

# Classification d'objets par réseaux de neurones

Alice LEBRUN

28 mars 2025

## Contents

|          |  |          |
|----------|--|----------|
| <b>1</b> | <b>Introduction</b>                        | <b>1</b> |
| <b>2</b> | <b>Visualisation de la base de données</b> | <b>2</b> |
| <b>3</b> | <b>Comparaison des classifications</b>     | <b>2</b> |
| <b>4</b> | <b>Améliorations possibles</b>             | <b>5</b> |

## 1 Introduction

Le but de ce projet est de construire un algorithme de classification (=classifieur) permettant de prédire l'état (efficace/inefficace) d'un objet décrit par trois paramètres (nombre d'heures d'utilisation quotidienne, consommation électrique et âge de l'objet en mois).

Le notebook jupyter NT306\_Alice\_LEBRUN\_sujet\_2.ipynb contient les fonctions de lecture de la base de données, de visualisation, de définition des fonctions de coût et des algorithmes d'optimisation pour la construction d'un tel classifieur basé sur la régression logistique, ainsi que la construction de classifieurs utilisant des perceptrons de complexité croissante.

Il contient l'ensemble des développements spécifiés dans le document evaluation-projet2.pdf et complète le canevas de notebook proposé sur le site <https://grosjean1.github.io/post/project>.

## 2 Visualisation de la base de données

Les différents objets présents dans la base de données sont dénommés Camera (1087 occurrences), Lights (1087 occurrences), Security System (1068 occurrences), Smart Speaker (1108 occurrences) et Thermostat (1039 occurrences).

On représente graphiquement cette base de données sur la figure (1).

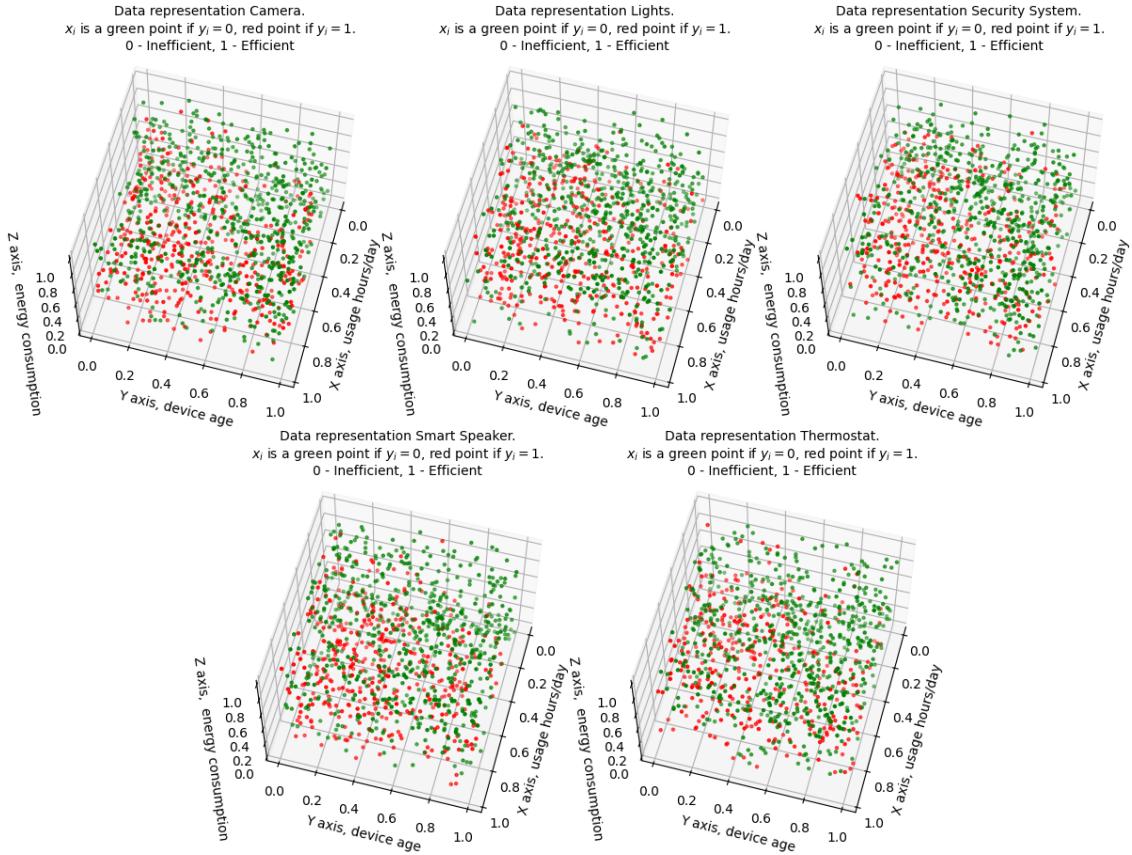


Figure 1: Visualisation de la base de données.

## 3 Comparaison des classifications

Dans cette section, nous comparons la performance des différents classificateurs (logistique=perceptron zéro couche, perceptron une couche et perceptron cinq couches) sur les différents types d'objets.

Le critère de performance log-loss d'un classifieur s'interprète de la manière suivante:

- Pour une classification parfaite (état prédit=état réel pour chaque instance) il vaut zéro;
- Pour une classification complètement fausse (état prédit opposé à l'état réel pour chaque instance) il vaut  $+\infty$
- Plus il est proche de zéro, plus la classification est performante.

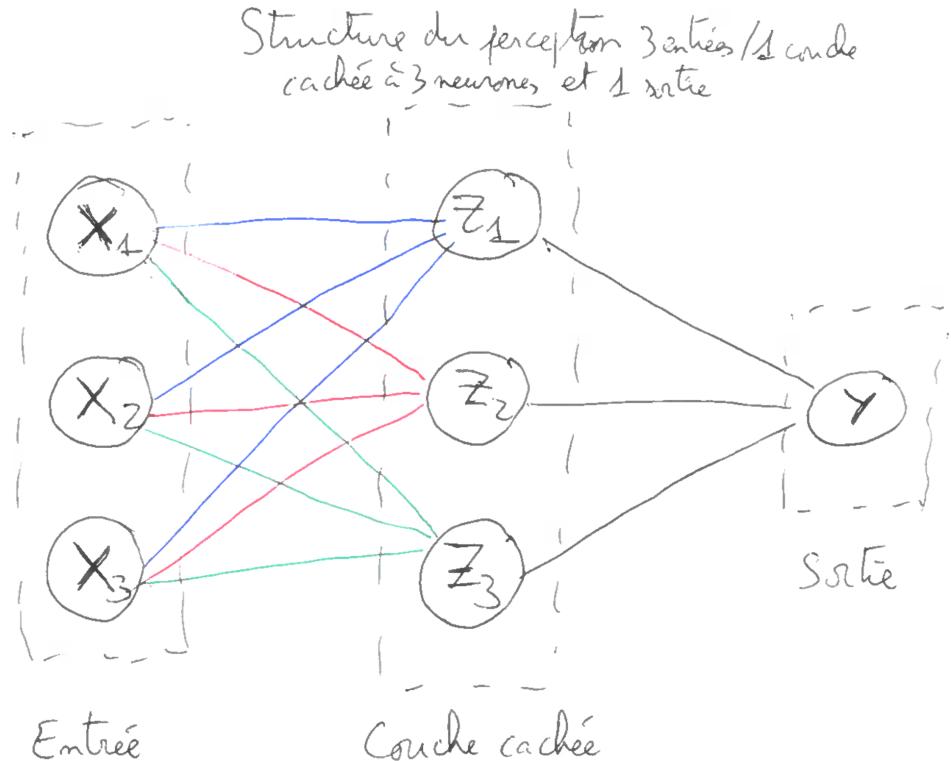


Figure 2: Structure du perceptron à 3 entrées, une couche cachée à 3 neurones et une sortie.

On constate dans le tableau 1 que les classificateurs les plus performants sont également les plus complexes, c'est-à-dire ceux ayant le plus grand nombre de paramètres.

La structure du classifieur perceptron à une couche cachée de trois neurones est indiquée sur la figure 2. Chaque neurone (ceux de la couche cachée et celui de la sortie) correspond à des coefficients à calculer:

- Autant coefficients de la combinaison linéaire reliant le neurone à ses antécédents. Ici, il y a autant de neurones dans la couche cachée que de paramètres en entrée du perceptron (3) donc il y a  $4 \times 3 = 12$  coefficients correspondant à ces combinaisons linéaires.
- Autant de biais que de neurones: il y a donc 4 coefficients de biais à calculer.

Au bilan, le perceptron à trois entrées, une sortie et une couche cachée à trois neurones comporte 16 coefficients.

|             | Camera       | Lights       | Security System | Smart Speaker | Thermostat   |
|-------------|--------------|--------------|-----------------|---------------|--------------|
| Logistic    | 0.611        | 0.599        | 0.600           | 0.612         | 0.597        |
| One layer   | 0.593        | <b>0.579</b> | 0.563           | 0.584         | <b>0.570</b> |
| Five layers | <b>0.557</b> | 0.591        | <b>0.530</b>    | <b>0.563</b>  | 0.581        |

Table 1: Performance des classifications selon l'indicateur log-loss. Plus la valeur est faible, meilleure est la classification. ( $\lambda = 10^{-4}$  pour la pénalisation  $L_2$  de tous les classificateurs)

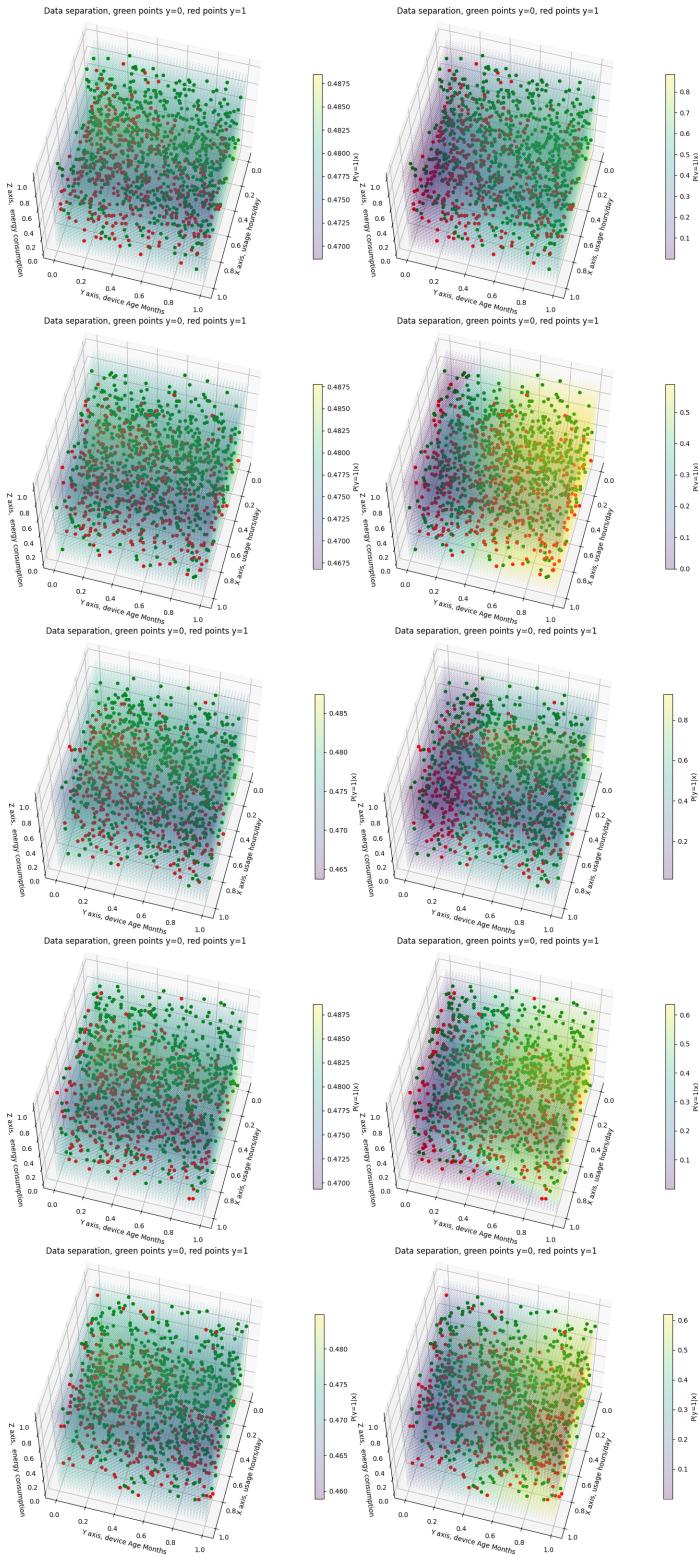


Figure 3: Visualisation du classifieur logistique (gauche) et perceptron 1 couche cachée (droite).

## 4 Améliorations possibles

L'apprentissage des classifieurs est faite en utilisant la totalité de la base. Mesurer leur performance sur cette même base est donc trompeur puisqu'on risque d'avoir construit un classifieur optimisé pour les instances de cette base d'apprentissage au détriment de sa capacité à classer correctement une nouvelle instance inconnue.

Une manière de se prémunir de ce risque serait de séparer la base de données en deux parties: une partie utilisée pour l'apprentissage (base d'apprentissage) et une partie (base de test) utilisée pour évaluer la performance du classifieur (performance de test).

Par ailleurs, la construction du classifieur fait intervenir un paramètre de régularisation qu'on peut vouloir optimiser. Une manière de procéder consiste à découper la base d'apprentissage en deux: une base d'apprentissage effectif et une base de validation.

Pour une valeur fixée du paramètre de régularisation on utilise la base d'apprentissage effectif pour obtenir la valeur des paramètres du classifieur, puis la base de validation est utilisée pour évaluer la performance du classifieur ainsi obtenu (performance de validation).

On fait varier la valeur du paramètre de régularisation dans un intervalle de manière à sélectionner la valeur conduisant à la meilleure performance de validation. On calcule alors la performance de test du classifieur optimal.

Pour la base considérée dans le projet, le temps d'apprentissage pour une valeur donnée du paramètre de régularisation est très court ( $<0.1s$ ), il serait donc possible de mettre en œuvre cette approche sans souffrir de temps d'apprentissage trop longs.