TIPE : Détection de la triche aux échecs

Estéban Collas, en association avec Eliott Fourriques

13 juillet 2025

Figure – source : Wikipedia

Sommaire

Objectifs du TIPE

Stratégie :

- Récupérer des parties analysées sur Lichess, IA VS Humain
- Entraîner un algorithme à détecter l'humain

Objectifs du TIPE :

- Détection de la triche aux échecs en ligne
- Étude du problème des Machines à Vecteur de Support
- Analyse du comportement des joueurs humains

Sommaire

Le principe des SVMs

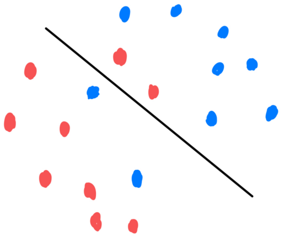


Figure – Problème de séparation à deux classes : trouver la fonction ϕ tel que $\phi(x) = 0$ sépare les deux classes avec le moins d'erreurs possible.

Le principe des SVMs

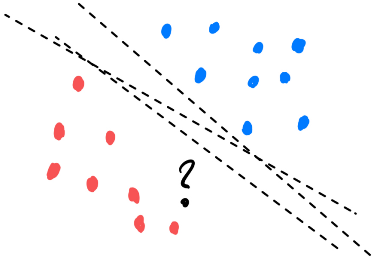


Figure – Problème de séparation linéaire à deux classes : quel est le meilleur hyperplan parmi tous ceux qui séparent les données ?

Le principe des SVMs

On cherche à maximiser la marge :

$$\mathrm{Marge}(\mathrm{H}) = \min_{x_i} \mathrm{d}(x_i, \mathrm{H})$$

Le principe des SVMs

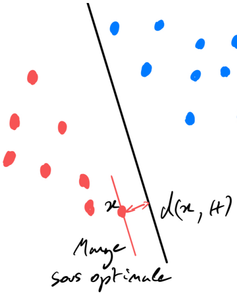


Figure – Exemple d'hyperplan sous-optimal

Le principe des SVMs

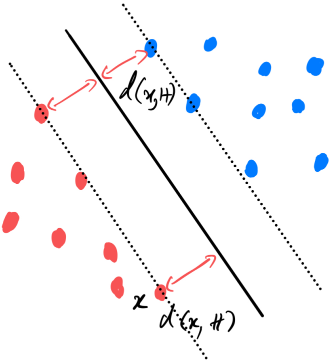


Figure – Exemple d'hyperplan optimal

Le principe des SVMs

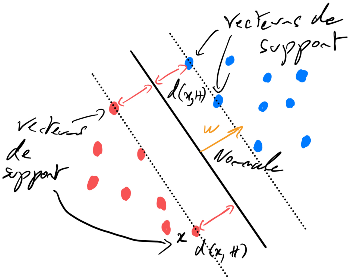


Figure – Représentation des vecteurs de support

Le principe des SVMs

Finalement, en posant ϕ la forme affine associée à notre hyperplan, on saura classifier un nouvel élement x :

- Si $\phi(x) = 0$, l'élément se trouve à la frontière, on ne connaît pas sa classe
- Si $\phi(x) > 0$, classe 1
- Si $\phi(x) < 0$, classe -1

Sommaire

Cas linéairement séparable

On se place dans le cas linéairement séparable. On cherche l'hyperplan de marge maximale, d'équation :

$$\phi(x) = \langle \omega, x \rangle + b = \omega^\top x + b$$

Si x_s est un vecteur de support et $H = \{x \,|\, \omega^\top x + b = 0\}$:

$$\text{Marge} = d(x_s, H) = \frac{|\omega^\top x_s + b|}{\|\omega\|}$$

Cas linéairement séparable

Les paramètres ω et b ne sont pas uniques. Pour tout $k \in \mathbb{R}^*$:

$$k\omega^\top x + kb = 0 \Leftrightarrow \omega^\top x + b = 0$$

On peut alors imposer la condition de normalisation $|\omega^\top x_s + b| = 1$ pour les vecteurs de support x_s :

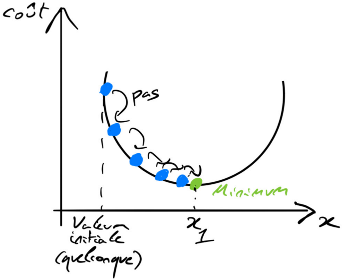
$$\text{Marge} = \frac{|\omega^\top x_s + b|}{\|\omega\|} = \frac{1}{\|\omega\|}$$

On arrive finalement au problème d'optimisation suivant :

$$\begin{cases} \min\limits_{\omega, b} \frac{1}{2}\|\omega\|^2 \\ \text{tel que } y_i(\omega \cdot x_i + b) \geq 1, \quad i = 1, \ldots, p \end{cases}$$

Sommaire

Résolution : Descente de gradient projeté



descenteDeGradient (α, ϵ) :

$x \leftarrow (0, 0, ..., 0)$
while $||\nabla f(x)|| \geq \epsilon$ **do**
  $x \leftarrow x - \alpha \nabla f(x)$
end while
return x

Résolution : Descente de gradient projeté

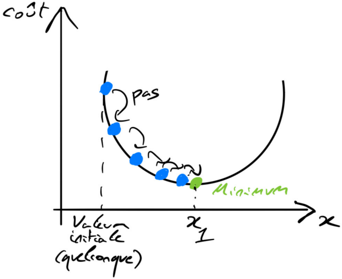
Rappel - Le problème d'optimisation est le suivant :

$$\begin{cases} \min\limits_{\omega, b} \frac{1}{2}\|\omega\|^2 \\ \text{tel que } y_i(\omega \cdot x_i + b) \geq 1, \quad i = 1, \dots, p \end{cases}$$

On projette alors sur

$$E = \{(\omega, b) \,|\, \forall i \in [|1, p|], y_i(\omega \cdot x_i + b) \geq 1\}$$

Résolution : Descente de gradient projeté



descenteDeGradientProjeté (α, ϵ, E) :

$x \leftarrow (0, 0, ..., 0)$

while $||\nabla f(x)|| \geq \epsilon$ **do**

$x \leftarrow x - \alpha \nabla f(x)$

$x \leftarrow P_E(x)$

end while

return x

Résolution : Descente de gradient projeté

Algorithme de projection alternée :

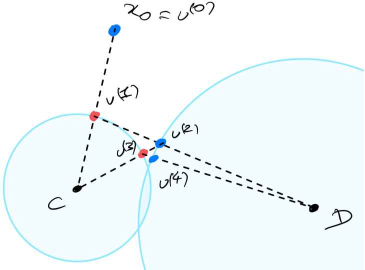


Figure – Projection Onto Convex Sets (POCS)

Résolution : Descente de gradient projeté

Algorithme de projection de Dykstra :

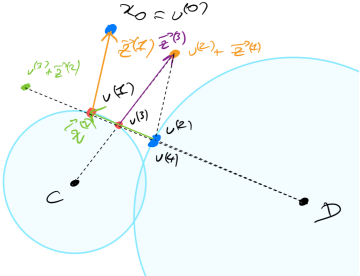


Figure – Algorithme de projection de Dykstra

Résolution : Descente de gradient projeté

Un résultat :

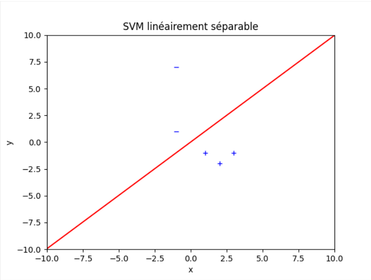


Figure – Classement de 5 points, cas linéairement séparable (pas $= 2 \cdot 10^{-6}$, limite $= 2 \cdot 10^{-1000}$)

Sommaire

Cas non linéairement séparable

Principe de la marge souple :

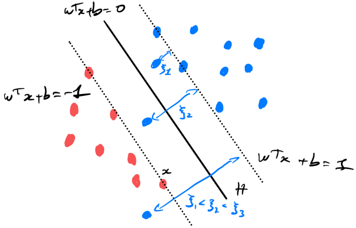


Figure – Visualisation des coûts des mauvais classements

Cas non linéairement séparable

J'ai choisi le "coût charnière" : $\xi_i = \max\left(0, 1 - y_i(\omega \cdot x_i + b)\right)$

Ainsi, le problème d'optimisation devient :

$$\min_{\omega, b} \begin{cases} \frac{1}{2}\|\omega\|^2 \\ \sum_{i=0}^{n} \xi_i \end{cases}$$

On introduit une variable C, pour avoir :

$$\min_{\omega, b} \frac{1}{2}\|\omega\|^2 + C\sum_{i=0}^{n} \xi_i$$

Quelques résultats :

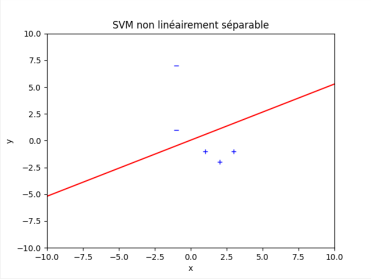


Figure – Classement de 5 points, cas non linéairement séparable (pas $= 2 \cdot 10^{-10}$, limite $= 2 \cdot 10^{-8}$, C $= 1 \cdot 10^{10}$)

Si on diminue C :

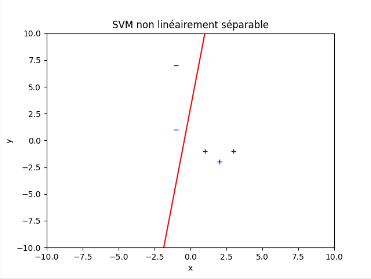


Figure – Classement de 5 points, cas non linéairement séparable (pas $= 2 \cdot 10^{-10}$, limite $= 2 \cdot 10^{-8}$, C $= 1 \cdot 10^{5}$)

Un exemple non linéairement séparable :

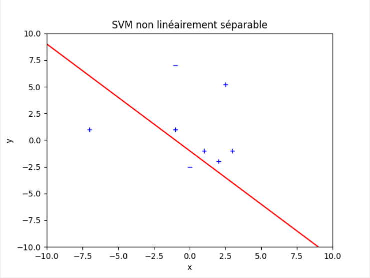


Figure – Classement de 8 points, cas non linéairement séparable (pas $= 2 \cdot 10^{-8}$, limite $= 2 \cdot 10^{-5}$, C $= 1 \cdot 10^{2}$)

Sommaire

Résultats

Malheureusement, la complexité de l'algorithme étudié est trop grande :

```
nombre d'éléments dans la classe 1 : 277
nombre d'éléments dans la classe -1 : 323
proportion de la classe 1 : 0.46166666666666667
nombre de points dans le training set : 60
nombre de points dans le testing set : 540
précision moyenne : 0.40370370370370373
duree : 1.4227503690000003
```

Idées d'amélioration :

- Descente de gradient à pas variable
- Limitation du nombre de coups considérés
- Limitation du temps d'exécution
- Utilisation d'une machine plus puissante
- Retour aux multiplicateurs de Lagrange

Deux conclusions possibles :

- Le jeu humain n'est pas si facile à différencier du jeu des IA
- Il faudrait entraîner mon algorithme sur plus de parties

Merci pour votre écoute !

Figure – source : Buzzwebzine