_{\{\eta(x) - \frac{1}{2} > 0\}} \quad \text{or} \quad h_B(x) = \text{sign}\left[\eta(x) - \frac{1}{2}\right]$$

Example 2 : linear classifier

$$h_\beta(x) = I_{\{x^T \beta > 0\}} \quad \text{or} \quad h_\beta(x) = \text{sign}[x^T \beta]$$

Overview

1 Introduction

- Basics in optimization
- Basics in classification

2 Logistic regression

- Classical logistic regression
- Regularized logistic regression

3 Support Vector Machines

- Linear SVM
- Kernel SVM

4 Theoretical guarantees

Logistic regression

Statistical learning with sparsity [HTW15]

From LM to GLM

Linear (regression) model

$$Y_i = x_i \beta + \varepsilon_i, \quad \varepsilon_i \sim \mathcal{N}(0, \sigma^2), \text{ i.i.d.} \Leftrightarrow Y_i | X_i = x_i \sim \mathcal{N}(x_i \beta, \sigma^2), \text{ ind.}$$
$$\Leftrightarrow \begin{cases} Y_i | X_i = x_i \sim \mathcal{N}(\mu_{x_i}, \sigma^2) \\ \mu_{x_i} = x_i^T \beta \end{cases}$$

Generalized linear model

$$\begin{cases} Y_i | X_i = x_i \sim \mathcal{B}(p_{x_i}), \text{ ind.} \\ p_{x_i} = g^{-1}(x_i^T \beta) \end{cases}$$

where $g(t) = \log \left[\frac{t}{1-t} \right]$ is the "logit" *link function*.

Note : Only $Y|x$ is considered.

Maximum likelihood inference

Y_1, \dots, Y_n independent cond. to x_1, \dots, x_n ,
 $Y_i|x_i \sim \mathcal{B}(p_{x_i})$, $\forall i = 1, \dots, n$

$$\Rightarrow \mathcal{L}(\beta) = \log \left\{ \prod_{i=1}^n p_i^{y_i} (1-p_i)^{1-y_i} \right\}$$

Proposition

$$\begin{aligned}\nabla \mathcal{L}(\beta) &= X^T(y - p), \\ H\mathcal{L}(\beta) &= -X^TDX,\end{aligned}$$

where $p = (p_1, \dots, p_n)$, $D = \text{diag}(p_i(1-p_i))$.

Note : No closed form solution for $\hat{\beta}$ but $\mathcal{L}(\beta)$ is concave.
 \Rightarrow Numeric optimization via Newton algorithm.

Newton method for LR

Main steps

- ★ 2nd order approximation

$$\widetilde{\mathcal{L}}_{(t)}(\beta) = \mathcal{L}(\widehat{\beta}^{(t)}) + \left[\nabla \mathcal{L}(\widehat{\beta}^{(t)}) \right]^T \left(\beta - \widehat{\beta}^{(t)} \right) + \frac{1}{2} \left(\beta - \widehat{\beta}^{(t)} \right)^T [H\mathcal{L}(\beta)] \left(\beta - \widehat{\beta}^{(t)} \right)$$

- ★ Define

$$\widehat{\beta}^{(t+1)} = \arg \max_{\beta} \widetilde{\mathcal{L}}_{(t)}(\beta)$$

Proposition

- i) $\widehat{\beta}^{(t+1)} = \widehat{\beta}^{(t)} + \left[X^T D_{(t)} X \right]^{-1} X^T (y - p_{(t)}),$
- ii) $\widehat{\beta}^{(t+1)}$ is also solution of

$$\arg \min_{\beta} \| X\beta - z_{(t)} \|_{D_{(t)}^{-1}}^2,$$

where $z_{(t)} = X\widehat{\beta}^{(t)} + D_{(t)}^{-1}(y - p_{(t)})$ and $p_{(t)} = (p_i(\widehat{\beta}^{(t)}))_{1 \leq i \leq n}$.

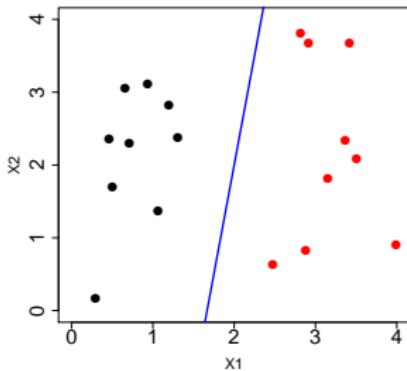
Logistic regression classifier

Proposition

The LR classifier is a linear classifier defined as

$$\hat{h}_{LR}(x) = I_{\{x^T \hat{\beta} > 0\}} \quad \text{where} \quad \hat{\beta} = \arg \max_{\beta} \mathcal{L}(\beta)$$

Separability : definition



Definition

A training set is separable if there exists β such that

$$\forall i | y_i = 1, x_i^T \beta > 0$$

$$\forall i | y_i = 0, x_i^T \beta < 0$$

Note 1 : \Leftrightarrow there exists a linear classifier h such that $L_n(h) = 0$,

Note 2 : discrete case : can be relaxed to a single cell.

Separability : consequence

Proposition

If the training set is separable, then

$$\begin{aligned}\mathcal{L}(\hat{\beta}) &= 0, \\ \text{and } \|\hat{\beta}\| &= +\infty.\end{aligned}$$

→ Even in the "small dimension" setting, regularization may be required.

From MLE to convex risk minimization

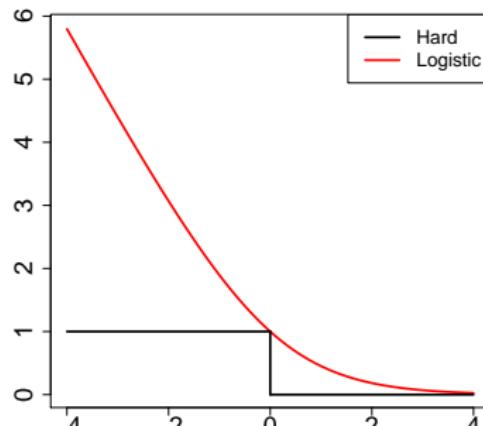
Proposition

Assume $Y_i = \pm 1, \forall i$. One has

$$\hat{h}_{LR}(x) = \text{sign}[x^T \hat{\beta}],$$

$$\text{with } \hat{\beta} = \arg \min_{\beta} \sum_{i=1}^n \ell_{LR}(y_i x_i^T \beta)$$

where $\ell_{LR}(t) = \log[1 + e^{-t}]$ is the logistic loss.



Regularized logistic regression

Definition

For any $\lambda > 0$ the regularized LR classifier is defined as

$$\hat{h}_{RLR}^\lambda(x) = \text{sign}\left[x^T \hat{\beta}_\lambda\right],$$

$$\text{with } \hat{\beta}_\lambda = \arg \min_{\beta} \sum_{i=1}^n \ell_{LR}\left(y_i x_i^T \beta\right) + \lambda R(\beta)$$

Ridge LR : $R(\beta) = \|\beta\|_2^2$

$\hat{\beta}_\lambda$ is always defined and unique.

Lasso LR : $R(\beta) = \|\beta\|_1$

$\hat{\beta}_\lambda$ is always defined and unique (under mild conditions, [T+13]).

Inference for regularized logistic regression

Recall in the low dimensional case

$$\hat{\beta}^{(t+1)} = \arg \min_{\beta} \|X\beta - z_{(t)}\|_{D_{(t)}^{-1}}^2,$$

where $z_{(t)} = X\hat{\beta}^{(t)} + D_{(t)}^{-1}(y - p_{(t)})$ and $p_{(t)} = (p_i(\hat{\beta}^{(t)}))_i$.

Solving regularized LR...

... is replaced with solving

$$\hat{\beta}_{\lambda}^{(t+1)} = \arg \min_{\beta} \left\{ \|X\beta - z\|_{D_{(t)}^{-1}}^2 + \lambda R(\beta) \right\} \quad (1)$$

hence boils down to regularized regression (at each step)!

Ridge LR : Solution of (1) has a closed form expression.

Lasso LR : use proximal/coordinate gradient descent.

Exact optimization for Lasso LR [SK03]

Solve

$$\hat{\beta}, \hat{\beta}_0 = \arg \min_{\beta, \beta_0} \left\{ \sum_{i=1}^n \ell_{LR}(y_i f(x_i)) + \lambda \|\beta\|_1 \right\}, \quad \text{where } f(x) = x^T \beta + \beta_0$$

First order conditions

Lead to the definition of the violation criterion :

$$\text{At point } \beta, \quad V_j = \begin{cases} |\lambda - F_j| & \text{if } \beta_j > 0 \\ |\lambda + F_j| & \text{if } \beta_j < 0 \\ \max(F_j - \lambda, -F_j - \lambda, 0) & \text{if } \beta_j = 0 \end{cases}$$

where

$$F_j = \sum_{i=1}^n \frac{e^{-y_i f(x_i)}}{1 + e^{-y_i f(x_i)}} y_i x_{ij}$$

Note : At point $\hat{\beta}$, $V_j = 0 \ \forall j$.

Solving for a single value of λ

Require: $(x_1, y_1), \dots, (x_n, y_n), \lambda$

Initialize β to β_{init} ; Set $\mathcal{A} = \{j / \beta_j \neq 0\}$

while There exists $j \notin \mathcal{A}$ s.t. $V_j \neq 0$ **do**

 Find $j_{max} = \arg \max_{j \in \mathcal{A}} V_j$

 Update $\mathcal{A} \leftarrow \mathcal{A} \cup \{j_{max}\}$

while there exists $j \in \mathcal{A}$ s.t. $V_j \neq 0$ **do**

 Find $j_{max} = \arg \max_{j \in \mathcal{A}} V_j$

 Optimize $L(\beta)$ w.r.t. $\beta_{j_{max}}$

 Recompute $V_j, j \in \mathcal{A}$

end while

end while

return β

Solving for a single value of λ

Require: $(x_1, y_1), \dots, (x_n, y_n), \lambda$

Initialize β to β_{init} ; Set $\mathcal{A} = \{j / \beta_j \neq 0\}$

while There exists $j \notin \mathcal{A}$ s.t. $V_j \neq 0$ **do**

 Find $j_{max} = \arg \max_{j \in \mathcal{A}} V_j$

 Update $\mathcal{A} \leftarrow \mathcal{A} \cup \{j_{max}\}$

while there exists $j \in \mathcal{A}$ s.t. $V_j \neq 0$ **do**

 Find $j_{max} = \arg \max_{j \in \mathcal{A}} V_j$

 Optimize $L(\beta)$ w.r.t. $\beta_{j_{max}}$

 Recompute $V_j, j \in \mathcal{A}$

end while

end while

return β

Solving for a single value of λ

Require: $(x_1, y_1), \dots, (x_n, y_n), \lambda$

Initialize β to β_{init} ; Set $\mathcal{A} = \{j / \beta_j \neq 0\}$

while There exists $j \notin \mathcal{A}$ s.t. $V_j \neq 0$ **do**

 Find $j_{max} = \arg \max_{j \in \mathcal{A}} V_j$

 Update $\mathcal{A} \leftarrow \mathcal{A} \cup \{j_{max}\}$

while there exists $j \in \mathcal{A}$ s.t. $V_j \neq 0$ **do**

 Find $j_{max} = \arg \max_{j \in \mathcal{A}} V_j$

 Optimize $L(\beta)$ w.r.t. $\beta_{j_{max}}$

 Recompute $V_j, j \in \mathcal{A}$

end while

end while

return β

Solving for a single value of λ

Require: $(x_1, y_1), \dots, (x_n, y_n), \lambda$

Initialize β to β_{init} ; Set $\mathcal{A} = \{j / \beta_j \neq 0\}$

while There exists $j \notin \mathcal{A}$ s.t. $V_j \neq 0$ **do**

 Find $j_{max} = \arg \max_{j \in \mathcal{A}} V_j$

 Update $\mathcal{A} \leftarrow \mathcal{A} \cup \{j_{max}\}$

while there exists $j \in \mathcal{A}$ s.t. $V_j \neq 0$ **do**

 Find $j_{max} = \arg \max_{j \in \mathcal{A}} V_j$

 Optimize $L(\beta)$ w.r.t. $\beta_{j_{max}}$

 Recompute $V_j, j \in \mathcal{A}$

end while

end while

return β

- ★ Sub-problem is str. convex $\Rightarrow L(\beta)$ decreases at each step,
- ★ Only a sparse vector to store.
- ★ $\beta_{init} = 0$ seems perfect.

Solving for a set of λ values

Require: $(x_1, y_1), \dots, (x_n, y_n)$, $\lambda_1 > \dots > \lambda_m$

for $k=1, \dots, m$ **do**

 Initialize β^{λ_k} to $\hat{\beta}^{\lambda_{k-1}}$; Set $\mathcal{A} = \{j \mid \beta_j^{\lambda_k} \neq 0\}$

while There exists $j \notin \mathcal{A}$ s.t. $V_j \neq 0$ **do**

 Find $j_{\max} = \arg \max_{j \in \mathcal{A}} V_j$

 Update $\mathcal{A} \leftarrow \mathcal{A} \cup \{j_{\max}\}$

while there exists $j \in \mathcal{A}$ s.t. $V_j \neq 0$ **do**

 Find $j_{\max} = \arg \max_{j \in \mathcal{A}} V_j$

 Optimize $L(\beta^{\lambda_k})$ w.r.t. $\beta_{j_{\max}}^{\lambda_k}$

 Recompute V_j , $j \in \mathcal{A}$

end while

end while

end for

return $\beta^{\lambda_1}, \dots, \beta^{\lambda_m}$

Take home message

Logistic regression

- ★ Belongs to the GLM family,
- ★ Is a linear classifier,
- ★ Is also an ECRM minimizer.
- ★ May require regularization **even in small dimension**

Inference

- ★ Can be performed easily,
- ★ But cannot be performed easily !
⇒ Pay attention to which package you use...

Overview

1 Introduction

- Basics in optimization
- Basics in classification

2 Logistic regression

- Classical logistic regression
- Regularized logistic regression

3 Support Vector Machines

- Linear SVM
- Kernel SVM

4 Theoretical guarantees

Support Vector Machines

Learning With Kernels [SS01]

Kernels Methods In Computational Biology [STV04]

Back to basics

Bayes classifier

$$h_B = \arg \min_h L(h)$$

- Requires $\mathbb{P}_{X,Y}$,
- $\mathcal{Y}^{\mathcal{X}}$ is... large!

Find a linear classifier

$$\hat{h}_f = \arg \min_{h_f} L_n(h_f), \quad \text{where } f(x) = x^T \beta + \beta_0$$

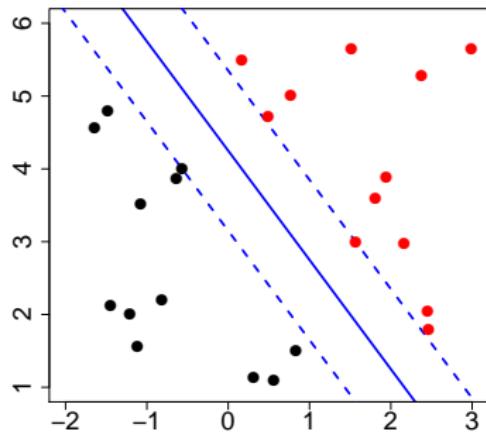
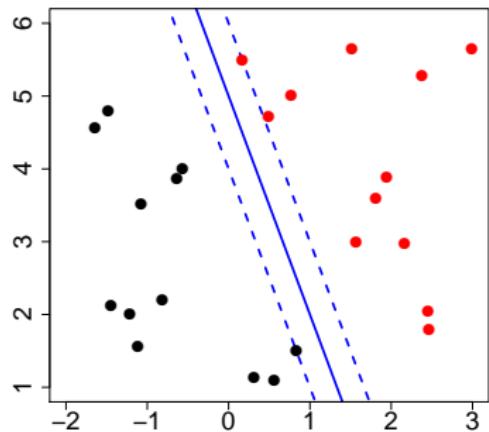
- Not unique (in two ways),
- Still NP hard to find in practice...

Find the *optimal* linear classifier

$$\hat{h}_f = \text{sign} [\hat{f}(x)] \quad \text{where } \hat{f}(x) = x^T \hat{\beta} + \hat{\beta}_0 \quad \text{and} \quad \hat{\beta}, \hat{\beta}_0 = \arg \min_{\beta, \beta_0} \text{Crit}(\beta, \beta_0)$$

Separating hyperplanes and margin

Consider a linearly separable dataset



Separating hyperplane : Any $\Delta : \{x^T \beta + \beta_0 = 0\}$ s.t.

$$y_i (x_i^T \beta + \beta_0) > 0.$$

Margin Smallest distance between a point and Δ .

Maximum margin hyperplane

If the dataset is linearly separable, choose $(\hat{\beta}_0, \hat{\beta})$ such that

$$\hat{\Delta} = \{x^T \hat{\beta} + \hat{\beta}_0 = 0\}$$

has **maximum** margin.

Proposition

$$(\hat{\beta}_0, \hat{\beta}) = \arg \min_{\beta_0, \beta} \frac{1}{2} \|\beta\|_2^2 \text{ u.c. } y_i(x_i^T \beta + \beta_0) \geq 1, \forall i$$

Note : Constraints account for - correct classification
- maximum margin
- Δ representation identification.

Non separable case

If the dataset is not linearly separable, relax constraints as follows

$$y_i(x_i^T \beta + \beta_0) \geq 1 - \varepsilon_i, \quad \forall i$$

where $\varepsilon_i \geq 0$, and penalize for the extend of margin violation.

Definition (Soft Margin SVM classifier)

$$\hat{h}_{SVM} = \text{sign}[\hat{\beta}_0 + x^T \hat{\beta}]$$

where $(\hat{\beta}_0, \hat{\beta})$ is solution of

$$\text{minimize} \quad \frac{1}{2} \|\beta\|_2^2 + C \sum_{i=1}^n \varepsilon_i$$

$$\text{subject to} \quad y_i(x_i^T \beta + \beta_0) \geq 1 - \varepsilon_i, \quad \forall i = 1, \dots, n$$
$$\varepsilon_i \geq 0, \quad \forall i = 1, \dots, n$$

Inference

Inference boils down to solving the following problem :

$$\text{minimize} \quad \frac{1}{2} \|\beta\|_2^2 + C \sum_{i=1}^n \varepsilon_i$$

$$\text{subject to} \quad \begin{aligned} y_i(x_i^T \beta + \beta_0) &\geq 1 - \varepsilon_i \\ \varepsilon_i &\geq 0 \end{aligned}$$

Associated primal problem :

$$\min_{\beta, \beta_0, \varepsilon} \max_{\alpha, \mu \geq 0} L(\beta, \beta_0, \varepsilon, \alpha, \mu)$$

where

$$L(\beta, \beta_0, \varepsilon, \alpha, \mu) = \frac{1}{2} \|\beta\|_2^2 + C \sum_{i=1}^n \varepsilon_i + \sum_{i=1}^n \alpha_i [1 - \varepsilon_i - y_i(x_i^T \beta + \beta_0)] - \sum_{i=1}^n \mu_i \varepsilon_i$$

Dual optimization problem

Dual problem :

$$\max_{\alpha, \mu \geq 0} \min_{\beta, \beta_0, \varepsilon} L(\beta, \beta_0, \varepsilon, \alpha, \mu)$$

Proposition

The SVM dual problem can be reformulated as

$$\text{maximize} \quad -\frac{1}{2} \alpha^T Q \alpha + \mathbf{1}_n^T \alpha$$

$$\text{subject to} \quad 0 \leq \alpha \leq C$$
$$\sum_{i=1}^n y_i \alpha_i = 0$$

where $Q_{ij} = y_i y_j x_i^T x_j$.

Sequential Minimal Optimization [Pla98]

Let B a subset of $\{1, \dots, n\}$. One has

$$\text{maximize} \quad \frac{1}{2} \begin{bmatrix} \alpha_B \\ \alpha_{\bar{B}} \end{bmatrix}^T \begin{bmatrix} Q_{BB} & Q_{B\bar{B}} \\ Q_{\bar{B}B} & Q_{\bar{B}\bar{B}} \end{bmatrix} \begin{bmatrix} \alpha_B \\ \alpha_{\bar{B}} \end{bmatrix} + \mathbf{1}_{|B|}^T \alpha_B + \mathbf{1}_{|\bar{B}|}^T \alpha_{\bar{B}}$$

$$\text{subject to} \quad 0 \leq \alpha_B \leq C, \quad 0 \leq \alpha_{\bar{B}} \leq C$$

$$Y_B^T \alpha_B + Y_{\bar{B}}^T \alpha_{\bar{B}} = 0$$

↔

$$\text{maximize} \quad -\frac{1}{2} \alpha_B^T Q_{BB} \alpha_B + U(\bar{B})^T \alpha_B + \Delta_1(\bar{B})$$

$$\text{subject to} \quad 0 \leq \alpha_B \leq C$$

$$Y_B^T \alpha_B = \Delta_2(\bar{B})$$

Apply with $|B| = 2$!

- ★ Simpler optimization problem,
- ★ Only 2 columns of Q need to be loaded at each step,
- ★ "Pairwise" coordinate descent.

Note : One can search for the "best" pair at each step...

From SVM to convex risk minimization

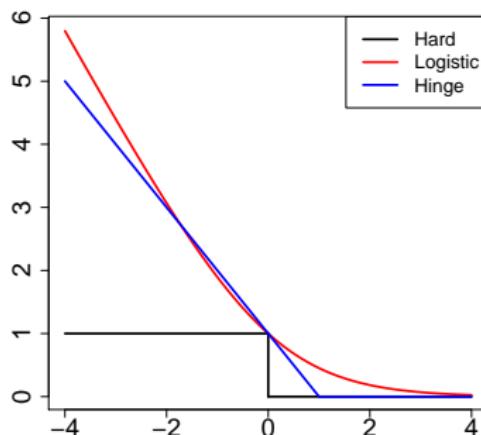
Proposition

Assume $Y_i = \pm 1, \forall i$. One has

$$\hat{h}_{SVM}^\lambda(x) = \text{sign} \left[x^T \hat{\beta}_\lambda + \hat{\beta}_{0\lambda} \right],$$

$$\text{with } (\hat{\beta}_{0\lambda}, \hat{\beta}_\lambda) = \arg \min_{\beta_0, \beta} \sum_{i=1}^n \ell_{SVM} (y_i [x_i^T \beta + \beta_0]) + \lambda \|\beta\|_2^2$$

where $\ell_{SVM}(t) = |1 - t|_+$ is the hinge loss.



So far...

SVM classifier

$$\hat{h}_{SVM}^{\lambda}(x) = \text{sign} \left[x^T \hat{\beta}_{\lambda} + \hat{\beta}_{0\lambda} \right],$$

$$\text{with } (\hat{\beta}_{0\lambda}, \hat{\beta}_{\lambda}) = \arg \min_{\beta_0, \beta} \sum_{i=1}^n \ell_{SVM} \left(y_i \left[x_i^T \beta + \beta_0 \right] \right) + \lambda \|\beta\|_2^2$$

- ★ Linear classifier with largest margin,
- ★ Linear classifier that minimizes the hinge loss.

Inference

$$\hat{\beta}_{\lambda} = \sum_{i=1}^n y_i \hat{\alpha}_i x_i$$

$$\text{with } \hat{\alpha} = \arg \max_{0 \leq \alpha \leq 1/\lambda} \left\{ -\frac{1}{2} \alpha^T Q \alpha + \mathbf{1}_n^T \alpha \right\} \text{ u.c. } \sum_{i=1}^n y_i \alpha_i = 0$$

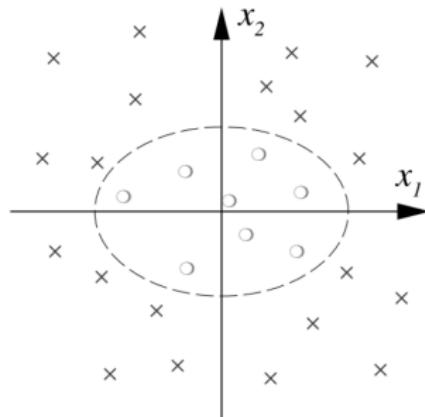
So far...

Restricted to $\mathcal{X} = \mathbb{R}^p$.

What about - text classification ?

- sequence classification ?
- pathway classification ?
- ...

Restricted to *linear* classification :



Naive way : transform and proceed (1/3)

Example 1 Document classification (e.g. Reuters dataset)

Bag of words

- ★ $\mathcal{Y} = \{1, \dots, M\}$, with M the number of document classes,
- ★ Apply transformation $\phi : \mathcal{X} = \{\text{documents}\} \rightarrow \mathbb{R}^d$

$$\phi(\text{doc}) = (N_{w_1}, \dots, N_{w_d}),$$

where N_{w_j} is the nb. of occurrence of word w_j in doc .

Characteristics

- ★ d is large ($\approx 35k$),
- ★ $\phi(\text{doc})$ is sparse (between 93 and 1263 words per doc).

Storing the $\phi(\text{doc})$'s is cheap !

Naive way : transform and proceed (2/3)

Example 2 Non-linear classification (e.g. Sphere example)



Apply transformation

$$x = (x_1, x_2)^T \mapsto \phi(x) = (x_1^2, x_2^2, \sqrt{2}x_1x_2).$$

Characteristics

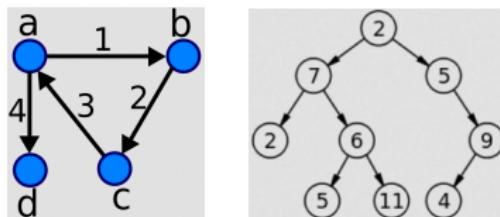
- ★ $x \in \mathbb{R}^p$ is "big", $\phi(x)$ is way bigger,
- ★ $\phi(x)$ is not sparse.

Storing the $\phi(x)$'s is **prohibitive**.

Naive way : transform and proceed (3/3)

Example 3 Structured data classification

Networks, trees



Sequences

5' ACTACTAGATTACTTACGGATCAGGTACTTAGAGGCTTGCAACCA 3'
||| ||| ||| ||| ||| ||| ||| ||| ||| |||
5' ACTACTAGATT----ACGGATC--GTACTTAGAGGCTAGCAACCA 3'

Finding $\phi: \mathcal{S} \rightarrow \mathbb{R}^d$ is **non-trivial**.

Smart way : kernel SVM

Combining CRM formulation + Inference leads to :

$$\hat{h}_{SVM}(x) = \text{sign} \left[\sum_{i=1}^n y_i \hat{\alpha}_i \langle \mathbf{x}_i, \mathbf{x} \rangle + C_{\hat{\alpha}} \right],$$

$$\text{with } \hat{\alpha} = \arg \max_{0 \leq \alpha \leq 1/\lambda} \left\{ \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{i'=1}^n y_i y_{i'} \alpha_i \alpha_{i'} \langle \mathbf{x}_i, \mathbf{x}_{i'} \rangle \right\}$$

$$\text{u.c. } \sum_{i=1}^n y_i \alpha_i = 0$$

The x_i 's only appear through scalar products.

⇒ Only need to compute $\langle \phi(x_i), \phi(x_j) \rangle$.

⇒ Only need to store the $n \times n$ Gram matrix.

Smart way : kernel SVM

Combining CRM formulation + Inference leads to :

$$\hat{h}_{SVM}(x) = \text{sign} \left[\sum_{i=1}^n y_i \hat{\alpha}_i \mathbf{k}(x_i, x) + C_{\hat{\alpha}} \right],$$

$$\text{with } \hat{\alpha} = \arg \max_{0 \leq \alpha \leq 1/\lambda} \left\{ \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{i'=1}^n y_i y_{i'} \alpha_i \alpha_{i'} \mathbf{k}(x_i, x_{i'}) \right\}$$

$$\text{u.c. } \sum_{i=1}^n y_i \alpha_i = 0$$

The x_i 's only appear through scalar products.

- ⇒ Only need to compute $\langle \phi(x_i), \phi(x_j) \rangle$.
- ⇒ Only need to store the $n \times n$ Gram matrix.
- ⇒ Only need to compute **some similarity** between x_i and x_j .

Kernels is what we need !

Kernels and RKHS

Definition

A kernel is a function $k : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ satisfying

$$\forall x_1, x_2, \quad k(x_1, x_2) = k(x_2, x_1),$$

$$\forall n \in \mathbb{N}, \quad \forall (x_1, \dots, x_n) \in \mathcal{X}^n, \quad (k(x_i, x_j))_{ij} \succeq 0$$

Hilbert space representation of a kernel

Proposition

For any kernel k on \mathcal{X} , there exists a Hilbert space $\mathcal{F} \subset \mathbb{R}^{\mathcal{X}}$ and a mapping $\phi : \mathcal{X} \rightarrow \mathcal{F}$ such that

$$\forall x, x' \in \mathcal{X}, \quad k(x, x') = \langle \phi(x), \phi(x') \rangle_{\mathcal{F}}$$

(Reproducing property)

Examples of kernels

Polynomial kernels

$\mathcal{X} = \mathbb{R}^p$, $k(x, y) = (\langle x, y \rangle + 1)^m$ ($m \in \mathbb{N}$).

A string kernel

$W_i = w_1 \circ \dots \circ w_{n_i}$, $w_i \in \mathcal{A}$: string of size n_i in alphabet \mathcal{A} .

$\mathcal{S}_q = \{s / s_1 \circ \dots \circ s_q, s_j \in \mathcal{A}\}$: set of strings with size q

$N_s(W_i)$: nb of occurrence of string s in W_i

$$k(W_i, W_j) = \sum_s N_s(W_i) N_s(W_j)$$

Note : $k(W_i, W_j)$ can be computed in time $n_i + n_j$.

More kernels

Gaussian (RBF) kernel : $\mathcal{X} = \mathbb{R}^p$, $k(x, y) = \exp\left(\frac{-d(x - x')^2}{2\sigma^2}\right)$.

Kernels for structured data : see [STV04] for many examples.

Polynomial kernel

$$1/(\langle x, y \rangle + 1)^m = \sum_{j=0}^m \binom{j}{m} \langle x, y \rangle^j \Rightarrow \text{prove that } \langle x, y \rangle^j \text{ is a kernel.}$$

2/ Define $\phi(x) = (\phi_s(x))_{s \in S_j}$ such that

$$\phi_s(x) = \left(x_1^{s_1} \times \dots \times x_p^{s_p} \right), \quad s \in S_j = \left\{ s_1, \dots, s_p \in \mathbb{N}^p / \sum_u s_u = j \right\}$$

2/ Define

$$\begin{aligned} \langle \phi(x), \phi(y) \rangle &= \sum_{s \in S_j} \phi_s(x) \phi_s(y) &= \sum_s \left(x_1^{s_1} \times \dots \times x_p^{s_p} \right) \left(y_1^{s_1} \times \dots \times y_p^{s_p} \right) \\ &= \sum_s ((x_1 y_1)^{s_1} \times \dots \times (x_p y_p)^{s_p}) \\ &= \left(\sum_{i=1}^p x_i y_i \right)^j = \langle x, y \rangle^j \end{aligned}$$

Note : Give access to non linear classification without additional computational cost.

Kernel trick

Proposition (Representer theorem ([STV04]))

Let \mathcal{X} be a set endowed with kernel k , and $\Psi : \mathbb{R}^n \rightarrow \mathbb{R}$. Then any solution of problem

$$\hat{f} = \arg \min_{f \in \mathcal{F}} \Psi(f(x_1), \dots, f(x_n)) + \lambda \|f\|_{\mathcal{F}}^2$$

has the form

$$\hat{f} = \sum_{i=1}^n \alpha_i k(x_i, \cdot).$$

Kernel trick applied to SVM

Proposition

Let k be a kernel on \mathcal{X} , and \mathcal{F} its associated RKHS. Consider the SVM classifier

$$\hat{h}_{SVM}^{\lambda}(x) = \text{sign}[\hat{f}^{\lambda}(x)]$$

where $\hat{f}^{\lambda} = \arg \min_{f \in \mathcal{F}} \left\{ \sum_{i=1}^n \ell_{SVM}(y_i f(x_i)) + \lambda \|f\|_{\mathcal{F}}^2 \right\}.$

Then $\hat{f}^{\lambda} = \sum_{i=1}^n y_i \hat{\alpha}_i k(x_i, \cdot),$

with $\hat{\alpha} = \arg \max_{0 \leq \alpha \leq \frac{1}{\lambda}} \left\{ \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j k(x_i, x_j) \right\}.$

⇒ Same computational efficiency as in the linear case...

Overview

1 Introduction

- Basics in optimization
- Basics in classification

2 Logistic regression

- Classical logistic regression
- Regularized logistic regression

3 Support Vector Machines

- Linear SVM
- Kernel SVM

4 Theoretical guarantees

Theoretical guarantees

Introduction to high-dimensional statistics [Gir14]

Connection with the Bayes classifier

Let ℓ be a convex loss function. Define

$$h_{f^*}(x) = \text{sign}[f^*(x)]$$

where $f^* = \arg \min_f \mathbb{E}[\ell(Yf(X))]$

h_{f^*} is "optimal" in the sense of loss ℓ .

How does h_{f^*} compare with h_B ?

Proposition

Assume that ℓ satisfies $\ell'(0) < 0$. Then

$$h_{f^*} = h_B.$$

ECRM on a ball

Consider the classifier based on the Empirical Convex Risk Minimizer (ECRM) on a ball of \mathcal{F} :

$$\hat{h}(x) = \text{sign} [\hat{f}(x)]$$

where

$$\begin{aligned}\hat{f} &= \arg \min_{f \in \mathcal{F}} \left\{ \sum_{i=1}^n \ell(y_i f(x_i)) + \lambda \|f\|_{\mathcal{F}} \right\} \\ &= \arg \min_{f \in \mathcal{F}_R} \left\{ L_n^\ell(f) \right\}\end{aligned}$$

\mathcal{F} : RKHS associated with kernel k ,
 \mathcal{F}_R : Centered ball of radius R in \mathcal{F} .

Upper bound for the risk

Theorem ([Gir14])

Let $R > 0$, and consider a loss function ℓ such that

- ℓ is positive, non-increasing and convex,
- ℓ is α -Lipschitz on $[-R, R]$,
- ℓ dominates the hard loss on \mathbb{R} .

Let \hat{h} the classifier previously defined, with kernel k satisfying $k(x, x) \leq 1, \forall x$.

With probability $1 - e^{-t}$, one has

$$L(\hat{h}) \leq L_n^\ell(\hat{f}) + \frac{4\alpha R}{\sqrt{n}} + C\sqrt{\frac{t}{2n}}$$

where C is a constant.

Step 1 : connect $L(\hat{h})$ and $L_n^\ell(\hat{f})$

$$\begin{aligned}\text{First note that } L(\hat{h}) &\leq L^\ell(\hat{f}) \\ &\leq L_n^\ell(\hat{f}) + \sup_{\mathcal{F}_R} |L_n^\ell(f) - L^\ell(f)|\end{aligned}$$

Theorem (McDiarmid)

If $g : \mathbb{R}^n \rightarrow \mathbb{R}$ satisfies $|g(z_1, \dots, z_i, \dots, z_n) - g(z_1, \dots, z_{i'}, \dots, z_n)| \leq b$,

$$\text{then } \mathbb{P}(g(Z_1^n) > \mathbb{E}[g(Z_1^n)] + u) \leq \exp\left(-\frac{2u^2}{nb^2}\right)$$

Applied to $g((X_i, Y_i)_1^n) = \sup_{\mathcal{F}_R} |L_n^\ell(f) - L^\ell(f)|$, one gets

$$L(\hat{h}) \leq L_n^\ell(\hat{f}) + \mathbb{E}\left[\sup_{\mathcal{F}_R} |L_n^\ell(f) - L^\ell(f)|\right] + \sqrt{\frac{t}{2n}} C$$

with prob. $1 - e^{-t}$ and $C = |\ell(-R) - \ell(R)|$

Step 2 : symmetrization and Rademacher complexity

Let $\{(X'_1, Y'_1), \dots, (X'_n, Y'_n)\}$ be a set of independent copies

$$\begin{aligned}\mathbb{E} \left[\sup_{\mathcal{F}_R} \left| L_n^\ell(f) - L^\ell(f) \right| \right] &= \mathbb{E} \left[\sup_{\mathcal{F}_R} \left| L_n^\ell(f) - \mathbb{E}' \left[L_n^{\ell'}(f) \right] \right| \right] \\ &\leq \mathbb{E}' \mathbb{E} \left[\sup_{\mathcal{F}_R} \left| L_n^\ell(f) - L_n^{\ell'}(f) \right| \right] \\ &\leq \frac{1}{n} \mathbb{E}' \mathbb{E} \left[\sup_{\mathcal{F}_R} \left| \sum_{i=1}^n \ell(Y_i f(X_i)) - \ell(Y_{i'} f(X_{i'})) \right| \right]\end{aligned}$$

Let $\sigma_1, \dots, \sigma_n$ uniform on $\{-1, 1\}$ and independent. Then

$$\begin{aligned}\mathbb{E} \left[\sup_{\mathcal{F}_R} \left| L_n^\ell(f) - L^\ell(f) \right| \right] &\leq \frac{1}{n} \mathbb{E}' \mathbb{E} \left[\mathbb{E}_\sigma \left[\sup_{\mathcal{F}_R} \left| \sum_{i=1}^n \sigma_i (\ell(Y_i f(X_i)) - \ell(Y_{i'} f(X_{i'}))) \right| \right] \right] \\ &\leq \frac{2}{n} \mathbb{E} \left[\mathbb{E}_\sigma \left[\sup_{\mathcal{F}_R} \left| \sum_{i=1}^n \sigma_i \ell(Y_i f(X_i)) \right| \right] \right]\end{aligned}$$

Step 3 : get rid of loss function and labels

Theorem (Contraction principle)

- Ω : bounded subset of \mathbb{R}^n
- $v : \mathbb{R} \rightarrow \mathbb{R}$, α -Lipschitz, $v(0) = 0$

$$\text{Then } \mathbb{E}_\sigma \left[\sup_{z \in \Omega} \left| \sum_{i=1}^n \sigma_i v(z_i) \right| \right] \leq 2\alpha \mathbb{E}_\sigma \left[\sup_{z \in \Omega} \left| \sum_{i=1}^n \sigma_i z_i \right| \right]$$

Application : $v = \ell$, $z_i = y_i f(x_i)$. Note that $|f(x)| \leq R$...

$$\begin{aligned} \frac{2}{n} \mathbb{E}_\sigma \left[\sup_{\mathcal{F}_R} \left| \sum_{i=1}^n \sigma_i \ell(y_i f(x_i)) \right| \right] &\leq \frac{4\alpha}{n} \mathbb{E}_\sigma \left[\sup_{\mathcal{F}_R} \left| \sum_{i=1}^n \sigma_i y_i f(x_i) \right| \right] \\ &\leq \frac{4\alpha}{n} \mathbb{E}_\sigma \left[\sup_{\mathcal{F}_R} \left| \sum_{i=1}^n \sigma_i f(x_i) \right| \right] \end{aligned}$$

Step 4 : bound the expectation over σ 's

Get rid of f : $\forall f \in \mathcal{F}_R$,

$$\begin{aligned}
 \left| \sum_{i=1}^n \sigma_i f(x_i) \right| &= \left| \sum_{i=1}^n \sigma_i \langle f, k(x_i, \cdot) \rangle_{\mathcal{F}} \right| = \left| \left\langle f, \sum_{i=1}^n \sigma_i k(x_i, \cdot) \right\rangle_{\mathcal{F}} \right| \\
 &\leq \|f\|_{\mathcal{F}} \left\| \sum_{i=1}^n \sigma_i k(x_i, \cdot) \right\|_{\mathcal{F}} \\
 &\leq R \left\| \sum_{i=1}^n \sigma_i k(x_i, \cdot) \right\|_{\mathcal{F}} \\
 \Rightarrow \mathbb{E}_{\sigma} \left[\sup_{\mathcal{F}_R} \left| \sum_{i=1}^n \sigma_i f(X_i) \right| \right] &\leq R \mathbb{E}_{\sigma} \left[\left\| \sum_{i=1}^n \sigma_i k(x_i, \cdot) \right\|_{\mathcal{F}} \right] \\
 &= R \mathbb{E}_{\sigma} \left[\sqrt{\sum_{i=1}^n \sum_{i'=1}^n \sigma_i \sigma_{i'} k(x_i, x_{i'})} \right] \\
 &\leq R \sqrt{\sum_{i=1}^n \sum_{i'=1}^n \mathbb{E}_{\sigma} [\sigma_i \sigma_{i'}] k(x_i, x_{i'})} \leq R \sqrt{R}
 \end{aligned}$$

Conclusion

A unified framework...

Two algorithms, one strategy : ECRM. Allows for

- the derivation of theoretical guarantees
- common optimization strategies
- application of classical penalties (group/fused lasso, EN,...)

... a very general kernel trick...

- allows for non-linear classification
- makes SVM and LR amenable to structured data
- useful in other application (kPCA, kClustering, kCCA,...)

... and still some specificities

- convex vs strictly convex (uniqueness of the solution, ...)
- regularization paths for the β s

What's next ? HD data vs big data

- primal vs dual
- stochastic gradient descent

-  Stephen Boyd and Lieven Vandenberghe.
Convex optimization.
Cambridge university press, 2004.
-  Edwin KP Chong and Stanislaw H Zak.
An introduction to optimization, volume 76.
John Wiley & Sons, 2013.
-  Luc Devroye, László Györfi, and Gábor Lugosi.
A probabilistic theory of pattern recognition, volume 31.
Springer Science & Business Media, 2013.
-  Jerome Friedman, Trevor Hastie, and Robert Tibshirani.
The elements of statistical learning, volume 1.
Springer series in statistics New York, 2001.
-  Christophe Giraud.
Introduction to high-dimensional statistics, volume 138.
CRC Press, 2014.



Trevor Hastie, Robert Tibshirani, and Martin Wainwright.

Statistical learning with sparsity : the lasso and generalizations.

CRC press, 2015.



Yuriii Nesterov et al.

Gradient methods for minimizing composite objective function,
2007.



John Platt.

Sequential minimal optimization : A fast algorithm for training
support vector machines.

1998.



Shirish Krishnaj Shevade and S Sathiya Keerthi.

A simple and efficient algorithm for gene selection using sparse
logistic regression.

Bioinformatics, 19(17) :2246–2253, 2003.

-  Bernhard Scholkopf and Alexander J Smola.
Learning with kernels : support vector machines, regularization, optimization, and beyond.
MIT press, 2001.
-  Bernhard Schölkopf, Koji Tsuda, and Jean-Philippe Vert.
Kernel methods in computational biology.
MIT press, 2004.
-  Ryan J Tibshirani et al.
The lasso problem and uniqueness.
Electronic Journal of Statistics, 7 :1456–1490, 2013.