RAPPORT DE STAGE - PROJET DE FIN D'ÉTUDES

Utilisation des données de monitorage dans la prédiction de la survie de patients sous ECMO-VV

Clément Delmaire-Sizes

18 Mars 2024 - 30 Août 2024

Département Sciences du Numérique - 3A

Sommaire

Listes des figures

1 Remerciements

Je tiens tout d'abord à remercier les personnes qui m'ont accompagnées le long de ce stage, en particulier:

- Michaël Poette, médecin anesthésiste-réanimateur au CHU de Toulouse, et encadrant principal de mon stage, pour son implication, sa supervision et son aide tout au long du projet.
- Robin Schwob et Jean-Marc Alliot, de la DSN du CHU de Toulouse, pour le temps qu'ils m'ont accordé, leur intérêt pour mon projet et les réponses apportées à mes questions.
- Le reste de l'équipe de la DSN, pour leur bienveillance et l'accueil qu'ils m'ont accordés.

2 Introduction

2.1 Contexte : Présentation de l'ECMO

L'oxygénation par membrane extracorporelle, plus connue sous l'acronyme **ECMO** (ExtraCorporeal Membrane Oxygenation) est une technique médicale avancée utilisée pour fournir un soutien cardiorespiratoire à des patients dont le cœur et/ou les poumons sont gravement défaillants. Elle est utilisée dans des situations critiques pour gérer par exemple des cas sévères de chocs cardiogéniques réfractaires, d'arrêts cardiaques réfractaires, ou encore de SDRA (syndromes de détresse respiratoire aiguë) réfractaires.

Le fonctionnement d'une ECMO peut être résumé par les trois étapes suivantes :

1. Circulation extracorporelle : le sang est prélevé du corps du patient par un système de canules insérées dans les grandes veines ou artères. Le sang est ensuite dirigé vers une machine ECMO.
2. Oxygénation et élimination du dioxyde de carbone : dans la machine ECMO, le sang passe à travers une membrane qui permet l'échange gazeux. L'oxygène est ajouté au sang et le dioxyde de carbone est retiré.
3. Réinjection dans le corps : le sang oxygéné est ensuite réintroduit dans le corps du patient par une autre canule.

Il existe deux types principaux d'ECMO :

- ECMO veino-artérielle (VA) : utilisée pour soutenir à la fois le cœur et les poumons. Le sang est prélevé d'une veine et réintroduit dans une artère. Elle est souvent employée en cas d'insuffisance cardiaque sévère ou d'arrêt cardiaque.
- ECMO veino-veineuse (VV) : utilisée principalement pour soutenir les poumons. Le sang est prélevé d'une veine et réintroduit dans une autre veine. Ce type est couramment utilisé en cas de défaillance respiratoire sévère.

Ces deux techniques d'ECMO sont expliquées schématiquement à l'aide de la figure 1.

L'étude menée lors de ce stage se concentre exclusivement sur les ECMO-VV.

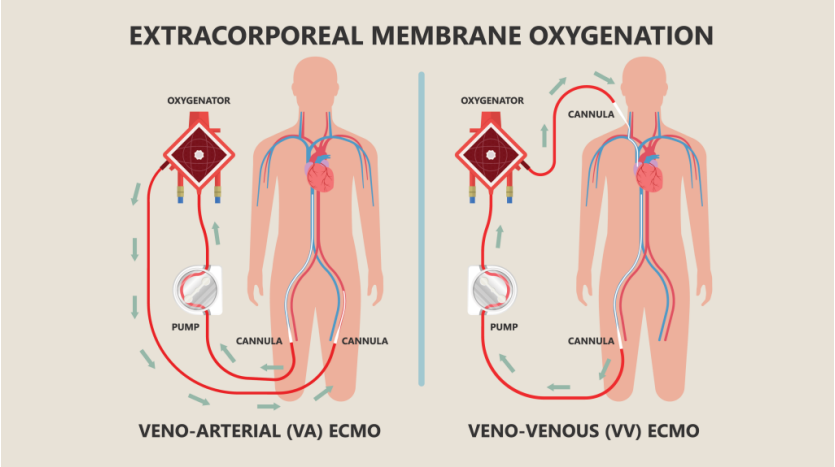

Figure 1: Schéma du fonctionnement des ECMO-VA et VV (schéma issu du site https://www.myamericannurse.com)

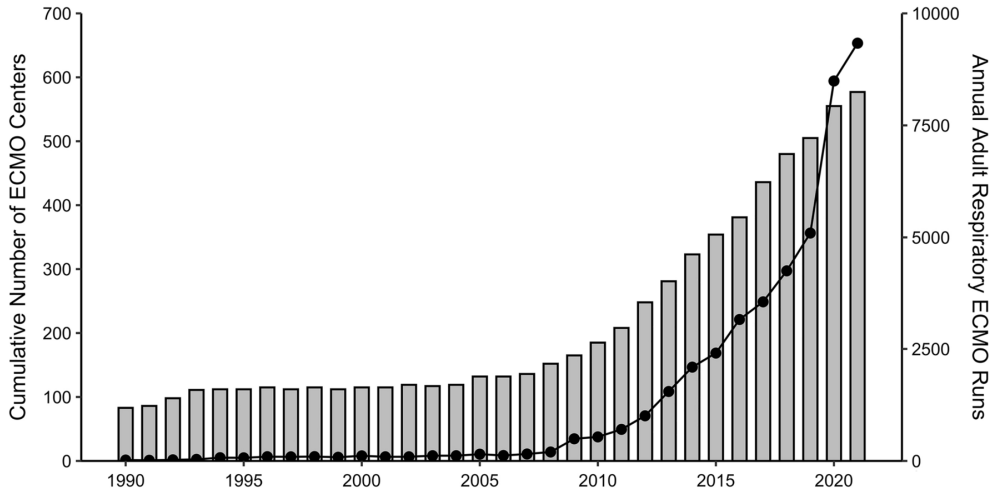

Figure 2: Nombre d'utilisations d'ECMO en fonction du temps (moyennes internationales) (figure extraite de l'article [1])

Comme le montre la figure 2, l'utilisation des techniques d'ECMO est de plus en plus fréquente, notamment depuis les années 2010.
Cependant, comme toute procédure invasive, cette technologie de sauvetage comporte des risques et des complications potentielles, tels que des saignements, des infections, ou des complications liées aux canules (thromboses, hématomes). De plus, de par les moyens techniques et humains mis en œuvres, l'ECMO se révèle être une technique extrêmement coûteuse.

2.2 Objectif

L'objectif de ce stage est alors le suivant: prédire, à l'aide de techniques d'apprentissage automatique, et à partir de l'évolution de certains signes vitaux, la mortalité hospitalière de patients sous ECMO-VV. La liste des signes vitaux et son choix seront explicités dans la suite de ce rapport. L'étude menée pendant ce stage s'intéresse uniquement à l'évolution de ces signes **après** l'implantation de l'ECMO.

Bien que cette étude soit à caractère observatoire et serve un but purement théorique, des applications pratiques seraient néanmoins envisageable et permettraient par exemple de réduire de nombreux coûts associés aux traitements de certains patients qui ne pourraient pas être sauvés. Dans le cas d'une saturation des installations, comme cela pourrait être le cas lors d'une pandémie telle que celle du covid-19, cela permettrait aussi de prioriser des patients qui ont de meilleures chances de survie.

2.3 Recherche existante

De nombreux articles s'intéressent à la prédiction de la mortalité hospitalière pré-installation d'une ECMO: c'est le cas par exemple des articles [12] ou [13] qui proposent des scores (respectivement les scores RESP et PRESERVE) permettant de prédire, à partir de critères pré-implantation, la mortalité hospitalière après la pose d'une ECMO-VV. La prédiction post-implantation est quant à elle nettement moins étudiée: nous n'avons en effet trouvé aucun article portant sur cette recherche particulière.

Cependant, la recherche est très riche concernant la prédiction de survie pour des patients en soins intensifs. Une grande partie de la bibliographie étudiée est basée sur ces recherches. En effet, celles-ci se rapprochent souvent de notre problématique en étudiant l'évolution de signes vitaux ou d'autres données au cours du séjour d'un patient. Pour réaliser la prédiction à partir de ces données, les articles proposent de nombreuses méthodes: certaines approchent le problème à l'aide de méthodes basées sur des règles [14], d'autres avec des techniques traditionnelles de machine learning [15, 16, 17, 18, 19]. Enfin, de nombreux articles récents utilisent des algorithmes de deep learning afin de réaliser leur apprentissage [15, 16, 17, 19, 7, 20, 21, 22, 23].

3 Présentation de l'environnement de travail et du projet

3.1 Présentation du CHU

Le Centre Hospitalier Universitaire (CHU) de Toulouse est une institution de santé et d'enseignement clé située à Toulouse, dans la région Occitanie. Comme les autres CHU en France, le CHU de Toulouse combine trois missions principales:

- Soins : le CHU offre des soins de santé à la population, allant des consultations de base aux traitements les plus complexes. Il dispose ainsi de nombreux services spécialisés (chirurgie, cardiologie, oncologie, etc.).
- Enseignement : en tant que centre universitaire, le CHU de Toulouse est également un lieu de formation pour les futurs médecins, infirmiers, et autres professionnels de santé.
- Recherche : ce CHU est aussi un lieu de recherche médicale. Ils participent au développement de nouvelles méthodes de traitement, de technologies médicales, et de médicaments.

Le CHU de Toulouse est constitué de plusieurs établissements et lieux de soins répartis sur quatre sites hospitaliers: les sites de Purpan, de Rangueil-Larrey, de l'Hôtel-Dieu/La Grave, et de Salies-du-Salat.

Parmi les sites du CHU Toulouse, certains possèdent des services de réanimations, qui sont des unités hospitalières spécialisées dans la prise en charge des patients en état critique, nécessitant des soins intensifs et une surveillance continue. Cependant, seul le site de Rangueil réalise des ECMO: c'est donc de ce site que sont tirées les données d'ECMO utilisées dans ce projet.

Avec 16 000 professionnels, dont 4000 médecins et 12 000 personnels non médicaux, le CHU est l'un des plus gros employeurs de la région, et est le quatrième hôpital de France en termes d'activité.

Le stage s'est déroulé sur le site de l'Hôtel-Dieu, qui regroupe la plupart des fonctions administratives du CHU de Toulouse. Plus précisément, le projet a été réalisé au sein des locaux de la direction des services numériques (DSN), direction qui regroupe tous les services liés à l'organisation et à la gestion des outils informatiques et numériques de l'établissement.

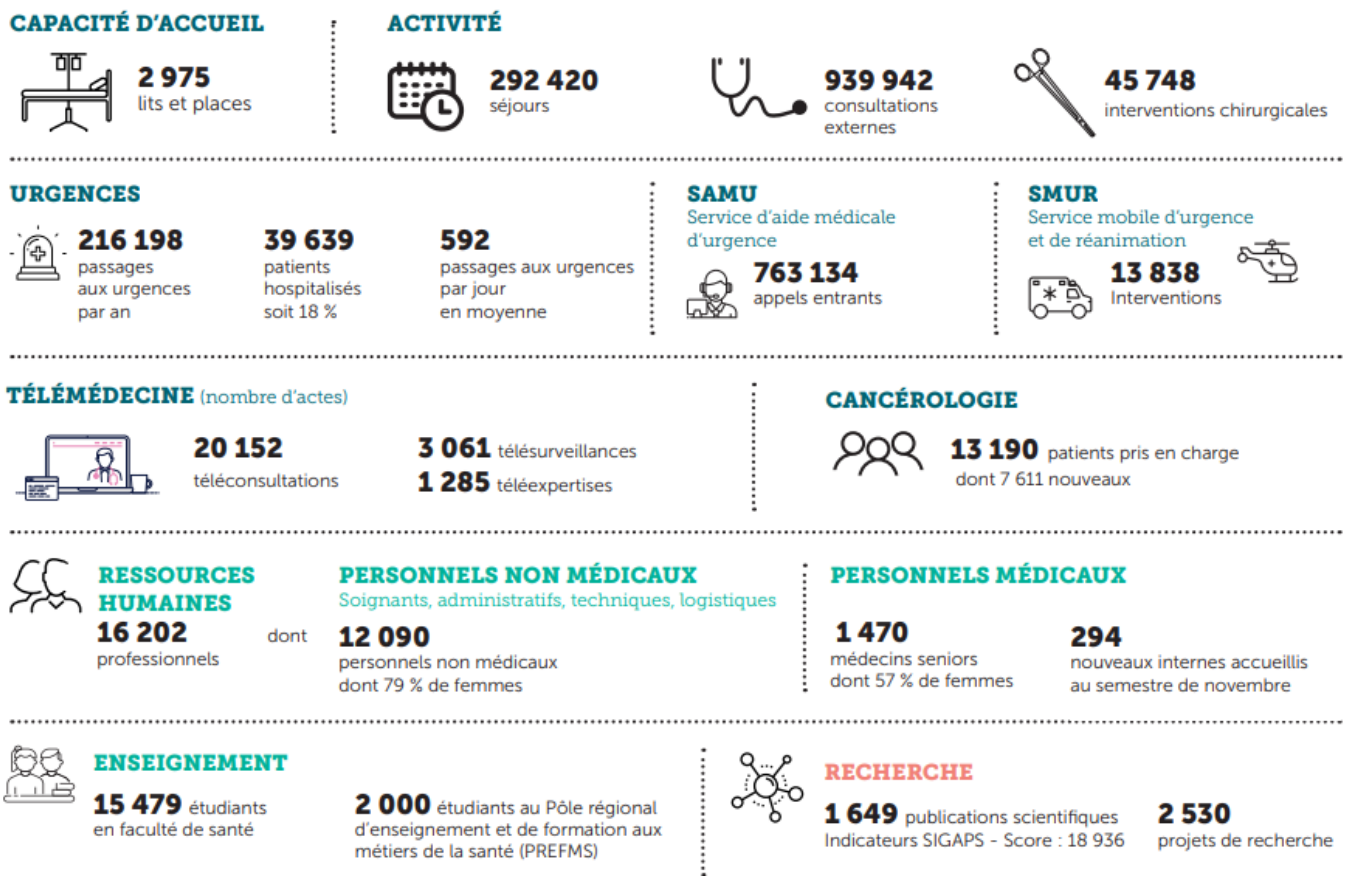

Figure 3: Chiffres clés du CHU de Toulouse (schéma tiré du projet d'établissement 2023-2028)

3.2 Le programme « IA pour la santé »

Initié par ANITI (Artificial and Natural Intelligence Toulouse Institute) et soutenu par la Région Occitanie, le programme « IA pour la santé » a pour objectif de fédérer les acteurs régionaux de la recherche, de la formation et de l'innovation autour de l'application de l'intelligence artificielle dans le domaine de la santé. Ce programme vise à renforcer la collaboration entre les chercheurs en IA et les professionnels de la santé.

Ce programme réunit plusieurs laboratoires de recherche régionaux spécialisés en numérique et en santé pour développer des techniques d'intelligence artificielle appliquées à la médecine, dans le but d'améliorer la précision des diagnostics, la qualité des soins, et le suivi des patients.

C'est dans le cadre de ce programme qu'a été financé le stage que j'ai effectué.

3.3 Organisation du travail et déroulé du stage

Plusieurs étapes ont marqué le déroulement de ce stage, les plus importantes sont les suivantes:

- La première étape a consisté en des travaux préliminaires au lancement de l'étude, en attendant les droits d'accès à la base de données. Cette étape a consisté en une remise à niveau en SQL (nécessaire pour extraire les données de la base du CHU), à l'étude des méthodes d'analyse des séries temporelles, et au début des recherches bibliographiques. Ces premières semaines m'ont aussi donné le temps d'obtenir l'accès à la base de données externe MIMIC IV, contenant notamment les données anonymisées de patients ayant fait un séjour en réanimation et potentiellement utilisables pour l'étude réalisée. Cet accès a pu être obtenu en passant des certifications liées aux recherches sur l'humain et à la confidentialité des données.
- Après l'obtention des droits d'accès, j'ai pu débuter l'extraction des données à partir de la base du CHU, puis effectuer le pré-traitement des données de sorte à constituer les datasets d'intérêt.
- L'analyse des données a constitué l'étape suivante du stage, consistant en l'élaboration de différentes techniques d'apprentissage automatique pour réaliser la tâche souhaitée.
- Enfin, la rédaction de ce rapport de stage a constitué la phase finale de ce stage.
- Une lecture régulière de ressources selectionnées suite à mes recherches bibliographiques a été nécessaire tout au long du stage afin d'être au maximum au point sur les recherches existantes liées à mon sujet d'étude.
- Des réunions avec mon tuteur Michaël Poette ont également été réalisées de manière hebdomadaire afin de suivre l'avancement de mon étude et réfléchir à la suite des travaux.

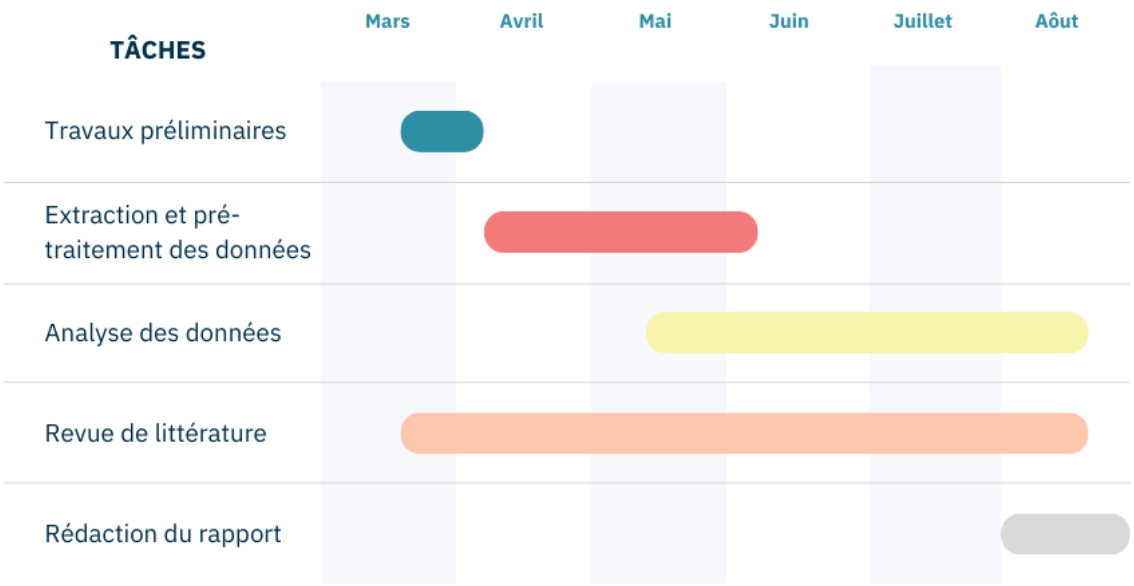

Figure 4: Diagramme des étapes principales du stage

4 Extraction des données

4.1 Dataset principal

Toutes les données des patients ayant côtoyé un des services de réanimation du CHU de Toulouse sont stockées dans la base de données ICCA (IntelliSpace Critical Care and Anesthesia) de ce même CHU. ICCA ayant été mise en place en 2014 à Toulouse, les données n'ont pu être extraites qu'à partir de cette date.

Dans un premier temps, l'objectif a été d'extraire le dataset contenant les patients ayant bénéficié d'une ECMO, dont les données peuvent alors être extraite de la base de données ICCA. Plus précisément, les patients doivent avoir subi une ECMO-VV, et ce, pendant au moins 5 jours consécutifs. De plus, seuls les patients de plus de 18 ans sont considérés.

Après extraction des identifiants des patients d'intérêt, on obtient au total **153 patients uniques**, ce qui semble peu mais qui en réalité est assez précieux car les ECMO constituent une opération rare et dont l'utilisation est récente.

4.2 Datasets supplémentaires

Comme il était envisagé de faire appel à des techniques de Machine Learning et en particulier à des techniques de Deep Learning, la faible taille du dataset (153 patients au maximum), qui sera possiblement encore réduite dans le cas de patients inutilisables (trop de données manquantes par exemple), ainsi que la complexité des variables qui le composent, rendent ces techniques moins pertinentes. C'est pourquoi il a été choisi d'extraire des données issues de nouvelles bases.

Les données de patients sous ECMO étant relativement rares, l'idée a été de chercher à extraire les données de patients en réanimation et ventilés (reliés à un appareil qui contrôle mécaniquement la ventilation du patient), qui sont nettement plus abondantes. Ces données ont été choisies car cette catégorie de patients comporte toutes les variables d'intérêt utilisées pour les ECMO-VV et restent relativement proches de celles-ci. De cette manière, il pourra être envisageable d'effectuer du Transfer Learning ou du Fine-Tuning sur certains modèles.

4.2.1 Données du CHU

Le CHU de Toulouse posséde plusieurs services de réanimation, il a été possible d'extraire certains patients des services de Rangueil et Purpan à partir de la base ICCA. La condition d'ECMO sur 5 jours consécutifs est remplacée par la condition d'être ventilé mécaniquement sur 5 jours consécutifs.

On obtient alors **679 patients utilisables**.

4.2.2 Données externes

Pour augmenter la taille des données d'entraînement, j'ai obtenu l'accès à la base de données MIMIC-IV [24], contenant en particulier les données de 65366 patients uniques en réanimation.

A partir de cette base, j'ai pû extraire **4140 patients** remplissant les conditions de ventilation nécessaires. (la base contient d'ailleurs 17 patients sous ECMO utilisables)

4.3 Variables extraites

De nombreuses variables entrent en jeu dans l'évolution d'un patient en réanimation. Dans le cadre de cette étude, il a été choisi de se concentrer sur un nombre restreint de ces variables. Le choix de celles-ci a été réalisé par mon tuteur médecin, le docteur Michaël Poette, et a évolué tout au long du stage.

Les variables extraites consistent en des variables dynamiques, qui évoluent de manière notable et rapide au cours du temps (les signes vitaux en font partie notamment), mais aussi en des variables statiques, qui restent figées ou très peu variables au cours du séjour du patient. Les parties suivantes explicitent les variables extraites ainsi que les raisons de leur choix.

4.3.1 Variables dynamiques

Dans l'ordre chronologique de leur ajout dans cette étude, voici la liste des variables dynamiques extraites ainsi que la justification de leur utilisation:

- La **fréquence cardiaque**, c'est-à-dire le nombre de battements du cœur par minute. Une fréquence cardiaque anormale, qu'elle soit trop basse (bradycardie) ou trop élevée (tachycardie), peut indiquer un problème cardiaque, une réponse au stress, une infection ou d'autres conditions médicales graves.
- La **fréquence respiratoire**, ou le nombre de respirations par minute. Une fréquence respiratoire élevée peut indiquer une détresse respiratoire, une infection, ou d'autres problèmes pulmonaires. Une fréquence respiratoire trop basse peut être un signe de dépression respiratoire. C'est un indicateur important de la fonction respiratoire et de l'état général du patient.
- Les **pressions artérielles**:
  - La **pression artérielle diastolique (PAD)**: celle-ci représente la pression dans les artères lorsque le cœur se relâche entre les battements. Une pression trop basse peut signaler un choc ou une défaillance organique, tandis qu'une pression élevée peut indiquer une hypertension, augmentant le risque de complications cardiovasculaires.
  - La **pression artérielle systolique (PAS)**, qui correspond à la pression dans les artères lors de la contraction du cœur. Sa valeur reflète la force du cœur. Des valeurs anormales peuvent indiquer des problèmes cardiaques ou vasculaires.
  - La **pression artérielle moyenne (PAM)**: il s'agit de la pression moyenne dans les artères pendant un cycle cardiaque complet. C'est une mesure importante pour s'assurer que les organes reçoivent un flux sanguin suffisant. Une PAM trop basse est critique car elle peut indiquer une hypoperfusion des organes.
- La **température** corporelle. En effet, la fièvre peut indiquer une infection, tandis qu'une température basse peut indiquer un choc ou une hypothermie.
- La **SpO2** (saturation pulsée en oxygène): celle-ci représente le pourcentage de l'hémoglobine dans le sang qui est saturé en oxygène. C'est un indicateur clé de la fonction respiratoire. Une SpO2 basse peut entraîner une insuffisance organique.
- La **diurèse**: il s'agit de la quantité d'urine produite sur une certaine période. C'est un indicateur de la fonction rénale. Une bonne diurèse est généralement un signe de bon fonctionnement rénal et de stabilité hémodynamique alors qu'une diurèse réduite peut indiquer une insuffisance rénale ou un choc.
- La **FiO2**: elle représente la fraction d'oxygène dans l'air inspiré par le patient. Elle est réglée par des dispositifs d'administration d'oxygène, comme les masques à oxygène ou les ventilateurs. Une FiO2 élevée est souvent nécessaire chez les patients en détresse respiratoire pour maintenir une SpO2 adéquate.
- La **compliance**: il s'agit de la compliance respiratoire (ou pulmonaire), c'est-à-dire la capacité des poumons à se distendre. Une compliance basse peut indiquer une rigidité pulmonaire, comme dans les cas de syndrome de détresse respiratoire aiguë (SDRA). C'est un indicateur important de la fonction pulmonaire et de la réponse à la ventilation mécanique. Comme cette variable est assez manquantes dans les différentes bases de données, la **Pression de Plateau**, la **Pression Expiratoire Positive (PEP)** ainsi que le **Volume Courant** ont été extrait car ils permettent de calculer la compliance à l'aide de la formule suivante :

$$Compliance = \frac{Volume_{Courant}}{Pression_{plateau} - PEP} \tag{1}$$

  .
- La présence de **dialyse**: la dialyse est une procédure pour éliminer les déchets et l'excès d'eau du sang lorsque les reins ne fonctionnent pas correctement. Elle indique alors une insuffisance rénale sévère, qui est un facteur de risque important pour la survie. La nécessité de dialyse signale une détérioration de l'état général du patient.

4.3.2 Variables statiques

Voici maintenant la liste des variables statiques utilisées dans ce projet:

- L'**âge** du patient. Les patients plus âgés ont généralement un pronostic moins favorable en raison de la diminution des réserves physiologiques et de la présence de comorbidités.
- Le **sexe** biologique du patient. Certaines pathologies et risques sont plus fréquents ou plus graves en fonction du sexe.
- La **taille** et le **poids** du patient. Ces deux variables peuvent être utilisées pour calculer l'indice de masse corporelle (IMC), qui est un indicateur important de l'état nutritionnel et de la santé générale.
- L'**IGS II** (Indice de Gravité Simplifié II) a aussi été extrait: il s'agit d'un score développé par Le Gall et al. [25], et utilisé en réanimation pour évaluer la gravité de l'état d'un patient à son admission dans une unité de soins intensifs. Ce score aide à prédire la probabilité de mortalité hospitalière, en se basant sur des paramètres cliniques et biologiques mesurés dans les premières 24 heures suivant l'admission.

4.3.3 Labélisation

La dernière étape d'extraction est celle de l'extraction du label "décès" ou "survie". On labélise ainsi un patient "décédé" si et seulement il y a une trace de son décès lors de son séjour en réanimation. Si cette extraction s'effectue aisément sur le dataset MIMIC, ce n'est pas le cas avec la base de données du CHU de Toulouse, car cette information n'est pas clairement stockée et son extraction a nécessité l'utilisation d'une source fiable à l'aide d'une base annexe du CHU.

Les statistiques de survie des patients selon les différents jeux de données sont les suivantes:

Table 1: Statistiques de survie des patients

	Nb Décès/Nb Patients
ECMO	55/153 (**35.9%**)
Ventilés CHU	132/560 (**23.6%**)
Ventilés MIMIC	1054/4140 (**25.5%**)

On remarque un déséquilibre des classes, notamment avec les datasets des patients ventilés.

5 Préparation des datasets

Afin de pouvoir traiter les données des patients à l'aide d'algorithmes d'apprentissage automatique, il a été nécessaire d'effectuer un pré-traitement de celles-ci. En effet, après extraction, les données peuvent avoir la forme suivante:

Temps après installation (minutes)	5	15	45	60	180	...	7130	7175
Valeur de la variable	81	65	70	0	68	...	57	58

Table 2: Exemple des données d'une variable après extraction

Or, on souhaiterait avoir, pour chaque variable, une valeur par heure, la plus fiable possible, et ce jusqu'à 120 $(5 * 24)$ heures. D'où la nécessité d'un pré-traitement.

Les étapes de ce pré-traitement des données sont détaillées dans les sous-parties suivantes.

5.1 Pré-traitement des données

La chaîne de pré-traitement des données est constituée de quatre étapes principales :

- la gestion des données aberrantes
- le moyennage horaire des variables
- la gestion des valeurs manquantes
- la standardisation des données

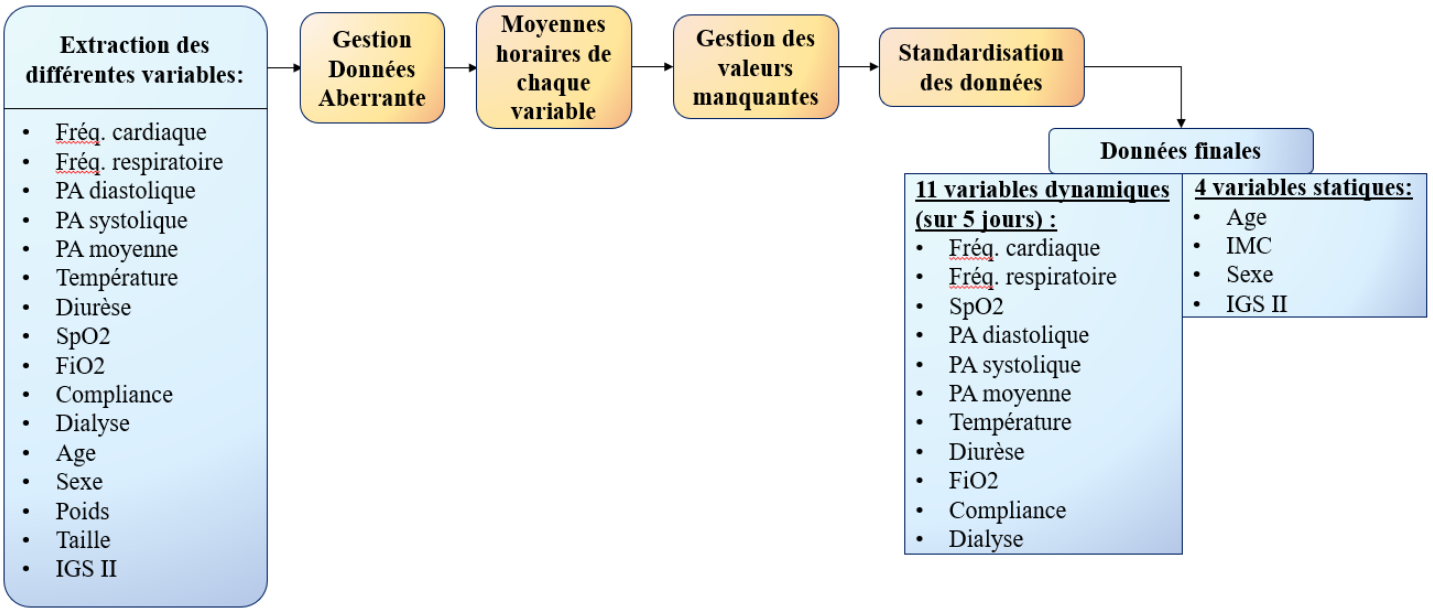

Figure 5: Schéma de la chaîne de pré-traitement des données

5.1.1 Gestion des données aberrantes

La première étape de cette chaîne de pré-traitement consiste à gérer les valeurs *'aberrantes'*, i.e. les valeurs dont on pense qu'elles sont dues soit à une erreur humaine lors la saisie de données de manière manuelle, notamment pour les données démographiques (âge, sexe, poids, taille,...), soit à des défaillances techniques liées aux équipements

de monitoring. Une manière simple de gérer ce problème est d'établir, pour chaque variable, un intervalle de valeurs pour lesquelles on considère la variable aberrante si elle n'appartient pas à cet intervalle: la valeur est alors supprimée des données. C'est cette solution qui a été utilisée.

La table suivante regroupe les différents intervalles utilisés, et validés par le médecin encadrant:

	MIN	MAX
Fréq. cardiaque (puls/min)	20	200
Fréq. respiratoire (resp/min)	5	50
PA diastolique (mmHg)	20	130
PA moyenne (mmHg)	30	200
PA systolique (mmHg)	40	230
Température (°C)	32	41
Diurèse (mL)	0	2000
SpO2 (%)	50	100
FiO2 (%)	20	100
Compliance (ml/mmHg)	3	150
Pression de Plateau (mmHg)	5	30
Pression Expiratoire Positive (mmHg)	2	22
Volume Courant (ml)	100	800
Poids (kg)	30	300
Taille (cm)	120	230

Table 3: Intervalles de valeurs pour les différentes variables

5.1.2 Moyennes horaires

L'objectif de la deuxième étape est d'avoir, pour chaque variable dynamique, une seule valeur par heure. Cela est effectué simplement en remplaçant la valeur à l'heure h par la moyenne des valeurs existante entre cette heure et l'heure suivante, ou par NaN (Not a Number) si aucune valeur n'existe durant cette période.

Dans le cas des variables statiques, on associe la valeur de la variable à la moyenne de toutes les valeurs existantes (si au moins une valeur existe, sinon NaN).

Lors de cette étape sont également effectuées certaines transformations des données, notamment:

- la fusion des données de pressions artérielles invasives et non invasives: on priorise les pressions invasives lorsqu'elles sont disponibles, et dans le cas contraire on garde la valeur non invasive.
- des changements d'unité, pour la température en particulier, qui est enregistrée en degré Fahrenheit dans la base MIMIC-IV
- les calculs de compliance pulmonaire à l'aide de la formule (1), lorsque celle-ci n'est pas enregistrée et qu'on possède les autres données nécessaires au calcul.
- la transformation de la diurèse, que l'on recalcule en mL/kg/heure, puis dont on fait la moyenne sur les 6 dernières heures.
- le calcul de l'IMC pour remplacer la taille et le poids.

5.1.3 Gestion des valeurs manquantes

5.1.3.1 Algorithme naïf

Une première méthode naïve pour gérer les valeurs manquantes est la suivante:

- Si aucune valeur n'existe pour la variable, on associe la valeur moyenne de la variable à toute la série temporelle
- Sinon, pour chaque valeur, si celle-ci est à équidistance des deux valeurs les plus proches, on lui associe la moyenne de ces deux valeurs. Sinon, on lui associe la valeur existante la plus proche.

Le problème de cette approche est qu'elle ne prend pas en compte les autres variables, qui peuvent donner des informations sur l'évolution de la variable manquante, et ce d'autant plus lorsqu'aucune valeur n'est présente dans la série temporelle.

5.1.3.2 Algorithme SAITS

La gestion des valeurs manquantes pour des séries temporelles multivariées est un sujet très étudié dans la recherche actuelle. Récemment, de nombreuses méthodes utilisant différentes techniques de Deep Learning ont été étudiées: certaines méthodes proposent par exemple l'utilisation de RNN (Recurrent Neural Network) [26, 27], d'autres tentent de prédire les valeurs manquantes avec des GAN (Generative Adversarial Network) [28] et d'autres ont aussi testé l'utilisation du mécanisme d'auto-attention avec de très bons résultats: c'est le cas de l'algorithme SAITS (Self-Attention-based Imputation for Time Series), proposé par Du et al.[2]. Cet algorithme, dont l'architecture est présentée figure 6, a été choisi en tant que seconde méthode pour prédire les valeurs manquantes.

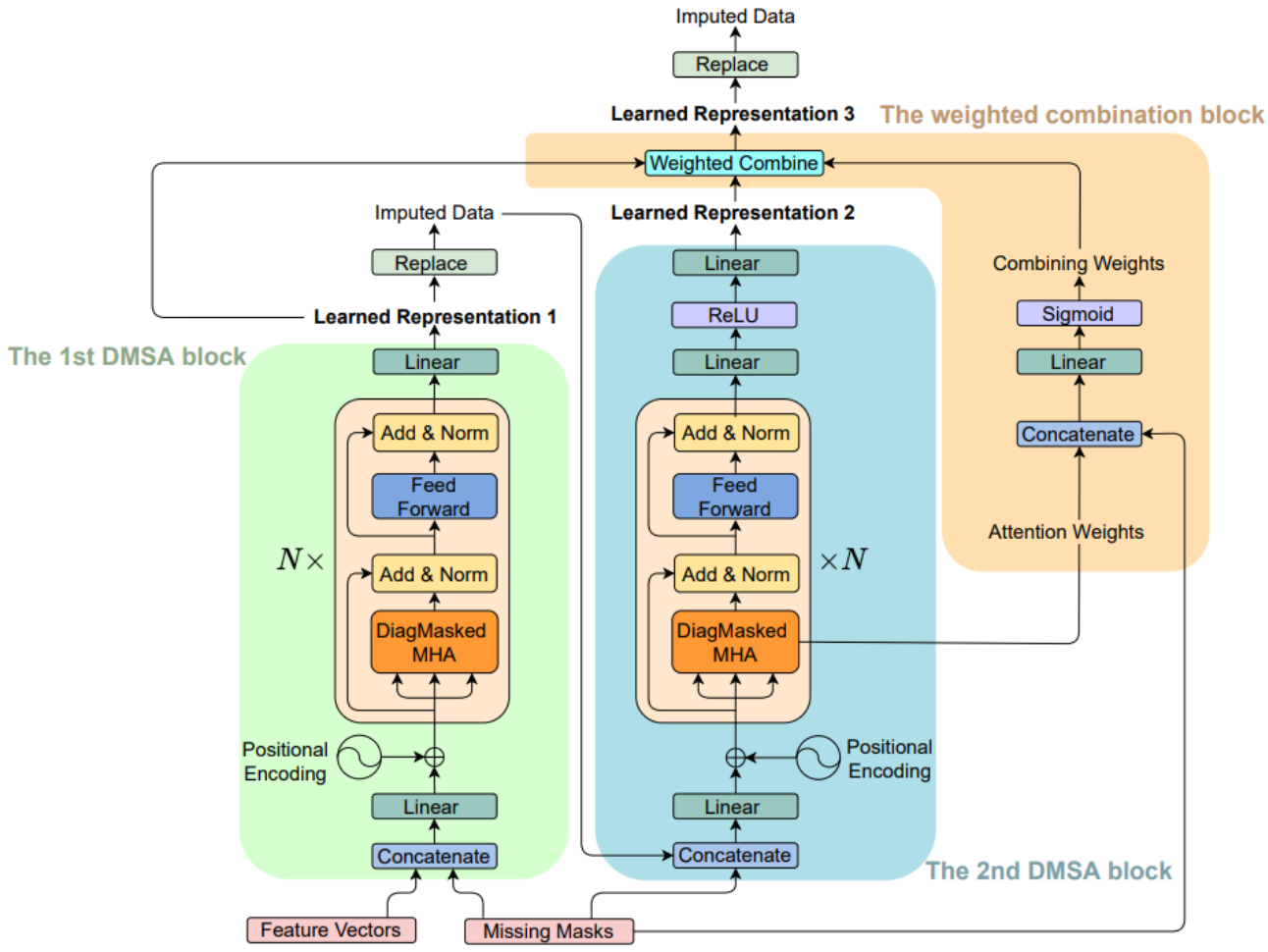

Figure 6: Architecture de l'algorithme SAITS (tirée de l'article [2])

5.1.3.3 Comparaison des algorithmes

Afin de comparer la performance de ces deux algorithmes, on ajoute aux séries temporelles des valeurs manquantes artificielles, puis on prédit la valeur réelle à l'aide d'un des algorithmes et on compare les valeurs prédites aux valeurs réelles à l'aide de l'erreur absolue moyenne (ou **MAE** pour Mean Absolute Error). Un schéma explicatif de la méthode est représenté figure 7.

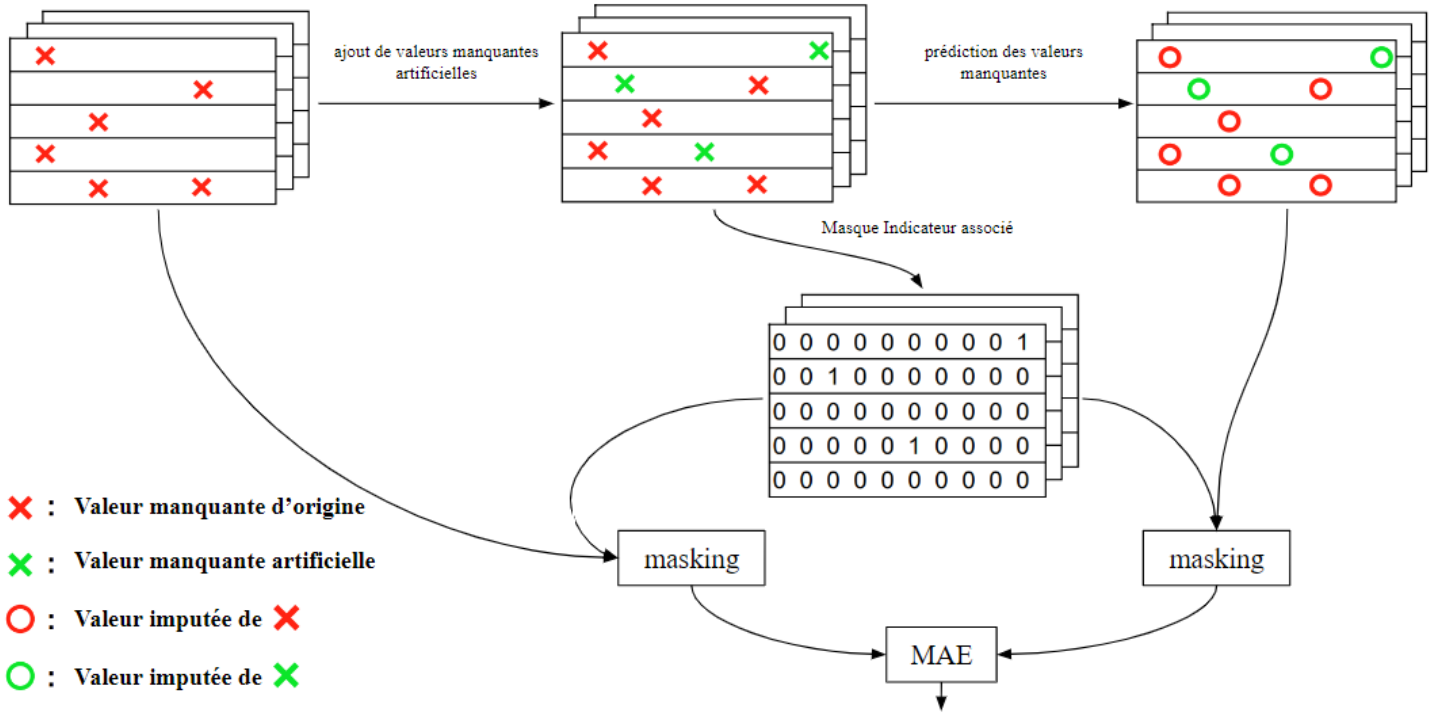

Figure 7: Schéma de la méthode de calcul de la performance des algorithmes d'imputation

En utilisant cette méthode et en entraînant l'algorithme SAITS sur la base MIMIC, les résultats sont les suivants sur les différentes base de données:

Table 4: Performances comparées des deux algorithmes

	MIMIC	Ventilés-CHU	ECMO
MAE - Algo Naïf	0.213	0.185	0.146
MAE - SAITS	**0.126**	**0.127**	**0.114**

5.1.4 Standardisation

La dernière étape du pré-traitement est la standardisation: pour chaque variable v, la moyenne μ_v et l'écart-type σ_v sont calculés à partir des données des patients ventilés. Pour chaque patient p et chaque variable v_p qui lui est associée, la nouvelle série temporelle prend la forme suivante:

$$v_p \leftarrow \frac{v_p - \mu_v}{\sigma_v} \tag{2}$$

5.2 Forme des données finales

Å la fin du pré-traitement, chaque patient est alors représenté par une série temporelle multivariée de taille $120 * 15$, pour les 15 variables sur 120 heures (si on considère les quatre variables statiques comme des variables dynamiques, ce qu'on ne fait pas toujours)

Le dataset principal ECMO a donc la forme suivante:

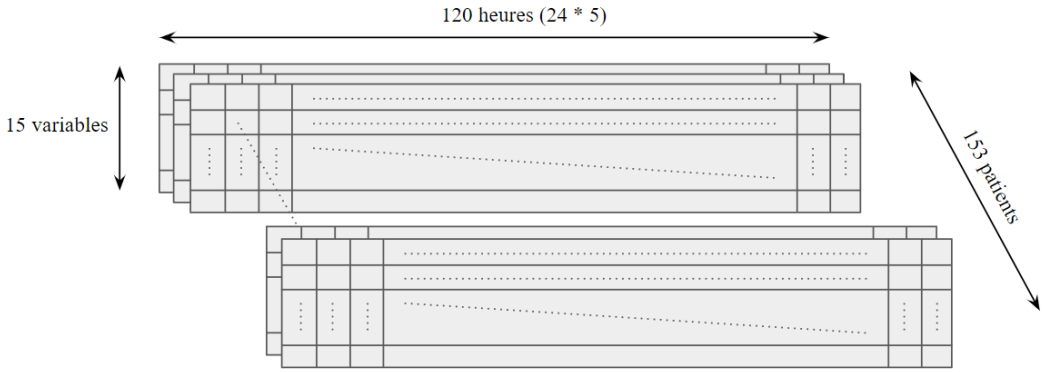

Figure 8: Forme du dataset ECMO

5.3 Statistiques sur les données

5.3.1 Statistiques générales

Une fois les données prêtes, nous avons pu faire quelques statistiques en fonction des trois cohortes à notre disposition: d'abord sur les données démographiques:

	ECMO	**Ventilés-CHU**	**MIMIC**
Hommes	108 **(71%)**	406 **(73%)**	2394 **(58%)**
Age	52.7 ± 13.1	60.2 ± 15.1	62.1 ± 16.2
IMC	30.9 ± 7.1	28.0 ± 6.0	30.6 ± 8.5

Table 5: Statistiques des données démographiques en fonction des cohortes

Puis sur les données des variables dynamiques:

	ECMO	**Ventilés-CHU**	**MIMIC**
Fréq. cardiaque (puls/min)	84.1 ± 20.0	83.2 ± 22.4	86.8 ± 18.4
Fréq. respiratoire (resp/min)	15.1 ± 5.6	20.8 ± 5.6	20.7 ± 5.8
PA diastolique (mmHg)	60.5 ± 9.8	62.0 ± 12.3	59.8 ± 12.8
PA moyenne (mmHg)	79.6 ± 12.0	82.4 ± 14.1	77.7 ± 14.4
PA systolique (mmHg)	119.0 ± 18.8	123.3 ± 21.6	118.2 ± 21.7
Température (°C)	36.7 ± 0.5	37.0 ± 0.7	37.2 ± 0.9
Diurèse (mL/kg/h)	0.888 ± 0.717	0.909 ± 0.722	1.124 ± 0.992
SpO2 (%)	94.9 ± 3.6	96.3 ± 3.1	97.4 ± 2.7
FiO2 (%)	54.0 ± 18.0	43.6 ± 15.3	48.4 ± 15.0
Compliance (mL/mmHg)	20.3 ± 15.1	46.0 ± 24.7	41.1 ± 17.1

Table 6: Moyennes et écarts-types des différentes variables dynamiques en fonction des cohortes

5.3.2 Statistiques en fonction de la survie

Des statistiques sur les variables en fonction de la survie ou non des patients ont été produites, dans le but d'anticiper l'importance de certaines de ces variables dans la prédiction de mortalité hospitalière. La table suivante (table 7) montre par exemple la différence entre les moyennes des variables des patients qui survivent et celles des patients qui décèdent à l'hôpital.

Table 7: Comparaison des moyennes de différentes variables en fonction de la survie des patients et des différentes cohortes

	ECMO		**Ventilés-CHU**		**MIMIC**	
	Survie	**Décès**	**Survie**	**Décès**	**Survie**	**Décès**
Hommes (nombre et proportion)	61 (62%)	47 (85%)	306 (71%)	100 (76%)	1816 (58%)	578 (55%)
Âge	50.0	57.6	58.7	65.2	60.8	66.0
IMC	30.4	31.9	28.1	27.8	30.7	30.2
Fréq. Cardiaque (puls/min)	82.9	86.2	82.6	85.1	86.4	88.0
Fréq. Respiratoire (resp/min)	15.4	14.6	20.8	21.0	20.5	21.3
PA Diastolique (mmHg)	61.1	59.3	62.7	59.9	60.6	57.4
PA Moyenne (mmHg)	80.2	78.6	83.0	80.4	78.6	75.1
PA Systolique (mmHg)	118.6	119.6	123.6	122.3	119.4	114.8
Température (°C)	36.8	36.7	37.1	36.9	37.3	37.0
Diurèse (mL/kg/h)	1.0	0.7	1.0	0.8	1.2	1.0
SpO2 (%)	95.1	94.5	96.4	96.0	97.5	97.1
FiO2 (%)	53.1	55.8	43.2	44.6	47.9	50.1
Compliance (mL/mmHg)	21.3	18.6	46.8	43.4	42.5	38.1

Les graphes suivants montrent quant à eux l'évolution moyenne de certaines variables en fonction du temps après l'admission, pour la cohorte des patients sous ECMO.

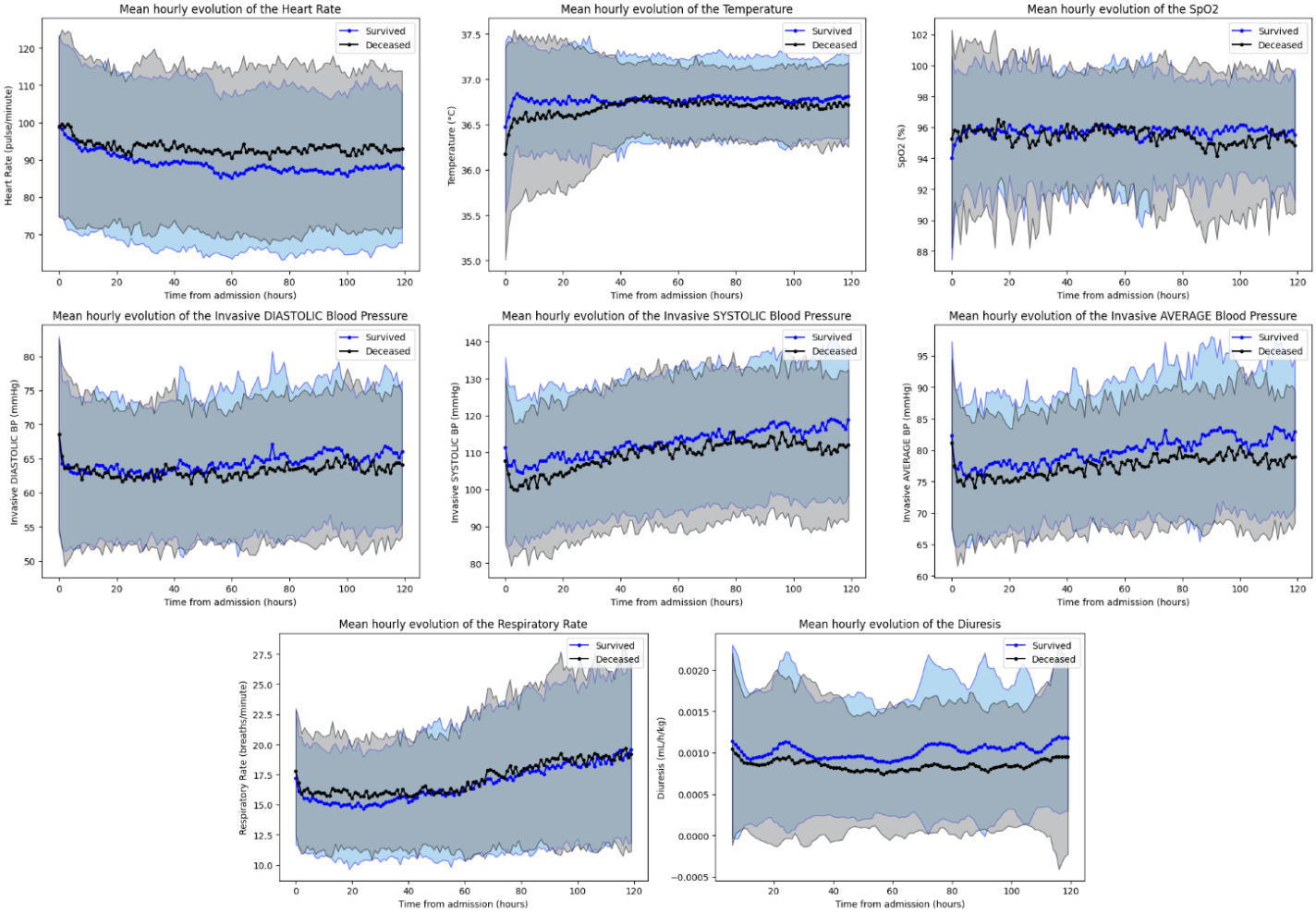

Figure 9: Évolutions moyennes de variables en fonction du temps (cohorte ECMO)

5.3.3 Statistiques sur les valeurs manquantes

Le problème des données manquantes étant assez récurrent dans le domaine des séries temporelles, de nombreuses statistiques ont été effectuées. Les deux figures suivantes (figures 10 et 11) permettent de rendre compte de l'ampleur du problème dans le cas de notre étude:

	ECMO	Ventilés-CHU	MIMIC
Fréq. Cardiaque	1	2	2
SpO2	7	6	2
PA Diastolique	1	3	2
PA Moyenne	1	5	2
PA Systolique	1	6	2
Fréq. Respiratoire	2	3	2
Température	20	17	67
Diurèse	0	0	73
FiO2	4	18	71
Compliance	16	51	89

Figure 10: Pourcentages de valeurs manquantes en fonction des différentes variables et des différentes cohortes

	ECMO	Ventilés-CHU	MIMIC
Fréq. Cardiaque	0	0,1	0
SpO2	11,1	0,1	0,1
PA Diastolique	0	0,1	0,5
PA Moyenne	0	0,7	0,4
PA Systolique	0	0,7	0,5
Fréq. Respiratoire	2	0,2	0,9
Température	33	0	8,7
Diurèse	0	0,2	46,6
FiO2	4	2,1	0,1
Compliance	20	44,7	80

Pourcentage de patients avec plus de 80% de valeurs manquantes pour une variable

	ECMO	Ventilés-CHU	MIMIC
Fréq. Cardiaque	0	0,1	0
SpO2	0	0,1	0
PA Diastolique	0	0,1	0
PA Moyenne	0	0,6	0
PA Systolique	0	0,5	0
Fréq. Respiratoire	0	0,1	0
Température	0	0	5,1
Diurèse	0	0,1	29,1
FiO2	0	0,8	0,1
Compliance	7,8	43,4	60,8

Pourcentage de patients avec plus de 90% de valeurs manquantes pour une variable

	ECMO	Ventilés-CHU	MIMIC
Fréq. Cardiaque	0	0	0
SpO2	0	0	0
PA Diastolique	0	0	0
PA Moyenne	0	0,4	0
PA Systolique	0	0,3	0
Fréq. Respiratoire	0	0	0
Température	0	0	2
Diurèse	0	0,1	12
FiO2	0	0,1	0
Compliance	5,2	39	31

Pourcentage de patients avec 100% de valeurs manquantes pour une variable

Figure 11: Pourcentages de patients avec plus d'un certain pourcentage de valeurs manquantes, par variable et par cohorte

6 Analyse des données

6.1 Algorithmes utilisés

Deux types d'algorithmes ont été étudiés lors de ce projet:

- Les algorithmes qui prennent en entrée des données agrégées des variables des patients: par exemple la moyenne, la variance, le maximum de chaque variable dynamique
- Les algorithmes qui prennent en entrée pour chaque patient l'ensemble de la série temporelle (les $15 * 120$ valeurs)

Tous les algorithmes ont été implantés en Python, à l'aide notamment de *Scikit-Learn* pour les algorithmes de Machine Learning traditionnels, et *PyTorch* pour les algorithmes de Deep Learning.

6.1.1 Algorithmes utilisant des données agrégées

Dans le cas de ces algorithmes, les fonctions d'agrégations finalement utilisées sont les suivantes:

- La moyenne
- L'écart-type
- Le maximum et le minimum
- La première et la dernière valeur
- Le coefficient d'asymétrie (skewness) et le kurtosis

Ces fonctions d'agrégation sont appliquées à chaque variable dynamique et dans un premier temps à toute la série temporelle. Dans un second temps, la moyenne, l'écart-type et le maximum/minimum sont aussi appliqués à différentes parties de la série temporelle: aux données des premières 12, 24, et 48 heures ainsi qu'à celles des dernières 12, 24, 48 et 72 heures.

Les principaux algorithmes utilisées sont décrits dans les sous-parties suivantes.

6.1.1.1 Régression logistique

La régression logistique est une technique de classification binaire qui modélise la probabilité qu'un échantillon appartienne à la classe 0 ou 1. Elle utilise une fonction sigmoïde pour transformer une combinaison linéaire des variables indépendantes en une probabilité. Les coefficients de cette combinaison sont ajustés pour minimiser une fonction de coût (telle que la log-vraisemblance négative). Une fois le modèle entraîné, il prédit la classe en comparant la probabilité calculée à un seuil (généralement 0,5). Si la probabilité dépasse ce seuil, l'échantillon est classé comme 1, sinon comme 0.

Cette technique simple de classification sert de ligne de base pour notre problème, permettant d'être comparées aux modèles plus complexes décrits dans la suite du rapport.

6.1.1.2 XGBoost

XGBoost (Extreme Gradient Boosting) [29] est un algorithme d'apprentissage automatique basé sur les arbres de décision, et est particulièrement utilisé pour les problèmes de classification et de régression. Le principe de l'algorithme est le suivant: XGBoost commence par entraîner un arbre de décision simple, appelé faible apprenant, sur les données. Une fois cet arbre construit, l'erreur de prédiction est calculée, par exemple à l'aide de la perte logarithmique en classification ou de la perte quadratique en régression. Pour ajuster les prédictions futures, XGBoost utilise les gradients de ces erreurs, qui indiquent la direction dans laquelle ajuster les prédictions pour minimiser l'erreur. Un nouvel arbre est ensuite formé pour corriger les erreurs des arbres précédents, et les prédictions de tous les arbres sont combinées de manière additive pour améliorer la précision globale du modèle.

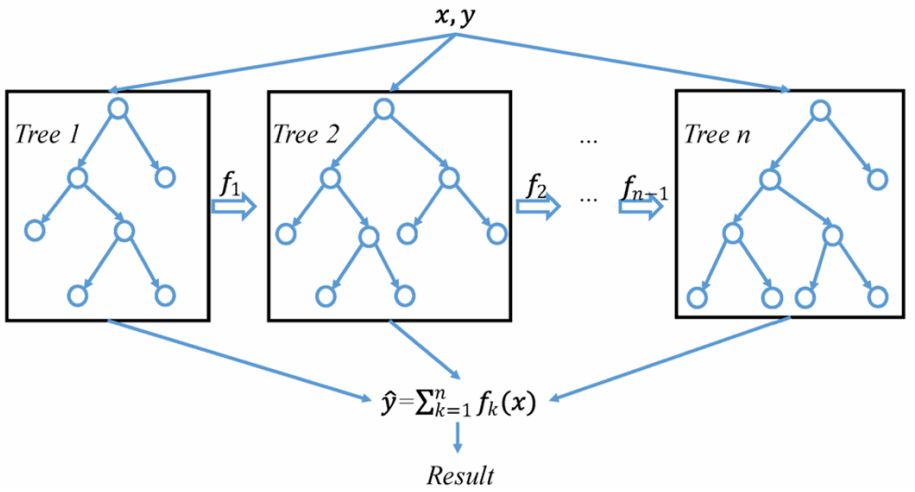

Figure 12: Architecture générale de XGBoost, tirée de l'article [3]

6.1.1.3 LGBM

LGBM (LightGBM, ou Light Gradient Boosting Machine) [30] est un algorithme d'apprentissage automatique basé sur le principe du gradient boosting, comme XGBoost, mais optimisé pour être plus rapide et plus efficace, surtout sur de grands ensembles de données. Contrairement à XGBoost qui construit les arbres niveau par niveau, LightGBM utilise une approche "leaf-wise" (par feuille), scindant les feuilles avec le plus grand gain de réduction de perte pour créer des arbres plus profonds et précis. Il ajoute séquentiellement des arbres de décision pour corriger les erreurs des précédents en minimisant la fonction de perte via le gradient boosting. Conçu pour être rapide, LightGBM utilise des techniques avancées comme la réduction de gradient, tout en intégrant des mécanismes de régularisation pour éviter le surapprentissage.

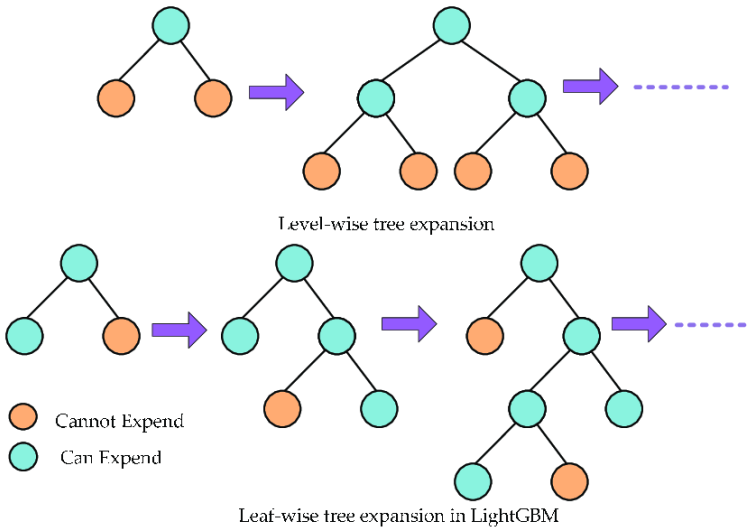

Figure 13: Schéma de l'expansion "leaf-wise" de l'arbre, tiré de l'article [4]

6.1.2 Algorithmes utilisant les données sous forme de séries temporelles

Les algorithmes utilisés pour traiter les séries temporelles entières ont été choisis après l'analyse de nombreux articles étudiant la classification de séries temporelles ainsi que ceux qui traitent des sujets similaires à ceux de notre étude. Quelques méthodes régulièrement mentionnées ou qui semblent obtenir les meilleures performances ont été testées pour notre problème: celles-ci sont décrites brièvement dans les sous-parties suivantes.

6.1.2.1 CNNs

La première méthode envisagée concerne les CNNs (Convolutional Neural Network, ou Réseau de neurones convolutifs), qui ont été initialement introduits dans l'article [31].

Les CNNs sont des réseaux de neurones conçus pour traiter des données structurées en grille, comme les images, ou dans notre cas des séries temporelles. Ils fonctionnent en appliquant des filtres, ou convolutions, sur les données pour détecter des caractéristiques locales. Ces caractéristiques sont ensuite réduites en taille à l'aide de couches de pooling, ce qui diminue la complexité tout en conservant les informations essentielles. Entre chaque couche, des fonctions d'activation comme la ReLU (Rectified Linear Unit) sont appliquées pour introduire de la non-linéarité, ce qui permet au réseau de capturer des relations complexes dans les données. Les CNNs sont composés de plusieurs couches de convolution et de pooling, suivies de couches entièrement connectées qui interprètent les caractéristiques extraites. Le réseau est entraîné par rétropropagation du gradient, un processus où l'erreur entre la prédiction du réseau et la vérité terrain est propagée en arrière à travers le réseau pour ajuster les poids des filtres.

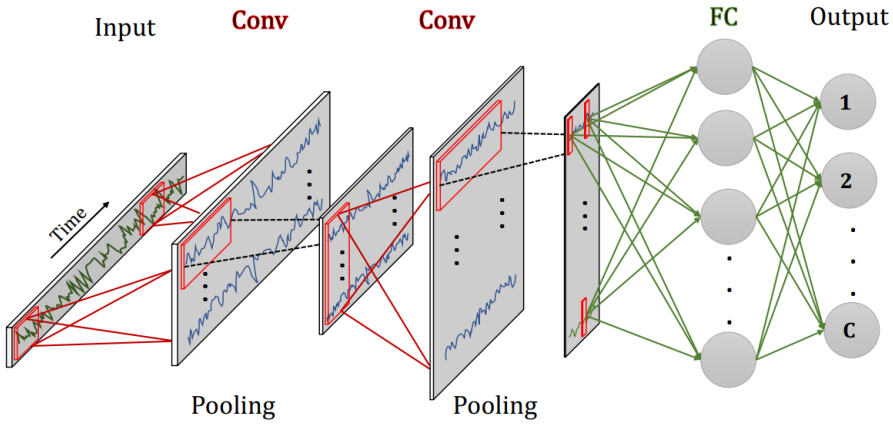

Figure 14: Architecture de LeNet (version spécifique pour les séries temporelles). Schéma tiré de l'article [5]

6.1.2.2 Hydra-MR

Dans leur article, Middlehurst et al. [32] comparent les performances de nombreux algorithmes existants pour traiter des problèmes de classification de séries temporelles. Cet article met notamment en lumière les performances supérieures, sur diverses bases de données, de deux algorithmes: Hive-Cote 2 (Hierarchical Vote Collective of Transformation-Based Ensembles) [33] et HYDRA-MR (HYbrid Dictionary–Rocket Architecture - MultiRocket) [6].

Hive-Cote 2 se base sur une approche ensembliste, i.e. qui combine les résultats de plusieurs modèles de classification pour prendre une décision finale. Hive-Cote 2 intègre plusieurs familles de classificateurs qui opèrent sur différentes transformations des séries temporelles. Hydra-MR est un algorithme qui combine des modèles basés sur les dictionnaires (cf figure 16) et sur les convolutions (cf figure 15). Un modèle basé sur les dictionnaires se concentre sur l'identification et l'extraction de motifs récurrents ou de sous-séquences dans les données, en les représentant sous forme de "mots" à partir d'un ensemble prédéfini de sous-séquences, appelé dictionnaire.

Compte tenu de temps de calcul extrêmement longs, l'algorithme Hive-Cote 2 n'a pu être testé qu'un nombre limité de fois et n'a donc pas pu être comparé aux autres méthodes présentées.

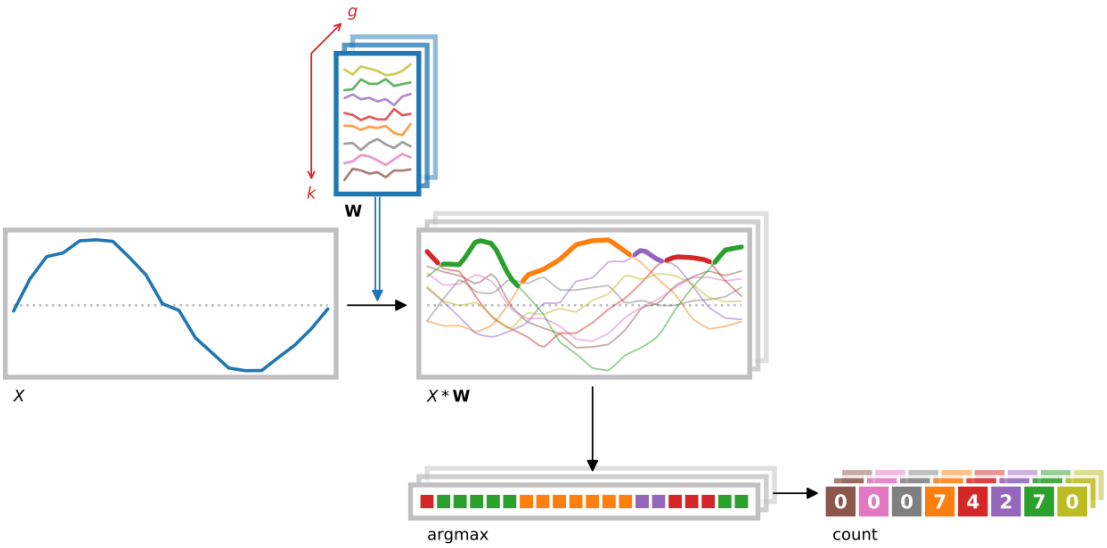

Figure 15: Hydra convolue la série temporelle avec un ensemble de filtres de convolutions aléatoires, et observe à chaque instant les filtres représentant les meilleures correspondances. Image issue de l'article d'origine [6]

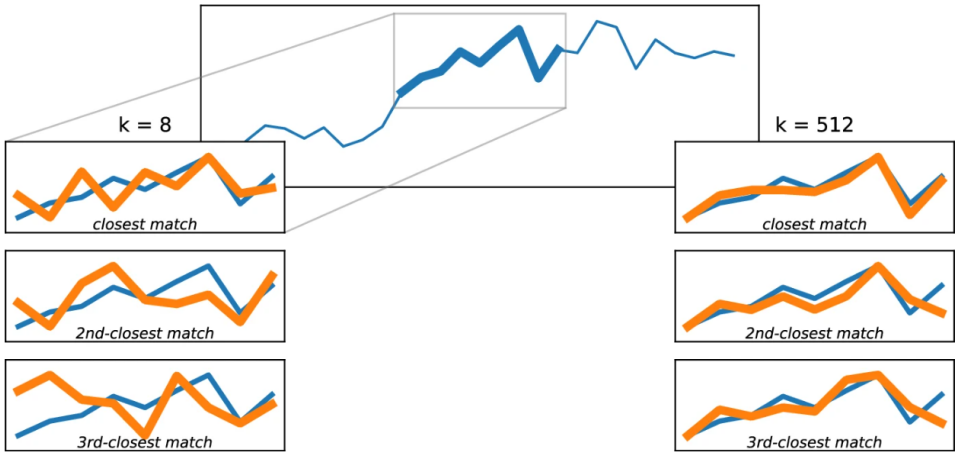

Figure 16: Comparaison d'une partie de la série temporelles à des "mots" du dictionnaire (plus ou moins précis selon les paramètres du modèle). Image issue de l'article d'origine [6].

6.1.2.3 LSTMs

Les LSTMs (Long Short-Term Memory) sont un type de réseau de neurones récurrents (RNN), introduits par Hochreiter et al. dans l'article [34], et conçus pour gérer efficacement les dépendances temporelles dans les séquences de données. Les LSTM utilisent une cellule de mémoire et trois types de portes (oublie, entrée, sortie) pour réguler le flux d'informations, permettant ainsi de conserver et d'utiliser des informations pertinentes sur de longues périodes tout en ignorant les données moins importantes. Cette architecture leur permet de mieux traiter les tâches impliquant des séquences longues, comme la traduction automatique, la prévision de séries temporelles, ou dans notre cas, la classification de séries temporelles.

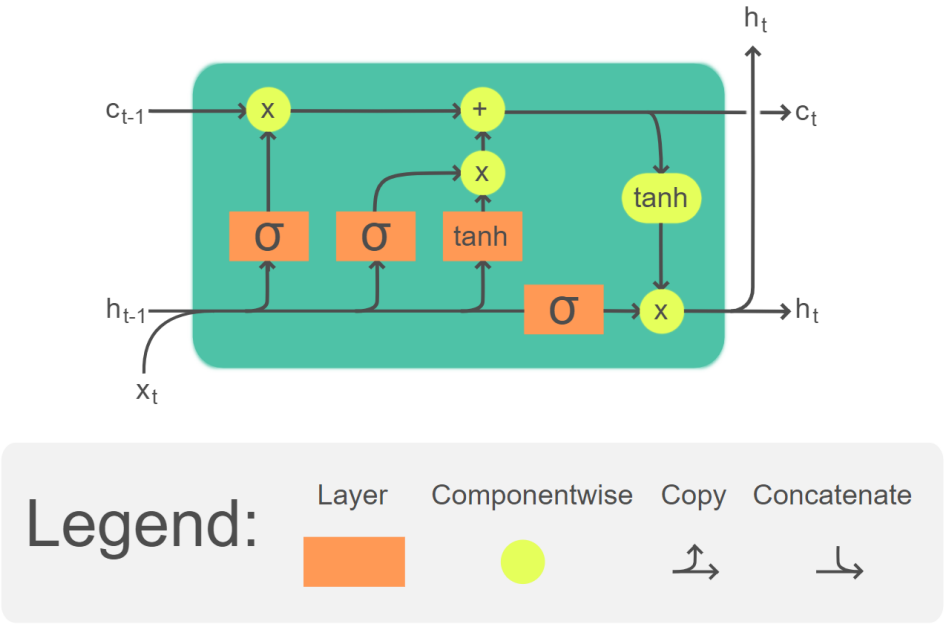

Figure 17: Une cellule de LSTM classique. Schéma tiré de la page Wikipédia sur les LSTMs (https://en.wikipedia.org/wiki/Long_short-term_memory)

Deux architectures différentes ont été testées. La première consiste à utiliser un unique réseau de LSTM pour traiter la série temporelle avant de faire passer les résultats en sortie de ce réseau par une couche linéaire pour réaliser la prédiction finale. La deuxième architecture est inspirée de celle utilisée par Ge et al. [7] (cf figure 18): celle-ci consiste cette fois à appliquer un réseau de LSTM à la série temporelle associée à chaque variable dynamique, puis à concaténer les résultats en sortie avec les variables statiques pour créer une prédiction finale.

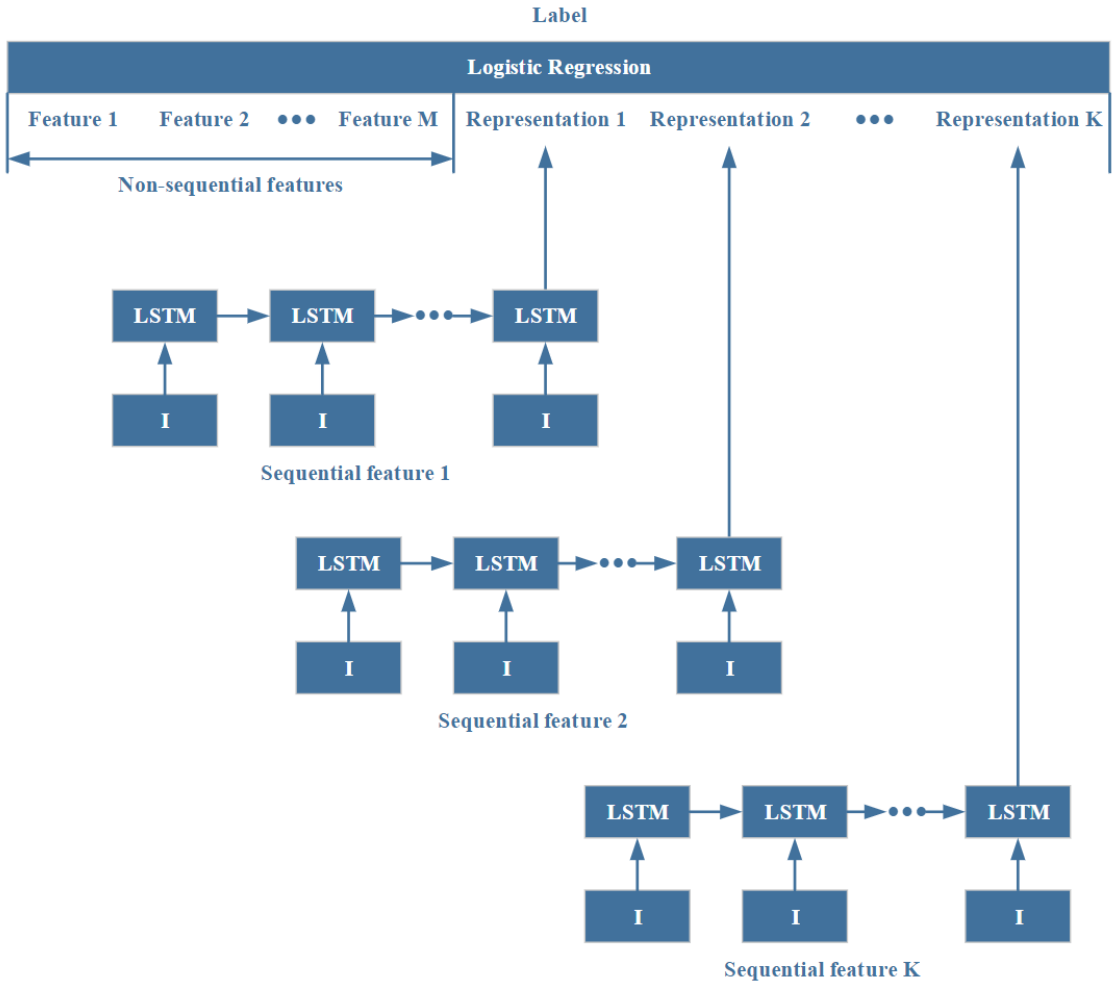

Figure 18: Architecture du Multi-LSTM utilisé. Figure provenant de l'article [7].

6.1.2.4 Inception-Time

Le dernier modèle utilisé est Inception-Time, imaginé par Fawaz et al. [8]. Il s'agit d'un CNN conçu spécifiquement pour la classification de séries temporelles. Inspirée du réseau Inception utilisé en vision par ordinateur, Inception-Time adapte ce concept aux séries temporelles en utilisant une combinaison de convolutions à différentes échelles pour capturer des caractéristiques variées des données.

L'architecture se compose de blocs Inception, qui appliquent des convolutions avec différents filtres de taille, permettant d'extraire des informations sur plusieurs résolutions temporelles simultanément. Cela rend le modèle particulièrement efficace pour traiter des séries temporelles complexes, où des motifs pertinents peuvent apparaître à différentes échelles de temps. L'architecture est schématisée figure 19.

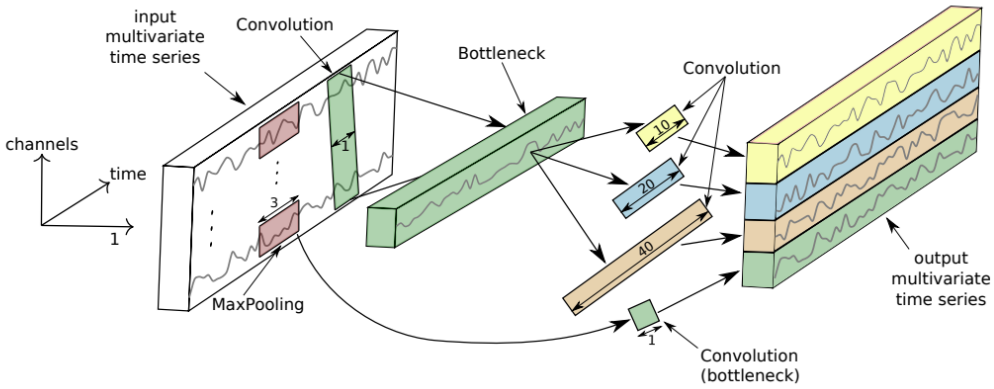

Figure 19: Architecture de Inception-Time. Image tirée de l'article [8]

6.2 Métriques utilisées

6.2.1 Métriques usuelles

Afin de comparer les différents modèles entre eux, il a fallu choisir des métriques pertinentes et adaptées à notre problème. Comme dans la plupart des articles médicaux, les principales métriques utilisées dans ce projet sont les suivantes:

- **L'AUROC** (Area Under the Receiver Operating Characteristic Curve), qui représente l'aire sous la courbe ROC (Receiver Operating Characteristic), sachant que celle-ci trace le taux de vrais positifs en fonction du taux de faux positifs pour différents seuils de classification. C'est cette métrique qui a été utilisée pour comparer les différents modèles, notamment de par sa capacité à évaluer un modèle lorsque les classes sont déséquilibrées, et parcequ'elle ne nécessite pas de fixer un seuil, contrairement aux métriques suivantes.
- Le **F1-Score**, qui combine la précision et le rappel en un seul indicateur. Sachant que la **précision** et le **rappel** (ou **sensibilité**) sont calculés comme suit:

$$Precision = \frac{Vrais\ Positifs}{Vrais\ Positifs + Faux\ Positifs}, \quad et \quad Rappel = \frac{Vrais\ Positifs}{Vrais\ Positifs + Faux\ Négatifs} \tag{3}$$

  Le F1-Score est calculé de cette manière:

$$F1 - Score = 2 * \frac{Précision * Rappel}{Précision + Rappel} \tag{4}$$

- La **spécificité** (ou taux de vrais négatifs), qui est calculée ainsi:

$$Spécificité = \frac{Vrais\ Négatifs}{Vrais\ Négatifs + Faux\ Positifs} \tag{5}$$

6.2.2 Calibration

Un enjeu de la prédiction de mortalité hospitalière consiste aussi en la calibration des modèles: il s'agit d'ajuster au mieux les probabilités de prédiction pour qu'elles reflètent correctement les véritables probabilités d'occurrence des événements (probabilité de décès dans notre cas).

- Une première métrique de calibration est le **score de Brier**, qui consiste simplement à effectuer la moyenne des carrés des différences entre les probabilités prédites p_i et les résultats observés o_i:

$$Score\ de\ Brier = \frac{1}{N}\sum_{i=1}^{N}(p_i - o_i)^2 \tag{6}$$

- Une métrique plus souvent utilisée dans les articles médicaux est celle du **test de Hosmer-Lemeshow** [35]. Ce test consiste à découper les prédictions en 10 groupes de taille égale selon la mortalité prédite, puis à calculer pour chaque groupe i créé, le score suivant:

$$HL_i = \frac{(P_i - O_i)^2}{P_i(1 - \frac{P_i}{N_i})} \ , \tag{7}$$

avec P_i le nombre de décès prédits dans le groupe i, O_i le nombre de décès observés dans ce groupe, et N_i la taille du groupe.

Le score final est obtenu en additionnant les scores des différents groupes.

6.3 Méthodologie

Il a été choisi de scinder chaque jeu de données en deux de manière à obtenir à chaque fois un ensemble d'entraînement (80% du dataset d'origine) et un ensemble de test (20% du dataset d'origine) complètement indépendants.

Dans le cas de Transfer Learning, les modèles sont entraînés sur **toutes** les données d'un dataset (souvent la concaténation des datasets des patients ventilés) et sont testés sur toutes les données des patients d'ECMO.

Enfin, dans le cas du finetuning, les modèles pré-entraînés sont ré-entraînés sur les données des patients sous ECMO en validation croisée (5 folds) pour éviter de tester seulement sur un échantillon restreint de cet ensemble qui pourrait amener à des résultats biaisés.

Pour les entraînements des modèles de Deep Learning, 10% du dataset d'entraînement est choisi comme ensemble de validation. A chaque epoch de l'entraînement, le modèle est testé sur l'ensemble de validation: les paramètres du modèle conservés à la fin de l'entraînement sont ceux du modèle lors de l'epoch où il obtient la meilleure AUROC sur l'ensemble de validation.

6.4 Résultats

6.4.1 Optimisation des modèles

6.4.1.1 Méthodologie d'entraînement

Afin d'optimiser les hyperparamètres ou l'architecture (dans le cas du Deep Learning) des modèles, la démarche a été assez similaire quels que soient les modèles: celle-ci a consisté à construire une grille de recherche des hyperparamètres et à tester les modèles avec les différentes combinaisons possibles pour observer lesquels étaient le plus performants.

Les comparaisons ont été effectuées sur les datasets des patients ventilés car les résultats sont plus fiables avec un grand jeu de données, et que la différence avec le dataset ECMO n'est pas si importante. La métrique de comparaison est l'AUROC. Quelques exemples de grilles de recherches sont présentés figures 20 et 21.

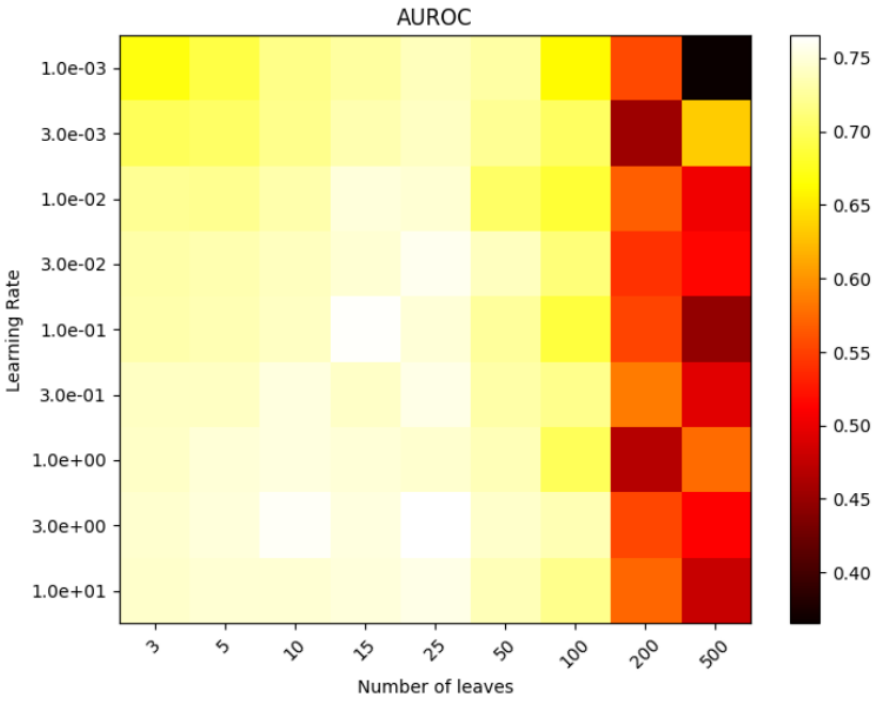

Figure 20: Grille de recherche avec 2 hyperparamètres pour LGBM

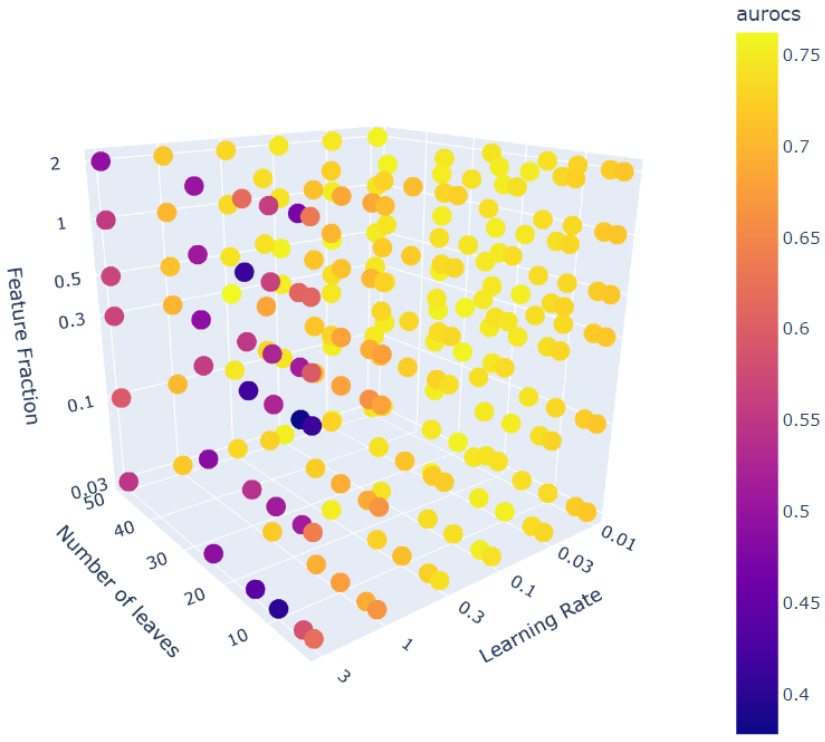

Figure 21: Grille de recherche avec 3 hyperparamètres pour LGBM

Toutefois, pour la plupart des modèles, il n'a pas été possible de tester les combinaisons de manière "exhaustive", de par des contraintes de temps et de ressources (notamment lorsque les modèles ont de nombreux hyperparamètres

importants, ou que leur temps d'exécution est important). Dans ces cas, les hyperparamètres ont été déterminés empiriquement en testant des combinaisons intuitivement plus intéressantes que d'autres, afin de minimiser les temps de recherche.

Ces recherches d'hyperparamètres m'ont semblé indiquer que les performances d'un modèle particulier relèvent plus de la qualité et de la quantité des données que des hyperparamètres ou de l'architecture de ce modèle.

6.4.1.2 Modèles finaux

- La **régression logistique** a été implémentée à l'aide de la fonction *LogisticRegression* de *sklearn*. Les paramètres retenus sont les suivants:
  - $solver =$ 'sag',
  - $penalty =$ '$l2$',
  - $max\_iter = 1000$,
  - $class\_weight = class\_weights$ (avec class_weights contenant les proportions des classes 0 et 1)
- L'algorithme **LGBM** a été implementé à partir du module *lightgbm* et de sa fonction *LGBMClassifier*. Les paramètres retenus sont les suivants:
  - $objective =$ '$binary$',
  - $metric =$ '$binary\_error$',
  - $boosting\_type =$ '$gbdt$',
  - $num_leaves = 15$,
  - $learning\_rate = 0.1$,
  - $feature\_fraction = 0.45$,
  - $n\_estimator = 100$,
  - $class\_weight = class\_weights$ (avec class_weights contenant les proportions des classes 0 et 1)
- L'algorithme **XGBoost** a été implémenté à l'aide de la fonction *XGBClassifier* du module *xgboost*. Les paramètres retenus sont les suivants:
  - $objective =$ '$binary : logistic$',
  - $eval\_metric =$ 'auc',
  - $learning\_rate = 0.06$,
  - $max\_depth = 10$,
  - $n\_estimator = 100$
- L'algorithme **Hydra-MR** a été implémenté à l'aide de la fonction *MultiRocketHydraRegressor* du module *aeon*. Les paramètres retenus sont les suivants:
  - $n\_kernels = 8$,
  - $n\_groups = 64$,
  - $n\_jobs = 1$
- Le **CNN** personnalisé a été implémenté à l'aide du module *torch*. Différentes architectures ont été testées, mais celle qui a obtenu les meilleures performances utilise des filtres de convolution à une dimension. L'architecture détaillé du CNN final est représentée sur la figure suivante (figure 22):

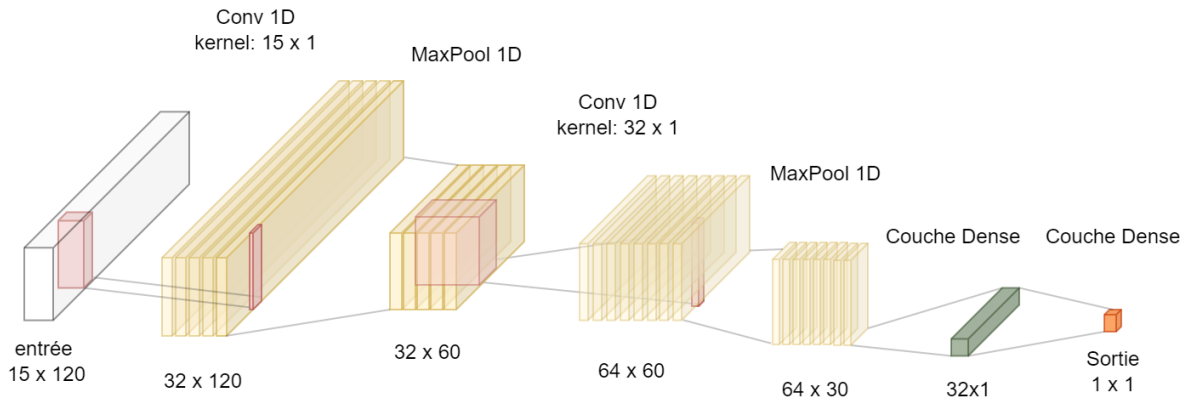

Figure 22: Architecture du CNN final

La fonction *ReLU* a été utilisée en tant que fonction d'activation en sortie des couches de convolution et de la première couche dense. Ce modèle utilise l'optimiseur Adam avec le paramètre $weight\_decay = 0.01$, un batch_size de 64 et la loss choisie est la *BCEWithLogitLoss* (pour Binary Cross-Entropy With Logit Loss) afin de gérer le déséquilibre des classes.

- Les deux réseaux de **LSTM** sont aussi implémentés avec *torch*. Le premier réseau utilise une seule couche de LSTM ($input\_size = 11, hidden\_size = 32, num\_layers = 1$) pour traiter toute les variables dynamiques. Le résultat du LSTM est concaténé aux variables statiques et passe par une couche dense pour calculer la prédiction finale.

  Le deuxième réseau (**"Multi-LSTM"**) utilise une couche de LSTM ($input\_size = 1, hidden\_size = 8, num\_layers = 1$) par variable dynamique. Les résultats des différentes couches sont concaténés aux variables statiques et passent par une couche dense pour calculer la prédiction finale.

  Les deux réseaux utilisent aussi l'optimiseur Adam ($weight\_decay = 0.01$ pour le premier réseau, et $weight\_decay = 0$ pour le second), la *BCEWithLogitLoss* et un batch_size de 64.

- L'implémentation de l'algorithme **Inception-Time** est basée sur le travail trouvé sur la page GitHub suivante: https://github.com/okrasolar/pytorch-timeseries/blob/master/src/models/inception.py . L'architecture retenue est la suivante:

  - 2 modules inception (composés chacun de 3 couches de convolution, un "bottleneck" (avec 2 canaux) et un connecteur résiduel)
  - des filtres de convolution de taille 41

6.4.1.3 Optimisation des entrées des modèles utilisant les agrégations

En ajoutant toutes les données agrégées utilisées pour les algorithmes de Machine Learning traditionnels, on obtient 401 valeurs en entrée au total. Afin d'optimiser le choix de ces valeurs, j'ai utilisé l'algorithme Boruta [36].

Boruta est un algorithme de sélection de caractéristiques qui identifie les variables les plus pertinentes dans un jeu de données en utilisant une approche basée sur l'importance des variables dans les forêts aléatoires. Le processus commence par la création de copies aléatoires des caractéristiques existantes, appelées *shadow features*, qui servent de référence pour évaluer l'importance des caractéristiques réelles. Ensuite, une forêt aléatoire est entraînée sur l'ensemble des caractéristiques réelles et *shadow*, permettant à Boruta de calculer l'importance de chaque caractéristique en fonction de sa contribution à la prédiction du modèle. Boruta compare alors l'importance des caractéristiques réelles à celle des *shadow features* et décide d'accepter, de rejeter ou de marquer comme indécise chaque caractéristique en fonction de cette comparaison.

L'algorithme a été implémenté à l'aide de la fonction *BorutaPy* du module Python *boruta*.

La table suivante donne les résultats avec LGBM (moyennes sur 10 entraînements, entraînements et tests sur le dataset des ventilés) selon les features conservées:

Table 8: Performances selon les features conservées

	AUROC
Features acceptées par Boruta	0.730 ± 0.006
5% des features avec le meilleur rang	0.738 ± 0.004
10% des features avec le meilleur rang	0.734 ± 0.008
25% des features avec le meilleur rang	0.740 ± 0.005
50% des features avec le meilleur rang	0.744 ± 0.005
100% des features	0.748 ± 0.008

Bien que la moitié des features suffise à obtenir un résultat très proche du résultat avec l'ensemble des features, j'ai choisi de conserver l'ensemble de ces features pour les tests suivants car les performances restent les meilleures dans ce cas.

6.4.2 Comparaison des différents modèles

6.4.2.1 Comparaison des AUROCs

Tous les résultats sont données sur la forme "moyenne ± écart-type" sur les différents entraînements effectués. Dans cette partie, les valeurs manquantes des datasets n'ont pas été imputées par l'algorithme SAITS.

La table suivante répertorie les résultats obtenus (AUROC sur l'ensemble de test) par les différents modèles, en entraînant sur 80% d'un dataset et en testant sur les 20% restants:

Entraînements et tests sur :	**LR**	**XGBoost**	**LGBM**	**CNN**	**LSTM**
Ventilés-MIMIC	0.720 ± 0.000	**0.794 ± 0.000**	0.793 ± 0.007	0.752 ± 0.010	0.743 ± 0.018
Ventilés	0.708 ± 0.000	**0.775 ± 0.000**	0.752 ± 0.009	0.738 ± 0.011	0.742 ± 0.006
ECMOs	0.591 ± 0.000	0.680 ± 0.000	0.653 ± 0.030	0.638 ± 0.071	0.585 ± 0.089

Multi-LSTM	**Inception-Time**	**Hydra-MR**
0.749 ± 0.007	0.759 ± 0.014	0.627 ± 0.009
0.748 ± 0.005	0.740 ± 0.007	0.638 ± 0.013
0.692 ± 0.068	0.596 ± 0.072	0.628 ± 0.088

Table 9: Comparaison des AUROCs (sur l'ensemble de test) selon les modèles

Sur les datasets *Ventilés-MIMIC* et *Ventilés*, *XGBoost* obtient les meilleurs résultats. Sur le dataset *ECMO*, c'est cependant le *Multi-LSTM* qui obtient la meilleure AUROC.

La table ci-dessous répertorie les résultats du Transfer Learning appliqué aux ECMOs selon les datasets sur lesquels les différents modèles sont entraînés:

Tests sur ECMO, entraînements sur :	**LR**	**XGBoost**	**LGBM**	**CNN**
Ventilés-MIMIC	0.672 ± 0.000	0.689 ± 0.000	0.691 ± 0.018	0.645 ± 0.013
Ventilés	0.699 ± 0.000	0.704 ± 0.000	**0.756 ± 0.018**	0.671 ± 0.016

LSTM	**Multi-LSTM**	**Inception-Time**	**Hydra-MR**
0.664 ± 0.016	**0.707 ± 0.017**	0.647 ± 0.017	0.576 ± 0.036
0.692 ± 0.013	0.713 ± 0.011	0.665 ± 0.023	0.588 ± 0.030

Table 10: **Transfer Learning** : Comparaison des AUROCs (test sur l'ensemble des ECMOs) selon les modèles et le dataset d'entraînement

Le Transfer Learning améliore nettement les résultats sur les ECMOs quels que soient les algorithmes, notamment sur LGBM qui obtient les meilleurs résultats. On remarque d'autre part que les entraînements sur le dataset *Ventilés* (contenant plus de patients) permettent chaque fois d'obtenir de meilleurs résultats.

La table ci-dessous répertorie cette fois les résultats du Finetuning appliqué aux ECMOs selon les datasets sur lesquels les différents modèles sont pré-entraînés:

Finetuning sur ECMO, entraînement sur :	**CNN**	**LSTM**	**Multi-LSTM**	**Inception-Time**
Ventilés	0.686 ± 0.150	0.701 ± 0.098	**0.743 ± 0.066**	0.693 ± 0.101

Table 11: **Finetuning** : Comparaison des AUROCs (test sur l'ensemble des ECMO) selon les modèles

Grâce au finetuning sur les ECMOs, chacun des algorithmes voit ses performances améliorées.

Les courbes ROC des meilleurs modèles sur le dataset *Ventilés* sont affichées ci-dessous:

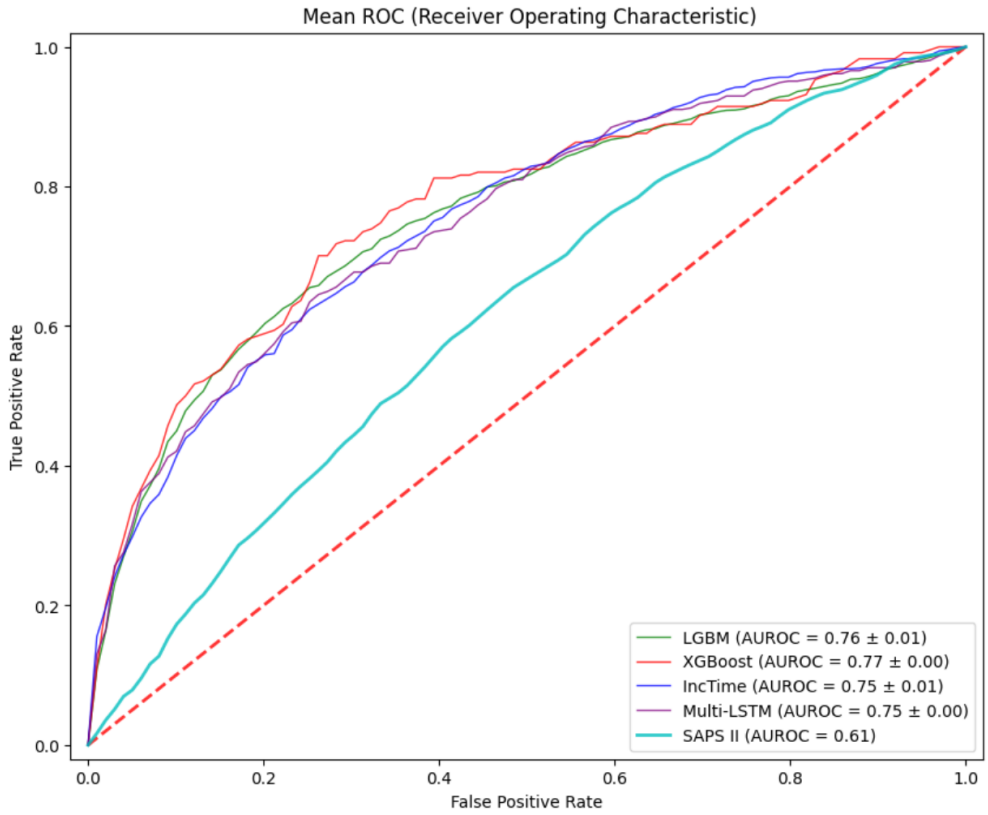

Figure 23: courbes ROC des meilleurs modèles, sur le dataset Ventilés

Les courbes ci-dessous sont celles de ces modèles entraînés sur le dataset *Ventilés* et testés sur le dataset *ECMO*:

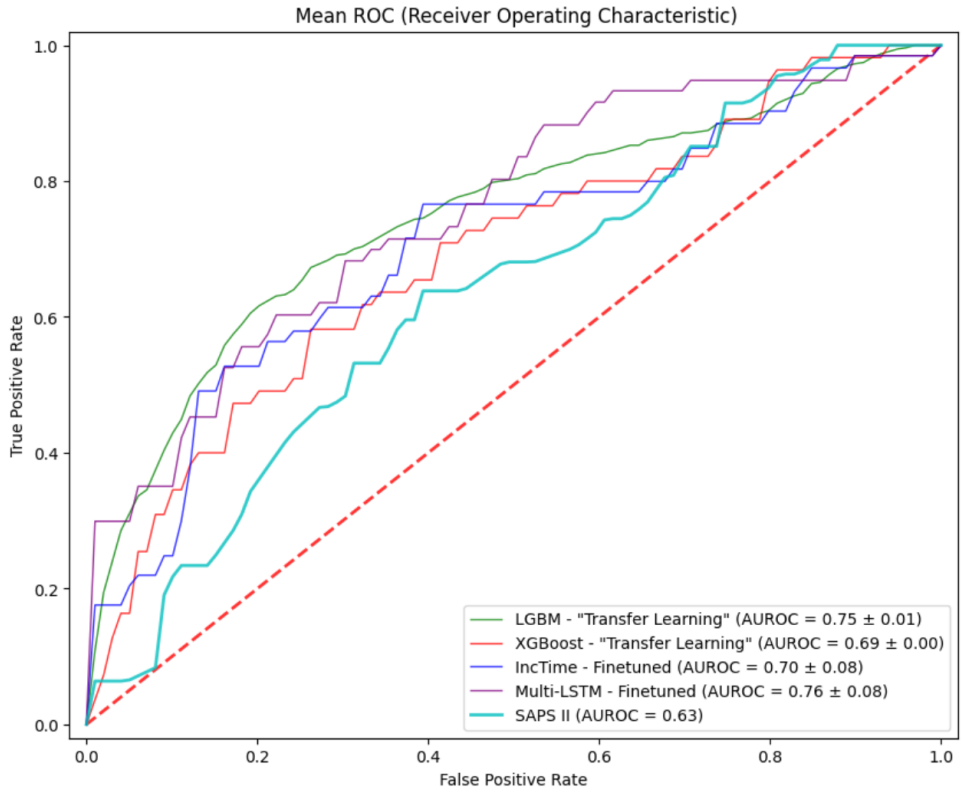

Figure 24: courbes ROC des meilleurs modèles, sur le dataset ECMO

6.4.2.2 Comparaison avec les données imputées par SAITS

On s'intéresse dans cette sous-partie aux résultats obtenus après imputation des valeurs manquantes par l'algorithme SAITS.
On peut alors comparer les trois nouvelles tables obtenues aux précédentes:

Entraînements et tests sur :	**LR**	**XGBoost**	**LGBM**	**CNN**	**LSTM**
Ventilés-MIMIC	0.746 ± 0.000	**0.812 ± 0.000**	0.802 ± 0.008	0.766 ± 0.011	0.741 ± 0.009
Ventilés	0.690 ± 0.000	0.748 ± 0.000	**0.751 ± 0.007**	0.734 ± 0.005	0.738 ± 0.009
ECMOs	0.436 ± 0.000	0.680 ± 0.000	0.666 ± 0.033	0.627 ± 0.064	0.588 ± 0.052

Multi-LSTM	**Inception-Time**	**Hydra-MR**
0.742 ± 0.006	0.765 ± 0.011	X
0.733 ± 0.003	0.731 ± 0.011	X
0.682 ± 0.083	0.602 ± 0.072	X

Table 12: Comparaison des AUROCs (sur l'ensemble de test) selon les modèles

Tests sur ECMO, entraînements sur :	**LR**	**XGBoost**	**LGBM**	**CNN**
Ventilés-MIMIC	0.660 ± 0.000	**0.715 ± 0.000**	0.694 ± 0.022	0.662 ± 0.014
Ventilés	0.693 ± 0.000	**0.746 ± 0.000**	0.742 ± 0.016	0.692 ± 0.016

LSTM	Multi-LSTM	Inception-Time	Hydra-MR
0.651 ± 0.011	0.680 ± 0.017	0.661 ± 0.011	X
0.688 ± 0.005	0.691 ± 0.010	0.671 ± 0.022	X

Table 13: **Transfer Learning** : Comparaison des AUROCs (test sur l'ensemble des ECMO) selon les modèles et le dataset d'entraînement

Finetuning sur ECMO, entraînement sur :	CNN	LSTM	Multi-LSTM	Inception-Time
Ventilés	0.716 ± 0.077	0.639 ± 0.037	0.718 ± 0.047	**0.732 ± 0.065**

Table 14: **Finetuning** : Comparaison des AUROCs (test sur l'ensemble des ECMO) selon les modèles

Si les performances sur le dataset *Ventilés-MIMIC* sont globalement meilleures ou égales à celles obtenues avec l'imputation naïve des valeurs manquantes, la différence est bien moins nette sur les autres datasets, ainsi que dans le cas du Transfer Learning et du finetuning.

6.4.2.3 Comparaison des scores de calibration

Afin d'évaluer la calibration des modèles, les courbes de Hosmer-Lemeshow ont été calculées avec différents modèles. Dans un premier temps pour l'entraînement et le test sur le dataset *Ventilés*:

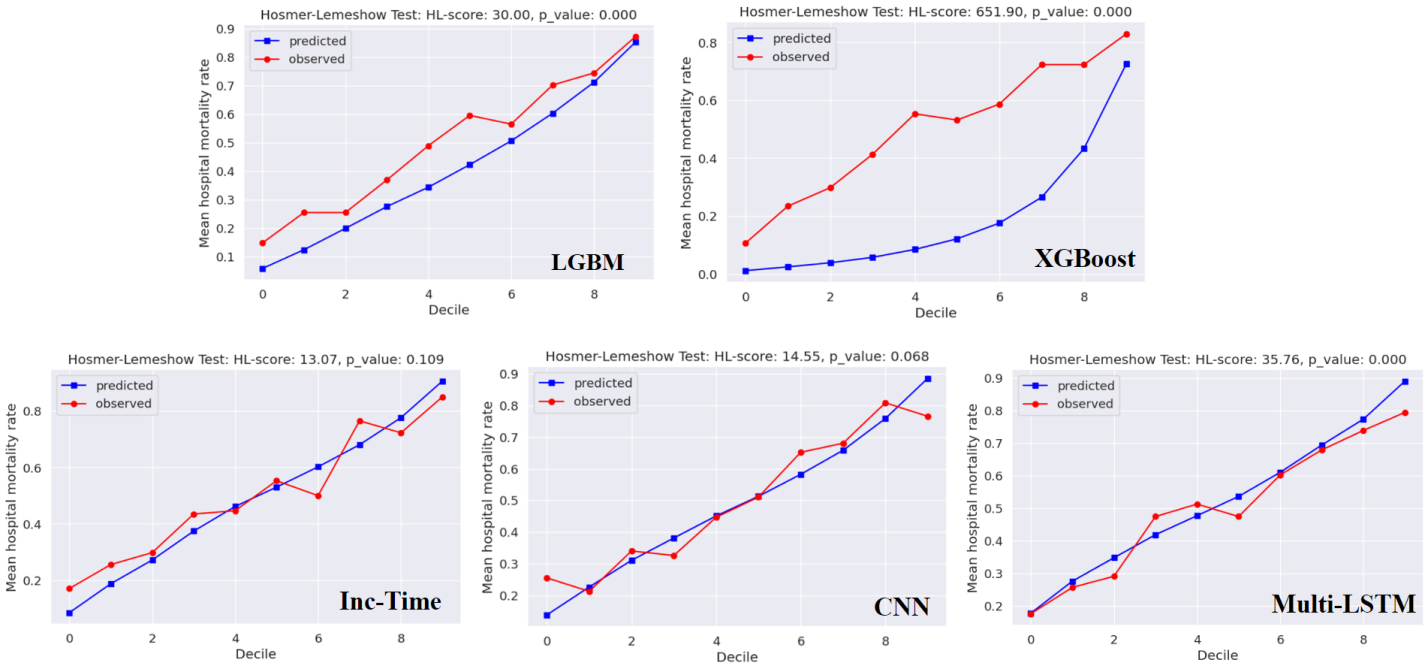

Figure 25: Courbes de Hosmer-Lemeshow (sur *Ventilés*)

Les courbes semblent indiquer que les modèles de Deep Learning permettent une meilleure calibration que *LGBM* et encore plus *XGBoost* (probablement du au fait que ce dernier modèle ne gère pas le déséquilibre des classes comme peut le faire *LGBM*)

Ces mêmes courbes ont également été calculées lorsque l'on effectue du Transfer Learning (entraînement sur *Ventilés* et test sur *ECMOs*)(Figure 26) et du Finetuning (Figure 27)

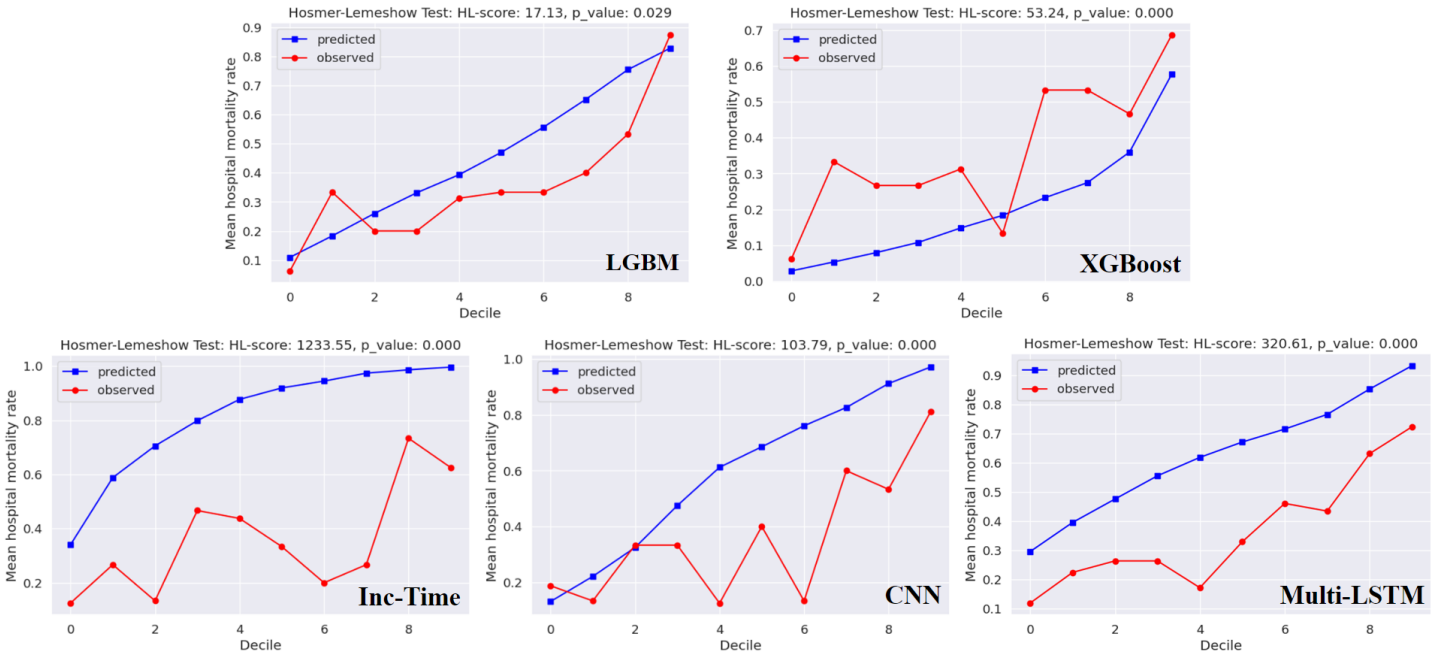

Figure 26: Courbes de Hosmer-Lemeshow (Transfer Learning)

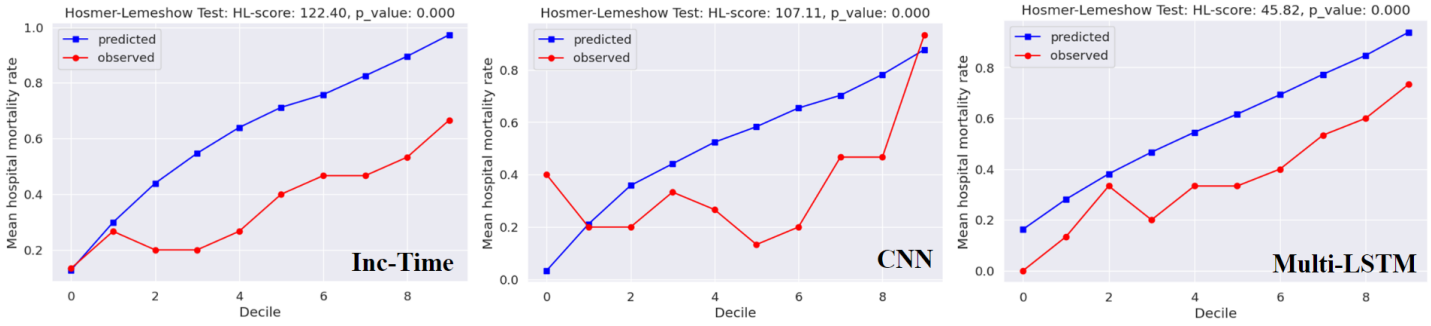

Figure 27: Courbes de Hosmer-Lemeshow (en finetunant sur les *ECMOs*)

Les courbes et les scores semblent montrer que le finetuning permet une calibration légèrement meilleure des modèles sur le dataset *ECMO* comparé au Transfer Learning. Malgé cela, la calibration reste nettement améliorable pour tous les modèles.

Les scores de Brier sont également répertoriés dans la table suivante:

Table 15: Scores de Brier selon les modèles et les entraînements

	Entraînement+Test sur *Ventilés*	Transfer Learning	Finetuning
LGBM	0.211	**0.202**	X
XGBoost	0.298	0.228	X
Multi-LSTM	**0.203**	0.245	**0.236**
Inception-Time	**0.203**	0.275	0.291
CNN	0.209	0.299	0.266

6.4.2.4 Comparaison des matrices de confusion

Si l'AUROC constitue une mesure intéressante de l'efficacité des modèles, permettant facilement de les comparer entre eux, cette métrique ne prend pas en compte le seuil qu'il est nécessaire de fixer pour classifier les patients en sortie des modèles. En effet, selon le seuil choisi, les différentes métriques qui peuvent intéresser un médecin varient fortement, comme le montre la figure suivante:

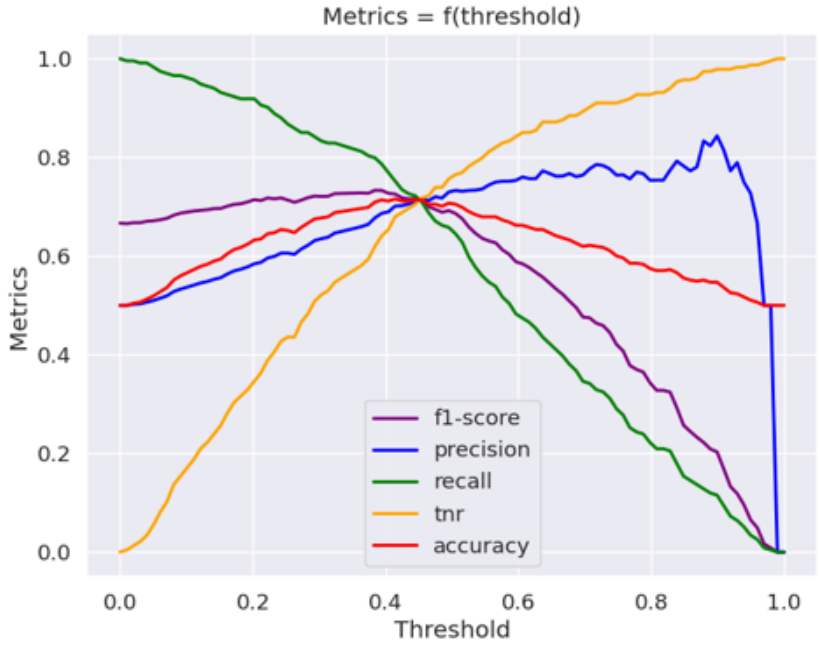

Figure 28: Différentes métriques en fonction du seuil choisi

Comme il n'est pas correct de choisir ce seuil sur l'ensemble de test a posteriori (i.e. après avoir calculé les courbes de la figure 28), le seuil optimal pour une métrique choisie est calculé à partir des résultats sur l'ensemble de validation (pour les modèles de Deep Learning) ou sur l'ensemble d'entraînement (pour les autres modèles). On peut alors obtenir les matrices de confusion sur l'ensemble de test en fonction de la métrique choisie pour optimiser le seuil (f1-score ou accuracy):

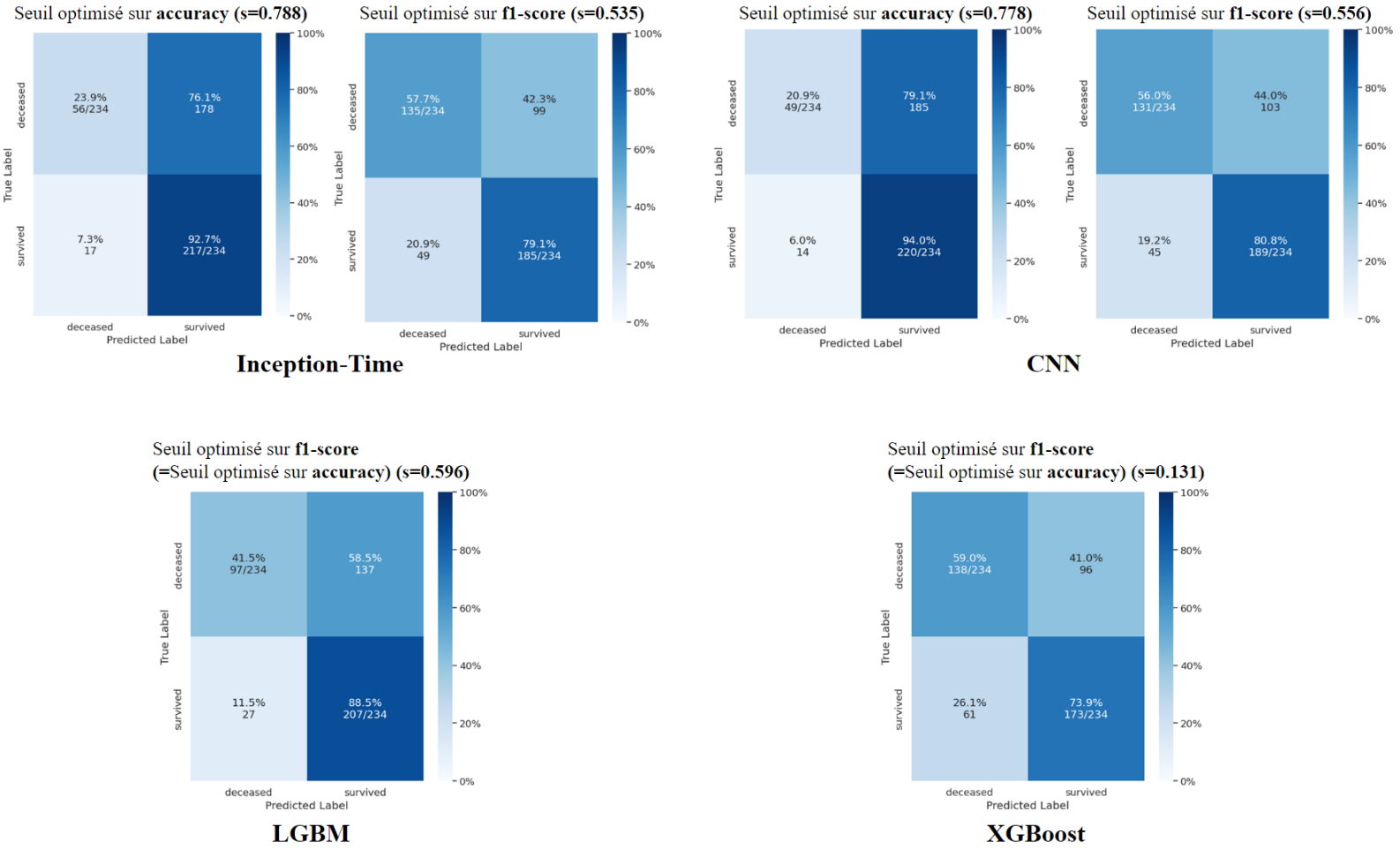

Figure 29: Matrices de confusion sur l'ensemble de test du dataset *Ventilés*, pour différents modèles

Dans le cas du Transfer Learning, le seuil n'est pas optimisé sur le dataset *ECMOs* mais encore sur celui des patients ventilés, car on ne fait que tester le modèle sur les patients sous ECMO. Les matrices de confusion obtenues sont les suivantes:

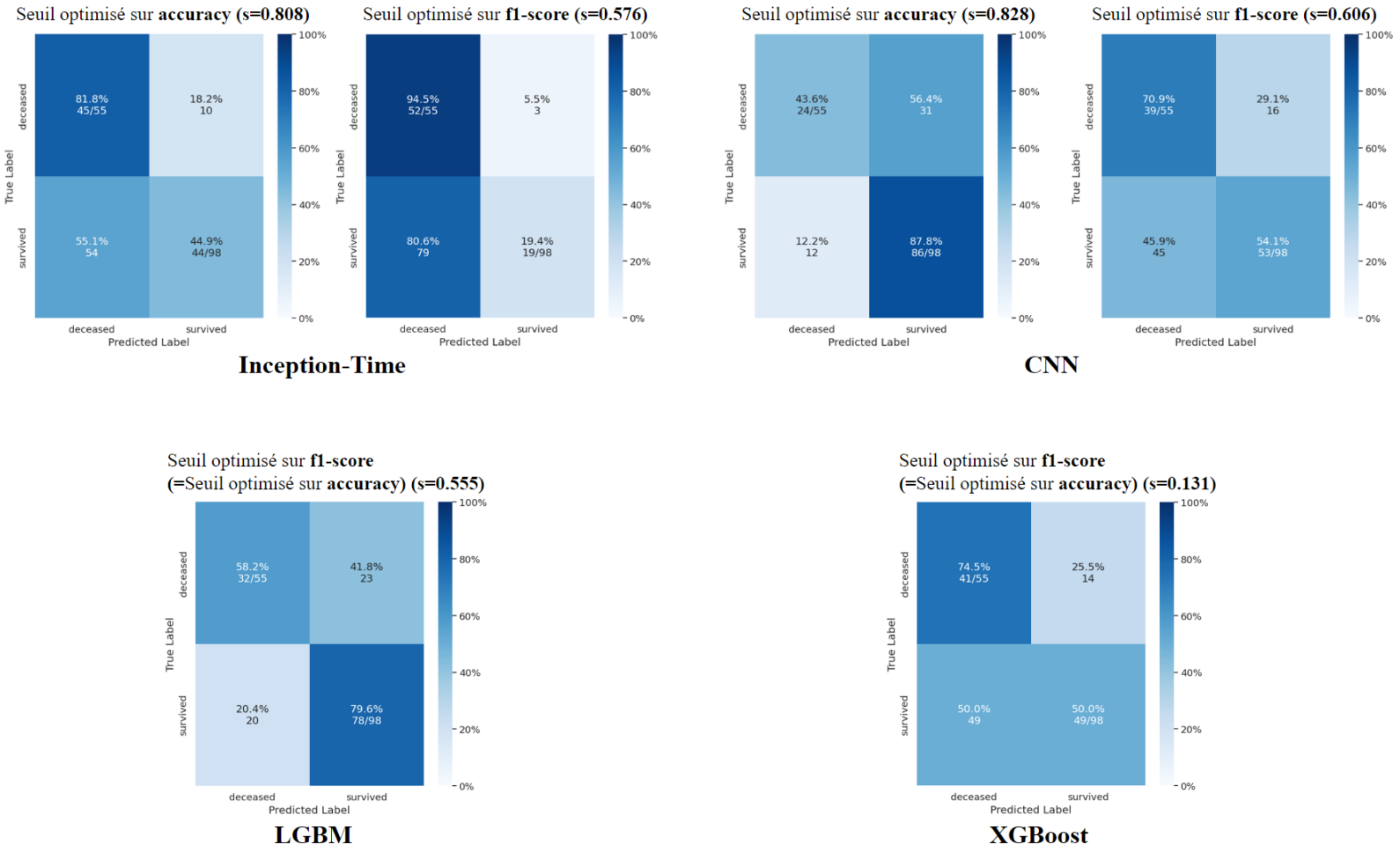

Figure 30: Matrices de confusion sur l'ensemble de test du dataset *ECMOs*, pour différents modèles

6.4.2.5 Comparaison avec augmentation des données

Quelques techniques simples d'augmentation des données ont été testées en fin du projet (en effet, il ne restait pas suffisamment de temps pour tester des techniques plus élaborées). Les trois techniques testées sont les suivantes:

- Le **window warping**: cette technique se base sur l'idée de déformer temporellement des fenêtres d'une série temporelle. Une partie de la série temporelle est alors étirée ou compressée dans le temps.

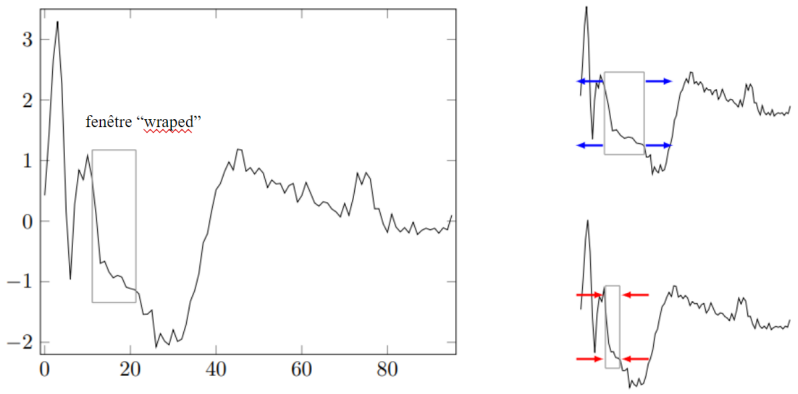

Figure 31: Exemple d'application de la technique de *window warping*, image tirée de l'article [9]

- Le **magnitude warping**: cette technique se focalise sur la modification de l'amplitude des valeurs dans une série temporelle plutôt que sur la déformation temporelle. Elle est particulièrement utile pour rendre un modèle plus robuste aux variations d'échelle ou de magnitude dans les données. Le "magnitude warping" consiste à modifier l'amplitude (ou la magnitude) des points d'une série temporelle de manière non linéaire, en appliquant une transformation continue sur les valeurs.

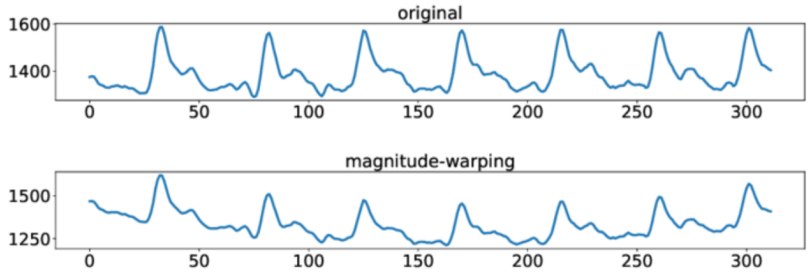

Figure 32: Exemple d'application de la technique de *magnitude warping*, image tirée de l'article [10]

- La technique **SPAWNER** (SuboPtimAl Warped time-series geNEratoR, imaginée par Kamycki et al.[37]): cette méthode identifie des segments significatifs de la série d'origine, les extrait, et les modifie ou les combine pour générer de nouveaux exemples.

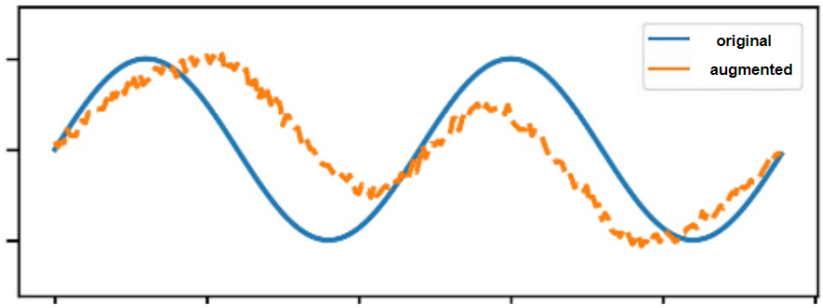

Figure 33: Exemple d'application de la technique *SPAWNER*, image tirée de l'article [11]

La table ci-dessous reporte quelques résultats obtenus à l'aide des différentes techniques d'augmentation des données. Les entraînements et les tests ont été réalisés sur le dataset *Ventilés* à l'aide du *CNN*.

Table 16: Performances selon les augmentations utilisées

	AUROC
Sans augmentation	0.737 ± 0.005
Avec le *window warping*	0.737 ± 0.010
Avec le *magnitude warping*	0.734 ± 0.016
Avec *SPAWNER*	0.740 ± 0.005
Avec les 3 techniques	0.744 ± 0.006

Avec le même modèle, on obtient ces résultats avec les différentes méthodes d'entraînements:

Table 17: Performances selon les augmentations utilisées, avec le **CNN**

	AUROC (sans augmentation)	AUROC (avec les 3 augmentations)
entraînement et test sur *ECMOs*	0.648 ± 0.065	0.645 ± 0.029
entraînement sur *Ventilés*, test sur *ECMOs*	0.671 ± 0.016	0.673 ± 0.008
entraînement sur *Ventilés*, finetuning sur *ECMOs*	0.686 ± 0.150	0.708 ± 0.119

On peut aussi comparer les résultats avec et sans augmentation (en utilisant les trois augmentations) sur les modèles *Inception-Time* et *Multi-LSTM*, en finetunant sur le dataset des *ECMOs*:

Table 18: Performances selon les modèles et l'utilisation d'augmentation (entraînements sur *Ventilés* et finetuning sur *ECMOs*
)

	AUROC (sans augmentation)	AUROC (avec les 3 augmentations)
Inception-Time	0.693 ± 0.101	0.696 ± 0.089
Multi-LSTM	0.743 ± 0.066	0.761 ± 0.085

L'ajout des trois augmentations semble ainsi améliorer légérement les résultats, notamment dans le cas du finetuning.

6.4.3 Différents scopes

Parmi les données utilisées dans les modèles jusqu'à présent, certaines ne sont pas toujours accessibles facilement (la compliance ou la diurèse par exemple). Quatre ensembles de variables plus ou moins complets ont été établis afin de comparer les résultats obtenus en utilisant ces différentes variables:

Table 19: Les quatre ensembles de variables établis

Ensemble 1 (Scope uniquement)	**Ensemble 2 (+Données démographiques)**	**Ensemble 3 (+Scope étendu)**	**Ensemble 4 (+Données DPI)**
Fréquence Card.	Fréquence Card.	Fréquence Card.	Fréquence Card.
Fréquence Resp.	Fréquence Resp.	Fréquence Resp.	Fréquence Resp.
PA Diastolique	PA Diastolique	PA Diastolique	PA Diastolique
PA Moyenne	PA Moyenne	PA Moyenne	PA Moyenne
PA Systolique	PA Systolique	PA Systolique	PA Systolique
SpO2	SpO2	SpO2	SpO2
Température	Température	Température	Température
-	**Age**	Age	Age
-	**Sexe**	Sexe	Sexe
-	**IMC**	IMC	IMC
-	**IGS**	IGS	IGS
-	-	**Compliance**	Compliance
-	-	**FiO2**	FiO2
-	-	-	**Diurèse**
-	-	-	**Dialyse**

Avec les modèles les plus performants, on obtient les résultats suivants sur l'ensemble *Ventilés* (imputé avec SAITS) en fonction de l'ensemble utilisé:

	Ensemble 1	**Ensemble 2**	**Ensemble 3**	**Ensemble 4**
LGBM	0.701 ± 0.006	0.710 ± 0.008	0.718 ± 0.002	0.751 ± 0.007
Multi-LSTM	0.692 ± 0.006	0.710 ± 0.004	0.710 ± 0.007	0.733 ± 0.003
Inception-Time	0.707 ± 0.008	0.712 ± 0.011	0.716 ± 0.006	0.731 ± 0.011

Table 20: AUROC des modèles performants sur l'ensemble *Ventilés* en fonction de l'ensemble utilisé

De la même manière, on obtient les résultats sur *ECMOs* par Transfer Learning:
Puis en finetunant sur les ECMOs:

6.4.4 Importance des variables

Dans le cas des modèles de Machine Learning qui utilisent des données agrégées, il est possible de mesurer facilement l'importance des différentes caractéristiques à l'aide de la fonction *feature_importances_* applicable directement au classifieur (LightGBM en l'occurence), ou à l'aide du module *shap* qui permet des calculer les SHAP Values

	Ensemble 1	**Ensemble 2**	**Ensemble 3**	**Ensemble 4**
LGBM	0.698 ± 0.013	0.719 ± 0.018	0.726 ± 0.020	0.742 ± 0.016
Multi-LSTM	0.640 ± 0.004	0.662 ± 0.022	0.688 ± 0.011	0.691 ± 0.010
Inception-Time	0.591 ± 0.014	0.622 ± 0.038	0.661 ± 0.010	0.671 ± 0.022

Table 21: AUROC des modèles performants sur l'ensemble *ECMOs* (Transfer Learning) en fonction de l'ensemble utilisé

	Ensemble 1	**Ensemble 2**	**Ensemble 3**	**Ensemble 4**
Multi-LSTM	0.623 ± 0.045	0.644 ± 0.095	0.729 ± 0.075	0.718 ± 0.047
Inception-Time	0.539 ± 0.100	0.617 ± 0.063	0.669 ± 0.050	0.732 ± 0.065

Table 22: AUROC des modèles performants sur l'ensemble *ECMOs* (Finetuning) en fonction de l'ensemble utilisé

(**SH**apley **A**dditive ex**P**lanations).

Appliquée à LightGBM, la fonction *feature_ importances_* calcule l'importance des caractéristiques en se basant sur le nombre de fois qu'une caractéristique est utilisée dans les splits des arbres. Chaque fois qu'une caractéristique est choisie pour un split, son importance est augmentée.

Les SHAP Values (introduites dans l'article[38]) s'appuient quant à elles sur le concept de valeurs de Shapley provenant de la théorie des jeux coopératifs. Dans ce contexte, les "joueurs" sont les caractéristiques d'un modèle, et la "récompense" est la prédiction du modèle. Les valeurs de Shapley mesurent la contribution marginale de chaque caractéristique à la prédiction, en tenant compte de toutes les combinaisons possibles des autres caractéristiques.

Le graphe suivant montre les features qui ont le plus d'importances selon la fonction *feature_ importances_* appliquée à LightGBM sur le dataset *Ventilés*:

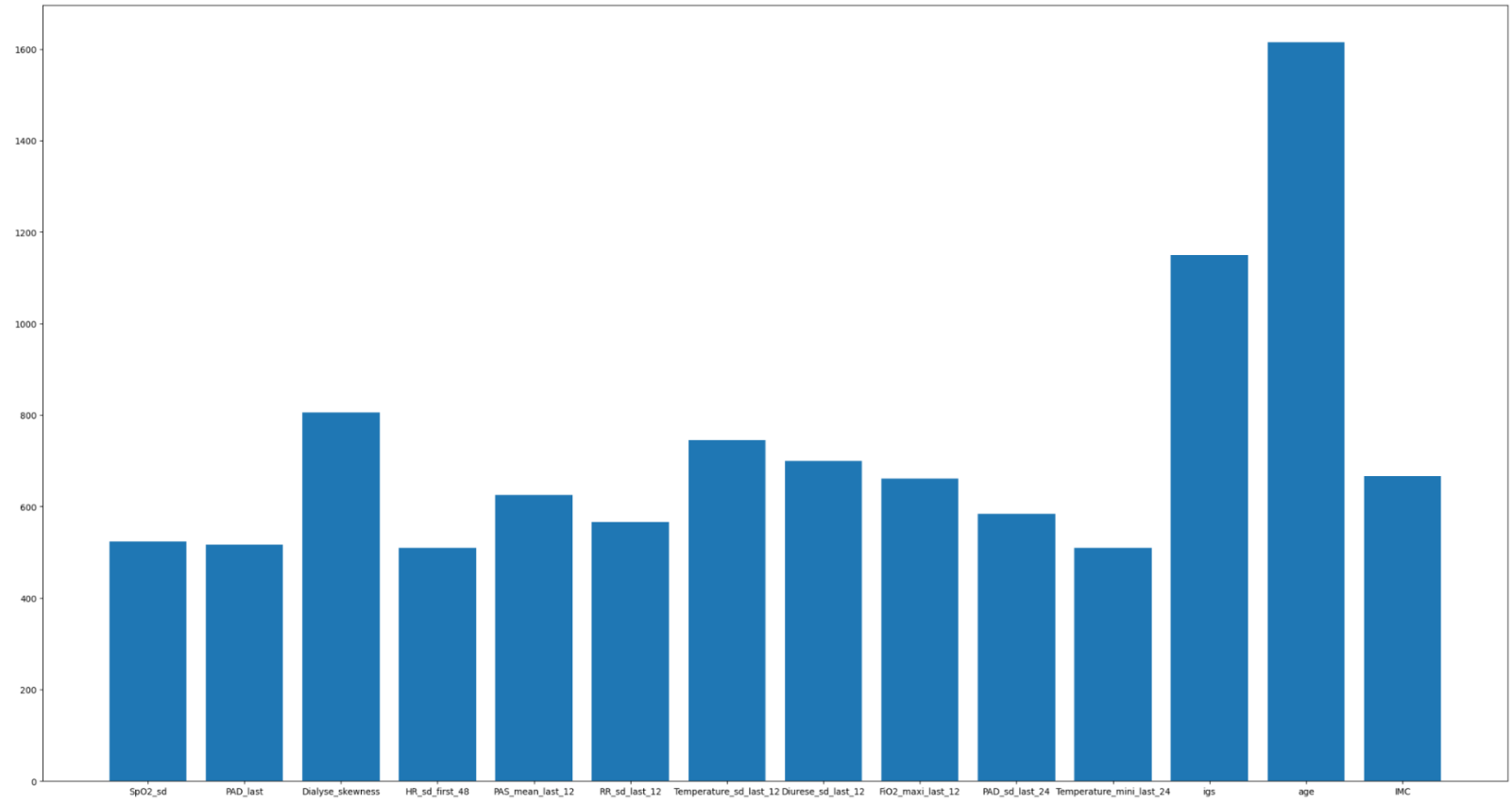

Figure 34: Features les plus importantes pour classifier les patients ventilés

L'importance des variables (dynamiques) peut-être calculée en ajoutant les importances de chaque agrégation pour toutes les variables:

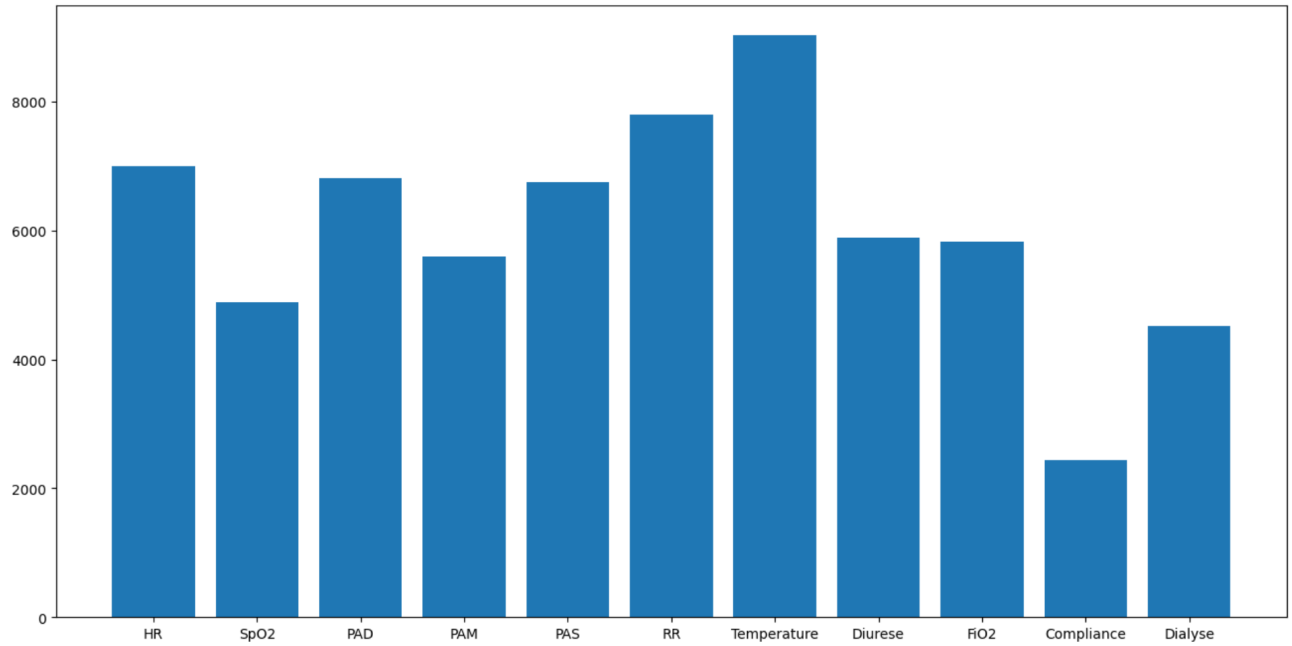

Figure 35: Importance de chaque variable (dynamique) pour classifier les patients ventilés

On peut de la même manière calculer l'importance de chaque agrégation en ajoutant leur importance pour chaque variable:

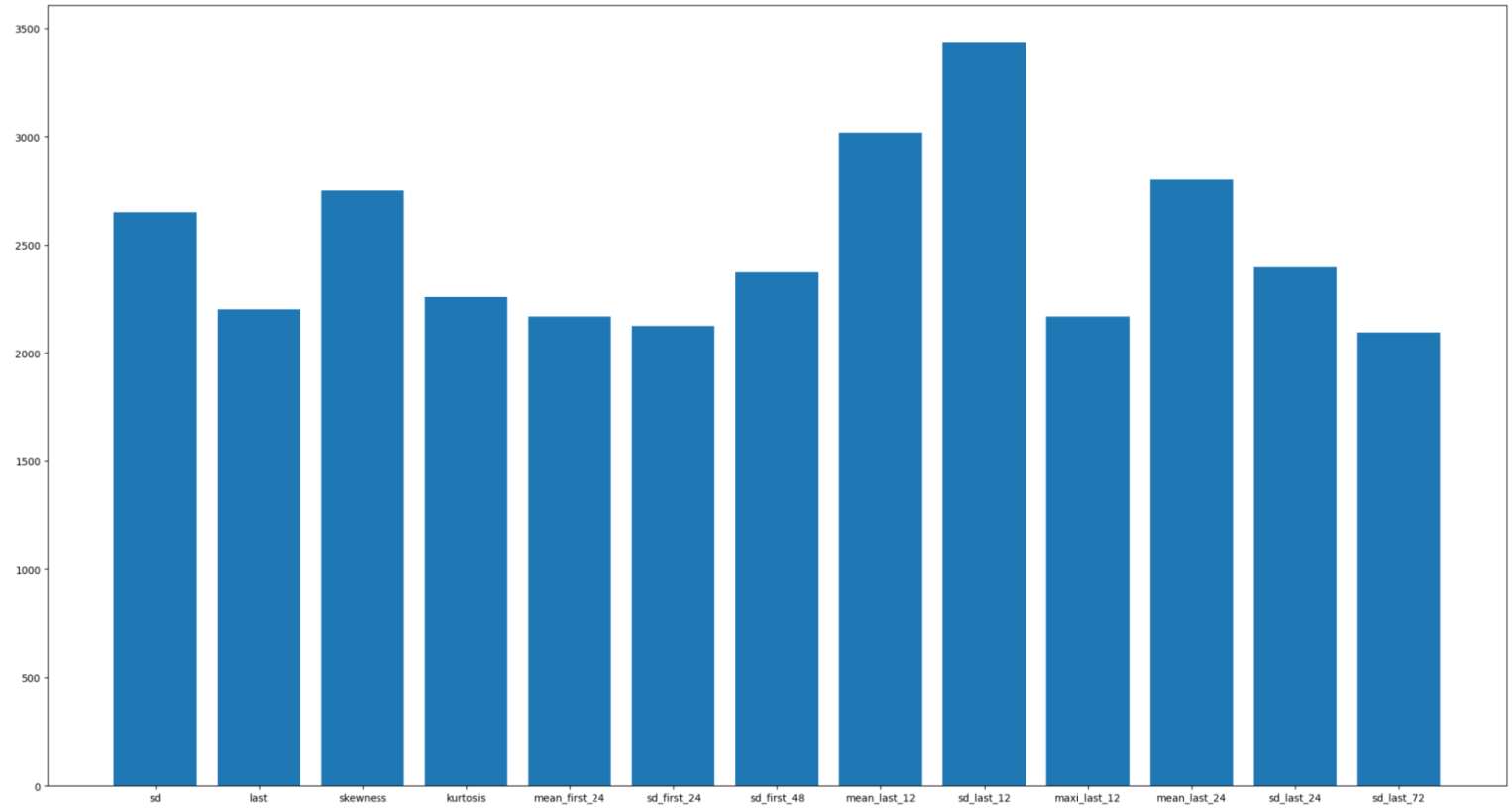

Figure 36: Agrégations les plus importantes pour classifier les patients ventilés

Grâce aux SHAP Values, calculées sur le dataset *Ventilés*, on peut observer l'impact de chaque caractéristique sur les prédictions du modèle (les caractéristiques sont classées par ordre décroissant d'importance) :

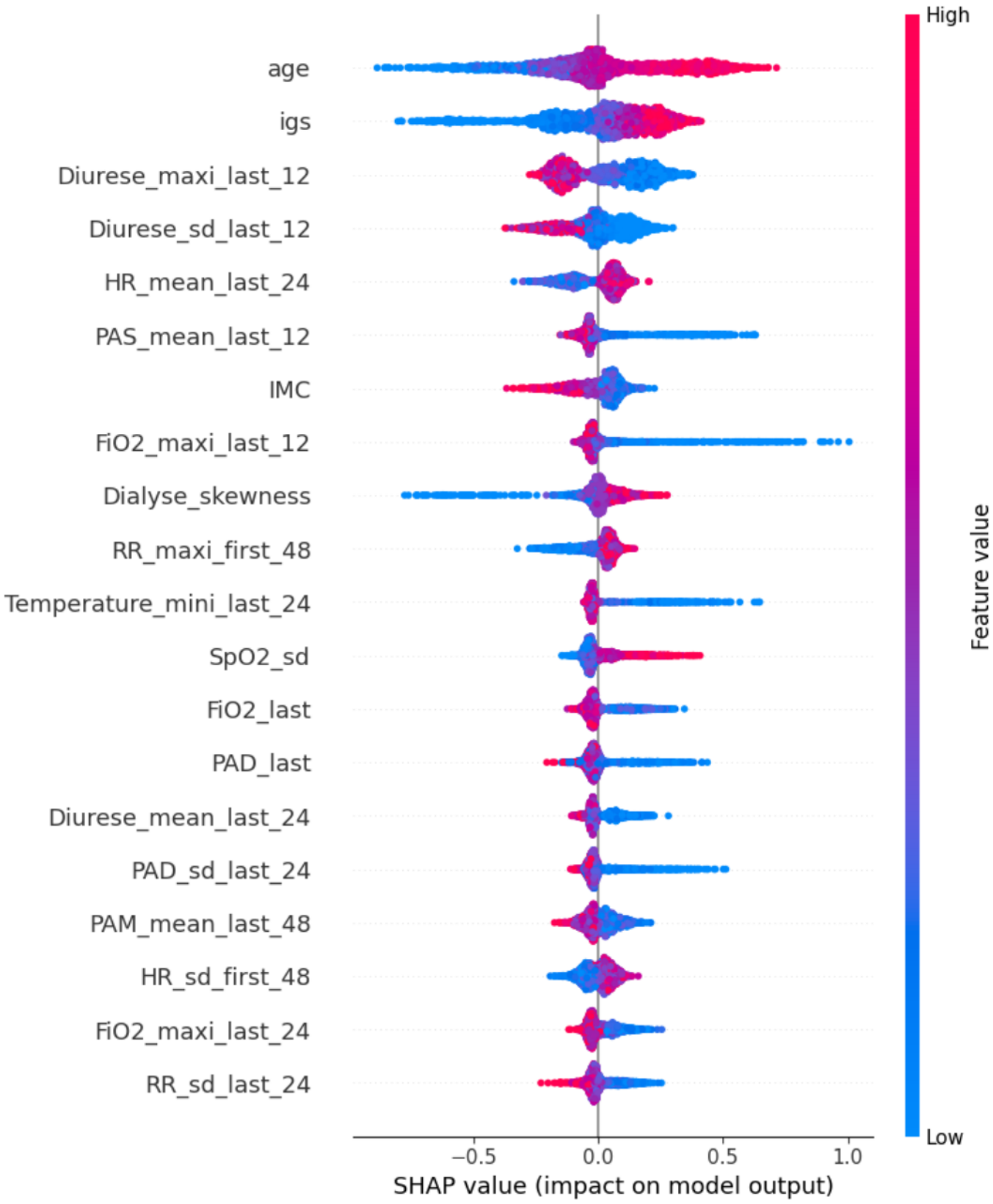

Figure 37: Importance des features, en utilisant les SHAP Values (dataset *Ventilés*)

Enfin, on peut créer le même graphe pour le dataset *ECMOs* cette fois:

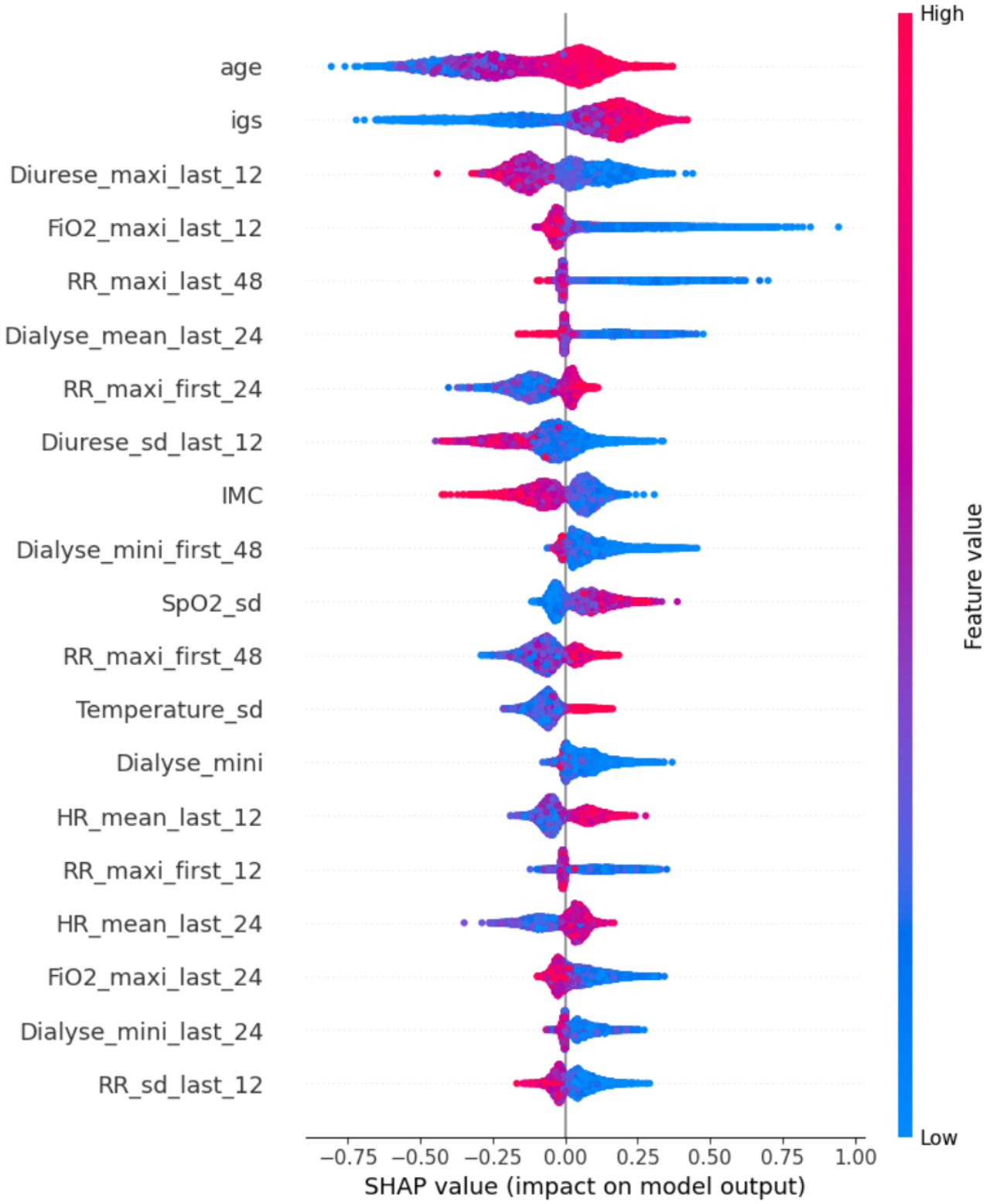

Figure 38: Importance des features, en utilisant les SHAP Values (dataset *ECMOs*)

6.5 Pistes d'amélioration des résultats

6.5.1 Retour sur la gestion des valeurs aberrantes

Durant mon stage, j'ai été amené à participer à une réunion avec Bernard Trillat, chef de projet à l'Hôpital Foch de Paris, et qui présentait son étude sur la détection d'artefact dans les données de pressions artérielles à l'aide de l'algorithme de Du et al. [39]. Ces recherches pourraient ainsi être appliquées au pré-traitement de nos données afin d'améliorer la qualité de celles-ci.

6.5.2 Augmentation/Génération des données

Le manque de données semblant être un des freins les plus importants à la performance des modèles, de nouvelles techniques plus élaborées d'augmentations des données pourraient aider à limiter ce problème. Une autre piste concerne la génération synthétique de données. En effet, ces dernières années, il a été proposé d'utiliser des GAN (Generative Adversarial Networks) ou des LLM (Large Language Model) pour créer de nouveaux jeux de données à partir des datasets d'origine. C'est le cas de l'article [40] qui utilisent un GAN pour synthétiser des données médicales, ou encore [41] qui utilisent quant à eux un LLM pour générer des séries temporelles.

7 Conclusion

En ce qui concerne les résultats de l'étude, les modèles entraînés ont permis de clairement surpasser le score IGS en terme d'AUROC. En effet, à l'aide de l'augmention des données et de l'imputation des valeurs manquantes par SAITS, on obtient, après entraînement sur les données de tous les patients ventilés et finetuning sur les patients sous ECMO, une AUROC de 0.76 avec le modèle *Multi-LSTM*, comparée à une AUROC de 0.63 pour l'IGS. Les modèles *LGBM* et *XGBoost* obtiennent eux aussi des performances similaires en étant entraînés sur les patient ventilés. Les SHAP values soulignent par ailleurs l'importance des données statiques ainsi que des données des dernières heures dans les prédictions de ces derniers modèles. Les performances des modèles sur les ECMOs, notamment en termes de calibration, restent cependant perfectibles. L'ajout de nouvelles données, la génération ou l'augmentation de données sont les solutions qui permettraient le plus probablement d'améliorer ces résultats.

Par ailleurs, d'un point de vue personnel, ce stage m'a permis d'éclairer ma vision du monde du travail, de mieux cerner celui de la recherche, et de développer mes compétences techniques et scientifiques.

References

[1] Patrick M. Wieruszewski, Jamel P. Ortoleva, Daniel S. Cormican, and Troy G. Seelhammer. Extracorporeal membrane oxygenation in acute respiratory failure. 9(1):109–126.

[2] Wenjie Du, David Côté, and Yan Liu. Saits: Self-attention-based imputation for time series. *Expert Systems with Applications*, 219:119619, June 2023.

[3] Yuanchao Wang, Z. Pan, J. Zheng, L. Qian, and Li Mingtao. A hybrid ensemble method for pulsar candidate classification. *Astrophysics and Space Science*, 364, 08 2019.

[4] Sheng Dong, Afaq Khattak, Irfan Ullah, Jibiao Zhou, and Arshad Hussain. Predicting and analyzing road traffic injury severity using boosting-based ensemble learning models with shapley additive explanations. *International Journal of Environmental Research and Public Health*, 19:2925, 03 2022.

[5] Navid Mohammadi Foumani, Lynn Miller, Chang Wei Tan, Geoffrey I. Webb, Germain Forestier, and Mahsa Salehi. Deep learning for time series classification and extrinsic regression: A current survey, 2023.

[6] Angus Dempster, Daniel F. Schmidt, and Geoffrey I. Webb. Hydra: Competing convolutional kernels for fast and accurate time series classification, 2022.

[7] Wendong Ge, Jin-Won Huh, Yu Rang Park, Jae-Ho Lee, Young-Hak Kim, and Alexander Turchin. An Interpretable ICU Mortality Prediction Model Based on Logistic Regression and Recurrent Neural Networks with LSTM units. *AMIA Annual Symposium Proceedings*, 2018:460–469, December 2018.

[8] Hassan Ismail Fawaz, Benjamin Lucas, Germain Forestier, Charlotte Pelletier, Daniel F. Schmidt, Jonathan Weber, Geoffrey I. Webb, Lhassane Idoumghar, Pierre-Alain Muller, and François Petitjean. Inceptiontime: Finding alexnet for time series classification. *CoRR*, abs/1909.04939, 2019.

[9] Arthur Le Guennec, Simon Malinowski, and Romain Tavenard. Data Augmentation for Time Series Classification using Convolutional Neural Networks. In *ECML/PKDD Workshop on Advanced Analytics and Learning on Temporal Data*, Riva Del Garda, Italy, September 2016.

[10] Joseph Azar, Abdallah Makhoul, Raphaël Couturier, and Jacques Demerjian. Deep recurrent neural network-based autoencoder for photoplethysmogram artifacts filtering. *Computers Electrical Engineering*, 92:107065, 06 2021.

[11] Elizabeth Fons, Paula Dawson, Xiao jun Zeng, John Keane, and Alexandros Iosifidis. Evaluating data augmentation for financial time series classification, 2020.

[12] Matthieu Schmidt, Michael Bailey, Jayne Sheldrake, Carol Hodgson, Cecile Aubron, Peter T. Rycus, Carlos Scheinkestel, D. Jamie Cooper, Daniel Brodie, Vincent Pellegrino, Alain Combes, and David Pilcher. Predicting survival after extracorporeal membrane oxygenation for severe acute respiratory failure. the respiratory extracorporeal membrane oxygenation survival prediction (resp) score. *American Journal of Respiratory and Critical Care Medicine*, 189(11):1374–1382, 2014. PMID: 24693864.

[13] Matthieu Schmidt, Elie Zogheib, Hadrien Rozé, Xavier Repesse, Guillaume Lebreton, Charles-Edouard Luyt, Jean-Louis Trouillet, Nicolas Bréchot, Ania Nieszkowska, Hervé Dupont, Alexandre Ouattara, Pascal Leprince, Jean Chastre, and Alain Combes. The PRESERVE mortality risk score and analysis of long-term outcomes after extracorporeal membrane oxygenation for severe acute respiratory distress syndrome. *Intensive Care Medicine*, 39(10):1704–1713, October 2013.

[14] Shaun Davidson, Mauricio Villarroel, Mirae Harford, Eoin Finnegan, João Jorge, Duncan Young, Peter Watkinson, and Lionel Tarassenko. Day-to-day progression of vital-sign circadian rhythms in the intensive care unit. *Critical Care*, 25(1):156, April 2021.

[15] Mucan Liu, Chonghui Guo, and Sijia Guo. An explainable knowledge distillation method with xgboost for icu mortality prediction. *Computers in Biology and Medicine*, 152:106466, 2023.

[16] Chih-Chou Chiu, Chung-Min Wu, Te-Nien Chien, Ling-Jing Kao, Chengcheng Li, and Chuan-Mei Chu. Integrating Structured and Unstructured EHR Data for Predicting Mortality by Machine Learning and Latent Dirichlet Allocation Method. *International Journal of Environmental Research and Public Health*, 20(5):4340, February 2023.

[17] Nora El-Rashidy, Shaker El-Sappagh, Tamer Abuhmed, Samir Abdelrazek, and Hazem M. El-Bakry. Intensive Care Unit Mortality Prediction: An Improved Patient-Specific Stacking Ensemble Model. *IEEE Access*, 8:133541–133564, 2020.

[18] Shinya Iwase, Taka-aki Nakada, Tadanaga Shimada, Takehiko Oami, Takashi Shimazui, Nozomi Takahashi, Jun Yamabe, Yasuo Yamao, and Eiryo Kawakami. Prediction algorithm for ICU mortality and length of stay using machine learning. *Scientific Reports*, 12(1):12912, July 2022.

[19] Leerang Lim, Ukdong Gim, Kyungjae Cho, Dongjoon Yoo, Ho Geol Ryu, and Hyung-Chul Lee. Real-time machine learning model to predict short-term mortality in critically ill patients: development and international validation. *Critical Care*, 28(1):76, March 2024.

[20] Stephanie Baker, Wei Xiang, and Ian Atkinson. Continuous and automatic mortality risk prediction using vital signs in the intensive care unit: a hybrid neural network approach. *Scientific Reports*, 10(1):21282, December 2020.

[21] Korbinian Randl, Núria Lladós Armengol, Lena Mondrejevski, and Ioanna Miliou. Early prediction of the risk of ICU mortality with Deep Federated Learning, December 2022. arXiv:2212.00554 [cs] version: 2.

[22] William Caicedo-Torres and Jairo Gutierrez. Iseeu: Visually interpretable deep learning for mortality prediction inside the icu. *Journal of Biomedical Informatics*, 98:103269, 2019.

[23] Ruoxi Yu, Yali Zheng, Ruikai Zhang, Yuqi Jiang, and Carmen C. Y. Poon. Using a multi-task recurrent neural network with attention mechanisms to predict hospital mortality of patients. *IEEE Journal of Biomedical and Health Informatics*, 24(2):486–492, 2020.

[24] Alistair E. W. Johnson, Lucas Bulgarelli, Lu Shen, Alvin Gayles, Ayad Shammout, Steven Horng, Tom J. Pollard, Sicheng Hao, Benjamin Moody, Brian Gow, Li-wei H. Lehman, Leo A. Celi, and Roger G. Mark. MIMIC-IV, a freely accessible electronic health record dataset. *Scientific Data*, 10(1):1, January 2023.

[25] Jean-Roger Le Gall, Stanley Lemeshow, and Fabienne Saulnier. A New Simplified Acute Physiology Score (SAPS II) Based on a European/North American Multicenter Study. *JAMA*, 270(24):2957–2963, 12 1993.

[26] Wei Cao, Dong Wang, Jian Li, Hao Zhou, Lei Li, and Yitan Li. Brits: Bidirectional recurrent imputation for time series, 2018.

[27] Jinsung Yoon, William Zame, and Mihaela Schaar. Estimating missing data in temporal data streams using multi-directional recurrent neural networks. *IEEE Transactions on Biomedical Engineering*, PP, 11 2017.

[28] Yonghong Luo, Ying Zhang, Xiangrui Cai, and Xiaojie Yuan. E^2gan: End-to-end generative adversarial network for multivariate time series imputation. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 3094–3100. International Joint Conferences on Artificial Intelligence Organization, 7 2019.

[29] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16. ACM, August 2016.

[30] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30:3146–3154, 2017.

[31] Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36(4):193–202, April 1980.

[32] Matthew Middlehurst, Patrick Schäfer, and Anthony Bagnall. Bake off redux: a review and experimental evaluation of recent time series classification algorithms. *Data Mining and Knowledge Discovery*, 38(4):1958–2031, April 2024.

[33] Matthew Middlehurst, James Large, Michael Flynn, Jason Lines, Aaron Bostrom, and Anthony J. Bagnall. HIVE-COTE 2.0: a new meta ensemble for time series classification. *CoRR*, abs/2104.07551, 2021.

[34] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9:1735–80, 12 1997.

[35] David W. Hosmer and Stanley Lemesbow. Goodness of fit tests for the multiple logistic regression model. *Communications in Statistics - Theory and Methods*, 9(10):1043–1069, 1980.

[36] Miron Kursa, Aleksander Jankowski, and Witold Rudnicki. Boruta - a system for feature selection. *Fundam. Inform.*, 101:271–285, 01 2010.

[37] Krzysztof Kamycki, Tomasz Kapuscinski, and Mariusz Oszust. Data augmentation with suboptimal warping for time-series classification. *Sensors*, 20(1), 2020.

[38] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 4765–4774. Curran Associates, Inc., 2017.

[39] Charles Huanghong Du, David Glick, and Avery Tung. Error-checking intraoperative arterial line blood pressures. *Journal of Clinical Monitoring and Computing*, 33(3):407–412, Jun 2019.

[40] Saloni Dash, Ritik Dutta, Isabelle Guyon, Adrien Pavao, Andrew Yale, and Kristin P. Bennett. Synthetic event time series health data generation. *CoRR*, abs/1911.06411, 2019.

[41] Alexandru Grigoraș and Florin Leon. Synthetic time series generation for decision intelligence using large language models. *Mathematics*, 12(16), 2024.