AutoEncoders pour l'imputation de données manquantes

Mariette DUPUY [1] & Marie CHAVENT [2] & Rémi DUBOIS [3]

[1] *IHU L'institut de Rythmologie Cardiaque (LIRYC) et Inria Bordeaux, France, mariette.dupuy@ihu-liryc.fr*
[2] *Inria et Institut de Mathématiques de Bordeaux, France, marie.chavent@u-bordeaux.fr*
[3] *IHU L'institut de Rythmologie Cardiaque (LIRYC) , France, remi.dubois@ihu-liryc.fr*

Résumé : Dans ce travail, nous nous intéressons à l'imputation des données manquantes via des méthodes basées sur les autoencoders. Les autoencoders sont des réseaux de neurones permettant de recoder les données dans un espace latent (encodage) avant de les reconstruire dans leur espace d'origine (decodage). Dans cet article, nous verrons comment ils peuvent être utilisés pour l'imputation des données manquantes et comparerons quatre méthodologies sur le jeu de données Boston Housing.

Mots-clés. Données manquantes, imputation, autoencoders

Abstract. In this work, we are interested in the imputation of missing data via methods based on autoencoders. Autoencoders are neural networks that recode data in a latent space (encoding) before reconstructing them in their original space (decoding). In this article, we will see how they can be used for missing data imputation and compare four methodologies on the Boston Housing dataset.

Keywords. Missing data, imputation, autoencoders

1 Introduction

La présence de données manquantes est un problème récurrent dans un monde où la collecte de données est permanente. C'est par exemple le cas pour des données construites sur l'enregistrement d'un signal par des capteurs, l'outil pouvant ne plus fonctionner sur certaines périodes et pouvant ainsi créer des données manquantes.

On retrouve deux approches classiques pour gérer les données manquantes : supprimer les observations possédant des valeurs manquantes ou les imputer. Il existe de très nombreuses méthodes d'imputation allant d'une simple complétion par la moyenne, à des méthodes par agrégation locale (de type k plus proches voisins), régressions locales, décomposition en valeurs singulières (SVD) ou encore inférence bayésienne. Récemment les Autoencoders (AE) et plus précisément les Denoising AutoEncoder (DAE) ont été utilisés comme méthode d'imputation (Pereira, R. *et al.* (2020)). Les méthodes citées dans cet article de review ne fonctionnent que dans le cas où il y a suffisamment de données observées pour séparer le jeu

de données en deux groupes: un groupe d'observations complètes utilisées dans une phase d'apprentissage et le groupe des observations incomplètes imputées lors de la phase de test.

Dans cet article, nous allons décrire et comparer deux méthodologies simples utilisant les AE et les DAE pour imputer des données manquantes dans le cas où il n'est pas possible d'extraire un groupe d'observations complètes pour l'apprentissage. La première consiste à simplement pré-imputer les données manquantes avant d'appliquer un AE ou un DAE et la seconde à prendre en compte en plus la présence de données manquantes dans la fonction de perte. Nous comparerons ces méthodes sur le jeu de données Boston Housing (Harrison, D. et Rubinfeld, D.L. (1978)) et présenterons quelques résultat préliminaires à titre d'illustration.

2 Autoencoder (AE)

On considère un échantillon de n observations $\mathbf{x}_i = (x_{i1}, ..., x_{ip})^T$ de $\mathbb{R}^p$ qui constituent une matrice de données $\mathbf{X}$ de dimension $n \times p$.

Un AE (Bengio Y. (2009)) est un réseau de neurones structuré en deux parties : une partie pour *encoder* les données et une partie pour les *décoder*. La FIGURE 1 représente par exemple un autoencoder basique schématisé par un réseau de neurones à trois couches.

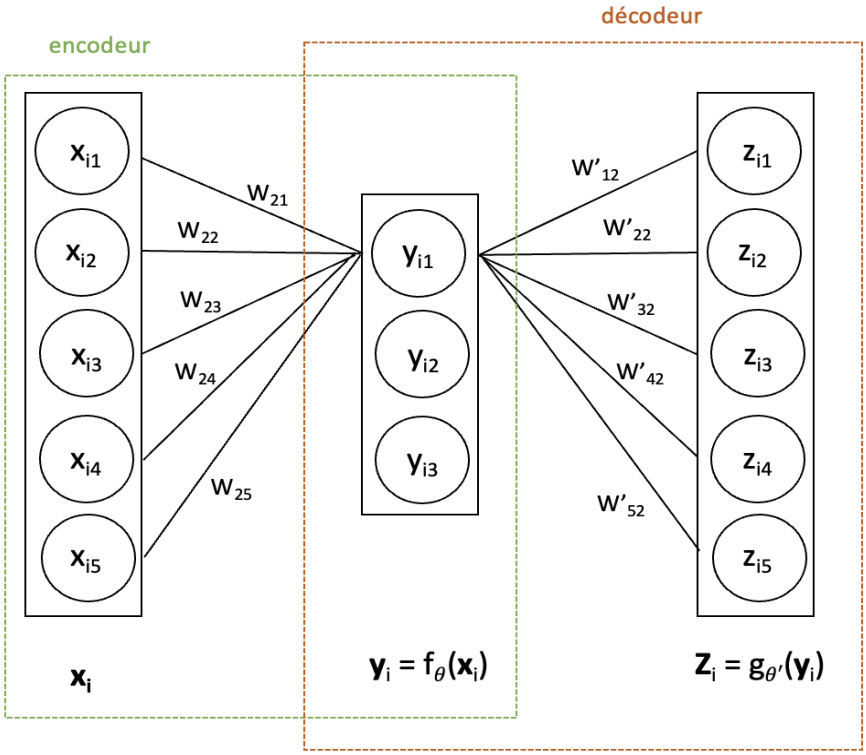


Figure 1: Schéma d'un autoencoder basique

Dans le cas particulier d'un AE basique, l'**encodeur** f_θ transforme une entrée $\mathbf{x}_i \in \mathbb{R}^p$ en un vecteur latent $\mathbf{y}_i \in \mathbb{R}^q$ aussi appelé vecteur caractéristique ou code :

$$\mathbf{y}_i = f_\theta(\mathbf{x}_i) = s(\mathbf{W}\mathbf{x}_i + \mathbf{b}),$$

où $\mathbf{W} \in \mathbb{R}^{q \times p}$ est une matrice de poids, $\mathbf{b} \in \mathbb{R}^p$ un vecteur de biais, s une fonction d'activation (e.g. ReLU ou sigmoïd). Lorsque $q \geq p$, l'étape d'encodage n'est plus une étape de réduction de dimension et le réseau est dit *overcomplete*.

Le **décodeur** $g_{\theta'}$ transforme ensuite le vecteur latent $\mathbf{y}_i \in \mathbb{R}^q$ en une sortie $\mathbf{z}_i \in \mathbb{R}^p$:

$$\mathbf{z}_i = g_{\theta'}(\mathbf{y}_i) = s(\mathbf{W}'\mathbf{y}_i + \mathbf{b}')$$

où $\mathbf{W}' \in \mathbb{R}^{p \times q}$ et $\mathbf{b}' \in \mathbb{R}^q$. On prendra souvent $\mathbf{W}^{'} = \mathbf{W}^T$.

En général, un AE possède plus de trois couches et sa structure est définie par le nombre K de couches cachées de l'encodeur et du décodeur (le réseau est souvent symétrique), le nombre de neurones par couche et la fonction d'activation s qui est en général la même sur toutes les couches cachées. Les paramètres $\theta = (\mathbf{W}_1, ..., \mathbf{W}_K, \mathbf{b}_1, ..., \mathbf{b}_K)$ et $\theta' = (\mathbf{W}'_1, ..., \mathbf{W}'_K, \mathbf{b}'_1, ..., \mathbf{b}'_K)$ devront alors minimiser l'**erreur de reconstruction** e.g. :

$$\theta, \theta' = \arg\min_{\theta,\theta'} \sum_{i=1}^{n} \|\mathbf{x}_i - (g_{\theta'} \circ f_\theta)(\mathbf{x}_i)\|^2 = \sum_{i=1}^{n} \underbrace{\|\mathbf{x}_i - \mathbf{z}_i\|^2}_{L(\mathbf{x}_i, \mathbf{z}_i)} = \|\mathbf{X} - \mathbf{Z}\|_F^2$$

où F est la norme de Frobenius.

Ce critère va favoriser la reconstruction des variables (colonnes de $\mathbf{X}$) de forte variance. La matrice de données $\mathbf{X}$ est donc généralement standardisée.

Denoising Autoencoder (DAE) Les DAE (Vincent, P. *et al.* (2008)) sont une variante des AE développée pour construire un réseau entraîné à débruiter une version corrompue des données d'entrée. Un DAE est donc un AE qui reçoit des données d'entrée corrompues, par exemple par un bruit Gaussien, et qui est entraîné à prédire les données d'origine non corrompues. Dans ce papier nous utiliserons le *Salt and Peper noise* (Vincent, P. *et al* (2010)) qui consiste, pour chaque entrée $\mathbf{x}_i \in \mathrm{R}^p$, à mettre une proportion $\mu \in [0, 1]$ de composantes aléatoirement à 0. On notera $N(\mathbf{x}_i)$ le vecteur $\mathbf{x}_i$ corrompu. Ce type de DAE est schématisé dans la FIGURE 2. On peut noter également qu'un DAE s'interprètent aussi comme un AE régularisée (Alain, G. et Bengio, Y. (2012)), le paramètre de régularisation étant dans notre cas la proportion μ.

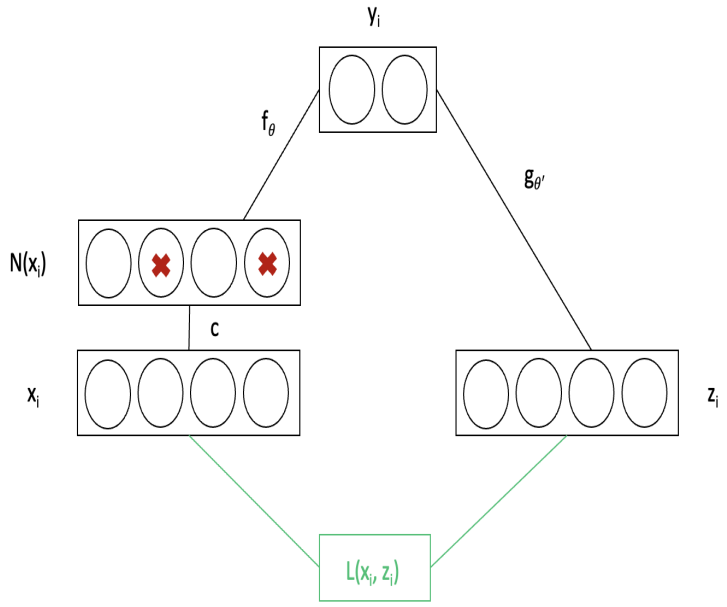


Figure 2: Représentation simplifiée d'un DAE. Ici $\mu = 0,5$. Les croix rouges représentent la corruption appliquée à $\mathbf{x}_i$.

3 AE et imputation de données manquantes

Nous nous plaçons ici dans le cas où la répartition des données manquantes ne permet pas d'extraire un groupe d'observations complètes suffisamment important pour entraîner le DAE.

Notations Soit $\mathbf{X}_{n\times p}$ une matrice de données incomplètes centrées (voir centrées-réduites) et $\Omega \subset \{1, ..., n\} \times \{1, ..., p\}$ l'ensemble des indices où les données ne sont pas manquantes. On définit une matrice $P_\Omega(\mathbf{X})$ de dimension $n \times p$ par :

$$P_\Omega(\mathbf{X}) = \begin{cases} \mathbf{X}_{ij} & \text{si } (i,j) \in \Omega \\ 0 & \text{si } (i,j) \notin \Omega \end{cases}$$

que l'on appellera projection de la matrice $\mathbf{X}$ sur les données présentes. De la même manière, on définit la projection complémentaire $P_\Omega^\perp(\mathbf{X})$ via $P_\Omega^\perp(\mathbf{X}) + P_\Omega(\mathbf{X}) = \mathbf{X}$. Les données imputées seront alors :

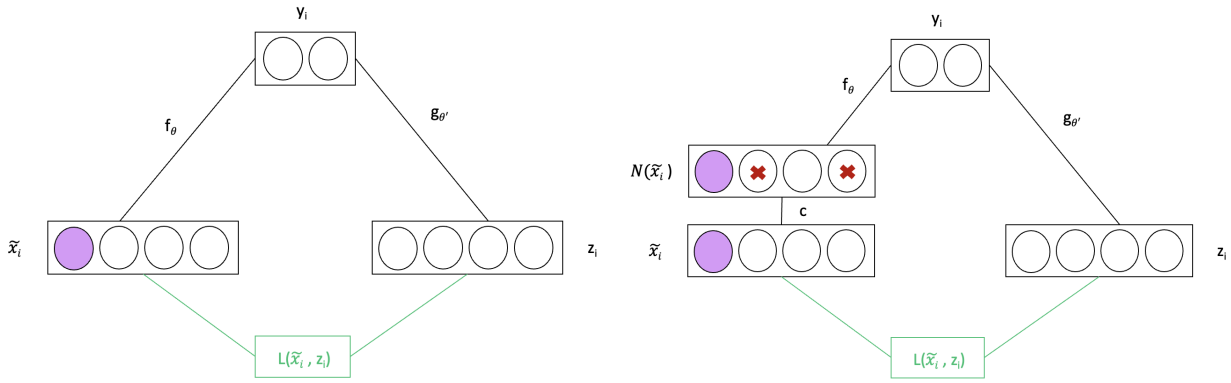
$$P_\Omega(\mathbf{X}) + P_\Omega^\perp(\mathbf{Z}),$$

où $\mathbf{Z}$ est la matrice reconstruite par le réseau. Par la suite, on notera :

- $\tilde{\mathbf{X}} = P_\Omega(\mathbf{X})$ la matrice des données pré-imputées par la moyenne (les données étant centrées),
- $N(\tilde{\mathbf{X}})$ la matrice des données corrompues en mettant aléatoirement une proportion μ de données non manquantes à 0 (Salt and Peper noise).

3.1 Application directe d'un AE et d'un DAE

Dans cette partie, nous appliquons directement l'AE et le DAE à la matrice des données pré-imputées $\tilde{\mathbf{X}}$. La FIGURE 3 décrit cette méthodologie.



(a) Méthode de l'AE simple

(b) Méthode du DAE simple

Figure 3: Représentations simplifiées de l'AE et du DAE appliqués directement aux données pre-imputées. Les ronds violets correspondent aux données manquantes remplacées par 0.

Comme le montre cette figure, l'erreur de reconstruction est optimisée sans tenir compte de la position des données manquantes dans la sortie $\mathbf{Z}$:

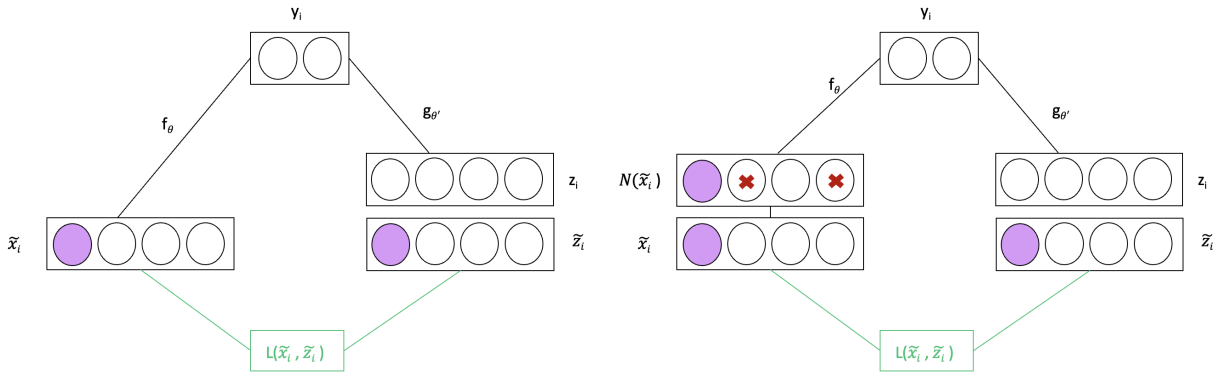
$$\mathcal{L} = \|P_\Omega(\mathbf{X}) - \mathbf{Z}\|_F^2$$

3.2 Prise en compte des données manquantes dans la fonction de perte

Afin de prendre en compte les données manquantes dans la fonction de perte, il suffit de mettre à 0 les valeurs de $\mathbf{Z}$ correspondant à des valeurs manquantes dans $\mathbf{X}$. On parlera alors de mAE (modified AE) et mDAE (modified DAE) (voir FIGURE 4).

L'erreur de reconstruction optimisée est alors :

$$\mathcal{L} = \|P_\Omega(\mathbf{X}) - P_\Omega(\mathbf{Z})\|_F^2$$



(a) Méthode de l'AE modifiée

(b) Méthode du DAE modifiée

Figure 4: Représentations simplifiées du mAE et du mDAE. Les ronds violets correspondent aux 0 ajoutés à l'emplacement des données manquantes.

4 Comparaison des approches

Dans cette section, les quatre approches AE, DAE, mAE et mDAE sont comparées à l'aide des données Boston Housing[1]. Ces données contiennent des informations collectées en 1978 par le *U.S Census Service* sur l'immobilier dans les alentours de la ville de Boston. Il contient 506 lignes et 14 colonnes. Chaque ligne représente une maison et les colonnes décrivent plusieurs informations telles que le nombre de chambre, la superficie de la maison, le prix du loyer,... On notera $\mathbf{X}_0$ cette matrice de données complètes.

La méthodologie consiste alors à ajouter aléatoirement ρ % de données manquantes à ces données complètes, à les centrer-réduire (en *sautant* les données manquantes) puis à appliquer les quatre méthodes AE, DAE, mAE, mDAE à cette matrice de données incomplète $\mathbf{X}$. La qualité de l'imputation est alors évaluée à partir des critères suivants :

- $MSE = \frac{1}{n}\|\mathbf{X}_0 - \mathbf{Z}\|_F^2$: la qualité de reconstruction globale.
- $MSE_{\Omega^\perp} = \frac{1}{n}\|P_\Omega^\perp(\mathbf{X}_0) - P_\Omega^\perp(\mathbf{Z})\|_F^2$: la qualité de reconstruction des données manquantes.
- $MSE_\Omega = \frac{1}{n}\|P_\Omega(\mathbf{X}_0) - P_\Omega(\mathbf{Z})\|_F^2$: la qualité de reconstruction des données non manquantes,

avec :

$$MSE = MSE_{\Omega^\perp} + MSE_\Omega.$$

Cette procédure est répétée B fois pour B tirages aléatoires de ρ % de données manquantes.

[1]https://www.kaggle.com/datasets/schirmerchad/bostonhoustingmlnd

L'architecture des AE et DAE utilisée ici est celle proposée par Gondara, L., et Wang, K. (2018) : une seule couche cachée pour l'encodeur et pour le décodeur (avec 21 neurones) et $q = 35$ neurones dans la couche *centrale* correspondant à l'espace latent. Le nombre de neurones des couches d'entrée et de sortie est $p = 14$, le réseau est donc de type *overcomplete*. L'algorithme utilisé pour la minimisation de la fonction de perte est l'algorithme *Adam*. Enfin, les codes ont été implémentés sous Python à l'aide du package PyTorch (Paszke, A. *et al.* (2019)).

Le Tableau 1 présente des résultats préliminaires pour la comparaison de ces quatre méthodologies.

	AE	DAE	mAE	mDAE
MSE	0.1110 ±0.0145	0.0906 ±0.0204	0.1026 ±0.0142	**0.0682±0.0092**
$MSE_{\Omega^\perp}$	0.1102 ±0.0146	0.0820 ±0.0205	0.1018 ±0.0143	**0.0546 ± 0.0090**
MSE_{Ω}	**0.0008 ±0.0004**	0.0086 ±0.0006	0.0008 ±0.0004	0.0137 ±0.0006

Table 1: Valeurs moyennes (± écart-type) obtenues avec $B = 20$ répétitions, $\rho = 10\%$ de données manquantes, $\mu = 10\%$ de corruption pour les DAE.

On observe d'abord dans ce tableau que les méthodes AE et DAE (appliquées simplement aux données pré-imputées) reconstruisent bien les données non manquantes (MSE_Ω proche de 0) et mal les données manquantes avec l'erreur globale de reconstruction presque entièrement portée par l'erreur sur les données manquantes ($MSE_{\Omega^\perp} \approx MSE$). Ce résultat était attendu puisque pour ces deux méthodologies, l'erreur de reconstruction est minimisée sans tenir compte des données manquantes dans la fonction de perte. On note également que le DAE reconstruit globalement mieux les données (avec une valeur de MSE plus petite qu'avec l'AE) ce qui là aussi était un résultat attendu, l'introduction de données corrompues dans le processus d'apprentissage permettant justement de fournir une reconstruction plus robuste au bruits (et donc aux données manquantes). D'ailleurs, la baisse du MSE avec la méthodologie DAE est principalement due à une baisse du $MSE_{\Omega^\perp}$ (qui passe de 0.11 à 0.07) et donc une erreur de reconstruction des données manquantes plus petite.

La méthode mAE donne des résultats sensiblement identiques à ceux de la méthode AE alors qu'on aurait pu penser qu'ils seraient meilleurs. En revanche, la méthode mDAE reconstruit ici mieux les données aussi bien globalement et que sur les données manquantes (les valeurs de MSE et $MSE_{\Omega^\perp}$ sont plus petites) avec une moins grande variabilité des résultats.

5 Conclusion

Dans cet article nous avons présenté et comparé quatre méthodologies qui utilisent les autoencoders pour imputer les données manquantes dans le cas particulier où une base d'apprentissage complète ne peut pas être extraite des données incomplètes.

La méthode que nous avons appelée mDAE reçoit des données d'entrée corrompues (comme les DAE) avant d'être entraînée à reconstruire les données d'origine (non bruitées)

en "sautant" les données manquantes. Cette simple modification semble donner des résultats prometteur d'après les résultats préliminaires obtenues avec les données Boston Housing. Ces premiers résultats seront enrichis en évaluant en particulier l'impact du choix du paramètre μ (proportion de corruption des données pour le DAE) qui joue le rôle d'un paramètre de régularisation. D'autres structures de données manquantes seront étudiées en mettant l'accent sur les données manquantes structurées en bloc issues de pannes de capteurs. Enfin, une variante itérative de la méthodologie mDAE sera également étudiée et comparée aux méthodes itérative d'imputation par SVD régularisée.

Bibliographie

Alain, G. et Bengio, Y. (2012), *What Regularized Auto-Encoders Learn from the Data Generating Distribution*, The Journal of Machine Learning Research, 15(1), 3563-3593.

Bengio, Y. (2009), *Learning Deep Architectures for AI*. Found Trends® Mach Learn.

Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; ... Chintala, S. (2019), *Pytorch: An imperative style, high-performance deep learning library*, Advances in neural information processing systems, 32.

Pereira, R.; Santos, M.; Rodrigues, .P; Abreu, .P (2020), *Reviewing autoencoders for missing data imputation: Technical trends, applications and outcomes*, Journal of Artificial Intelligence.

Vincent, P.; Larochelle, H.; Lajoie, I.; Bengio, Y.; Manzagol, P-A. (2010), *Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion* J Mach Learn Res.

Vincent, P.; Larochelle, H.; Bengio, Y.; Manzagol, P-A. (2008), *Extracting and composing robust features with denoising autoencoders*, Proc 25th Int Conf Mach Learn - ICML.