

Perceptron (Classification linéaire)

Cours 4
ARF Master DAC

Nicolas Baskiotis

`nicolas.baskiotis@lip6.fr`

`http://webia.lip6.fr/~baskiotis`

équipe MLIA, Laboratoire d'Informatique de Paris 6 (LIP6)
Sorbonne Université - Université Pierre et Marie Curie (UPMC)

S2 (2017-2018)

Résumé des épisodes

Notions abordées

- Ensemble d'apprentissage : $\mathcal{D} = \{(\mathbf{x}^i, y^i) \in \mathcal{X} \times \mathcal{Y}\}_{i=1, \dots, N}$, $\mathcal{X} \in \mathbb{R}^d$
- Ensemble de test
- Régression : $\mathcal{Y} \in \mathbb{R}$
- Apprentissage supervisée : \mathcal{Y} discret
- Apprentissage non supervisée : pas de \mathcal{Y}
- Apprentissage : trouver $f : \mathcal{X} \rightarrow \mathcal{Y}$ qui fait le moins d'erreurs
- Erreur : 0 – 1, moindres carrés
- Sélection de modèle : validation croisée

Approches et algorithmes

- Estimation de densité
- Arbres de décision
- Classifieur bayésien, décision bayésienne, vraisemblance
- Régression logistique
- Descente de gradient

Plan

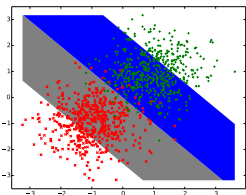
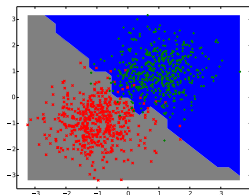
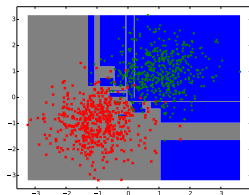
1 Apéro

2 Perceptron

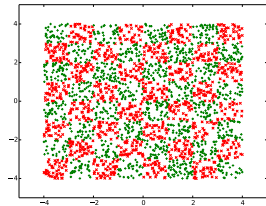
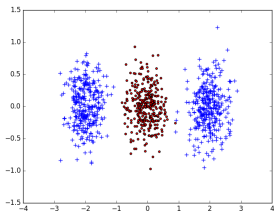
3 Interlude géométrique

On réfléchit un peu

Quelle approche correspond à quelle frontière ?

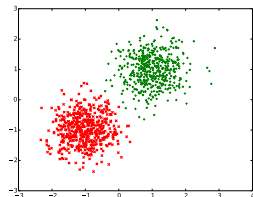


Et pour ces cas ?

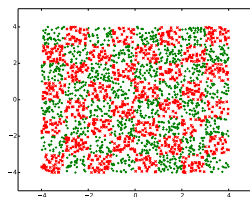
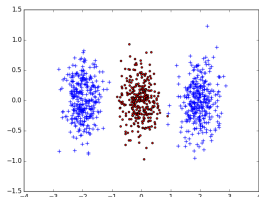


Espace linéairement séparable

Linéairement séparable



Non linéairement séparable



Conclusion (temporaire)

Importance :

- de l'espace de fonctions considéré (choisi a priori)
 - du paramétrage des algorithmes
- notion d'expressivité ...

à suivre.

Plan

1 Apéro

2 **Perceptron**

3 Interlude géométrique

Inspiration biologique

Le cerveau

- Robuste, tolérant aux fautes
- Flexible, sait s'adapter
- Gère les informations incomplètes
- Capable d'apprendre

Composé de neurones !

- 10^{11} neurones dans un cerveau humain
- 10^4 connexions par neurones
- Potentiel d'action, neuro-transmetteurs, période réfractaire
- Signaux excitateurs / inhibiteurs

Problèmes

- Opacité des raisonnements
- Opacité des résultats

Inspiration biologique

Le cerveau

- Robuste, tolérant aux fautes
- Flexible, sait s'adapter
- Gère les informations incomplètes
- Capable d'apprendre

Composé de neurones !

- 10^{11} neurones dans un cerveau humain
- 10^4 connexions par neurones
- Potentiel d'action, neuro-transmetteurs, période réfractaire
- Signaux excitateurs / inhibiteurs

Problèmes

- Opacité des raisonnements
- Opacité des résultats

Inspiration biologique

Le cerveau

- Robuste, tolérant aux fautes
- Flexible, sait s'adapter
- Gère les informations incomplètes
- Capable d'apprendre

Composé de neurones !

- 10^{11} neurones dans un cerveau humain
- 10^4 connexions par neurones
- Potentiel d'action, neuro-transmetteurs, période réfractaire
- Signaux excitateurs / inhibiteurs

Problèmes

- Opacité des raisonnements
- Opacité des résultats

Historique

Prémisses

- McCulloch et Pitts (1943) : 1er modèle de neurone formel. Base de l'IA
- Règle de Hebb (1949) : apprentissage par renforcement du couplage synaptique

Premières réalisations

- Adaline (Widrow-Hoff, 1960)
- Perceptron (Rosenblatt, 1958-1962)
- Analyse de Minsky et Papert (1969)

Développement

- Réseau bouclé (Hopfield 1982)
- Réseau multi-couches (1985)

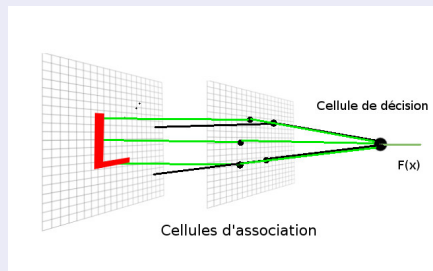
Deuxième renaissance

- Réseaux profonds (2000-)

Le perceptron de Rosenblatt (1960)

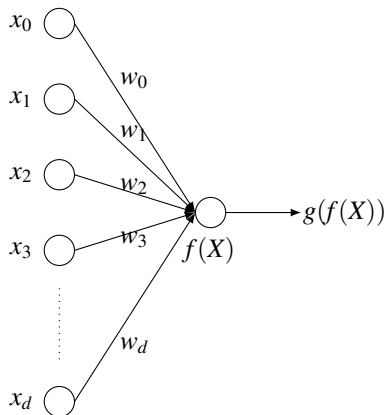
L'idée

- Reconnaissance de forme (*pattern*) entre deux classes
- Inspirée du cortex visuel



- Chaque cellule d'association produit une sortie $f_i(S)$ en fonction d'un stimulus
- La cellule de décision répond selon une fonction seuil $f_d(\sum w_i f_i(S_i))$

Formalisation



Le perceptron considère

- $f(\mathbf{x}) = \sum_{i=1}^d x_i w_i = \langle \mathbf{x}, \mathbf{w} \rangle$
- Fonction de décision :
 $g(x) = \text{sign}(x)$
- Sortie : $g(f(\mathbf{x})) = \text{sign}(\langle \mathbf{x}, \mathbf{w} \rangle)$

Algorithme d'apprentissage

Algorithme du perceptron

- Initialiser au hasard \mathbf{w}
- Tant qu'il n'y a pas convergence :
 - ▶ pour tous les exemples (x^i, y^i) :
 - ★ si $(y^i \times \langle \mathbf{w}, \mathbf{x}^i \rangle) < 0$ alors $\mathbf{w} = \mathbf{w} + \epsilon y^i \mathbf{x}^i$
- Décision : $f(x) = \text{sign}(\langle \mathbf{w}, \mathbf{x} \rangle)$

Théorème de convergence (Novikov, 1962)

- Si
 - ▶ $\exists R, \forall x : \|x\| \leq R$
 - ▶ les données peuvent être séparées avec une marge ρ
 - ▶ l'ensemble d'apprentissage est présenté au perceptron un nombre suffisant de fois
- alors après au plus R^2/ρ^2 corrections, l'algorithme converge.

Autre formulation

A quoi correspond la règle de mise à jour :

- Si $(y < \mathbf{w} \cdot \mathbf{x} >) > 0$ ne rien faire
- Si $(y < \mathbf{w} \cdot \mathbf{x} >) < 0$ corriger $\mathbf{w} = \mathbf{w} + y\mathbf{x}$?

Autre formulation

A quoi correspond la règle de mise à jour :

- Si $(y < \mathbf{w} \cdot \mathbf{x}) > 0$ ne rien faire
- Si $(y < \mathbf{w} \cdot \mathbf{x}) < 0$ corriger $\mathbf{w} = \mathbf{w} + y\mathbf{x}$?

Hinge loss

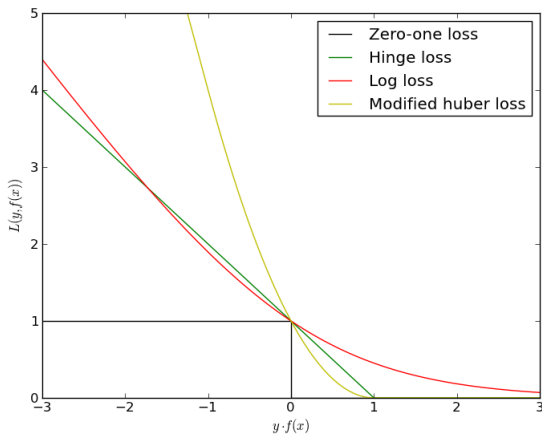
$$l(f(x), y) = \max(0, \alpha - yf(x))$$

Descente de gradient !

- $l(f_{\mathbf{w}}(x), y) = \max(0, -y < \mathbf{w} \cdot \mathbf{x} >)$
- $\nabla_{\mathbf{w}} l(f_{\mathbf{w}}(x), y) = \begin{cases} 0 & \text{si } (y < \mathbf{w} \cdot \mathbf{x}) > 0 \\ -yx_i & \text{sinon} \end{cases}$

Pourquoi ce changement dans la fonction de coût ?

Et pourquoi pas d'autres erreurs ?

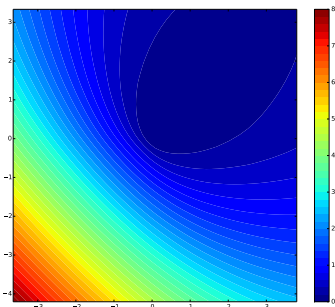
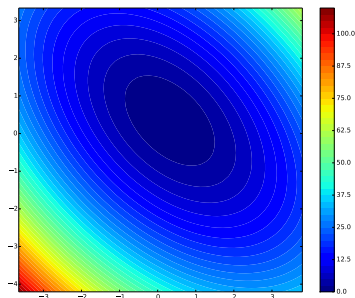


Exploration de l'espace des solutions

Vision duale de l'espace des exemples

- $R_{hinge}(f_{\mathbf{w}}) = \mathbb{E}(l_{hinge}(f_{\mathbf{w}}(x), y)) = \mathbb{E}(\max(0, -f_{\mathbf{w}}(x)y))$
- $R_{mse}(f_{\mathbf{w}}) = \mathbb{E}(l_{mse}(f_{\mathbf{w}}(x), y)) = \mathbb{E}((f_{\mathbf{w}}(x) - y)^2)$

Pour un ensemble fixé de données, le risque est vu comme une fonction de \mathbf{w} .



Variantes de l'algorithme

- Cas hors-ligne (ou batch) :
Pour chaque époque (correction de w), on itère sur toute la base d'exemples
- Cas en-ligne (stochastique) :
Une correction de w est faite par rapport à un exemple tiré au hasard dans la base.
- hybride : mini-batch
Des petits sous-ensembles d'exemples sont tirés au hasard, la correction se fait selon le gradient calculé sur ces exemples.

Avantages et inconvénients ?

- Batch : plus stable, plus rapide
- Stochastique : bien meilleure tolérance au bruit !

Plan

1 Apéro

2 Perceptron

3 Interlude géométrique

Considérations géométriques

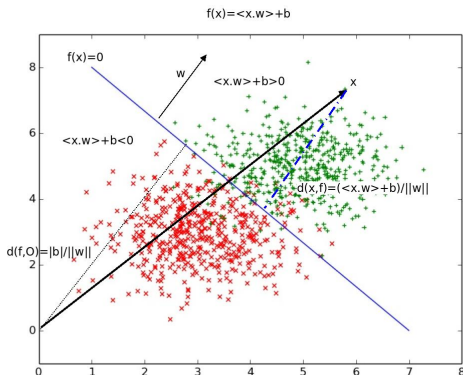
Soit y la sortie attendue :

- Que représente w par rapport à la séparatrice ?
- Que représente $\langle w \cdot x \rangle$?
- Que représente $y \langle w \cdot x \rangle$?

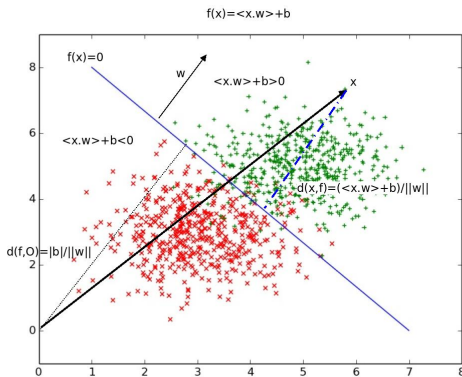
Considérations géométriques

Soit y la sortie attendue :

- Que représente w par rapport à la séparatrice ?
- Que représente $\langle w \cdot x \rangle$?
- Que représente $y \langle w \cdot x \rangle$?
- A quoi correspond la règle de mise à jour :
 - ▶ Si $(y \langle w \cdot x \rangle) > 0$ ne rien faire
 - ▶ Si $(y \langle w \cdot x \rangle) < 0$ corriger $w = w + yx$?



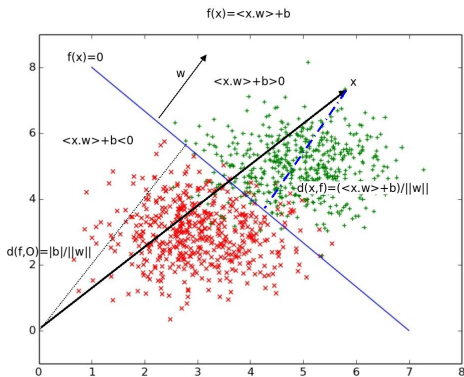
Considérations géométriques



Rappel

Espace de fonction linéaire : $f_w(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} = \sum_{i=0}^d w_i x_i$, prédiction : $\text{sign}(f_w(\mathbf{x}))$

Considérations géométriques

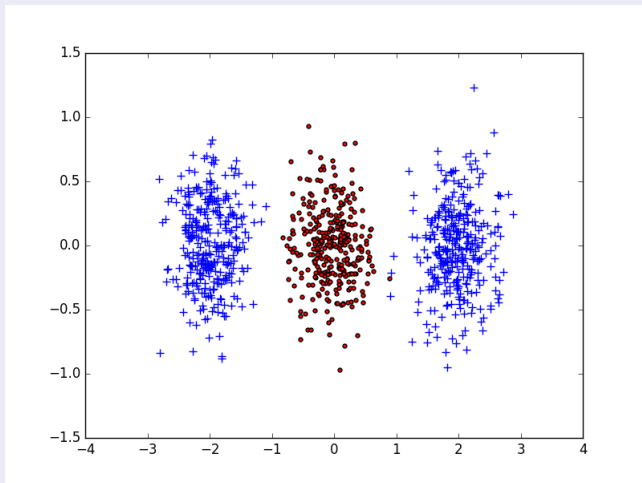


Questions

- Solution unique ?
- Certaines solutions meilleures que d'autres ?

Problèmes “durs”

Non linéairement séparable

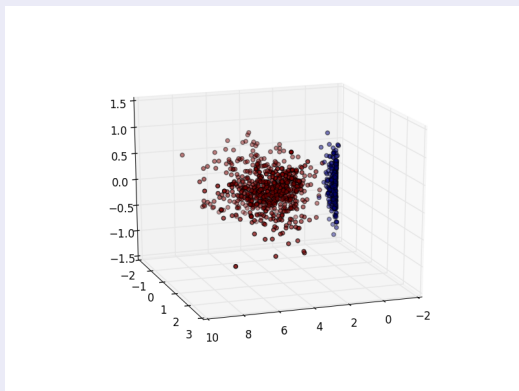


Que faire ?

Problèmes “durs”

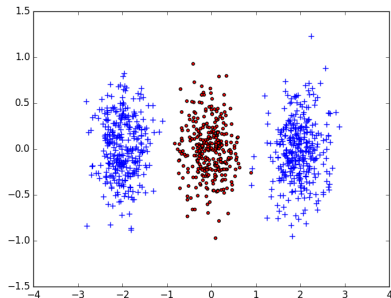
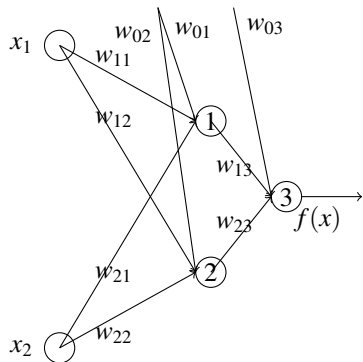
Transformation de la représentation

- On augmente d'une dimension : $(x_1, x_2) \rightarrow (x_1^2, x_1x_2, x_2)$



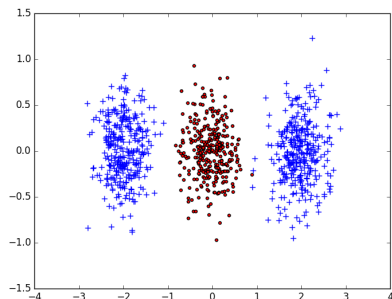
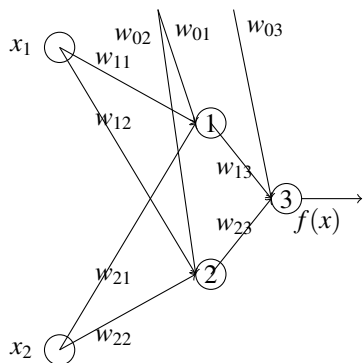
- Le problème est de nouveau séparable linéairement !
- Autre solution ?

Deux neurones



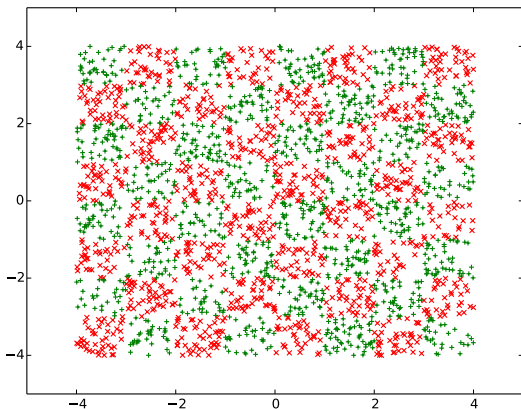
- Combiner des neurones \rightarrow augmente l'expressivité
 - Création de dimensions nouvelles, de nouveaux features
- \Rightarrow Est-il facile d'apprendre ses réseaux ?

Deux neurones



- Combiner des neurones \rightarrow augmente l'expressivité
 - Création de dimensions nouvelles, de nouveaux features
- \Rightarrow Est-il facile d'apprendre ses réseaux ?

Et pour ce problème ?



Si vous aviez droit à n'importe quelle fonction dans un neurone ?