Justice Prédictive et Réseaux de Neurones*

Gildas, Gilles, Guillaume, Jacky, Sébastien, Stéphane
EMT Mines Alès et Chrome Université de Nîmes

Preliminary version

Abstract

1 Notations

Matrice $\mathbf{X}$ de taille $n \times p$ des variables (inputs) X_j, $j = 1, \ldots, p$.

$\mathbf{X} = [x_{ij}]$ avec i l'individu (observation) étudié $i = 1, \ldots, n$ et j la dimension tel que $j = 1, \ldots, p$.

$\mathbf{x}_i$ $i$ème ligne de $\mathbf{X}$: individu i avec p dimensions

Vecteur des sorties (outputs) $\hat{\mathbf{Y}}$ $n \times s$

Matrice des pondérations (poids) $\mathbf{W}$ de taille $s \times p$

Vecteur des pondérations associé à l'individu i $\mathbf{w}_i$ de taille $1 \times p$. On notera $\mathbf{w}_i(t)$ la pondération associée à l'individu i à l'étape d'apprentissage (itération) t.

2 Perceptron

2.1 Perceptron simple

Le modèle du Perceptron (simple / multiple) consiste à appliquer une ou plusieurs couche de neurones afin de séparer de manière linéaire (dans le cas

*Cette recherche a été menée dans le cadre du contrat #**** du Minsitère de la culture...

simple) ou non linéaire (dans le cas multiple) des nuages de points individus que l'on souhaite classer et dont on souhaite prédire le comportement. Le modèle est binaire dans le cas simple, $\hat{Y}_i = 0, 1$, il est multinomial dans le cas multiple $\hat{Y}_{ki} = 0, 1$ pour chaque classe créée telle que $k = 1, \ldots, K$.

L'idée est de choisir de manière aléatoire des poids que l'on associe aux signaux d'entrée *i.e.* aux variables X_j. Une observation $\mathbf{x}_i$, prise au hasard, est associée aux pondérations aléatoires $\mathbf{w}$, de telle sorte que le modèle linéaire $\mathbf{x}_i\mathbf{w}^\intercal + c$ (avec c une constante que l'on appelle le biais) permet de prédire si l'individu est classé dans le groupe de modalité 0 ou 1. Il sera classé dans le groupe de modalité 0, si $\mathbf{x}_i\mathbf{w}^\intercal + c \leq 0$, il sera classa dans le groupe de modalité 1, si modalité 1 si $\mathbf{x}_i\mathbf{w}^\intercal + c > 0$. En d'autres termes:

$$\hat{Y}_i \equiv f(\mathbf{x}_i\mathbf{w}^\intercal + c) = \begin{cases} 0 & \text{si } \mathbf{x}_i\mathbf{w}^\intercal + c \leq 0 \\ 1 & \text{si } \mathbf{x}_i\mathbf{w}^\intercal + c > 0 \end{cases} \tag{1}$$

Évidemment, puisque les poids ont été choisis de manière aléatoire, il n'y a aucune raison pour que l'individu i dont on connait le classement (observé) Y_i, soit correctement classé. L'erreur de classement, ou erreur de prévision $Y_i - \hat{Y}_i$ doit permettre de réviser les pondérations, afin de classer correctement un autre individu pris au hasard dans l'échantillon. La technique du gradient descendant issue de la minimisation de l'erreur quadratique moyenne donne une révision de la dimension j du poids $\mathbf{w}$ telle que:

$$w_j(t+1) = w_j(t) + \lambda(Y_i - \hat{Y}_i)f'(\cdot)x_j \tag{2}$$

où λ est un constante d'apprentissage. Une fois les pondérations mises à jour, un nouvel individu est pris au hasard afin de d'obtenir une estimation de son classement $\hat{Y}_i$ et de réviser à nouveau les pondérations, et ainsi de suite. Ainsi, la frontière linéaire $\mathbf{x}_i\mathbf{w}^\intercal(t)$ est révisée à chaque itération $t$de telle sorte qu'elle puisse au mieux séparer le nuage de points des individus associés à la modalité 0 avec celui des individus associés à la modalité 1.

Faisons une application de ce modèle sur les troubles anormaux du voisinage, pour lesquels les nuisances suivantes ont été sélectionnées parmi 49 textes relatifs à des décisions de justice en appel : abris, acoustique, arrachage, bruits, cloison, clôture, cuisine, eaux, empiétement, emplacement, fenêtre, gouttière, haie, limite. L'utilisation du perceptron simple permet de classer les décisions selon deux modalités : 0 lorsque le juge déboute la demande de l'appelant (ou lorsque ce dernier ne peut statuer) ; 1 lorsque le juge

accepte la demande. La structure du réseau est illustrée dans la Figure 1 où l'on constate que chaque terme caractérisant une nuisance va alimenter un neurone qui lui-même, après calibrage des pondérations, aura une influence sur le classement des 49 textes selon les modalités 0 / 1.

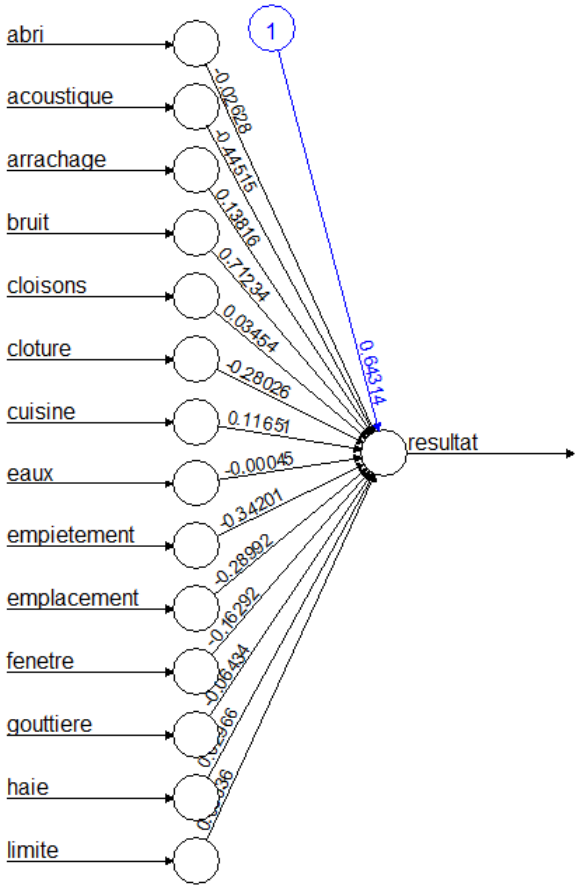

Figure 1: Perceptron (linéaire)

En entraînant le modèle sur la totalité de la base, on constate que les erreurs du modèle sont au nombre de 8 ; l'erreur étant la différence entre le code observé d'un texte (0 / 1) et le code tel qu'il estimé par le perceptron

(0 / 1). Pour chaque texte mal prédit, cette erreur vaut -1, indiquant que le modèle prévoit à tort une issue positive du procès.

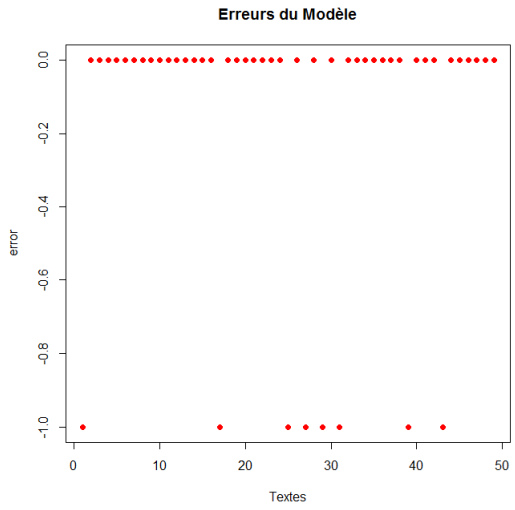

Figure 2: Erreurs du Perceptron

Le modèle peut être comparé à un modèle de régression linéaire où les paramètres sont estimés à l'aide du critère de minimisation de la variance des erreurs. Le modèle de régression linéaire s'écrit:

$$\hat{Y}_i = \mathbf{x}_i \hat{\mathbf{w}}^\intercal + \hat{c}, \tag{3}$$

où les pondérations $\hat{\mathbf{w}}$ sont données par :

$$\hat{\mathbf{w}} = \arg\min \mathrm{Var}(Y - \hat{Y}) \tag{4}$$

Sur notre jeu de données, le modèle est équivalent, en effet, les mêmes huit décisions sont mal classées, alignés à la valeur -1 en bas du graphique ci-dessous.

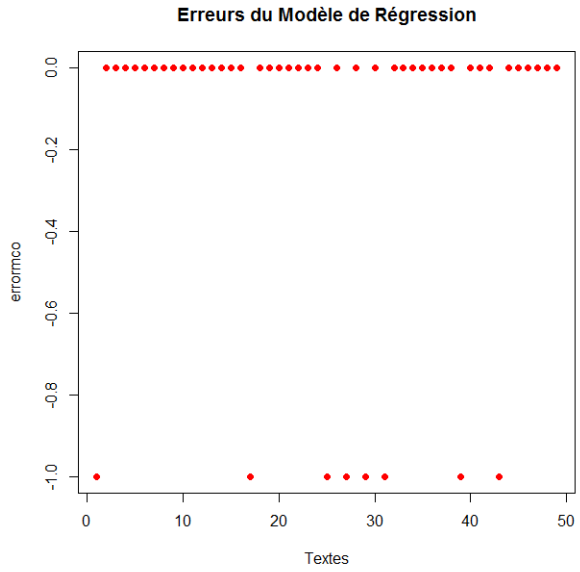

Figure 3: Erreurs du modèle de Régression

Il est clairement établit dans la littérature économétrique et statistique que le modèle de régression linéaire évalue mal la variable prédite, notamment lorsque la réponse est binaire, impliquant dans la plupart des cas que cette dernière à une forme logistique. Néanmoins, dans cet exemple précis, le modèle de régression logistique n'améliore pas les résultats, le modèle fera les mêmes erreurs de classement que le modèle de régression classique précédent, comme en témoigne le graphique des erreurs suivant.

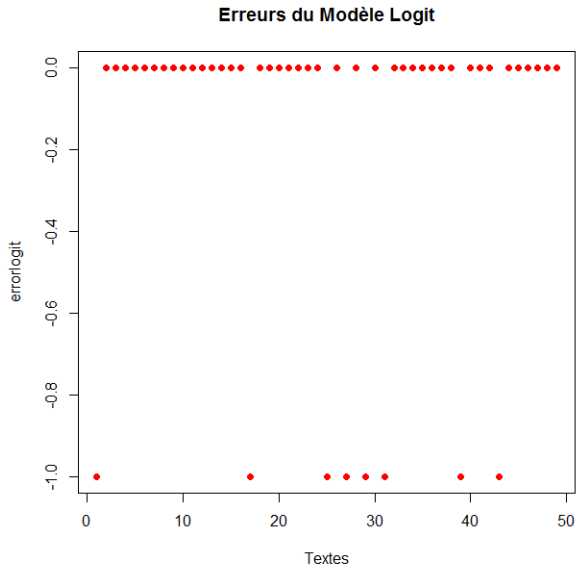

Figure 4: Erreurs du modèle Logit

Dans le cadre des réseaux de neurone, la fonction de sortie, créant l'output $\hat{Y}_i$, peut elle aussi être logistique, il s'agit du perceptron simple logistique :

$$f(u) = \frac{1}{1 + e^{-u}} \tag{5}$$

En appliquant cette fonction de sortie logistique, les résultats restent très bons, comme en témoigne des le graphique des erreurs suivant.

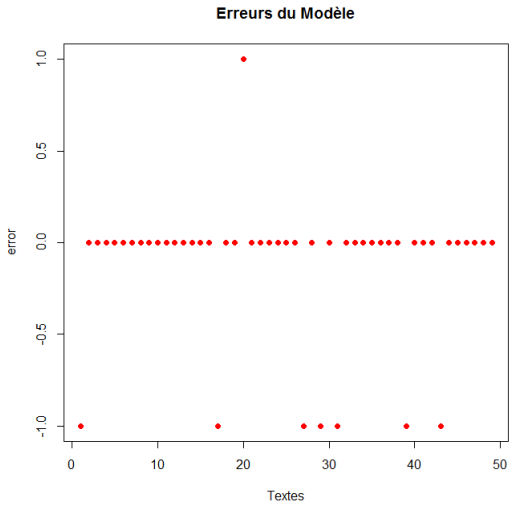

Figure 5: Erreurs du Perceptron simple (sotie log)

L'utilisation de la fonction de sortie logistique implique que les pondérations vont être mises à jour *via* l'équation 2. Les poids associés aux neurones changent, comparés au modèle linéaire, comme le montre le graphique suivant :

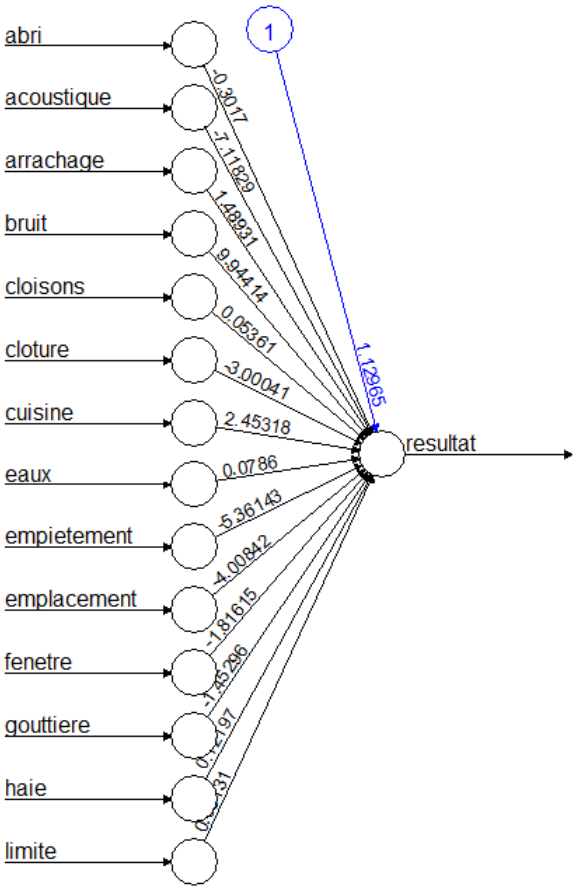

Figure 6: Perceptron Logistique

On constate que les neurones qui semblent avoir une influence sur la décision du juge, comme le "bruit", gardent un poids important : $0,71$ dans le perceptron linéaire, contre $9,94$ dans le perceptron logistique. Dans chacun des modèles, ces pondérations étant les plus fortes, elles exercent ainsi une influence non négligeable sur le résultat, comparé par exemple aux problèmes de clôture dont les pondérations sont seulement estimées à hauteur de $-0,28$ et de -3 dans les perceptrons linéaire et logistique, respectivement.

2.2 Perceptron multi-couches

Dans le cadre d'un perceptron multi-couches, une couche cachée est d'abord estimée, comme nous l'avons précédemment vu, cette couche permettant d'alimenter une autre couche de neurones nous donnant le résultat final de classification de nos individus. Plusieurs configurations sont possibles, par exemple, comme le montre la figure suivante, il est possible d'utiliser une couche cachée de deux neurones.

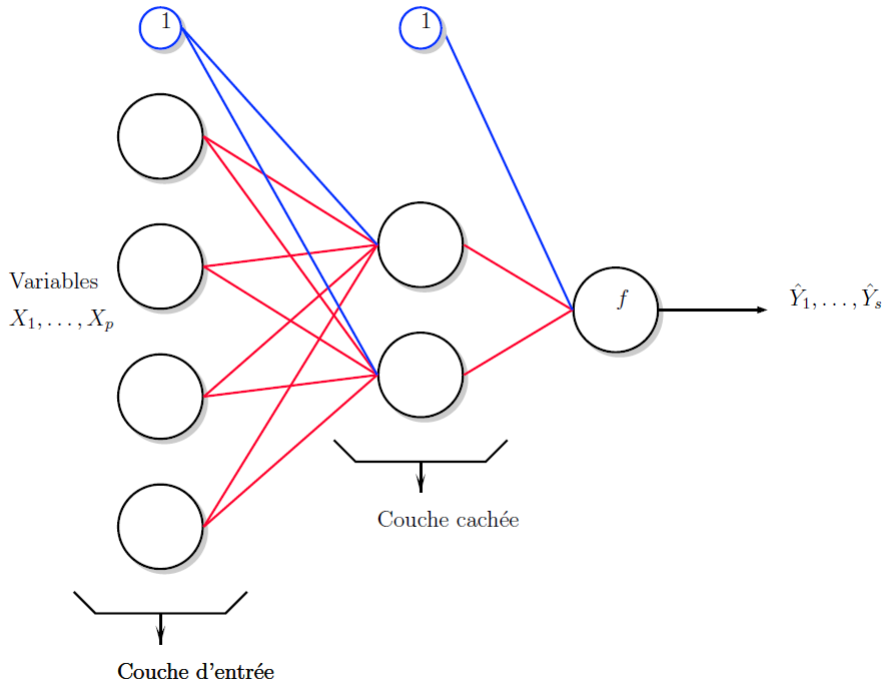

Figure 7: Perceptron Multicouches

3 Carte de Kohonen

Les cartes de Kohonen (1982) ont permis de réaliser des progrès dans le traitement d'image. L'idée est d'organiser les informations reçues par les neurones (matérialisés par des cercles dans la figure ci-dessous) regroupés à l'intérieur d'une carte dont les propriétés géométriques peuvent changer (forme rectangulaire, hexagonale, etc.). Chaque neurone va recevoir le signal X_j de chaque neurone j. Les individus proches en terme de signal X_j seront regroupés dans un même neurone ou dans des neurones connexes. Des pondérations sont générées (de manière aléatoire, de manière floue, etc.) de telle sorte que chacune est comparée aux signaux de départ (pris au hasard). Le neurone compétitif retenu sera celui qui minimisera la distance entre la pondération et le signal d'entrée. L'opération est réitérée afin de mettre à jour les pondérations avec un certain pas d'apprentissage. Au fur et à mesure de l'apprentissage, chaque neurone va se spécialiser en captant toute ou partie des variables (signaux) X_j de départ. Chaque neurone nous donnera une classification des signaux d'entrée que nous pourrons repérer dans la carte.

Prenons l'exemple des troubles du voisinage. L'idée étant de regrouper des décisions de justice similaires. Les signaux d'entrée sont les variables bruits, ensoleillement, haies, nuisances, plantations, clôtures, etc. Voyons comme une carte hexagonale peut permettre d'organiser les signaux d'entrée.

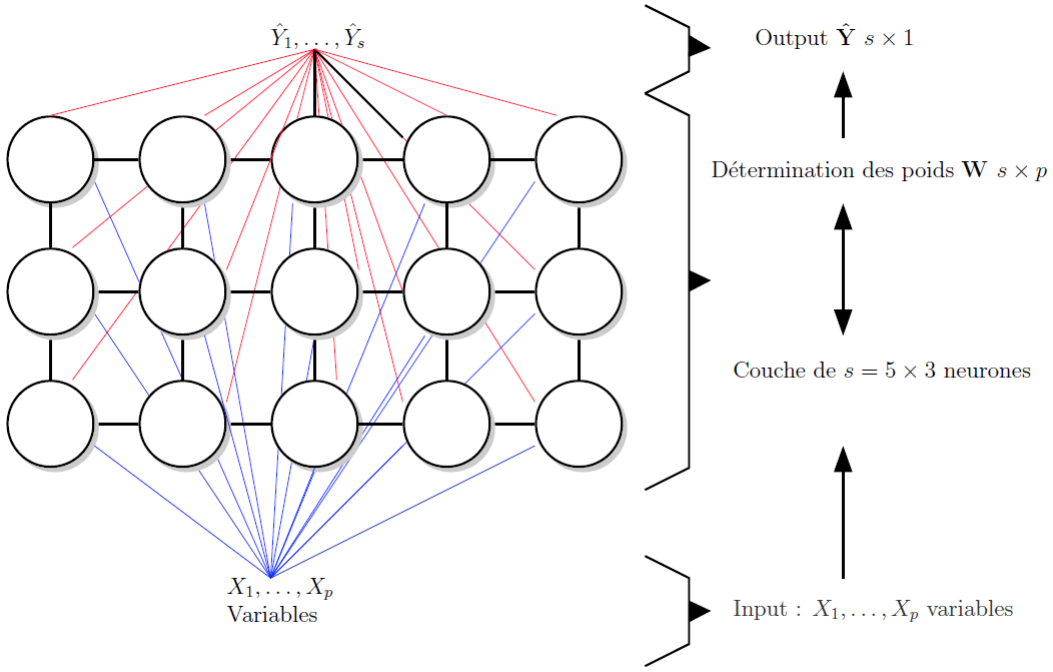

Figure 8: Carte de Kohonen

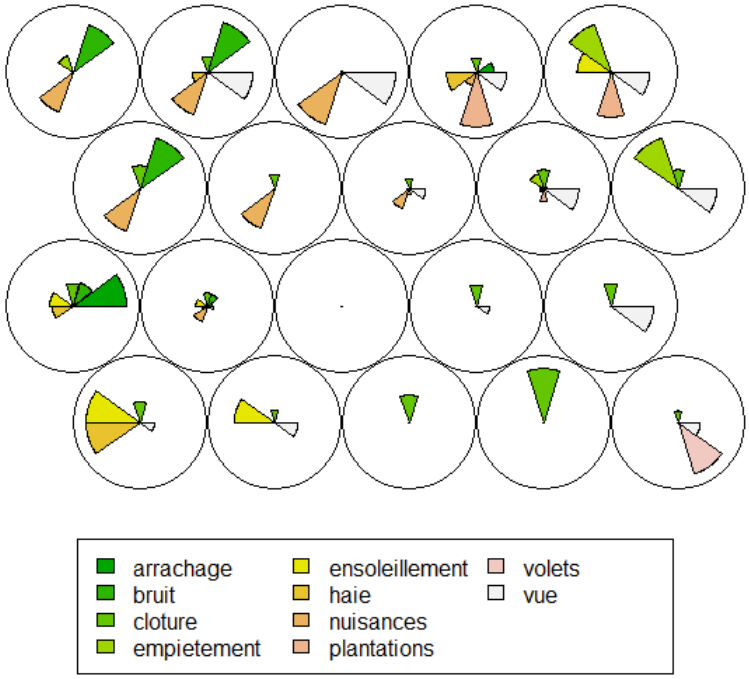

Figure 9: Troubles du voisinage

Il semble que les problèmes d'ensoleillement et de haies soient regroupés, ce qui semble logique (voir le neurone en bas à gauche). Les problèmes de vue et de volets font aussi l'objet d'un regroupement (neurone en bas à droite). Bruits et nuisances (et éventuellement vue) sont regroupés dans les neurones en haut à gauche. Enfin, empiétement et plantations sont regroupés dans le neurone en haut à droite. Lorsque les termes, évoquant les circonstances dans lesquelles les troubles de voisinages apparaissent, sont en faible nombre, alors les résultats sont cohérents comme nous pouvons le voir à la Figure 9 ci-dessus. Lorsque le nombre de terme dépasse 200, les résultats deviennent très difficiles voire impossibles à analyser.

References

Goodfellow I., Bengio Y., et A. Courville (2016), *Deep Learning*, MIT Press. Parizeau, M. (2006), *Réseaux de neurones*, Université Laval.