Optimisation Numérique pour la Classification

Application au Dataset Titanic

Fatimetou Limam Abeid
C16689

Université NKTT

January 12, 2026

Plan

Contexte du problème

Le dataset **Titanic** est un problème classique de **classification binaire supervisée**.

Chaque observation correspond à un passager décrit par :

- âge,
- sexe,
- classe sociale,
- tarif,
- etc.

Variable cible :

$$y = \begin{cases} 1 & \text{survie} \\ 0 & \text{décès} \end{cases}$$

Objectif mathématique

L'objectif est de construire un classifieur capable de prédire la survie d'un passager en minimisant une fonction de perte.

Nous transformons ce problème statistique en **problème d'optimisation convexe** afin de :

- obtenir des garanties théoriques,
- comparer différentes méthodes,
- relier pratique et théorie du cours.

Formulation mathématique

On considère :

$$\{(x_i, y_i)\}_{i=1}^{n}$$

avec :

$$x_i \in \mathbb{R}^d, \quad y_i \in \{-1, +1\}$$

On cherche :

$$w \in \mathbb{R}^d$$

Fonction objectif

Fonction logistique régularisée :

$$F(w) = \frac{1}{n} \sum_{i=1}^{n} \log\left(1 + e^{-y_i x_i^T w}\right) + \frac{\lambda}{2} \|w\|_2^2$$

- Perte logistique : classification
- Régularisation L2 : stabilité et forte convexité

Régularité — Classe C^2

La fonction :

$$\ell(z) = \log\left(1 + e^{-z}\right)$$

est de classe C^∞.

Or :

- $z = y_i x_i^T w$ est linéaire en w,
- $\|w\|_2^2$ est polynomiale.

Conclusion :

$$F \in C^2(\mathbb{R}^d)$$

Convexité

- $\ell(z)$ est convexe (log-sum-exp),
- composition avec une application linéaire conserve la convexité,
- $\|w\|_2^2$ est convexe.

Conclusion :

F est convexe

Forte convexité

Le Hessien du terme quadratique :

$$\nabla^2 \left(\frac{\lambda}{2} \|w\|^2 \right) = \lambda I$$

Le Hessien de la perte logistique est semi-défini positif.

$$\nabla^2 F(w) \succeq \lambda I$$

Conclusion :

F est λ-fortement convexe

Gradient explicite

$$\nabla F(w) = -\frac{1}{n}\sum_{i=1}^{n}\frac{y_i x_i}{1 + e^{y_i x_i^T w}} + \lambda w$$

- gradient exact,
- implémenté directement dans le code Python.

Gradient Lipschitzien

$$\nabla^2 F(w) = \frac{1}{n} X^T D(w) X + \lambda I$$

avec :

$$0 \preceq D(w) \preceq \frac{1}{4} I$$

Constante de Lipschitz :

$$L = \frac{1}{4n} \|X\|_2^2 + \lambda$$

Descente de gradient

$$w_{k+1} = w_k - \alpha \nabla F(w_k) \quad \text{avec } \alpha \leq \frac{1}{L}$$

Convergence linéaire :

$$F(w_k) - F(w^{)} \leq C(1-\alpha\lambda)^k$$

Motivation du SGD

- Gradient complet : $O(nd)$
- SGD : $O(d)$ par itération

Adapté aux grands jeux de données comme Titanic.

Algorithme SGD

$$w_{k+1} = w_k - \alpha_k \left(-\frac{y_{i_k} x_{i_k}}{1 + e^{y_{i_k} x_{i_k}^T w_k}} + \lambda w_k \right)$$

Pas décroissant :

$$\alpha_k = \frac{\alpha_0}{1 + k}$$

RMSProp et Adam

RMSProp

- normalisation adaptative,
- réduction des oscillations.

Adam

- momentum,
- RMSProp,
- convergence rapide et stable.

Régularisation L1

$$\Phi(w) = f(w) + \lambda \|w\|_1$$

- non dérivable en 0,
- favorise la sparsité.

Opérateur proximal

$$\text{prox}_{\lambda\|\cdot\|_1}(v) = \text{soft-thresholding}(v)$$

$$(\text{prox}(v))_j = \begin{cases} v_j - \lambda & v_j > \lambda \\ 0 & |v_j| \leq \lambda \\ v_j + \lambda & v_j < -\lambda \end{cases}$$

ISTA et FISTA

ISTA

$$O(1/k)$$

FISTA

$$O(1/k^2)$$

FISTA converge beaucoup plus rapidement.

Conclusion

- problème Titanic bien posé,
- application directe des chapitres 1 à 4,
- résultats théoriques confirmés expérimentalement.

Lien fort entre optimisation, théorie et données réelles.