Écosystème multi-agents pour l'intégration d'approches symboliques et numériques : Application à l'amélioration du suivi de patients insuffisants cardiaques

Nadia Abchiche-Mimouni, Laure Bedu, Gilles Bellido, Pascal Goube, Sendah Nouili, Farida Zehraoui

Laboratoire IBISC, univ. Evry, université Paris-Saclay

Rapport de Recherche

Écosystème multi-agents pour l'intégration d'approches symboliques et numériques :

Application à l'amélioration du suivi de patients insuffisants cardiaques

Nadia Abchiche-Mimouni[*], Laure Bedu[*], Gilles Bellido[*], Pascal Goube[+], Sendah Nouili[+], Farida Zehraoui[*]

[*]Laboratoire IBISC

+Centre Hospitalier Sud Francilien (CHSF)

Juillet 2021

RR-IBISC_20210715-01

Préambule

Ce rapport de recherche a été rédigé dans le cadre d'un projet de collaboration entre le service de cardiologie du CHSF (Centre hospitalier Sud Francilien) et le laboratoire IBISC (Informatique, BioInformatique, Systèmes Complexes).
Il regroupe, notamment, des travaux réalisés dans le cadre du stage de Master 2 en Informatique de Gilles Bellido (Année 2016-2017) et des stages de 2ème année d'école d'ingénieur (ENSIIE) de Laure Bedu (Année 2016-2017) et Quentin Picholet (Année 2018-2019). Qu'ils soient remerciés pour leur contribution.

Table des matières

Chapitre 1

Introduction

Ce mémoire se place dans le cadre d'un stage voyant la collaboration entre le laboratoire Informatique, Biologie Intégrative et Systèmes Complexes *(IBISC)* d'Évry Val d'Essonne, et le Centre Hospitalier Sud Francilien *(CHSF)* sur le site de l'hôpital Sud Francilien de Corbeil-Essonnes. Dirigé par Franck Delaplace, le laboratoire *IBISC* abrite des chercheurs se penchant sur le développement de formalismes, de méthodes et d'outils informatiques tournés vers la compréhension de systèmes complexes, vivants ou artificiels. Parmi eux, nous retrouvons Nadia ABCHICHE-MIMOUNI, membre de l'équipe *IRA2*, et Farida ZEHRAOUI, membre de l'équipe *AROBAS*, qui sont les deux encadrantes impliquées sur ce sujet de stage. Ces équipes travaillent respectivement sur des thématiques de systèmes d'assistance à la personne et sur des problématiques algorithmiques, notamment dans le cadre de l'apprentissage automatique.

Inauguré en 2012, le *CHSF* est un groupement hospitalier assurant aujourd'hui la couverture médicale de près de 600 000 personnes dans l'Essonne. Il est composé de plusieurs établissements dont l'hôpital Sud Francilien de Corbeil-Essonnes. Ce dernier regroupe un grand nombre de services différents. C'est dans le cadre d'un récent projet de l'établissement appelé Consultation Semi-urgente d'Insuffisance Cardiaque *(CSIC)* que nous collaborons dans le cadre de ce papier avec le *CHSF*. La *CSIC* est née en début d'année 2017 suite à l'observation d'une explosion du nombre d'hospitalisations pour cause d'insuffisance cardiaque sur la période 2011-2015. Avec un passage de 588 séjours en 2011 à 1063 en 2015, cela représente une augmentation d'environ 80% en seulement 4 ans. De plus, parmi ces 1063 insuffisants cardiaques, 27,93% ont été intégrés dans le cadre d'une réhospitalisation. L'objectif de la *CSIC*, composée de spécialistes de cardiologie et d'infirmiers, est, par conséquent, de déceler les signes précoces de décompensation cardiaque avec pour double objectif :

— d'éviter des complications graves et potentiellement handicapantes pour les malades.
— de désengorger le service des urgences, dont proviennent un grand nombre des hospitalisations enregistrées pour insuffisance cardiaque .

L'insuffisance cardiaque est une condition clinique grave, souvent induite par des malformations cardiaques ou des problèmes d'hypertension mal traités. Cela se traduit alors par une incapacité du cœur à pomper correctement le sang et à le faire circuler dans l'ensemble du corps. Lorsque ce dysfonctionnent s'aggrave, il affecte l'ensemble du métabolisme et on parle alors de décompensation cardiaque. On répertorie un nombre conséquent de signes avant coureurs de complications chez le malade : fatigue constante, difficultés respiratoires (Dyspnée) causées par l'accumulation de fluide dans les poumons, prise de

poids rapide par rétention d'eau etc. Si elle est traitée trop tard, l'insuffisance peut s'avérer mortelle et capable de causer des séquelles graves, entraînant par là même une augmentation de la probabilité de réhospitalisation. En 2015, c'est environ 26 millions d'insuffisants cardiaques qui ont été enregistrés de par le monde [1] et ce chiffre est en hausse depuis plusieurs années. En plus d'être un bilan lourd au niveau humain, on constate un impact important de l'insuffisance cardiaque sur le budget hospitalier mondial. Ainsi, les soins liés à cette condition représentent pour l'Europe et pour l'Amérique du Nord entre 1 et 2% de leurs dépenses totales respectives [2]. L'hospitalisation apparaît comme la principale responsable de ce coût faramineux, avec comme facteurs aggravants des séjours moyens variants entre 5 et 10 jours ainsi qu'une forte probabilité de réadmission dans les 1 à 2 mois suivants [3, 4].

Notre objectif est le développement d'un système de support pour le personnel de la *CSIC* afin de réduire la charge de travail ainsi que d'améliorer le parcours de soin du malade. Le soutien apporté devra s'effectuer à deux échelles. La première sera la facilitation de tâches logistiques, comme le transfert d'informations entre membres de la *CSIC*, pour une optimisation du temps et de l'effort de travail fourni par le praticien. La seconde consiste en la création d'un outil d'aide à la décision qui apportera un avis supplémentaire sur lequel le praticien pourra s'appuyer lors de son service. Pour remplir ce cahier des charges, nous avons sélectionné deux domaines de l'intelligence artificielle : l'apprentissage automatique et les systèmes multi-agents. L'apprentissage automatique ou *Machine Learning* offre une large panoplie d'algorithmes permettant de s'adapter à une grande variété de problèmes. De plus, il est à même de pouvoir exploiter le volume conséquent de données hospitalières restant à ce jour inutilisé par le *CHSF*. Les systèmes multi-agents apportent la possibilité de représenter de manière efficace le projet de la *CSIC* et son fonctionnement, sans pour autant en abandonner la complexité. De plus, les agents possèdent un certain nombre de qualité permettant de représenter les différents acteurs du système.

Nous allons dans un premier temps aborder des notions d'apprentissage automatique et des systèmes multi-agents afin que le lecteur puisse posséder les éléments nécessaires à la compréhension du travail effectué lors de ce stage. Nous y aborderons le vocabulaire ainsi que les différentes techniques communément répandues dans ces deux domaines. Ensuite nous étudierons la manière dont ils sont exploités dans le monde médical et dans la lutte contre les accidents cardiaques. Une fois cette étape de mise en contexte terminé, nous nous intéresserons au cadre de travail de la *CSIC* et de nos objectifs à son égard. Nous y détaillerons les données fournies par le service ainsi que les différents modèles d'apprentissage automatique que nous avons pu expérimenter. Nous passerons ensuite vers une optique plus orientée multi-agents. Nous décrirons une proposition d'architecture ainsi que les choix derrière la structure de cette dernière. Rapidement, nous parlerons de l'implémentation qui a été effectuée pour ensuite ouvrir vers les perspectives de notre travail. Nous conclurons enfin sur les accomplissements et les perspectives de ce stage.

Chapitre 2

État de l'art

Dans cette partie, nous allons nous consacrer à la présentation des deux domaines choisis pour la création de notre système de soutien au personnel médical de la *CSIC* : l'apprentissage automatique et les systèmes multi-agents. Classification et régression. L'un comme l'autre présentent des intérêts spécifiques pour le problème que nous abordons et il est donc essentiel de pouvoir fournir au lecteur les clés pour comprendre le travail qui sera présenté plus avant dans ce mémoire. Pour cela seront discutés notamment les fondements, les objectifs et les challenges classiques associés au milieu.

2.1 Apprentissage automatique

2.1.1 Généralités

Généralement associé au domaine de l'intelligence artificielle, l'apprentissage automatique est une branche dont l'objectif est de permettre à une machine d'apprendre à partir de données et de pouvoir ensuite effectuer des prédictions. Cette approche a notamment pour intérêt de pouvoir résoudre des problèmes qu'il serait trop difficile et/ou coûteux de traiter par des techniques algorithmiques plus classiques. Un autre avantage est la capacité des techniques d'apprentissage automatique de pouvoir effectuer cette tâche sans supervision par un être humain.

Sous-domaine de l'intelligence artificielle, l'apprentissage automatique ou *Machine Learning* est une branche née de la volonté d'entraîner des machines à effectuer des tâches typiquement attribuées à des êtres humains. On peut penser par exemple à la reconnaissance de chiffres et/ou de lettres ou au jeu de Go. Concrètement, cette approche consiste à faire apprendre des concepts ou des règles à la machine à partir de jeux de données. Une fois cette étape finie, on peut alors donner de nouveaux exemples à la machine pour qu'elle puisse en prédire les sorties. Ce qui est recherché au cours de ce processus, c'est le développement de la capacité à généraliser de la machine à travers les concepts appris. Cette manière d'opérer permet notamment de résoudre des problèmes complexes qui se révéleraient trop coûteux en terme de temps et/ou de ressources par des moyens algorithmiques classiques.

L'apprentissage automatique a le vent en poupe ces dernières années et son utilisation se démocratise dans de nombreux domaines comme la bourse, la robotique, la reconnaissance d'images etc. Un des facteurs de ce succès se trouve être la capacité des algorithmes d'apprentissage à pouvoir égaler, voire surpasser les performances humaines. La base de données MNIST [5] est un jeu classique de données dans le milieu de l'appren-

tissage automatique et représente un problème classique de la reconnaissance d'images. Elle contient 60 000 exemples dédiés à l'apprentissage d'écriture manuscrite de chiffres et 10 000 exemples supplémentaires pour la prédiction. Sur ce jeu de données et dans l'état de l'art, le taux d'erreur le plus faible obtenu lors de la classification des images par des machines se limite à 0,21%. De plus, comme son nom le suggère, l'apprentissage automatique présente l'avantage de pouvoir effectuer l'étape d'apprentissage avec peu, voire pas, de supervision humaine.

L'apprentissage peut se subdiviser en trois catégories selon le type de donnée auquel on fait face. Le cas de MNIST présenté précédemment permet un apprentissage dit supervisé. Dans un apprentissage supervisé, chacun des exemples utilisés pour le processus d'apprentissage possède une étiquette indiquant la valeur de sortie qui serait attendue lors d'une prédiction. La machine possède donc des informations sur les concepts qu'elle doit apprendre. Pour MNIST, les étiquettes sont les chiffres que la machine devrait reconnaître pour un exemple donné. Pour des applications réelles, l'apprentissage supervisé est le plus difficile à mettre en place car, compte tenu de la quantité de données collectées de nos jours, il est rare de pouvoir posséder des exemples étiquetés en quantité suffisante. Aussi, la quantité de travail que représenterait l'étiquetage manuel est si colossale qu'elle se révèle rarement concevable. C'est dans ce cas qu'intervient l'apprentissage semi supervisé. Il consiste à travailler conjointement avec des données avec et sans étiquettes. La machine peut alors s'orienter dans son apprentissage grâce aux informations des données étiquetées et développer sa capacité à généraliser sur les données non étiquetées. Globalement il est observé que l'association des deux catégories de données tend à améliorer la précision des modèles. Finalement, lorsqu'il n'existe pas de données étiquetées pour un jeu de données, alors on a recours à l'apprentissage non supervisé. Dans cette situation, la machine va devoir déduire par elle-même les critères et les concepts pertinents, regroupant par exemple des exemples semblant correspondre à un même profil.

Les problèmes traités par l'apprentissage sont de trois ordres : la classification, le *clustering* et la régression. La classification englobe l'ensemble des problèmes où l'on va rechercher à prédire une classe, une catégorie. Dans le cas des données MNIST, les classes que l'on cherche à prédire correctement sont l'ensemble des chiffres allant de 0 à 9. Il est possible d'effectuer de la classification de manière supervisée et semi-supervisée. Le *clustering* peut être vu comme de la classification en milieu non supervisé. L'objectif est de découvrir et de rassembler les cas proches afin de former des groupes, les *clusters*. Tout comme la classification, la régression a besoin d'un paradigme supervisé ou semi-supervisé. Nous cherchons ici à prédire des valeurs continues. Typiquement on peut vouloir prédire le prix d'un bien immobilier en fonction de critères comme sa localisation, la superficie etc. Il est ici nécessaire de posséder des étiquettes pour avoir une échelle sur laquelle pourra s'appuyer la machine pour ses prédictions.

Comme dans tout domaine, l'apprentissage automatique possède son lot de challenges. Pour revenir sur le fait que les algorithmes peuvent apprendre sans besoin d'intervention humaine, il reste cependant une étape préalable qui reste essentielle au développement d'un modèle prédictif : le paramétrage. C'est un problème complexe que de définir les bons paramètres pour un algorithme et il n'existe pas pour l'heure de règle d'or sur le sujet. Parmi ces paramètres, une des questions récurrentes est la sélection du nombre d'*"epochs"*, c'est-à-dire le nombre d'étapes d'apprentissage et de rectification des erreurs effectué par l'algorithme. Une intuition serait d'avoir un grand nombre d'"epochs" ce qui laissera au modèle prédictif plus de temps pour apprendre de ses erreurs et donc de s'améliorer. Le problème auquel on fait face dans ce cas de figure est celui du sur-apprentissage. Le modèle

perd sa faculté à généraliser et se contente de mémoriser les données d'apprentissage. Jusque-là, nous avons surtout abordé des problèmes relatifs au modèle mais il existe aussi des problèmes dont les origines sont les données elles-mêmes. Il n'est pas rare de constater la présence d'exemples dont certaines valeurs sont absentes. Plus difficile à repérer, les données peuvent être bruitées et devenir porteuses d'informations erronées. Cela soulève de question d'indice de confiance en la qualité des données. Certains domaines quant à eux ont plus de difficultés à obtenir la quantité nécessaire d'exemples pour l'apprentissage ou fournir des données qui sont biaisées vers un certain type de classe.

Avant de conclure cette partie consacrée à la présentation de l'apprentissage automatique, nous allons présenter quelques algorithmes classiques du domaine nécessaires pour la compréhension du travail exposé plus tard. On s'intéressera plus particulièrement à leurs applications sur des problèmes de classification puisque cela se révélera, a posteriori, plus pertinent pour le lecteur.

2.1.2 Arbre de décision et forêt d'arbres

L'arbre de décision est une technique d'apprentissage supervisé qui peut être utilisée pour des problèmes de régression ou de classification [6, 7]. Simple à construire, un arbre de décision est composé de trois éléments : des nœuds, des branches et des feuilles. Un nœud contient un paramètre d'entrée sélectionné à partir des données d'apprentissage auquel est appliqué une condition. En fonction de cette dernière, le nœud est suivi par un certain nombre de branches qui correspondent aux différentes réponses possibles. Si la condition est une vérification d'égalité ou d'inégalité comme *(Statut==Marié ?)*, alors il existera deux branches arborant *TRUE* pour l'une et *FALSE* pour l'autre. Si la condition porte sur la valeur même de caractéristique comme *(Statut ?)*, on dénombrera autant de branches que de valeurs potentielles du paramètre : Célibataire, Marié, Veuf ... Les branches peuvent mener à deux éléments : un nouveau nœud, auquel cas on répète le processus, ou une feuille. La feuille est l'élément atteint lorsque le modèle considère qu'il peut se prononcer. Elle contient la classe que l'on cherche à prédire, c'est-à-dire la valeur de la classe.

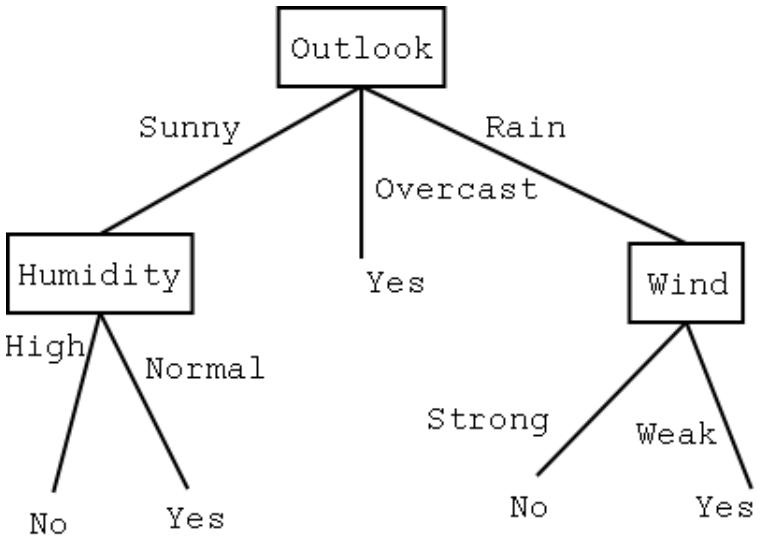


FIGURE 2.1 – Arbre de décision pour savoir si l'on peut jouer au tennis en fonction de la météo

Le plus grand avantage de l'arbre de décision est sa facilité d'interprétation par l'utilisateur. Il suffit de suivre les branches pour comprendre le raisonnement du modèle prédictif pour une solution retournée. Cependant, la construction de l'arbre s'appuie sur des heuristiques comme les algorithmes gloutons, ce qui ne permet pas de s'assurer de l'optimalité de l'arbre retourné. De plus, les arbres de décision sont particulièrement intolérants aux problèmes de données manquantes et leur capacité à généraliser dépend fortement de la complexité du problème à résoudre. Pour mitiger ces lacunes, les forêts d'arbres de décision sont souvent utilisés. Un autre avantage des arbres de décision réside dans l'absence d'hyper-paramètres. La phase de paramétrage est remplacé par une sélection des critères pris en entrée par l'arbre afin d'obtenir des ramifications pertinentes.

Les forêts d'arbres font partie des méthodes dites d'ensemble. Ces dernières consistent en l'association de plusieurs modèles différents pour améliorer les résultats d'une prédiction. Ici les algorithmes associés sont des variations d'arbres de telle sorte que chaque arbre est différent. Lorsqu'un exemple est donné en entrée de la forêt, chaque modèle va fournir le résultat de sa prédiction puis va s'opérer un vote pour définir quelle réponse a été majoritaire. Cette approche permet d'apporter une part de randomisation au processus d'apprentissage et d'améliorer la capacité de généralisation des arbres de décision. Il existe bien sûr un certain nombre de variantes de cette méthode qui vise à pousser plus loin la randomisation et les performances obtenues mais nous ne les aborderons pas ici.

2.1.3 Machines à vecteurs de support

Les machines à vecteurs de support ou Support Vector Machines (SVMs) sont des algorithmes principalement utilisés dans le cadre de l'apprentissage supervisé [8, 9]. Bien qu'il existe des versions capables de résoudre des problèmes de régression, leur application est plus courante pour les problèmes de classification. Pour un sujet de prédiction à deux classes, l'objectif du SVM va être de trouver un hyperplan capable de séparer au mieux les éléments des deux catégories. Puisqu'il existe une infinité d'hyperplans possibles, on évalue la qualité du séparateur en maximisant les marges. Les marges sont calculées en fonction de la distance séparant l'hyperplan de l'élément le plus proche de chaque classe (Distance en fonction d'une droite perpendiculaire à l'hyperplan). Plus les marges sont grandes, meilleure est la séparation entre les classes. Dans le meilleur des cas, comme dans la figure 2.2, les classes sont linéairement séparables. Souvent, on doit relâcher les contraintes pour trouver un hyperplan. Ceci correspond à une tolérance à la présence d'exemples incorrectement classifiés de chaque côté de la séparation. Lorsque les classes ne peuvent pas être séparées d'une manière linéaire et que le relâchement de contraintes ne suffit pas, l'utilisation de noyaux est nécessaire. Un noyau permet de transformer le problème en le transposant dans un espace de dimension supérieure pour le rendre linéairement séparable.

Lorsque le problème est multi-classes, il existe plusieurs stratégies pour trouver un hyperplan séparateur comme le *One-Versus-All* où l'on confronte une classe à l'ensemble des autres en même temps ou la stratégie *One-Versus-One* où l'on confronte les classes deux à deux. La grande difficulté des SVMs réside dans le choix de l'hyperplan séparateur optimal car comme expliqué précédemment, il existe une infinité de possibilités. Aussi, contrairement à ce que peuvent suggérer les figures, il n'est pas toujours possible de visualiser les données que ce soit à cause de la quantité de données ou de leurs dimensions.

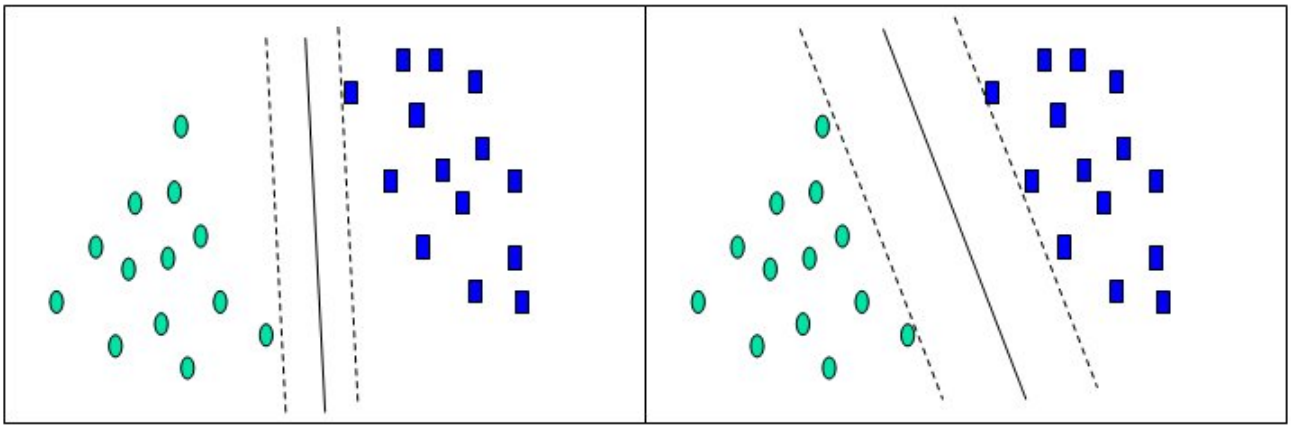


FIGURE 2.2 – Exemple de séparations linéaires possibles entre les ronds verts et les carrés. La séparation de droite est meilleur que celle de gauche puisque la marge séparant chaque classe de la droite se révèle plus grande.

2.1.4 Apprentissage profond

L'apprentissage profond ou *Deep Learning* est une jeune sous-branche de l'apprentissage automatique qui commence à se populariser du fait de ses très bons résultats [10, 11]. Elle est particulièrement prometteuse dans les domaines de la reconnaissance d'images ou de sons, du traitement du langage naturel, des jeux etc. Les modèles d'apprentissage profond, aussi appelés Réseaux de neurones, montrent une plus grande capacité d'abstraction que leurs homologues classiques, ce qui leur permet d'atteindre des performances impressionnantes. Par exemple, le programme *AlphaGo* de *DeepMind* a réussi à battre le meilleur joueur de Go, Ke Jie, en mai 2017, un jeu réputé pour sa complexité en matière de nombre de coups possibles. Comme leur nom l'indique, les réseaux de neurones sont conceptuellement inspirés par le fonctionnement du cerveau et des neurones qui le composent. L'architecture classique, le *Multi Layer Perceptron* (*MLP*), est un graphe dirigé composé de neurones disposés en couches inter-connectées (figure 2.4). La première couche, la couche d'entrée, contient un nombre de neurones égal au nombre de paramètres prédictifs dans l'exemple à prédire. C'est à cette couche que sont fournis les éléments nécessaires à la prédiction. Il existe ensuite un certain nombre de couches dites cachées dont le nombre de neurones peut être plus ou moins grand. Le processus d'abstraction prend place dans ces couches, les représentations étant construites sur les données fournies par les couches précédentes. La dernière couche possède une taille correspondant au nombre de sorties désirées. Le neurone est un élément recevant une certaine quantité d'informations, pondérées, pouvant provenir directement de l'exemple à prédire ou de la couche précédente (figure 2.3). Le neurone possède une fonction d'activation qui définit un seuil à partir duquel le neurone s'activera. Selon les valeurs des informations reçues, le seuil peut être passé ou non. S'il l'est, alors le neurone s'active et transmet à la couche suivante le résultat de son activation. Sinon, le neurone restera tout simplement inactif. Lors de l'apprentissage, à chaque *epoch*, il va y avoir une retro-propagation de l'erreur et l'on va pouvoir connaître les neurones qui en sont responsables. Les poids associés à chacun de ces neurones sont alors modifiés de sorte à rectifier au mieux l'erreur à la prochaine étape. Le processus présenté permet de comprendre le fonctionnement basique d'un réseau de

neurones mais il existe bien sûr de nombreux algorithmes différents avec leurs propres subtilités.

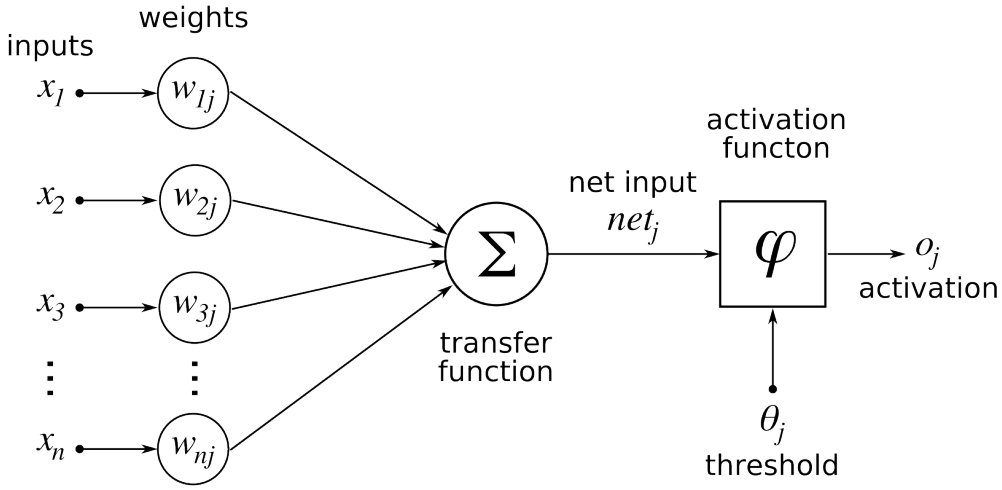


FIGURE 2.3 – Représentation d'un neurone. Il reçoit des informations pondérées en entrée dont la somme active ou non sa fonction d'activation. Ici, la condition d'activation est un seuil.

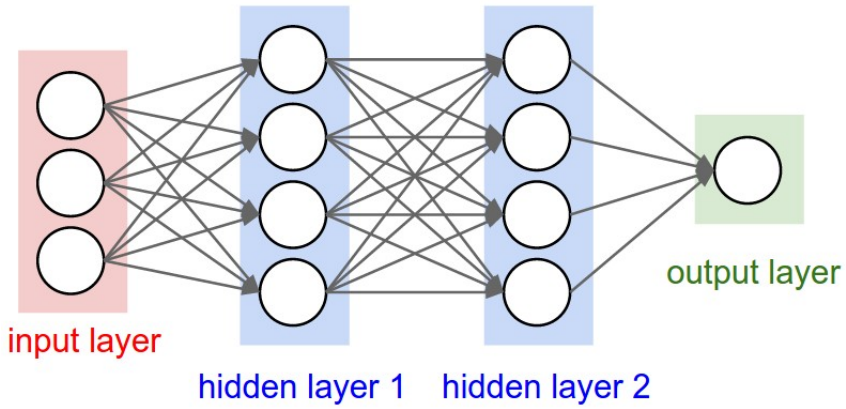


FIGURE 2.4 – Représentation d'un réseau de neurones à deux couches cachées.

Une des critiques récurrentes concernant l'apprentissage profond est son aspect de boîte noire. Bien que l'on sache que l'information est abstraite par les différentes couches cachées, le problème repose dans l'interprétation des réseaux de neurones afin de déterminer ce qu'il y a été appris. L'interprétation des abstractions découvertes par le l'algorithme d'apprentissage constitue le sujet de nombreux travaux. L'extraction d'informations depuis un réseau de neurones constitue une véritable problématique de recherche et nombreux sont les chercheurs se penchant sur la question. La seconde difficulté réside dans le besoin d'une quantité massive de données pour réunir les conditions idéales pour l'entraînement d'un modèle d'apprentissage profond. Dans des domaines, comme la reconnaissance d'image ou du son, qui disposent d'une masse phénoménale d'exemples cela ne représente pas une barrière mais rares sont les autres domaines pouvant se targuer d'avoir autant de ressources.

2.2 Systèmes multi-agents

On définit un système multi-agents comme étant un système composé de plusieurs entités intelligentes appelées *Agents* qui interagissent les unes avec les autres au sein d'un environnement partagé. S'il n'existe pas de consensus sur une définition du terme agent, Wooldridge a été l'un des premiers à en décliner une définition opérationnelle [12]. Selon Wooldridge, le terme *agent* caractérise un système informatique matériel ou (plus souvent) logiciel qui comporte les caractéristiques suivantes :

- Autonomie : l'agent agit sans intervention extérieure (humain ou logiciel) et est capable de contrôler ses actions et ses états internes.
- Capacités sociales : l'agent interagit avec d'autres agents (pouvant être des êtres humains) à l'aide d'un langage de communication dédié.
- Réactivité : l'agent perçoit son environnement et réagit aux changements de ce dernier.
- Pro-activité : l'agent n'agit pas simplement aux stimuli de son environnement, il est aussi capable de mettre en œuvre des comportements dirigés par des buts internes et de prendre des initiatives.

Le paradigme de la programmation orientée agent offre les moyens de travailler avec des systèmes complexes dans des environnements potentiellement distribués. On en trouve des applications dans l'étude des systèmes complexes comme la biologie cellulaire mais également dans des domaines touchant à la logistique de transport de biens, au diagnostic de systèmes etc. Contrairement aux systèmes multi-agents, la notion d'agents et les concepts l'entourant ne font pas encore consensus. Responsable de la variété de la définition de l'agent se trouve la large gamme d'application des systèmes multi-agents, qui ne partagent pas forcément les mêmes exigences.

Cependant, s'il existe des qualités de l'agent que l'on peut considérer comme essentielles, c'est sa capacité à être autonome et à interagir avec les autres agents et/ou l'environnement. L'agent doit être intelligent, se révéler capable de fonctionner sans intervention humaine. Chez un agent simple, on trouvera plutôt une intelligence réactive de l'agent qui se contentera de répondre aux changements et aux stimuli de son environnement. Cette catégorie d'agents se révèle parfaite pour simuler et expliquer des systèmes modérément complexes comme le fonctionnement d'une fourmilière recherchant de la nourriture. Plus un agent se complexifie, plus on s'attend à ce qu'il se révèle proactif dans ses actions. Il ne se contente plus de répondre à son environnement mais est capable de prendre des initiatives, bien souvent orientées par un but qui lui est propre. Cette intelligence permet d'éviter de "coder en dur" les réponses au système, ce qui ferait exploser les coûts en terme de mise en place et de maintien. Dans la littérature, on distingue 2 types d'agents : réactifs et cognitifs. Mais il est courant de concevoir des agents hybrides, combinant ainsi capacités cognitives et capacités réactives.

L'interaction soulève l'aspect de la communication au sein du paradigme multi-agents. Selon les applications, l'acte communicatif entre agents peut revêtir plus ou moins d'importance. Si dans des modèles simples suivant une situation Proie-Prédateur, l'approche concurrentielle des agents implique peu voire pas d'échanges entre les différents agents du système, cela change radicalement dans des modèles se voulant coopératifs. Dans les systèmes coopératifs, chaque agent va posséder ses propres aptitudes lui permettant de remplir des tâches spécifiques et des objectifs qui vont orienter ses actions [13]. Selon les situations, les objectifs des différents acteurs du système peuvent être, partiellement ou

complètement, divergents ou concordants. Ainsi les agents doivent être capables de communiquer les uns avec les autres pour coopérer afin d'atteindre un objectif commun ou de négocier pour trouver des compromis satisfaisant chaque participant [14]. La coopération peut prendre la forme d'une demande de service auprès d'un autre agent pour traiter une tâche ne rentrant pas dans le domaine de compétence de l'agent. Cela peut également être le partage d'une tâche en sous-tâches réparties entre un groupe d'agents pour un traitement plus rapide [15]. C'est cette capacité qui est recherchée et qui est nécessaire pour pouvoir appréhender la complexité des systèmes étudiés. Il n'existe pour l'instant pas de réelles conventions régissant les communications entre agents. Cependant KQML [16] et ACL-FIPA [17], deux langages de communication orientés agent (*ACL : Agent Communication Language*), offrent tout de même un certain nombre de normes utiles permettant la mise en place d'échanges intelligents. ACL-FIPA se pose comme successeur de *KQML* et se popularise parmi les développeurs de systèmes multi-agents.

Les challenges liés au domaine du multi-agents se révèlent nombreux. Ils découlent de la variété des tâches nécessaires à la mise en place d'un système multi-agents. De plus, comme expliqué précédemment, l'absence de standardisation pour l'ensemble du domaine constitue un facteur de complexité supplémentaire. Tout un pan de la recherche se fait dans l'élaboration de l'architecture du système. Le développement d'une architecture peut être subdivisé en plusieurs tâches. Il faut définir les différents types d'agents nécessaires au fonctionnement du système et définir leurs rôles au sein de ce dernier. Cette première étape permet d'établir une hiérarchisation qui permet de visualiser la dynamique du système : Qui communique avec qui ? Y-a-t-il des appariements maître-esclave ? L'environnement est-il distribué ? Existe t-il des cloisons entre les agents ? etc. Les agents nécessitent également le développement d'une architecture interne. Pour un système relativement simple, l'agent pourra être simplement muni de règles ou d'automates à états finis pour répondre aux événements de son environnement. Cependant, cette tâche se révèle aussi complexe que le système dans lequel va évoluer l'agent. Pour être proactif, un agent se doit d'avoir des connaissances et des buts. Cela soulève des questions sur la manière de modéliser ces deux éléments pour qu'ils puissent être exploités mais également pour que l'agent puisse apprendre et évoluer. Cette intelligence s'applique également à l'aspect social de l'agent, qui va devoir voter, négocier, informer d'autres acteurs du système. Cela amène aux protocoles de communication ainsi qu'à la mise en place de mécanismes permettant aux différents agents de pouvoir s'exprimer et surtout de pouvoir se comprendre.

Une fois ces étapes finies émerge la nécessité de trouver des mécanismes de validation et de vérification du système développé et ce en l'absence de normes. La constante évolution d'un système multi-agents ne permet pas la mise en place de tests comme cela pourrait être le cas avec un système implémentant une approche déterministe, puisqu'il n'existera pas réellement d'état normal de référence. Pour ce qui est des communications inter-agents, on retrouve bien évidemment des problématiques de sécurité des communications et dans des milieux sensibles comme la médecine, la vie privée.

Après cette présentation des domaines employés, les deux sections suivantes ont pour objectif de mettre en avant la richesse des applications du monde médical impliquant l'usage de l'apprentissage automatique, des systèmes multi-agents voire des deux domaines. La section apprentissage automatique se concentre sur les techniques employées et le contexte dans lequel elles sont appliquées. La section systèmes multi-agents se consacrera aux différents types de systèmes multi-agents existants. Dans cette seconde partie, il y sera parfois fait mention d'apprentissage automatique mais nous nous contenterons de mettre en lumière sa présence. Cette décision tient au fait que peu de chercheurs éla-

borent cette association des deux domaines, préférant se concentrer sur l'un ou sur l'autre. Aussi, notre sujet ayant trait aux conditions cardiaques, nous avons essayé d'intégrer le plus d'exemples possibles liés à la cardiologie tout en évitant de nuire à la diversité des applications du domaine ; avec l'objectif de mieux situer nos futurs travaux concernant la CSIC.

2.3 Apprentissage automatique et monde médical

De par ses objectifs et ses capacités, le domaine de l'apprentissage automatique a, depuis ses débuts, suscité l'intérêt de chercheurs et d'acteurs provenant de tous les horizons. Parmi eux, on retrouve un grand nombre de personnes affiliées, de près ou de loin, au monde médical, attirées par la perspective de pouvoir exploiter le large volume de données à disposition. Ce constat est encore plus vrai aujourd'hui que par le passé puisqu'il existe une réelle demande de développement d'outils se basant sur l'apprentissage automatique pour le milieu médical. Deux facteurs, liés aux progrès technologiques, en sont les principaux responsables. Le premier est la facilitation de la récolte de données et, par conséquent, la quantité phénoménale d'informations n'attendant que d'être exploitées. En effet, que ce soit un hôpital, un médecin de ville ou un fabricant de *Pacemaker*, toute entité du corps médical récolte d'une manière ou d'une autre des données potentiellement porteuses de connaissances. Cette abondance ainsi que l'hétérogénéité font émerger le problème de l'impossibilité pour un être humain de traiter et apprendre des informations récoltées. C'est en çà que les algorithmes d'apprentissage automatique interviennent puisqu'ils offrent les clés pour pouvoir exploiter la connaissance enfouie dans ce véritable océan de données. Cette qualité est en partie la conséquence des progrès en terme de puissance de calcul des ordinateurs modernes. Également liée à cette évolution technologique, l'apparition de nouveaux algorithmes constitue quant à lui le second facteur de l'intérêt porté par le monde médical à l'apprentissage automatique. La libération de contraintes d'ordre matériel a entraîné une diversification des algorithmes, inenvisageables par le passé. Un des exemples les plus probants reste l'apprentissage profond, qui a dû attendre près de 30 ans pour obtenir un cadre propice à son exploitation. Désormais, il est possible d'apprendre à partir de signaux, d'images, de corpus littéraire etc. Cette richesse se révèle particulièrement attractive pour le milieu qui travaille sur des électrocardiogrammes (Signal), sur des radios (Image) et un grand nombre d'autres médiums.

Au-delà de cela, l'apprentissage automatique se place comme un moyen de diminuer l'impact de la faillibilité humaine. Ainsi, aux États Unis et en 2004, il a été estimé qu'entre 32000 et 98000 patients décédés annuellement des suites d'erreurs médicales dans les hôpitaux [18]. Ces dernières entraînent près de 24 millions de jours supplémentaires d'hospitalisation pour un coût estimé de 4,6 milliards. Ces constats mettent en évidence les coûts en terme de vies humaines, de temps et d'argent. De plus, ces erreurs contribuent elles-mêmes à perpétuer un cercle vicieux. Ces hospitalisations évitables représentent autant de temps ne pouvant être consacré aux autres patients, augmentant ainsi les risques d'erreurs médicales et ainsi de suite. Parmi les coupables derrière ces accidents, on retrouve des problèmes d'organisation de l'établissement [19], la fatigue du personnel, des biais cognitifs etc. [20, 21]. Les algorithmes d'apprentissage automatique offrent donc la possibilité de fournir des diagnostics supplémentaires, avertir de combinaisons de médicaments à éviter etc. tout en réduisant la charge de travail du personnel médical. Sur cette partie, nous nous sommes surtout concentré sur la partie professionnelle du monde médical mais l'apprentissage automatique suscite aussi un intérêt du côté des patients. Avec l'essor de

la médecine personnalisée, nombreuses sont les personnes soucieuses de leur état de santé et qui désirent se surveiller.

Avant de nous intéresser aux techniques d'apprentissage automatique dans le milieu médical et le contexte dans lesquels elles sont utilisées, il est important de développer brièvement la notion de dossiers médicaux électroniques (*DME*s) ou *E-Health Record*s. Les *DME*s remplissent, comme les dossiers médicaux classiques, le rôle de trace de l'état du patient lors de es différentes interactions avec le milieu médical. La différence fondamentale tient dans leur informatisation et dans la centralisation des informations du patient. Théoriquement, par l'intermédiaire d'un *DME*, il est possible d'accéder à des résultats de laboratoire, aux vaccins, aux allergies, aux antécédents et ainsi de suite. Du point de vue du praticien, cela implique notamment une facilité d'accès aux informations présentes et passées du patient et évite la perte de papiers. Du point de vue du *Data Scientist*, le *DME* représente une source riche de par son caractère agrégateur d'informations et surtout permettant une facilitation du travail d'extraction de données. Si le *DME* en tant que tel n'est pas lié au domaine de l'apprentissage automatique, nous tenions à le présenter car de nombreuses recherches présentées par la suite s'appuieront sur ce type de système. Dans leurs travaux, Menachemi et Collum estiment que l'introduction de *DME*s peut réduire de 55% les erreurs graves de traitement et jusqu'à 83% lorsque couplé à un système d'aide à la décision [22]. En France, c'est le Dossier Médical Partagé qui fait office de *DME*, même s'il ne fait pas encore l'unanimité auprès des professionnels de la santé.

Rôle emblématique du médecin, le diagnostic est la première tâche à laquelle on pense lorsque l'on parle des applications de l'apprentissage automatique dans le domaine médical. Cependant, ce terme ne renvoie pas à la complexité et la diversité des formes que cette tâche peut revêtir. En diagnostiquant, le médecin peut chercher à identifier un type de tumeur sur une radiographie, tenter de repérer la présence d'un souffle au cœur ou bien isoler la cause d'une fièvre. En excluant le choix des algorithmes, il existe deux approches différentes au développement d'outil d'aide au diagnostic : l'outil spécialisé et l'outil générique. *Intelligent Heart Disease Prediction System*(*IHDPS*), développé par Palaniappan et al, fait partie de cette première catégorie car il s'intéresse à la résolution d'un problème bien particulier [23]. *IHDPS* est un système d'aide à la décision en ligne ayant pour objectif l'évaluation des risques d'accidents cardiaques. Là où la plupart des outils se concentrent sur un unique algorithme pour effectuer la prédiction, *IHDPS* propose en sortie les résultats de trois différents algorithmes : un réseau de neurones, un arbre de décision et un classifieur bayésien naïf. Ce dernier est un classifieur statistique qui considère qu'il n'existe pas de dépendances entre les attributs de la prédiction. On évalue la manière dont les attributs et leurs valeurs impactent sur la classe prédite (Figure 2.5). Tout comme l'arbre de décision, le classifieur bayésien naïf offre une explication du choix du résultat final. Cette propriété se révèle très intéressante car d'une part, il est estimé que la clarté des systèmes d'aide à la décision se révèle tout aussi important que leurs performances puisqu'elle augmente les chances d'adoption de ces systèmes par les praticiens [24][25]. D'une autre, elle peut être exploitée pour extraire de la connaissance en parallèle de la tâche d'aide à la décision. On peut effectuer ainsi une hiérarchie des attributs en fonction de leur pouvoir prédictif. Cela permet un objectif secondaire qui est la réduction du nombre de tests médicaux à effectuer et de gagner en rapidité. Ainsi, à partir des quinze attributs de départ, Palaniappan et al [23] montrent que quatre suffisent à obtenir 99,61% de chance de déceler les patients souffrant d'une maladie cardiaque. Parmi les travaux s'intéressant en particulier aux maladies cardiaques, on retrouve souvent l'utilisation d'algorithmes empruntés à l'apprentissage profond et aux techniques d'ensemble

(*Bagging*,*Boosting*, *Random Forest*) [26][27].

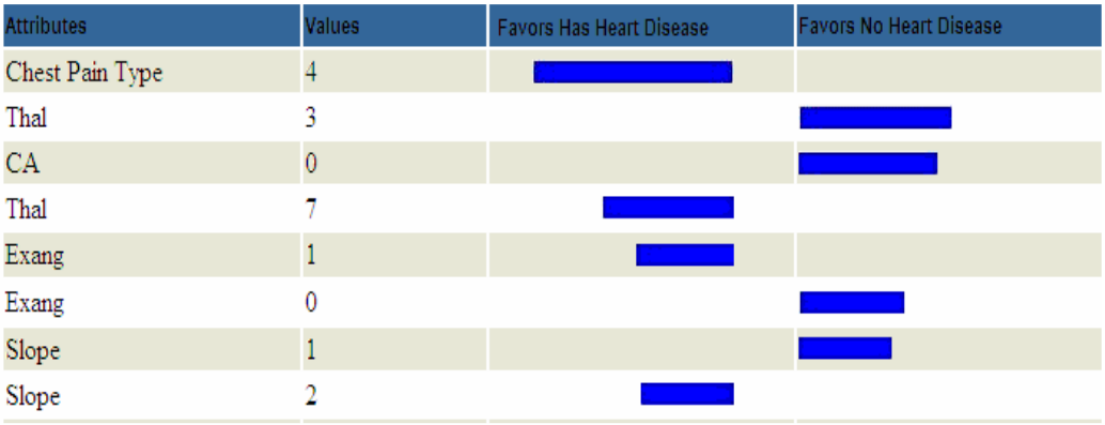


FIGURE 2.5 – Dans ce classifieur, *Thal* = 7 est favorable à la présente de maladie et *Thal* = 3 est défavorable à la présente de maladie. Aussi, *Thal* = 7 est plus impactant dans son verdict favorable que *Exang* = 1 ou *Slope* = 2.

Les motivations derrière le développement d'un outil de prédiction générique sont de gagner en efficacité que ce soit en terme de ressources ou de temps. Ainsi, au lieu de devoir développer un ou plusieurs modèles prédictifs par pathologie et multiplier les étapes d'entraînement, de paramétrage etc., on crée un seul et unique modèle sur lequel se concentrer. Cette approche offre aussi l'avantage de pouvoir potentiellement découvrir de nouvelles relations entre des symptômes grâce à la prise en compte d'attributs qui seraient habituellement omis dans un outil spécialisé. Le système proposé par Lipton et al [28] se place typiquement dans ce cadre puisqu'il utilise des *DME*s de patients étant passé par les unités de soins intensifs. Ce service, comme celui des urgences, présente une large variété de profils de patients différents. L'algorithme employé dans ce papier est intéressant car il soulève une nouvelle fois la valeur en terme d'informations qu'apportent les *DME*s. C'est un *Long Short-Term Memory* (*LSTM*), une variante de réseau de neurones dit récurrents. Ces derniers ont la propriété de garder en mémoire et de prendre en compte un certain nombre de cas précédents celui nouvellement fournis dans leur prédiction. Cette capacité se révèle particulièrement intéressante puisqu'elle permet d'exploiter l'historique médicale fourni par le *DME* du patient, permettant une meilleure compréhension de la situation du patient. La prédiction du système est multi-étiquettes afin de prendre en compte que les différentes pathologies ne sont pas mutuellement exclusives.

La présentation du concept de réseaux de neurones récurrents nous permet d'aborder une autre application de l'apprentissage automatique qui va être la prédiction du pronostic du patient. Pour des conditions comme l'insuffisance cardiaque, les patients se révèlent en effet particulièrement fragiles après une hospitalisation. Cet état de fait additionné à un arrêt prématuré de la prise en charge du malade peut avoir de graves conséquences, menant à une réadmission. Afin d'éviter d'inutiles complications pour la santé ainsi que la génération de frais supplémentaires pour les deux partis (patient et professionnel de la santé), il faut donc être capable de déterminer les risques liés à la sortie d'un patient du domaine hospitalier. Pour déterminer cela, il faut prendre en compte l'historique du malade puisqu'il peut y exister des facteurs de risques que l'état actuel du patient ne révèle pas. Parmi les outils implémentant ce type d'approche, on retrouve *Deepcare* de Pham et al. [29] et *Doctor AI* de Choi et al. [30]. Travaillant sur un *LSTM*, Pham et al. font l'observation qu'un des challenges dans la prédiction de l'évolution de maladies

est l'irrégularité d'un point de vue temporel. Les admissions d'un patient ne suivent pas de règles particulières et se révèlent porteuses de méta-informations (Ex : Si une admission n'était pas prévue, alors elle dénotait d'une urgence). Aussi, chaque maladie possède son propre rythme et sa propre manière d'évoluer. La particularité de *Deepcare* tient dans la possibilité de paramétrages temporels permettant de calquer au mieux la dynamique de chaque maladie. La phase d'apprentissage s'est effectuée sur des données d'un hôpital australien, privées des cas possédant des informations manquantes ou moins de deux admissions. Pham et al. ont réussi à obtenir, respectivement, une précision de 79% et de 74% dans la prédiction de réadmission non prévue pour des patients diabétiques (dans les douze mois) et pour des patients souffrant de troubles mentaux (dans les trois mois). *Doctor AI*, quant à lui, se propose de prédire à la fois les diagnostics de la visite suivante mais également les traitements correspondant aux symptômes prédits. Choi et al., utilisant un réseau de neurones récurrents, observe que *Doctor AI* voit augmenter ses performances proportionnellement au nombre de cas observés pour un même patient. De plus, la quantité d'admissions se révèle être porteuse d'informations. Une historique bien fournie laisse présumer d'une certaine fragilité de la santé du patient contrairement à une historique plus clairsemée. Les résultats affichent une sensibilité à 79,5% démontrant la capacité du modèle à prédire de manière satisfaisante les futures maladies.

La liste des travaux se penchant sur l'apprentissage automatique dans le milieu médical restant encore large, nous allons conclure cette partie en présentant un certain nombre de travaux que nous avons jugés intéressants et reflétant de la diversité des problèmes abordés. Miotto et al. [31] ainsi que Thodoroff et al. [32] s'intéressent au procédé appelé *Feature Learning*. L'objectif du *Feature Learning* est la sélection de critères de manière automatique et la découverte de nouvelles connaissances découlant de ces derniers (Dépendance, Corrélation etc.). Cela évite également la présence de biais potentiels entraînés par une sélection manuelle des critères. Pour les deux travaux, le but est l'obtention de modèles plus robustes pour améliorer la performance des prédictions de, respectivement, l'apparition de nouvelles maladies et de crises d'épilepsie. À noter que Thodoroff travaille sur une combinaison intéressante d'imagerie et de séries temporelles (représentations spectrales et spatiales d'électroencéphalogrammes au cours du temps). Pour ce qui est des traitements, différents problèmes sont abordés comme celui de pouvoir déterminer correctement les dosages des substances utilisées [33] et les interactions entre médicaments lors des essais cliniques [34]. Ces deux aspects représentent des enjeux très importants que ce soit d'un point de vue financier ou de qualité des soins. On retrouve également l'utilisation de l'apprentissage profond dans la découverte de nouveaux phénotypes pour construire de meilleures représentations des maladies [35, 36]. Jusque-là, les applications présentées se destinaient à l'usage des professionnels de la santé mais avec l'arrivée du paradigme de la médecine personnalisé, les patients se révèlent également demandeur d'outils permettant de se pencher sur leur santé. Dans le cadre de l'arythmie, Oresko et al. [37] se sont penchés sur le développement d'un système sur smartphone permettant au patient d'analyser les données de ses électrocardiogrammes et le prévenir en cas d'anomalie dans son rythme.

2.4 Systèmes multi-agents et monde médical

Face à la grande popularité de l'apprentissage automatique, l'utilisation des systèmes multi-agents pour répondre à des problématiques médicales peut sembler relativement marginale. Il n'en est rien puisqu'une grande quantité de travaux s'intéresse aux propriétés de ces systèmes répondant à de nombreux besoins des systèmes informatiques liés à la

santé. Le monde médical est bien connu pour sa complexité et un environnement comme celui des hôpitaux est, tous les jours, le théâtre d'un nombre incalculable d'actions et d'interactions de la part des acteurs (Patient, Médecin, Administration, ...) le composant. Un autre détail important est que, malgré les protocoles et les procédures mis en place dans le milieu de la santé pour standardiser les processus, les facteurs inconnus et humains ajoutent à la complexité et l'imprédictibilité de l'état du système. Les agents permettent, en représentant les différents acteurs des environnements simulés, de pouvoir représenter efficacement et de visualiser l'état du système. De plus, leur autonomie permet d'observer l'évolution de l'état du système sans avoir besoin d'intervenir ni d'avoir de connaissance sur ce que ce dernier devrait être. Les objectifs des systèmes multi-agents se révèlent particulièrement variés, ce qui rend difficile la définition du type de l'application étudiée. Pour cet état de l'art, afin d'offrir une vision de la variété des systèmes multi-agents, nous allons ici reprendre, en partie, la classification proposée par Isern et Moreno dans leur *survey* [38]. Le choix de cette classification se base sur la pertinence et la robustesse des définitions fournies par ses auteurs. Pour ce faire, ils définissent deux grands critères complémentaires pour caractériser un système multi-agents. Le premier critère s'intéresse à la portée du système multi-agents. On définit trois sous-catégories : l'approche centrée patient, l'approche centrée personnel et l'approche centrée organisation. La première a pour objectif de fournir des outils et des services directement aux patients, avec souvent une emphase sur la personnalisation des soins leur étant apportés. La seconde concerne les systèmes soutenant le personnel dans l'exécution de leurs tâches quotidiennes, prenant souvent la forme d'assistant personnel. La troisième vise à fournir les outils nécessaires à une organisation pour simuler et améliorer son fonctionnement de manière générale. Ce second critère comporte à son tour sept sous-catégories définissant les aspirations d'un système multi-agents. Contrairement aux catégories du premier critère, ces dernières ne se révèlent pas mutuellement exclusives et un système peut remplir des fonctions appartenant à plusieurs d'entre elles.

Les systèmes de gestion des données, comme leur nom l'indique, visent à faciliter toutes les actions relatives aux données, comme le prétraitement et la récupération. De par le caractère virtuel et distribué des *DME*s, leurs mises en place dans un contexte multi-agents se révèlent parfaitement en accord avec cette classification. Un exemple d'un tel système est le système *MAID* (*Multi-Agent System for Integration of Data*) [39] développé par Cruz-Correia et al. Ce dernier consiste en un certain nombre d'agents qui ont pour objectif d'intégrer les données fournies par les différents systèmes hérités de l'hôpital et de les agréger dans un *DME*. Les dossiers sont stockés dans un dépôt central auquel peut accéder les utilisateurs. Ces différents agents s'assurent en permanence de l'actualité des informations mises à disposition des utilisateurs. Si une information demandée par un utilisateur n'est pas présente sur le dépôt, alors les agents s'organisent les uns les autres afin d'accélérer l'arrivée de cette dernière jusqu'à l'utilisateur. Dans un autre contexte, le système *CHIS* (*Contexte-aware Hospital Information System*) [40] offre quant à lui un système omniprésent d'informations à l'ensemble de l'hôpital. Il est ainsi possible pour les praticiens et infirmiers de pouvoir avoir accès à des services basiques comme l'accès et le partage d'informations entre membres du personnel. Mais il est également possible, selon le niveau d'autorisation de l'individu, d'observer la localisation du matériel et du personnel par l'intermédiaire d'agents localisateurs.

Les systèmes tournés vers la simulation ont pour but de fournir des outils simulant le fonctionnement d'un environnement. À partir de ces simulations, il est alors possible d'observer des failles ou d'isoler les causes de dysfonctionnement dans le système. L'objec-

tif est en quelque sorte de diagnostiquer la simulation afin d'améliorer la situation réelle qu'elle émule. Silverman et al. [41] se sont penchés sur cette approche pour étudier la santé mentale et le bien-être de la population de Philadelphie. Les agents de la simulation sont définis par des motivations et un état (comportant l'état psychologique, physiologique et socio-économique) évoluant au cours du temps, qui influent tout deux sur les actions effectuées. L'objectif est de permettre une meilleure allocation des ressources pour un meilleur suivi des patients et une minimisation des risques d'hospitalisation. Les résultats prennent la forme de nouvelles consignes et de pratiques de travail pour atteindre les objectifs fixés. La simulation de la propagation d'un virus au sein d'un service des urgences, quant à elle, permet d'étudier différents scénario de manière détaillée et de pouvoir mettre en place des protocoles appropriés de lutte contre l'infection. Laskowski et al. [42] soulignent le fait que le paradigme multi-agents apporte des éléments nouveaux dans la compréhension du fonctionnement de la propagation avec notamment des faits contre-intuitifs qui n'auraient pas été mis en lumière en temps normal.

Les systèmes de surveillance et d'alarmes se placent dans un contexte où les patients doivent être observés de manière permanente comme pour les services gériatriques ou les insuffisants cardiaques. On retrouve plutôt ces systèmes intégrés à des infrastructures hospitalières, où le personnel ne peut consacrer son temps à l'observation d'un seul patient. Ces systèmes se basent donc sur l'usage de capteurs, qui enregistrent en continu les informations du patient et repèrent les éventuelles anomalies. Kafali et al. propose un système de ce type avec leur système $COMMODITY_{12}$ [43], qui est un système dédié au personnel de santé à l'intention des patients diabétiques. $COMMODITY_{12}$ s'appuie sur les données physiologiques du patient qui sont récoltées par l'intermédiaire de différents appareils portables. Ces informations sont ensuite soumises au jugement d'agents experts, qui les analysent en se basant sur des connaissances médicales. Ce sont ces agents qui émettent les recommandations et les avertissements au patient afin de l'informer de sa situation. Ils possèdent également la capacité de communiquer avec des professionnels de la santé afin d'obtenir de l'aide si nécessaire. Les systèmes de plateformes de soin, recoupent une partie de ses objectifs avec les systèmes de surveillance et d'alarmes dans la mesure où ils utilisent souvent des capteurs pour effectuer des diagnostics sur l'état de santé du patient. Cependant la différence tient dans le fait que ces systèmes se placent dans un contexte très personnel puisqu'ils s'intégrent au quotidien et/ou au domicile du patient. Ainsi, les agents promeuvent la télémédecine en permettant une surveillance à distance du patient par le praticien ou en permettant la mise en place de visite à distance par les médecins. Su et al. [44] appliquent ce concept pour l'observation du développement fœtal à distance. Cette approche permet à la femme enceinte de pouvoir obtenir des informations sur fœtus quand elle le désire sans avoir besoin de se déplacer (donc de se fatiguer). C'est également vrai pour l'obstétricien qui peut garder un œil sur l'ensemble de ses patientes avec facilité.

Les systèmes de planification usent des agents afin de définir et de faciliter l'émergence d'une organisation optimisée des plannings et/ou des ressources. Les critères de cette optimisation peuvent être très variés s'appuyant sur des considérations budgétaires, des préférences du personnel, la localisation du matériel, etc. *CAMAP* (*Context-Aware Multi-Agent Planning*) développé par Ferrando et al. [45] est un système prenant avantage de l'autonomie et de l'intelligence des agents le composant. Lorsqu'un agent propose un plan, il exprime clairement ses opinions pour les soumettre aux jugements de ses pairs. Il débute une négociation. Chacun des agents impliqués dans cette dernière est ensuite libre de présenter des arguments ou des contre-arguments contre la ligne de conduite proposée. La capacité la plus importante des agents présentés dans ces travaux est leur capacité

à abandonner et modifier leurs comportements en prenant en compte le contexte de la négociation et les arguments des autres agents.

Les systèmes d'aide à la décision peuvent prendre plusieurs formes mais ont tous pour objectif d'apporter les moyens de soutenir le personnel médical dans les différents processus décisionnels. Certains systèmes implémentent des systèmes dont l'intérêt va être de fournir des éléments comme des dossiers similaires, des antécédents,... du patient pour éclairer le choix du praticien. D'autres vont s'appuyer sur des outils comme des techniques d'apprentissage automatique pour fournir des réponses et accélérer la prise de décision. La définition de la marche à suivre en fonction de la place dans le *workflow* du patient et du médecin peut aussi être sujet à un outil d'aide à la décision. Singh et al. [46] propose un système où l'utilisateur, par l'intermédiaire de son agent, charge des agents brokers de récupérer des dossiers similaires à celui qu'il étudie dans des bases de connaissances. Ces dernières sont constituées d'expériences passées de différents praticiens. Les brokers formatent ces dossiers et les agrègent en une grande réponse pour l'utilisateur. *HealthAgents* [47] est quant à lui un système visant à diagnostiquer les nouveaux cas de tumeurs. Pour ce faire, l'agent de l'utilisateur communique avec des agents classifieurs chacun localisé dans des centres médicaux différents. Les données n'étant pas identiques dans ces centres, ces agents classifieurs disposent donc d'un point de vue unique sur le cas qu'on lui propose. Le praticien n'a plus qu'à interpréter les résultats reçus et à évaluer la confiance associée à chacun d'eux.

Finalement, le dernier aspect se concentre sur l'implémentation de systèmes sécurisés. L'évolution dans un paradigme distribué oblige en effet la présence d'un certain nombre de contraintes et ce tout particulièrement dans le monde médical. Ainsi il faut qu'un système soit toujours disponible puisqu'une panne peut se révéler lourde de conséquences pour le personnel et les patients. D'un autre côté, la sécurité et le respect de la vie privée se doivent d'être une priorité au vu de la sensibilité des informations qui sont manipulées. Le paradigme multi-agents apporte bien entendu des contraintes spécifiques comme la présence d'agents malveillants ou la mise en place de protocoles sécurisés pour les communications inter-agents. Le grand problème que soulèvent Isern et Moreno [38] est le nombre relativement faible de travaux se concentrant sur ces problématiques. Un grand nombre de systèmes sont proposés sans garantie d'un point de vue de la sécurité soit pour cause de contrainte technique ou par omission. Cet état de fait entraîne l'obligation d'effectuer un travail a posteriori dans le but de rectifier ces lacunes.

Chapitre 3

Notre approche

Notre objectif est le développement d'une architecture multi-agents intégrant des modèles d'apprentissage automatique pour répondre aux besoins de la *CSIC*. L'intérêt de cette approche est d'offrir un système capable de s'adapter à un environnement évolutif comme celui des hôpitaux. Les agents suppléeront à des tâches du quotidien comme l'échange des informations entre les différents acteurs de l'environnement. Pour des tâches plus compliqués comme l'aide à la décision, l'intégration d'algorithmes d'apprentissage au sein des agents se pose comme un solution idéale permettant d'exploiter les données médicales des hôpitaux attendant d'être utilisées. Ces données contiennent des informations importantes sur le profil de chaque patient et peuvent permettre de personnaliser leur parcours de soin. Notre système permettrait aux acteurs du système hospitalier de prendre en compte cet aspect et de mieux s'adapter aux besoins des patients. Nous utiliserons parfois le terme écosystème pour décrire l'environnement de part sa complexité et le grand nombre d'acteurs le composant. Ce chapitre sera divisé de la manière suivante. Nous allons tout d'abord élaborer nos propos sur le projet de la *CSIC*. Nous y présenterons les membres le composant ainsi que son fonctionnement. Cette étape permettra d'obtenir une vision précise du projet afin de mieux discuter de nos choix concernant la direction de nos travaux. Nous étudierons ensuite les plans sur lesquels le projet pourrait bénéficier d'aide à la décision. Une fois isolés, nous analyserons les données à notre disposition et définiront des premiers modèles potentiels pour répondre à ces besoins. Finalement, nous élaborerons l'architecture multi-agents à laquelle viendront s'intégrer les modèles précédemment développés.

3.1 Organisation de la CSIC

Notre travail se place dans le contexte de la prévention de l'insuffisance cardiaque et, plus précisément, dans le cadre de la *CSIC* mise en place par le CHSF. Ce projet est une unité expérimentale ayant débuté ses activités en janvier 2017. Son objectif est d'évaluer l'impact d'un dispositif dédié à la détection de décompensation cardiaque sur le désengorgement du service des urgences et le taux d'hospitalisation des insuffisants cardiaques. Dû au caractère nouveau de la *CSIC*, ses membres ne lui sont pour l'instant pas entièrement dédiés et proviennent de services annexes comme la cardiologie. Le personnel est composé d'infirmiers et de médecins, remplissant chacun des rôles bien définis.

La procédure de consultation semi-urgente débute toujours par un appel téléphonique auprès de la *CSIC* afin d'être mis en relation avec l'infirmier de garde. L'origine de ces appels est variée et peut provenir du patient insuffisant cardiaque lui-même, d'un de ses

proches ou de son médecin traitant par exemple. Une fois la connexion établie entre les deux partis, l'infirmier va suivre un formulaire et demander les informations nécessaires à sa complétion. L'objectif de ce formulaire, créé par le personnel de la CSIC, va être de déterminer si une consultation est nécessaire pour stabiliser le malade. Ainsi, l'infirmier récolte dans un premier temps les informations personnelles du patient tel que son nom, son sexe, son âge et ainsi de suite. Cette étape d'identification terminée, il dispose ensuite d'un certain nombre de critères qui vont lui permettre d'évaluer l'état du patient. Ces critères sont des signes avant-coureurs de décompensation cardiaque dont on évalue la présence chez le malade (*True* si présent, *False* sinon).

Nos discussions avec le personnel de la CSIC nous ont permis de déterminer que tous les critères ne possèdent pas le même poids dans la détection de décompensation. La table 3.1 illustre cette observation. Un indice "sérieux" permet de pencher très fortement pour la nécessité d'une consultation. Un indice modéré ne peut déterminer à lui seul si le malade souffre de décompensation cardiaque et requiert la présence d'autres indices. Finalement un indice faible peut être un facteur s'ajoutant à un indice préexistant mais peut être un indicateur d'un tout autre problème qu'une décompensation. Cette hiérarchisation des symptômes reste cependant à moduler par le fait que le facteur humain rentre en compte lors du jugement de l'infirmier. La détresse de l'interlocuteur, par exemple, constitue un critère important dont la quantification est subjective.

Critère du formulaire téléphonique	**Indice de décompensation cardiaque**
Prise de 2-3kg en moins d'une semaine	Sérieux
Enflement des membres inférieurs	Modéré
Fatigue	Modéré
Activité physique limitée	Modéré
Difficulté respiratoire en position allongée	Modéré
Toux	Faible
Fièvre supérieure à 38,5°C	Faible
Palpitations	Faible
Sensation de malaise ou de vertige	Faible
Sensation de voile noir devant les yeux	Faible

TABLE 3.1 – Observation de la gravité d'un symptôme pour la détection d'une insuffisance cardiaque.

Lorsque l'infirmier possède d'assez d'éléments pour juger de l'admissibilité du patient au sein de la CSIC, trois cas de figures sont envisageables. L'état du patient ne correspond pas au profil attendu par l'infirmier (i.e. malade en début de décompensation). Son état peut ne pas impliquer le type de compétences du personnel de la CSIC, auquel cas il est redirigé vers un professionnel de la santé adapté à ses besoins. Aussi, il est possible que la décompensation soit trop avancé, dans ce cas, il est redirigé vers un service comme le SAMU pour une prise en charge de toute urgence. Seconde possibilité, le malade est jugé admissible dans le cadre d'une CSIC. L'infirmier lui propose alors une consultation dans les 36 heures et lui indique une série d'examens à effectuer dans le laboratoire de son choix. Les coordonnées du laboratoire sont enregistrées et l'appel est terminé. Enfin, dans le cas où l'infirmier n'arrive pas à déterminer si l'état du patient relève de la CSIC, il peut contacter le médecin de garde afin d'obtenir un avis complémentaire menant à un des dénouements précédents.

Champs	Indice sur stabilité
Ajustement thérapeutique	Défavorable
Suivi biologique prescrit	Neutre
Examens complementaires prescrits	Neutre
Informations données sur l'insuffisance	Neutre
Compte-rendu de sortie remis	Non pertinent
Retour vers médecin traitant	Neutre
Retour avec programmation HDJ	Défavorable
Retour avec consultation hospitalière	Défavorable
Education thérapeutique proposé	Défavorable
Passage aux urgences évité	Défavorable

TABLE 3.2 – Indices permettant d'évaluer la stabilité d'un patient à partir du bilan de sortie de ce dernier.

Pour la consultation, le patient arrive avec les résultats des examens demandés lors de l'appel et est pris en charge par l'infirmier de garde. Ce dernier dispose une nouvelle fois d'un formulaire, aux champs plus pointus que ceux du questionnaire précédent. Le but n'est pas ici de diagnostiquer précisément le mal du patient mais de définir si le patient a besoin de se faire hospitaliser ou non. Pour ce faire, on relève comme précédemment les informations nécessaires à l'identification du malade. Le médecin intervient ensuite pour ausculter le patient. Il relève toutes les données relatives aux antécédents du patient, des traitements en cours ainsi que des données physiologiques comme la pression artérielle ou le poids. Si le médecin n'est pas satisfait des observations effectuées, il lui est possible de demander des examens complémentaires. Sinon il peut décider de programmer une hospitalisation ou de renvoyer le patient à son domicile. Dans le premier cas, le patient n'est plus du ressort de la CSIC et le processus de suivi du patient s'arrête là. Pour un retour à domicile, un certain nombre de champs est disponible pour obtenir une vision plus précise des conditions de ce retour. Chacun d'entre eux permet de se faire une idée sur la stabilité du patient au moment de la visite. On retrouve ainsi les champs de la table 3.2.

Le champ "Compte-rendu de sortie" est classé comme étant "Non pertinent" dans l'évaluation de la stabilité car il dénote simplement du fait qu'une fiche ait été fourni au patient après la consultation. Les champs jugés comme "Neutre" ne permettent pas d'obtenir d'informations sur l'état du patient puisqu'ils dépendent du contexte dans lequel s'est effectué la consultation. N'ayant pas accès à cette information par le moyen des formulaires, ces champs sont inutilisables dans la détermination de l'état du patient. Enfin les indices "Défavorables" sont des marqueurs d'instabilité et plus ils sont nombreux, plus on peut être certain que le patient était dans un état critique. Il est à noter que les indices associés à chaque champ dans les tables 3.1 et 3.2 sont issus de notre analyse. Ils nous serviront plus tard lors de la phase d'analyse des données et lors de la phase d'apprentissage. Cependant, d'un point de vue médical, l'ensemble de ces champs est essentiel et apporte leur lot de connaissances, exploitables par le personnel médical.

Pour une représentation visuelle du processus de la CSIC, un organigramme créé par un ancien doctorant au laboratoire IBISC est mis à disposition dans l'annexe A. De même, les formulaires téléphoniques et les formulaires de consultation semi-urgente peuvent également y être trouvés, respectivement dans l'annexe B et C. Ces derniers seront détaillés plus amplement dans la partie suivante, qui se concentre sur leur analyse et celle des données récoltées par la CSIC.

3.2 Analyse des formulaires et des données

Notre premier objectif consistait à développer un système d'aide à la décision pour le personnel de la CSIC. Pour ce faire, nous avons décidé de nous tourner vers l'emploi de techniques d'apprentissage automatique. Cette approche présentait comme avantage de pouvoir se passer, en partie, des connaissances du personnel médical. La retranscription de ces dernières par des professionnels de la santé représente en effet un lourd investissement en terme de temps, temps dont ne disposent pas forcement les experts médicaux. Exploitant les informations fournies par les formulaires du personnel, l'idée était de construire des modèles pouvant s'insérer naturellement dans la dynamique de travail de la CSIC. Ce choix fut motivé par une volonté de faciliter l'adoption du système par le corps médical. En effet, le changement des habitudes de travail demeure, aujourd'hui encore, une des raisons expliquant la faible implantation de l'apprentissage automatique au sein du milieu médical.

Lors de l'extraction des dossiers associés à chaque formulaire, nous fîmes le constat qu'il n'existait qu'un nombre très restreint d'exemples. On ne dénombre que 42 et 34 dossiers respectivement pour le formulaire téléphonique et pour le formulaire de consultation semi-urgente. Les principales causes derrière cette rareté des données sont tout d'abord la relative jeunesse et le caractère expérimentale du projet. Aussi la communication entourant l'existence du projet en est encore à ses débuts. Un des premiers problèmes émergeant de la quantité de données fut la non-représentativité de la population visée par la *CSIC*. Nous entendons par là que l'échantillon que nous possédons présente le risque de véhiculer une réalité biaisée. Ainsi, les modèles entraînés n'apprendront pas les concepts généraux pour atteindre l'objectif du formulaire mais les concepts spécifiques à l'éventuelle tranche de population des dossiers. Aussi, il est impossible de pouvoir créer un ensemble d'entraînement et de validation puisque le nombre d'exemples se révèle déjà trop restreint. Pour compenser ces points, l'idée d'exploiter les bases de données de l'hôpital afin d'enrichir la base d'apprentissage avec des cas similaires fut envisagée pour les questionnaires de consultation semi-urgente. Ce traitement ne pouvait cependant être transposé pour les formulaires téléphoniques car il est nécessaire que le patient soit sur place pour ajouter son cas aux bases de données du CHSF. Nous reviendrons sur certains de ces aspects lors de l'analyse en détail des informations recueillies pour chaque type de formulaire.

Dans un premier temps, nous allons nous concentrer sur le formulaire téléphonique. Lors d'une première et rapide analyse, nous avons pu observer un certain nombre de défauts dans les données. Comme c'est souvent le cas dans le milieu médical, certains champs n'étaient pas renseignés et entraînaient la présence de valeurs manquantes. Concernant ces dernières, il est important de faire la différence entre les valeurs manquantes aléatoires et les valeurs manquantes non aléatoires. La première catégorie consiste en des absences qui ne sont pas explicables par d'autres variables observables ou par un phénomène extérieur. On retrouve ainsi des exemples dont des champs comme le sexe ou les antécédents ne sont pas remplis. L'absence de ces valeurs peut s'expliquer très probablement par un oubli de

la part du praticien et peut donc être considérée comme un événement dû au hasard. La seconde décrit des variables dont l'absence de valeur se trouve être liée à la raison pour laquelle elle est manquante. Pour clarifier ce point, un certain nombre des critères de la table 3.1 manquait dans certains exemples alors que d'autres possédaient la valeur *True*. On peut donc en déduire que le praticien avait jugé nécessaire de ne renseigner un champ que si la réponse était positive. Cette catégorie d'absence est la plus facile à "réparer" puisque l'absence même de remplissage du champ nous indique la valeur qu'il aurait dû prendre. Ainsi, pour les champs de la table 3.1, nous décidâmes de remplacer toutes valeurs manquantes par *False*. Plus gênantes, les valeurs manquantes aléatoires n'offrent pas d'indices quant à la valeur normalement attendue. L'analyse a révélé deux champs concernés par ce cas de figure : l'âge et le verdict d'admissibilité du patient au sein de la CSIC. Le problème se révèle plus complexe que le postulat de base puisque le premier champ est une valeur prédictive alors que le troisième est la valeur à prédire. Pour ne pas biaiser nos futurs résultats, il fût décidé d'éliminer les trois cas dont on ne connaissait pas le verdict final. Concernant l'âge, deux approches furent employées, la suppression des exemples concernés et le remplacement par l'âge moyen.

La table 3.4 résume l'ensemble des statistiques effectuées sur les exemples de formulaires téléphoniques. On peut tout d'abord attester d'un certain déséquilibre de classe pour les champs de sortie. En effet, près de 70% des patients ont été jugés compatibles avec les exigences de la *CSIC* en terme de consultation tandis que les 30% restant se sont répartis parmi les deux classes restantes. Ce point soulève une vigilance accrue quant aux résultats qui seront obtenus par les modèles d'apprentissage car même un modèle stupide ne prédisant que des consultations pourra obtenir 70% de précision.

Champ	**Pourcentage**
Homme : Femme :	51% 49%
0-20 ans : 20-40 ans : 40-60 ans : 60-80 ans : 80-100 ans :	00% 02% 08% 59% 31%
Connu du CHSF	87%
Appel d'un patient en décompensation	72%
Appel d'un patient posant un autre problème cardiaque	13%
Appel d'un patient pour un autre motif	15%
Prise de 2-3kg en moins d'une semaine	21%

Champ	**Pourcentage**
Enflement des membres inférieurs	28%
Fatigue	49%
Activité physique limitée	38%
Difficulté respiratoire en position allongée	54%
Toux	44%
Fièvre supérieure à 38,5°C	41%
Palpitations	38%
Sensation de malaise ou de vertige	33%
Sensation de voile noir devant les yeux	36%

TABLE 3.3 – Statistiques sur les valeurs observées pour chaque champ du formulaire téléphonique.

Patient admis en consultation	67%
Patient orienté vers un service d'urgence	13%
Patient orienté vers un autre service	20%

TABLE 3.4 – Statistiques sur les occurrences de chaque classe.

Pour ce qui est de l'apprentissage automatique appliqué aux formulaires téléphoniques, quatre algorithmes différents furent sélectionnés à partir de deux critères de sélection. Pour le premier critère qui est la traçabilité de la décision notre choix s'est porté sur le modèle bayésien naïf et l'arbre de décision simple. En effet, les praticiens optent souvent pour des modèles légèrement moins performants mais dont il est possible d'expliquer le résultat. Le second critère se révèle tout simplement être les performances des modèles. Les deux derniers modèles sont les forêts d'arbres et les *SVM*s. Nous avons utilisé la bibliothèque Scikit-Learn [48] puisque l'ensemble des techniques choisies sont disponibles par l'intermédiaire de cette bibliothèque. L'implémentation de l'arbre de décision sous Scikit-Learn ne prenant en entrées que des valeurs numériques, toutes les valeurs booléennes de notre base d'apprentissage a subi des transformations de telle sorte que *False* soit égal à 0 et *True* soit égal à 1. De même, les champs "Homme" et "Femme" furent remplacés par "Homme ?" afin de réduire le nombre de champs. Pour ce qui est des champs à classes multiples comme les tranches d'âge, les différents types d'appels ou le verdict final, un autre prétraitement fut appliqué. En effet, en remplaçant les n différentes classes par des valeurs allant de 0 à n, on crée une hiérarchie des classes aux yeux des modèles. La solution pour éviter cela réside en un procédé appelé le *one hot encoding*. Prenons trois classes a, b et c, on représente la valeur d'un champ par un vecteur de taille 3. Ainsi les vecteurs [1,0,0], [0,1,0] et [0,0,1] indiquent respectivement l'appartenance aux classes a, b et c.

Puisque la sélection des paramètres pour des modèles comme les *SVM*s ou les forêts d'arbres de décision est une étape essentielle dans l'obtention des meilleures performances, nous avons employé des *grid*s. Ces dernières nous permettent de spécifier des valeurs possibles pour chaque paramètre et d'effectuer automatiquement les tests sur l'ensemble des combinaisons possibles. Ainsi pour les forêts d'arbres de décision, nous avons pu définir trois critères différents. Le premier était le nombre total d'arbres composant la forêt. Pour ce faire, nous avons sélectionné, arbitrairement, des quantités plus ou moins grandes afin d'observer l'évolution des performances en fonction de la taille de la forêt. Il est possible que d'autres valeurs auraient pu apporter de meilleurs résultats que ceux exposés plus tard dans cette partie. Cependant, la recherche est une tâche se révélant coûteuse en terme de temps et dont l'effet bénéfique sur les résultats n'est pas assuré. Nous avons par conséquent décidé de nous en tenir aux valeurs du tableau ci-dessous.

Nombre d'arbres dans la forêt :	10	50	100	250	500	1000

TABLE 3.5 – Ensemble des valeurs testées pour le paramètre "Nombre d'arbres".

Le second paramètre s'intéresse aux échantillons utilisés par la construction des différents arbres de la forêt. Deux options existent : avec ou sans *Bootstrap*. Le *Bootstrap* désigne un procédé statistique qui permet à partir d'un ensemble de données à créer un grand nombre de nouveaux ensembles dérivés de cet ensemble de base. Ainsi, on peut par

exemple sélectionner aléatoirement un certain nombre d'éléments pour créer un nouvel ensemble, tout en permettant la sélection d'un même élément plusieurs fois. Chacun des arbres apprend donc à partir de cas légèrement différents ce qui permet une meilleure capacité de généralisation pour la forêt. Enfin le dernier concerne la manière dont un arbre va évaluer la pertinence d'un branchement possible en terme de gain d'informations. Deux critères furent tester, à savoir le coefficient de Gini et l'entropie. Le coefficient de Gini est une mesure de l'égalité interclasses qui peut varier entre 0 et 1. Plus l'on tend vers 0 alors plus les classes se révèlent équilibrées en terme de présence dans l'ensemble de données. Inversement, plus l'on tend vers 1, plus on se rapproche d'un déséquilibre dans la représentation des différentes classes. L'entropie essaie quant à elle de maximiser le gain d'informations en déterminant les nœuds permettant la meilleure discrimination. Pour ce qui est des SVMs, les critères sélectionnés furent le type de noyau utilisé pour la séparation des différentes classes ainsi que la tolérance relative au critère d'arrêt de recherche du séparateur. Les quatre noyaux utilisés furent le noyau linéaire, le noyau polynomial, le noyau gaussien et le noyau sigmoïde.

La taille de notre ensemble de données ne nous permettant pas sa séparation en un ensemble d'apprentissage et en un ensemble de validation, nous nous sommes tourné vers la validation croisée (ici 10-*Folds*) afin d'améliorer le processus d'apprentissage de nos modèles. Le principe est de diviser dans un premier temps le jeu de données en K sous-ensembles différents (K-*Folds*). Ensuite notre modèle va apprendre sur chaque combinaison possible de K-1 sous-ensembles différents tandis que la validation s'effectuera sur le sous-ensemble restant. Il ne reste plus qu'à effectuer une moyenne des résultats obtenus par chacune des K itérations pour estimer les performances globales du modèle. Cette méthode présente l'avantage de n'exclure aucun exemple et donc de ne pas perdre de l'information, ce qui est particulièrement important compte tenu des caractéristiques de nos données. Nous avons calculé trois mesures différentes afin d'évaluer les performances de nos modèles. L'*Accuracy* détermine la justesse des valeurs qui ont été prédites par rapport aux véritables valeurs attendues. Dans un paradigme bi-classe dont les valeurs possibles sont "Positif" et "Négatif", on calcule l'*Accuracy* de la manière suivante

$$ACCURACY = \frac{TP + TN}{P + N}$$

où TP, TN les ensembles respectifs de "Positif" et "Négatif" correctement prédits et P, N les ensembles respectifs de "Positif" et "Négatif" prédits. Pour nos données, on se base sur les ensembles de classes prédites correctement et les ensembles totaux de classes prédites. La mesure suivante est la *Precision*. Elle calcule, pour chaque classe, quelle est le pourcentage de valeurs correctement prédites sur l'ensemble prédit pour cette classe. On la calcule de la manière suivante

$$PRECISION = \frac{TP}{TP + FP}$$

où FP l'ensemble de "Positif" appartenant en réalité à une autre classe. Enfin, la dernière mesure est le *Recall* qui calcule le pourcentage de valeurs correctement prédites sur le véritable ensemble des valeurs appartenant à cette classe. Le *Recall* s'exprime de la manière suivante

$$RECALL = \frac{TP}{TP + FN}$$

où FN l'ensemble de "Positif" qui ont été prédites dans une classe différente de "Positif". Pour rappel, nous avons testé chacun des algorithmes précédemment cités avec quatre

variations du jeu de données original, dont les résultats sont décrits dans les tableaux ci-dessous.

	Accuracy	**Precision**	**Recall**
Arbre de décision :	0.47	0.69	0.45
Forêt d'arbres :	0.63	0.68	0.63
Bayésien naïf :	0.71	0.65	0.71
SVM :	0.67	0.55	0.67

TABLE 3.6 – Performances des meilleurs modèles avec les données d'origine, hors suppression des exemples à "Âge" manquants. Stratégie : 10-*Fold*

	Accuracy	**Precision**	**Recall**
Arbre de décision :	0.43	0.61	0.44
Forêt d'arbres :	0.53	0.60	0.56
Bayésien naïf :	0.67	0.58	0.67
SVM :	0.67	0.55	0.67

TABLE 3.7 – Performances des meilleurs modèles avec les données où les âges manquants ont été remplacés par l'âge moyen de la population. Stratégie : 10-*Fold*

	Accuracy	**Precision**	**Recall**
Arbre de décision :	0.47	0.64	0.39
Forêt d'arbres :	0.63	0.70	0.61
Bayésien naïf :	0.68	0.68	0.68
SVM :	0.68	0.75	0.68

TABLE 3.8 – Performances des meilleurs modèles avec les données avec suppression des exemples à "Âge" manquants et la transformation de l'âge en tranches d'âge. Stratégie : 10-*Fold*

	Accuracy	**Precision**	**Recall**
Arbre de décision :	0.49	0.66	0.43
Forêt d'arbres :	0.57	0.63	0.53
Bayésien naïf :	0.61	0.56	0.61
SVM :	0.66	0.53	0.66

TABLE 3.9 – Performances des meilleurs modèles avec les données où les âges manquants ont été remplacés par l'âge moyen de la population et la transformation de l'âge en tranches d'âge. Stratégie : 10-*Fold*

On remarque dans un premier temps, et ce pour l'ensemble des jeux de données, une très faible *Accuracy* atteignant au mieux 71%. Ces résultats sont décevants car nous rappelons que 67% des patients sont admis en consultation semi-urgente. Ainsi le meilleur résultat obtenu ne dépasse que de 4 points le score qu'obtiendrait un classifieur naïf qui ne prédirait que "Consultation semi-urgente". Cependant, il n'est pas étonnant d'obtenir de telles performances compte tenu du faible volume de données, du déséquilibre de classes

observé et de la non-représentativité de notre échantillon. Parmi les observations intéressantes sur les résultats, on remarque que l'arbre de décision est systématiquement le moins bon des quatre modèles proposés. Cela peut s'expliquer par le fait qu'il est beaucoup plus sensible aux problèmes cités précédemment ce qui influe forcément sur son niveau de performance. Aussi, les modèles travaillant avec les données d'origine semblent obtenir de meilleurs résultats que les modèles travaillant avec des données ayant subi un prétraitement. Pour être plus précis, ceux où les exemples à âge manquant étaient supprimés se révèlent plus performants que ceux usant d'une valeur de remplacement. L'utilisation de la moyenne d'âge de la population comme substitut semble donc avoir une influence négative sur les performances de nos modèles. D'une manière générale, on peut penser que la perte potentielle d'informations imputable aux différents traitements puisse être trop importante compte tenu du faible nombre d'exemples pour espérer améliorer les capacités de généralisation des modèles. La SVM et le classifieur bayésien naïf semblent être les deux modèles les plus performants. Si ce n'est le fait que le classifieur bayésien de la table 3.2 semble être le meilleur modèle (hors *Precision*), il reste cependant difficile de déterminer lequel des deux algorithmes se révèlent le plus pertinent dans l'absolu pour répondre aux problématiques de la consultation semi-urgente.

Afin d'améliorer les performances des modèles précédents, nous avons tenté de changer de stratégie de validation croisée afin de nous tourner vers du *LeaveOneOut*. Il s'agit d'exclure un unique exemple à chaque étape de validation afin de s'en servir d'ensemble de validation. Nous espérions ainsi améliorer les capacités de généralisation de nos modèles en comparaison avec la stratégie 10-*Fold*. Nous obtinrent une amélioration de nos performances. Ces dernières sont retranscrites dans les tables 3.2, 3.2, 3.2 et 3.2. Hormis l'arbre de décision simple, les autres modèles semblent en effet montrer de meilleures scores peu importe la situation ou le critère observé. Afin de vérifier ces affirmations et observer l'impact du déséquilibre de classe sur les résultats de nos modèles, nous avons décidé d'intégrer des matrices de confusion en plus de nos différentes mesures. Ces matrices nous permettent d'observer les quantités d'exemples correctement et incorrectement prédits pour en déduire les capacités de classification des modèles. Nos matrices seront à interpréter de la manière décrite dans la table 3.2. On peut observer, de manière générale, chez nos modèles une bonne capacité à prédire la classe majoritaire. Les deux autres classes se révèlent cependant un challenge. Leurs pourcentages de prédictions correctes n'atteignent même pas 50% et ce dans l'ensemble des matrices présentées dans les tables 3.2 à 3.2. Ce constat est particulièrement vrai pour la classe la moins représentée, la Redirection vers les urgences. Les différents *SVM*s en arrivent même à prédire uniquement la classe majoritaire à l'exception de celui de la table 3.2, qui est le seul à utiliser un noyau linéaire. Les scores avoisinant les 70% sont donc entraînés par un apprentissage biaisé vers la prédiction d'une Consultation nécessaire. Nous avons donc décidé d'allier le *LeaveOneOut* avec un système de pondération des différentes classes dans l'optique de contre-balancer ce déséquilibre.

Prédiction/Vraie valeur	Vers la consultation	vers un autre service	Vers les urgences
Vers la consultation :	**Correct**	Incorrect	Incorrect
vers un autre service :	Incorrect	**Correct**	Incorrect
Vers les urgences :	Incorrect	Incorrect	**Correct**

TABLE 3.10 – Forme prise par nos matrices de confusion

	Accuracy	Precision	Recall
Arbre de décision :	0.35	0.38	0.41
Forêt d'arbres :	0.73	0.78	0.73
Bayésien naïf :	0.68	0.70	0.73
SVM :	0.68	0.68	0.68

Arbre décision : $\begin{pmatrix} 12 & 8 & 5 \\ 4 & 1 & 3 \\ 1 & 3 & 0 \end{pmatrix}$

Forêt d'arbres : $\begin{pmatrix} 22 & 2 & 1 \\ 5 & 3 & 0 \\ 2 & 2 & 0 \end{pmatrix}$

Bayésien naïf : $\begin{pmatrix} 25 & 0 & 0 \\ 6 & 2 & 0 \\ 3 & 1 & 0 \end{pmatrix}$

SVM : $\begin{pmatrix} 25 & 0 & 0 \\ 8 & 0 & 0 \\ 4 & 0 & 0 \end{pmatrix}$

TABLE 3.11 – Performances des meilleurs modèles avec les données d'origine, hors suppression des exemples à "Âge" manquants. Stratégie : *LeaveOneOut*

	Accuracy	Precision	Recall
Arbre de décision :	0.59	0.49	0.51
Forêt d'arbres :	0.72	0.72	0.72
Bayésien naïf :	0.64	0.64	0.69
SVM :	0.68	0.68	0.68

Arbre décision : $\begin{pmatrix} 17 & 7 & 2 \\ 5 & 1 & 2 \\ 1 & 4 & 0 \end{pmatrix}$

Forêt d'arbres : $\begin{pmatrix} 24 & 2 & 0 \\ 5 & 2 & 1 \\ 2 & 2 & 1 \end{pmatrix}$

Bayésien naïf : $\begin{pmatrix} 26 & 0 & 0 \\ 5 & 2 & 1 \\ 3 & 2 & 0 \end{pmatrix}$

SVM : $\begin{pmatrix} 22 & 2 & 2 \\ 4 & 2 & 2 \\ 2 & 1 & 2 \end{pmatrix}$

TABLE 3.12 – Performances des meilleurs modèles avec les données où les âges manquants ont été remplacés par l'âge moyen de la population. Stratégie : *LeaveOneOut*

Arbre décision : $\begin{pmatrix} 20 & 4 & 1 \\ 4 & 1 & 3 \\ 2 & 1 & 1 \end{pmatrix}$

	Accuracy	Precision	Recall
Arbre de décision :	0.59	0.49	0.51
Forêt d'arbres :	0.70	0.70	0.70
Bayésien naïf :	0.73	0.73	0.73
SVM :	0.68	0.68	0.68

Forêt d'arbres : $\begin{pmatrix} 24 & 1 & 0 \\ 4 & 3 & 1 \\ 2 & 2 & 0 \end{pmatrix}$

Bayésien naïf : $\begin{pmatrix} 25 & 0 & 0 \\ 7 & 1 & 0 \\ 2 & 2 & 0 \end{pmatrix}$

SVM : $\begin{pmatrix} 25 & 0 & 0 \\ 8 & 0 & 0 \\ 4 & 0 & 0 \end{pmatrix}$

TABLE 3.13 – Performances des meilleurs modèles avec les données avec suppression des exemples à "Âge" manquants et la transformation de l'âge en tranches d'âge. Stratégie : *LeaveOneOut*

Arbre décision : $\begin{pmatrix} 20 & 4 & 2 \\ 4 & 1 & 3 \\ 2 & 1 & 1 \end{pmatrix}$

	Accuracy	Precision	Recall
Arbre de décision :	0.56	0.54	0.51
Forêt d'arbres :	0.67	0.67	0.67
Bayésien naïf :	0.69	0.69	0.72
SVM :	0.67	0.67	0.67

Forêt d'arbres : $\begin{pmatrix} 25 & 1 & 0 \\ 4 & 2 & 2 \\ 2 & 3 & 0 \end{pmatrix}$

Bayésien naïf : $\begin{pmatrix} 26 & 0 & 0 \\ 6 & 0 & 2 \\ 2 & 3 & 0 \end{pmatrix}$

SVM : $\begin{pmatrix} 26 & 0 & 0 \\ 8 & 0 & 0 \\ 5 & 0 & 0 \end{pmatrix}$

TABLE 3.14 – Performances des meilleurs modèles avec les données où les âges manquants ont été remplacés par l'âge moyen de la population et la transformation de l'âge en tranches d'âge. Stratégie : *LeaveOneOut*

Nous avons tout d'abord calculé les poids associés à chaque classe dans les différentes situations possibles. L'intérêt de ces poids est leur association avec les exemples de la classe respective afin d'équilibrer la phase d'apprentissage. Ainsi un modèle sera plus pénalisé si il fait une erreur sur une catégorie rare que sur une catégorie commune. Cela oblige nos modèles à apprendre ces classes moins courantes. Pour des raisons techniques, nous n'avons ici évalué que les performances des forêts d'arbres et du *SVM* dans les conditions que sont la pondération des classes et la validation croisée *LeaveOneOut*. Les performances obtenues par les forêts d'arbres semblent se maintenir, avec quelques variations selon les situations.

Ainsi le modèle de 3.2 affiche une augmentation pour tous ses indices de mesure alors que le reste enregistre une diminution ou une stagnation de ses différents scores. Encore une fois, la Redirection vers les urgences s'avère difficile à prédire pour les forêts et seule le modèle de 3.2 réussit une prédiction correcte sur cette classe. Concernant les *SVM*s, on peut faire deux observations intéressantes. L'ensemble des modèles présentés use d'un noyau linéaire, comme le *SVM* le moins biaisé de l'expérimentation précédente. Il semblerait donc que le choix de ce noyau soit le plus efficace pour nos données actuelles. Aussi, tous nos *SVM*s, sauf un, enregistrent une chute drastique de performances, retombant aux alentours des 60%. Les matrices de confusion semble montrer un réel effort pour apprendre les autres classes, entraînant un nombre conséquent de classification incorrecte pour les Consultations. L'unique *SVM* n'ayant pas régressé (table 3.2) affiche une nette amélioration de ses performances avec un gain de 5 points. Nous pensons que l'utilisation des tranches d'âges est le facteur de cette amélioration. Nous nous appuyons sur le fait que les tables 3.2 et 3.2 revelent de meilleures performances respectivement que 3.2 et 3.2. Hors la différence entre ces derniers est la transformation de l'âge en tranche d'âge. Ces résultats, bien que légèrement biaisés, augurent de bonnes perspectives quant à l'obtention d'un classifieur performant. Il est certain que l'addition incrémentale de nouvelles données pourra améliorer les performances de nos différents modèles.

	Accuracy	**Precision**	**Recall**
Forêt d'arbres :	0.73	0.73	0.73
SVM :	0.59	0.59	0.59

Forêt d'arbres : $\begin{pmatrix} 23 & 1 & 1 \\ 5 & 3 & 0 \\ 1 & 3 & 0 \end{pmatrix}$

SVM : $\begin{pmatrix} 12 & 11 & 2 \\ 4 & 3 & 1 \\ 1 & 2 & 1 \end{pmatrix}$

TABLE 3.15 – Performances et matrices de confusion associées aux meilleurs modèles avec les données d'origine, hors suppression des exemples à "Âge" manquants. Stratégie : *LeaveOneOut* et pondération

	Accuracy	**Precision**	**Recall**
Forêt d'arbres :	0.67	0.69	0.69
SVM :	0.56	0.56	0.56

Forêt d'arbres : $\begin{pmatrix} 25 & 1 & 0 \\ 5 & 2 & 1 \\ 2 & 3 & 0 \end{pmatrix}$

SVM : $\begin{pmatrix} 14 & 10 & 2 \\ 4 & 2 & 2 \\ 1 & 2 & 2 \end{pmatrix}$

TABLE 3.16 – Performances et matrices de confusion associées aux meilleurs modèles avec les données où les âges manquants ont été remplacés par l'âge moyen de la population. Stratégie : *LeaveOneOut* et pondération

	Accuracy	Precision	Recall
Forêt d'arbres :	0.70	0.68	0.62
SVM :	0.73	0.73	0.73

Forêt d'arbres : $\begin{pmatrix} 22 & 2 & 1 \\ 34 & 3 & 1 \\ 1 & 3 & 0 \end{pmatrix}$

SVM : $\begin{pmatrix} 18 & 6 & 1 \\ 3 & 3 & 2 \\ 2 & 1 & 1 \end{pmatrix}$

TABLE 3.17 – Performances et matrices de confusion associées aux meilleurs modèles avec les données avec suppression des exemples à "Âge" manquants et la transformation de l'âge en tranches d'âge. Stratégie : *LeaveOneOut* et pondération

	Accuracy	Precision	Recall
Forêt d'arbres :	0.72	0.74	0.69
SVM :	0.59	0.59	0.59

Forêt d'arbres : $\begin{pmatrix} 25 & 1 & 0 \\ 3 & 3 & 2 \\ 1 & 3 & 1 \end{pmatrix}$

SVM : $\begin{pmatrix} 19 & 6 & 1 \\ 3 & 3 & 2 \\ 2 & 2 & 1 \end{pmatrix}$

TABLE 3.18 – Performances et matrices de confusion associées aux meilleurs modèles avec les données où les âges manquants ont été remplacés par l'âge moyen de la population et la transformation de l'âge en tranches d'âge. Stratégie : *LeaveOneOut* et pondération

Concernant les informations contenues dans les formulaires de consultation semi-urgente, nous nous sommes appuyés sur les observations recueillies sur les données des formulaires téléphoniques. Ainsi, nous avons procédé à une rapide analyse du nombre d'exemples et des valeurs associées. Tout comme pour les formulaires, le nombre d'exemples se révélait une nouvelle fois limité. Cette observation concorde avec la jeunesse du projet ainsi que le nombre d'appels téléphoniques. Souvent une personne appelante est admise pour une consultation semi-urgente. Cependant, le manque de standardisation des réponses s'est révélé plus important que pour les premiers formulaires. Les raisons derrière cela sont sûrement le nombre plus important de champs ainsi que le fait qu'un certain nombre de ces champs soit ouvert. Ainsi, pour le champ Fraction d'Éjection Ventriculaire Gauche (*FEVG*), on trouvait parfois des valeurs réglementaires à savoir des entiers mais certains exemples contenaient simplement la mention normale. Le problème est que la *FEVG* dépend de facteurs comme l'âge et nous ne disposons pas de l'expertise pour en déterminer la valeur. Aussi, bien qu'il y ait peu d'exemples ici, cela pose un véritable problème à grande échelle puisque cela demande des compétences et du temps afin de vérifier l'ensemble des exemples. De la même manière, certains praticiens ne complétaient pas certains champs ou n'utilisaient pas les mêmes unités de mesure pour un même champ. Ces cas se sont révélés bien trop nombreux pour que nous envisagions d'effectuer l'apprentissage sur les données telles quelles. Après discussions avec les membres de la *CSIC* sur les difficultés rencontrées, l'option d'utiliser les historiques médicaux conservés par le *CHSF* fut abordé. Cette approche présentait l'avantage d'enrichir la base de données avec un grand nombre d'exemples. Cependant, cela demanderait un grand travail de prétraitement pour obtenir des modèles les plus proches possibles de ceux souhaités pour les formulaires. Malheureusement cette piste dût être abandonnée car les systèmes du

CHSF appartiennent à de nombreux éditeurs différents et se révèlent très hétérogènes. Réunir et recouper les informations pour chaque patient représentait un investissement en terme de temps que nous ne pouvions raisonnablement demandé au service informatique de l'hôpital. Après discussion avec un rythmologue appartenant au projet de la *CSIC*, celui-ci partagea l'intérêt que pourrait avoir le projet à pouvoir identifier les alarmes cardiaques chez les patients possédant des appareils comme des *pacemakers*. En effet, un grand nombre de patients insuffisants cardiaques sont connus du *CHSF* et sont suivis par des médecins faisant parfois partie du projet. Nous avons donc expérimenté dans cette voie afin de découvrir ce qu'il pouvait être fait et en utilisant de l'apprentissage profond.

3.3 Les données PhysioNet

Après propositions de différentes sources de données aux médecins, notre choix s'est porté sur une base d'informations mise à disposition par PhysioNet. PhysioNet est une plateforme offrant l'accès à de larges collections de données médicales, en particulier des signaux physiologiques. Chaque année et ce depuis 2000, est proposé sur le site un challenge cherchant à développer et améliorer des algorithmes pour traiter un problème médical spécifique. Nombre de chercheurs participent à ces challenges et font ensemble avancer le domaine de l'apprentissage automatique appliqué à la médecine. Le challenge qui nous a intéressé date de 2015 et se nomme "*Reducing False Arrhythmia Alarms in the ICU*". On compte pour ce dernier 20 publications et 28 *softwares* proposés par des chercheurs. Le sujet du défi consistait à définir des modèle permettant de réduire le nombre de fausses alarmes cardiaques enregistrées par une unité de soin intensive. En effet, les fausses alarmes cardiaques impactent négativement sur la qualité de vie des praticiens et des patients ainsi que sur la qualité des soins. Cinq types d'alarmes différentes ainsi que leurs caractéristiques sont définies par le sujet : l'asystolie, la bradycardie extrême, la tachycardie extrême, la tachycardie ventriculaire et la fibrillation ventriculaire.

Ce challenge présentait comme spécificité un partenariat avec *MATLAB*. Ainsi chaque exemple appartenant à la base de données était enregistré sous la forme de deux fichiers. Le premier de ces fichiers est enregistré sous le format *MATLAB Data*. Il contient les signaux sous forme de séries temporelles en unité digitale. Les valeurs de ces séries temporelles contiennent uniquement des entiers et ont besoin d'un pré-traitement afin d'obtenir les véritables valeurs en unité physique. La fonction *RDMAT* de la boîte à outils *WFDB MATLAB* est une solution pour effectuer cette transformation. Plus d'informations sur ces signaux sont disponibles grâce au second fichier de type *.hea* pour *header*. On y trouve des informations sur la façon dont l'exemple a été échantillonné ainsi que le type d'alarme déclenchée et sa véracité. De plus, chaque signal est associé à un acronyme permettant de déterminer le sujet de l'observation. Ainsi on trouve *ABP* pour *Arterial lood Pressure*, *PLETH* pour *Plethysmogram* et ainsi de suite. Il est à noter que tous les exemples ne présentent pas les mêmes signaux observés. Une fois cette rapide analyse de la structure des données avec lesquels nous allions travailler, nous nous sommes penchés sur les travaux des différents participants.

Parmi l'ensemble des *softwares* proposés, le meilleur score obtenu par un modèle s'élève à 81,39%. Ce score est calculé à partir de la formule

$$SCORE = \frac{TP + TN}{TP + TN + FP + 5 * FN}$$

où l'on peut observer le poids beaucoup plus important attribué aux erreurs Faux Négatif

comparé aux Faux Positifs. Ce score n'est mentionné qu'à titre indicatif puisqu'il nous sera impossible de nous comparer avec les participants. En effet, le test de validation utilisé par les auteurs du challenge était et demeure caché du public. Le papier que nous avons jugé le plus intéressant est celui écrit par Ansari et al. [49]. Avec un bon score s'élevant à 74,48%, l'intérêt des travaux d'Ansari et al. réside dans la manière concise de décrire les étapes de prétraitement et les techniques employées pour détecter les différents pics des séries temporelles. Afin de ne pas nous attarder sur la méthodologie de ces prétraitements, qui ne constitue pas notre domaine d'intérêt, notre décision fut de se servir des méthodes mises à disposition par Ansari et al. dans leur *software* afin de nous concentrer sur la prédiction.

Deux soucis s'imposèrent à nous. Premièrement, le code fourni contenait un certain nombre d'erreurs empêchant l'exploitation de tous les types d'alarmes. Ne pouvant déterminer les origines de ces erreurs et n'ayant pas réussi à établir de contact avec les auteurs, nous nous concentrâmes sur les alarmes de type Tachycardie Ventriculaire dont la vaste majorité des exemples nous était disponible. Le second problème émergea du fait que nous ne possédions pas d'experts à notre disposition afin de nous assurer de la normalité des pics détectés par le prétraitement d'Ansari et al. Notre solution pour pallier à ce problème fut de nous tourner vers des méthodes de *clustering*. Puisque nous connaissions l'aspect d'un battement de cœur normal et anormal pour une tachycardie ventriculaire, nous pouvions déterminer à partir de l'allure moyenne des pics de nos clusters leur appartenance à l'une ou l'autre catégorie. Pour ce faire, nous avons essayé un nombre variable de clusters afin de déterminer ceux les plus pertinents en terme de forme et de taille. Notre meilleur résultat s'avéra être la division en trois clusters comme le montre la figure 3.1. Nous avons ainsi déterminé que les deux clusters de gauche correspondaient à des battements anormaux grâce à l'allure obtus des courbes. Le cluster de droite quant à lui présente un pic plus aigu similaire à l'aspect normal d'un battement. En réalité, les deux meilleures choix pour le nombre de clusters consistaient soit à deux (figure 3.2) soit à trois clusters (figure 3.1). Cependant, l'allure du cluster "normal" ne semblait pas être impacté négativement par l'augmentation du nombre de clusters tandis que les profils de clusters semblaient s'affiner. C'est pour cela que notre décision finale se porta sur ce nombre. On remarquera un déséquilibre des classes avec un plus grand nombre de pics jugés normaux, ce qui semble être une observation normale.

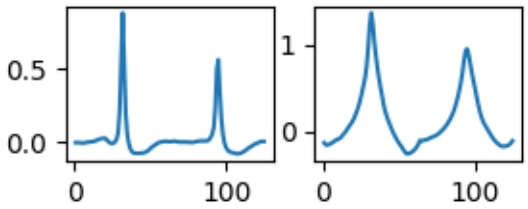


FIGURE 3.2 – Clusters obtenus pour un nombre de clusters fixé à deux. Taille des clusters de gauche à droite : 1077, 5513.

Pour la construction de modèles d'apprentissage profond, nous avons usé de la bibliothèque *Python Keras* [50]. Elle consiste en une surcouche pour *Theano* [51] ou *Tensorflow* [52] facilitant le développement et l'expérimentation de réseaux de neurones. Deux modèles furent testés dans le cadre de nos recherches, un *Multi Layer Perceptron* simple possédant une unique couche cachée et un *Multi Layer Perceptron* à deux couches cachées. L'objectif était ici d'observer l'impact de couches supplémentaires sur le problème

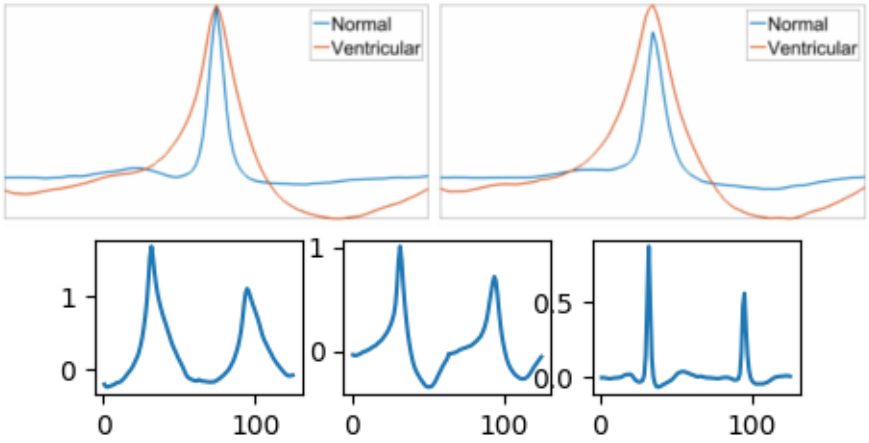


FIGURE 3.1 – En haut : l'allure d'un battement normal en bleu et deux allures de battements trahissant une tachycardie ventriculaire. En bas : Les trois clusters que nous avons isolé, avec deux profils anormaux à gauche et un profil normal à droite. Taille des clusters de gauche à droite : 1566, 379, 4645.

de détection de pics anormaux. Chaque série temporelle à évaluer étant divisée en 125 périodes, nos deux modèles possèdent par conséquent autant de neurones d'entrée. Les réseaux de neurones se terminent par un seul neurone de sortie indiquant la normalité ou l'anormalité de l'exemple donné en entrée. Comme précédemment, nous avons fait usage d'une *grid* afin de tester un grand nombre de combinaisons de paramètres pour obtenir les meilleurs résultats. La validation croisée 10-*Fold* fut employée afin d'évaluer les performances de l'algorithme. De même, des poids furent assignés à chaque classe afin de contrebalancer le déséquilibre observé.

Nombre de neurones dans la couche cachée :	50	100	125	150	250

TABLE 3.19 – Ensemble des valeurs testées pour le paramètre "Taille de la couche cachée".

Les paramètres les plus évidents de nos grilles furent les taille des couches cachées des réseaux de neurones. Nous avons fait varier la taille possible entre 50 et 250 neurones afin d'observer l'impact de la diminution ou l'augmentation par rapport à la quantité de départ. Puisque le nombre de combinaisons finales se révéla important, nous avons décidé de réduire le nombre de valeurs possibles aux cinq mentionnées dans la table 3.19. Le second critère s'intéresse à la manière d'initialiser les poids associés aux neurones de chaque couche. Parmi l'ensemble des possibilités offertes par *Keras*, notre choix s'est porté sur une distribution suivant une loi normale et une autre suivant une distribution aléatoire. Afin de ne pas complexifier d'avantage le modèle, nous sommes parti du principe que toutes les couches emploieraient la même méthode d'initialisation. Pour ce qui est de la retro-propagation de l'erreur, nous avons testé différentes manières pour calculer l'erreur qui sera minimisée par l'algorithme. Nous avons ainsi essayé l'erreur absolue moyenne, l'erreur quadratique moyenne et l'entropie croisée. Afin de minimiser l'erreur, la descente du gradient stochastique (*SGD*) fut bien entendu utilisée mais aussi une variante faisant usage de l'élan pris par l'algorithme (*RMSprop* [53]) et une autre s'appuyant sur des estimations adaptatives (Adam [54]). La fonction d'activation *ReLu* (*rectified linear unit*) fut utilisée comme fonction d'activation pour nos neurones [55]. Ce choix découle de la popularité de cette fonction d'activation dans le milieu de l'apprentissage profond et nous

voulions donc expérimenter avec cette dernière dans un premier temps.

Les résultats obtenus à partir du *MLP* simple se révélèrent très bons avec le modèle le plus performant atteignant 98% d'*Accuracy*. Ces résultats étonnants nous firent envisager dans un premier temps la présence de sur-apprentissage par notre réseau de neurones. Cependant, au vu de l'utilisation de la validation croisée, ce risque se révélait assez mince. Notre supposition fut que le choix de nos clusters s'était révélé pertinent ce qui expliquait les performances observées. Connaître les véritables étiquettes aurait été utile afin d'évaluer cette hypothèse. Aussi, ces chiffres n'apparaissent si aberrants lorsque l'on prend en compte les éléments suivants. Il ne faut pas comparer nos résultats avec ceux des différents participants du challenge. Le challenge demandait de pouvoir distinguer les fausses et vrais alarmes parmi 5 types d'alarme. Notre modèle est spécialisé sur une seule de ces catégories. Aussi, nous ne distinguons pas la véracité des alarmes mais simplement la normalité des battements de cœur. La prédiction effectuée par les participants relève donc d'une étape postérieure à notre prédiction et donc potentiellement plus complexe. Il est également possible que nos performances se révèlent moins impressionnantes si nous avions pu tester notre modèle sur les données de validation cachées au public.

Concernant le *MLP* à deux couches, les performances obtenues se placent dans la continuité de celles observées précédemment. Le meilleur *MLP* réussit à atteindre 96,5% d'*Accuracy* mais dans l'ensemble, les performances se révèlent sensiblement moins bonnes. De plus, nos observations semblent montrer que les modèles obtiennent de meilleures résultats lorsque les deux couches cachées font la même taille. Il serait intéressant, dans l'éventualité où d'autres types d'alarme soient évalués par nos modèles, de vérifier si ce phénomène se réitère ou non. Nous pensons que cette baisse de performance peut être évitée grâce à l'usage du *DropOut* ou de techniques de régularisation (*L1*,*L2*,...). Les techniques de régularisation ont pour objectif de pénaliser les valeurs de poids (sous certaines conditions) afin d'éviter le sur-apprentissage. Cela se serait révélé particulièrement utile puisque nous n'avons pas de traces évaluant le déroulement de l'apprentissage au fur et à mesure des *epochs*. Le *DropOut*, quant à lui, consiste à désactiver aléatoirement un certain pourcentage de neurones à chaque étape pour éviter également le sur-apprentissage. Un autre indicateur pour vérifier la vraisemblance des résultats que nous avons obtenu serait de les comparer à ceux d'algorithmes plus classiques comme *SVM*s, les forêts d'arbres etc.

Ces résultats et ces perspectives concluent notre partie sur l'apprentissage automatique. Nous nous concentrerons dans la suite de ce mémoire au développement du système multi-agents qui verra leur intégration.

Accuracy	***Optimizer***	***Loss***	**Initializer**	**Couche**
98%	Adam	Erreur quadratique	Normale	100 neurones

TABLE 3.20 – Paramètres et performance du meilleur *MLP* simple.

Accuracy	***Optimizer***	***Loss***	**Initializer**	**Couche 1**	**Couche 2**
96,5%	Adam	Erreur quadratique	Normale	250 neurones	250 neurones

TABLE 3.21 – Paramètres et performance du meilleur *MLP* à deux couches.

3.4 L'architecture multi-agents

En parallèle du travail d'analyse des données de la *CSIC* et de l'entraînement des modèles d'apprentissage automatique, notre tâche consistait à développer un système capable d'accompagner le personnel du projet dans le processus de consultation semi-urgente. Pour ce faire, nous avons sélectionné l'emploi des systèmes multi-agents. Une des raisons derrière ce choix est la complexité de l'environnement hospitalier de manière générale. Chaque acteur du système possède ses propres caractéristiques, ses points forts, ses préférences et ainsi de suite. Couplés au nombre de spécialités et à la diversité des tâches à effectuer, ces propriétés du système désignaient les agents comme une solution parfaite. Il est en effet possible par leurs intermédiaires de représenter cette hétérogénéité puisque chaque agent possède des caractéristiques lui étant propre. Aussi, ils offrent un avantage non négligeable pour la modélisation de systèmes complexes comme celui du CHSF. Un système modélisé et représenté de manière figée demande un coût énorme en terme de temps et de ressources à mettre en place. Une fois cette mise en place terminée, la maintenance représente également un challenge de taille. On compte dans ce coût des aspects techniques comme le codage du modèle de représentation mais également des aspects de recherche comme la spécification de tous les cas de figure possibles. Cette manière de faire est particulièrement fragile puisque chaque cas de figure non-pris en compte entraîne des coûts de modification de mise à jour du système, avec tous les potentiels conflits que cela peut faire émerger. Si l'on peut espérer que ces types d'événements resteront rares, un hôpital implique un environnement en perpétuelle évolution où l'imprévu a une place omniprésente. Les agents, de part leur capacité d'adaptation, permettent de développer des systèmes robustes et capables de s'adapter à des situations inattendues. De plus si les rôles d'un acteur ou d'une classe d'acteurs viennent à changer, il suffit de modifier les agents concernés en conséquence pour modifier le système. Ces propriétés attendues dans un tel système constituent le challenge de la modélisation multi-agents. Notre sujet étant pensé comme les prémisses d'une thèse, un certain nombre des choix expliqués dans cette section seront motivés par une volonté de posséder une flexibilité permettant l'élargissement du système à d'autres services du CHSF. Aussi, bien que l'hôpital n'envisage pour l'instant pas l'usage de *smartphones* ou *PDA*s, l'architecture présentée prend en compte une grande mobilité des acteurs du système et donc part du postulat que les agents seront capables de les suivre où qu'ils soient. Ce parti pris ne pose pas de problèmes particuliers par rapport à la situation réelle (utilisation via ordinateur) puisque cela pourrait être simplement représenté comme un appareil ne se déplaçant pas.

Dans un premier temps, afin d'obtenir une modélisation pertinente du système étudié, il faut le comprendre et déterminer les règles le régissant, les acteurs y étant présents. Pour notre part, nous nous sommes d'abord concentrés sur l'étude des différents profils impliqués de près ou de loin avec le processus de consultation semi-urgente. Ainsi, nous nous sommes basés sur nos discussions avec les différents membres de la CSIC ainsi que l'organigramme de l'annexe A pour isoler les agents nécessaires au bon fonctionnement de la consultation semi-urgente. On retrouve bien sûr les deux professionnels de la santé que sont l'infirmier et le médecin. De part leur place au sein du processus de prise en charge du patient, leur présence se révèle indispensable pour une simulation réaliste de la situation. Il existera donc deux types d'agent, l'agent *Nurse* et l'agent *Doctor*, associés respectivement à l'infirmier et au médecin. Le projet de la *CSIC* étant le premier "service" à être modélisé, il n'existe pas d'autres types d'infirmier ou de médecins dans le cadre du système. Les agents que nous présentons ici sont donc des archétypes spécialisés pour

la consultation semi-urgente. Le rôle des agents *Nurse* et *Doctor* consiste à épauler leurs partenaires humains dans leurs tâches quotidiennes. Ainsi l'agent *Nurse* devra permettre une simplification de la saisie des informations du patient par l'infirmier. Aussi, en intégrant un ou des modèles d'apprentissage automatique, on attend de l'agent qu'il serve de conseiller à l'infirmier dans l'admission des patients. Cependant, le rôle de l'agent se confinera au conseil puisque le décisionnel doit reposer sur les épaules d'un être humain. En effet, au delà de problèmes éthiques que cela pourrait poser, un algorithme ne peut disposer de l'ensemble des informations disponibles au médecin, comme le ton de l'interlocuteur par la voix. L'optimisation du temps du personnel hospitalier étant un facteur important pouvant réduire les coûts et les erreurs médicales, notre agent doit également être capable d'améliorer cet aspect du travail d'infirmier. Ainsi l'agent pourra automatiser la prise de rendez-vous de consultation, transmettre les dossiers au médecin concerné etc. Le rôle de l'agent *Doctor* se révèle quant à lui assez similaire à celui de l'agent *Nurse*. Il se concentre sur la saisie des informations des formulaires de consultation semi-urgentes et sert de conseiller sur la marche à suivre vis à vis du patient. L'enjeu de la prise de décision est de déterminer si le patient doit être hospitalisé ou non. Aussi, l'agent se chargera d'avertir le médecin en temps réel de nouvelles consultations, de consultations imminentes et de la transmission du dossier du patient à examiner.

Puisqu'il est au centre de la démarche, un autre acteur qu'il est évident d'inclure au sein de l'architecture est le patient lui-même. Avec l'émergence et la popularisation des *smartphones*, il n'est pas difficile de concevoir qu'une personne souffrant d'insuffisance cardiaque installe un agent sur son appareil. L'objectif de cet agent *Patient* sera de pouvoir renseigner les professionnels de la santé efficacement en leur permettant d'accéder aux informations du malade. Ces informations seront pour la plupart d'ordre médical comme ses antécédents, ses traitements etc. Il pourrait cependant être utile de disposer d'informations plus générales par exemple ses préférences afin de disposer d'un parcours de soin personnalisé et plus approprié. Au delà de cela, on peut envisager la connaissance des numéros de téléphone des proches pour pouvoir les contacter rapidement. Si les proches disposent aussi d'un agent, on peut imaginer qu'ils pourraient également recevoir une notification par cet intermédiaire. Cette mise à disposition des informations se révélerait particulièrement intéressante dans le cas où le patient est inconscient. Bien que nous ne nous sommes pas penchés sur cet aspect, la sécurité entourant les agents *Patient* se doit d'être particulièrement élevée. Là où la sensibilité des informations accessibles aux agents *Nurse* et *Doctor* se voit mitigée par le cadre restreint qu'est l'hôpital, on suppose ici que le patient se déplace n'importe où avec son agent et ses informations personnelles. Aussi, les questions de vie privée se posent et le patient doit pouvoir définir les informations qu'il souhaite partager ou non. Dans le cadre de la consultation semi-urgente, suite à une requête de l'agent *Nurse*, l'agent *Patient* enverra de manière automatiquement les informations nécessaires au remplissage du formulaire téléphonique. De plus, si il y a réellement besoin d'une consultation, il pourra recevoir directement un récapitulatif des examens demandés par l'infirmier à l'intention de son utilisateur. Il existe également la possibilité d'effectuer des avertissements pour rappeler au patient la date et l'heure de ses rendez-vous.

Deux profils supplémentaires d'acteurs humains furent également identifiés, celui du laborantin et celui de l'externe. Le laborantin désigne les différents individus chargés de s examens et des différentes analyses associées. Dans le cadre théorique de la simulation, ces derniers peuvent soient être des laborantins internes au *CHSF* ou des laborantins externes au *CHSF*. Cependant, dans la réalité, seuls les laborantins internes pourraient

avoir accès à un agent spécifique, appelé *Technician*, puisque la connexion directe avec un laboratoire externe est une procédure lourde. En effet, cela demande souvent un investissement financier conséquent et la mise en place de protocoles de sécurité pour l'échange d'informations. L'agent *Technician* servira de relais entre les laborantins et les médecins afin que les résultats des examens circulent plus facilement au sein de l'hôpital. Ainsi, lorsque toutes les analyses sont terminées, le médecin référent pour le patient obtient automatiquement un bilan transmis par l'agent du laborantin. Cela permet d'éviter des déplacements et donc de gagner du temps. Le second profil est en réalité un ensemble de profils différents réunis sous le même nom. L'*External* couvre toutes les possibilités d'un intervenant extérieur à l'hôpital comme le SAMU, le médecin traitant, le cardiologue traitant etc. Ce regroupement s'explique par le fait que ces profils n'interviennent ne disposeront pas d'une implémentation dans notre version du système multi-agents. Ils ne remplissent donc aucun rôle particulier et n'influencent pas le déroulement de la consultation. Ainsi, pour ces raisons et leur non-appartenance au *CHSF*, ils ne sont donc pas concernés par le développement actuel du système multi-agent. Malgré cela, les agents *External* sont présents d'un point de vue conceptuel pour donner plus de réalisme à la simulation de la consultation semi-urgente.

Une fois ces profils définis, il nous a fallu creuser un peu plus le mode de fonctionnement des différents agents, en particulier des agents *Nurse* et *Doctor*. La problématique tournait autour de la façon dont serait encapsulé l'apprentissage automatique par les agents. Premièrement, un agent devrait-il disposer d'un seul modèle ou bien d'une combinaisons de plusieurs d'entre eux ? Le dernier cas offre des avantages non négligeables. D'une part, certains modèles peuvent se révéler plus efficaces face à certains cas de figure, servant alors d'experts en la matière. De plus, si une majorité d'entre eux obtiennent un même résultat alors cela peut constituer un signe de confiance envers le système du point de vue du praticien. Ces avantages peuvent cependant rapidement tourner à l'inconvénient. En effet, rien ne peut permettre au praticien d'identifier quelles sont les modèles experts dans un cas donné. De plus, même si cela lui était possible, le praticien n'a pas à obtenir de connaissances supplémentaires pour évaluer les performances des différents modèles. Cela nuirait à l'objectif premier de l'agent qui est de simplifier la prise de décision de l'utilisateur. Cette prise de décision se révèle encore plus difficile si les différents modèles n'arrivent pas à un consensus pour la décision finale. Un tel résultat présente le risque d'entraîner une confusion du praticien et donc de ralentir sa prise de décision, mais aussi de réduire la confiance qu'il place dans le système. De plus, plusieurs modèles impliquent que lors des mises à jours des classifieurs de l'agent, chaque modèle doit être ré-entraîné individuellement. Cette étape peut donc s'avérer coûteuse en terme de temps et de ressources, bien que la quantité actuelle de données ne poserait pas de tels soucis. Afin de simplifier la modélisation et la prise en main du système par les praticiens, nous avons pour ces raisons décidé de ne définir qu'un seul modèle par agent.

Cet aspect de mise à jour soulève une seconde question : comment va se dérouler l'étape d'entraînement ? Pour pouvoir l'effectuer, les différents agents devront disposer d'un accès direct au contenu de la base de données recueillant tous les dossiers patients. Cela implique un potentiel risque en terme de sécurité dans le cas où un agent malveillant ou un utilisateur malveillant accède au système. Une solution serait de passer par l'envoi de messages sécurisés mais encore une fois le volume de données peut s'avérer être un obstacle en terme de temps et de ressources. Une autre problématique est l'hétérogénéité des résultats chez un même profil d'agent. Il faut ici comprendre que, le système étant décentralisé, les mises à jours s'effectuent au gré de l'utilisateur. Il existera potentiellement

des divergences entre les résultats fournis par deux agents pour un cas donné puisqu'ils ne posséderont pas la même version du modèle. Cela peut donc constituer un motif supplémentaire de méfiance envers le système. Cependant, ce risque se fera de moins en moins prévalent au fur et à mesure que la base d'apprentissage s'élargit. Un modèle robuste, se basant sur des centaines de cas, ne devrait en effet pas être affecté de manière radicale face à l'addition de nouveaux exemples. Dans le cas contraire, il faudrait remettre en question le dit-modèle. Notre solution fut de créer un nouvel agent pour s'occuper de l'apprentissage automatique, l'agent *Data Manager* ou *DM*. Le rôle du *DM* va être de servir d'intermédiaire entre la base de données et les autres agents du système. Ce sera donc lui qui effectuera l'entraînement des différents modèles. Un agent *Nurse* cherchant à se mettre à jour enverra une requête demandant un nouveau modèle au *DM*. La requête comportera simplement la date associée à la dernière mise à jour de l'agent. Dans le cas où cette date est évaluée comme récente, le *DM* comprendra que l'utilisateur n'est pas satisfait par le modèle actuel et entamera à nouveau une phase d'apprentissage. Cette dernière finie, deux cas de figure se présentent. Soit les performances du nouveau modèle se révèlent meilleures ou équivalentes à celle du dernier modèle enregistré auquel cas le modèle sera stocké en interne comme la dernière version du classifieur puis envoyé à l'agent *Nurse*. Soit les performances se révèlent nettement moins bonnes et le *DM* informe par un message qu'il est préférable de garder le modèle actuel. Pour le cas où la date est assez ancienne, on transmet simplement le modèle stocké par le *DM*. Cette approche répond à la problématique de la mise à jour des modèles. Au lieu de devoir lancer autant d'instances d'apprentissage que d'agents, il suffit simplement de contacter le *DM*.

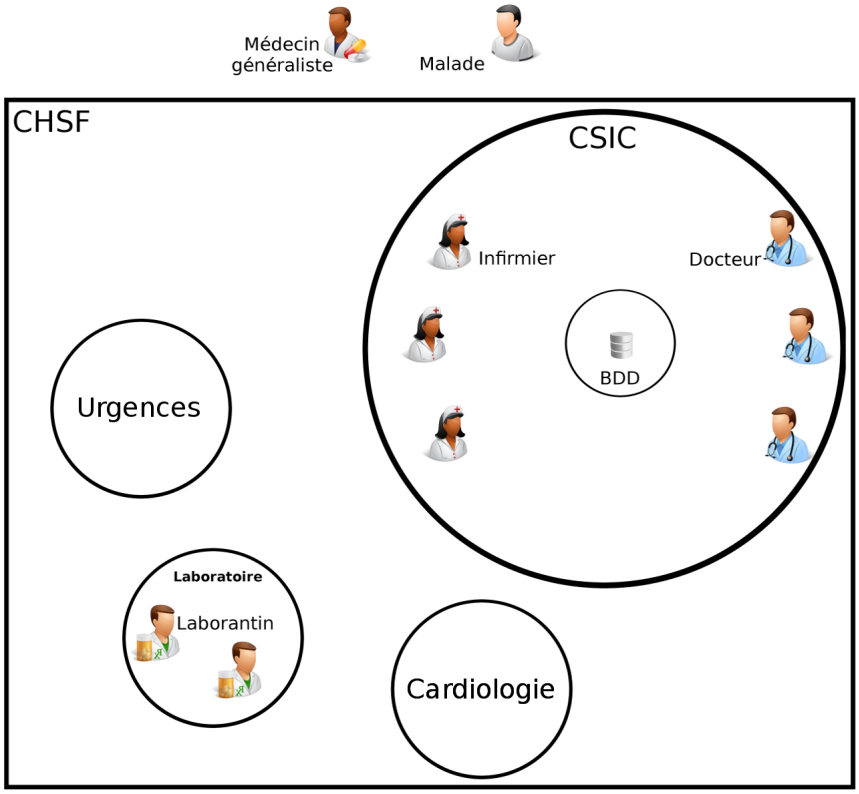


FIGURE 3.3 – Modélisation de l'écosystème de la *CSIC* avec ses différents acteurs.

On réduit à la fois les coûts en terme de temps et de ressources mais également la proximité avec les données. Cette solution n'empêche cependant pas les agents d'avoir

des modèles différents. L'hétérogénéité des modèles se révèle toutefois moins importante puisqu'au lieu d'avoir de nombreux modèles différents, on peut ici réfléchir en terme de "versions" du modèle. Pour mitiger ce phénomène, notre solution est de donner à l'agent *DM* un registre associant chaque agent l'ayant contacté et à une version du classifieur. L'agent *DM* endosse alors le terme d'observateur et pourra suggérer aux agents possédant une version trop ancienne de la remplacer par la dernière. Bien que cette approche ne règle pas de manière définitive le problème d'hétérogénéité, les variations observées ne devraient se révéler que mineures et n'affecteront pas la perception par l'utilisateur. Un autre déclencheur de mise à jour pourrait également être le pourcentage d'erreurs rapportés par les utilisateurs lors de l'ajout de nouveaux exemples, indiquant la nécessité de classifieurs plus adaptés.

La position du *DM* vis à vis de la base de données le place comme le régulateur des requêtes des autres agents. Ainsi, ce sera à lui que seront envoyés les nouveaux cas de la consultation semi-urgente et que les demandes de dossiers patients seront adressés (par les agents). Concernant ce dernier point, nous avons en effet considéré que les agents *Nurse* et *Doctor* ne devaient pas garder en mémoire les informations des patients rencontrés. À la place, ils conserveront un numéro de dossier qui leur sera fourni par le *DM* lors de l'ajout de l'exemple à la base de données. Ce choix s'appuie sur deux points. Si un autre agent vient à mettre à jour le cas d'un patient sur la base de données (Mauvaise information par exemple), l'agent *Nurse* ayant rencontré ce patient ne disposera pas des informations les plus récentes. Les conséquences de cette non-cohérence pourrait alors résulter en des erreurs médicales évitables. De plus, le stockage des informations représente un risque de sécurité si une personne mal intentionnée accède à la mémoire de l'agent. Puisque les informations échangées sont sensibles, la sécurisation des messages se révélera donc extrêmement importante. Cette centralisation de l'accès à la base de données au niveau du *DM* implique toutefois que s'il venait à défaillir, il serait impossible d'interagir avec les données. Un tel cas de figure paralyserait l'ensemble du système, ce qui s'oppose à nos souhaits d'un système robuste. Nous avons donc décidé dans notre modélisation finale de définir plusieurs agents *DM* de sorte à parer à cette éventualité. L'approche présente de plus l'avantage de diviser le flux de requêtes des agents à travers l'ensemble des *DM*s. Cette répartition du "trafic" nous a également inspiré pour aller plus loin. En effet, si tous les agents *DM* prenaient en charge tous les types de modèles alors cela nécessiterait beaucoup de communications pour assurer la cohérence des classifieurs proposés. Ici, il existera un *DM* référent pour chaque type de modèle nécessaire aux agents.

Finalement, nous avons défini un dernier agent dont le rôle est simplement d'accomplir un service pour les autres agents de la *CSIC*. Cet agent appelé *Planner* encapsule l'agenda contenant les vacations des membres de la *CSIC*. L'objectif est de proposer automatiquement des vacations pour les futures consultations. Si la vacation n'arrange pas le patient, l'infirmier le fait savoir à son agent qui transmet à son tour le message au *Planner*. Ce dernier propose alors des vacations jusqu'à ce que le patient soit pris en charge, dans la limite des 36 heures imposée par le projet. De plus, le *Planner* dispose d'un registre répertoriant l'ensemble des membres de la *CSIC*. Ainsi il est capable de notifier immédiatement l'agent du médecin concerné par la vacation venant d'être alloué à la consultation semi-urgente. Bien que le problème de dépendance à l'agent *Planner* se pose de la même manière que pour l'agent *DM*, nous avons décidé dans un premier de ne définir qu'un seul et unique *Planner*. Cette décision a été motivé par le fait que les problèmes de cohérence pouvant découler de multiples *Planners* et les efforts pour les éviter se révéleraient trop lourds pour une fonction "mineure" du système. Le temps de remise en service du *Planner*,

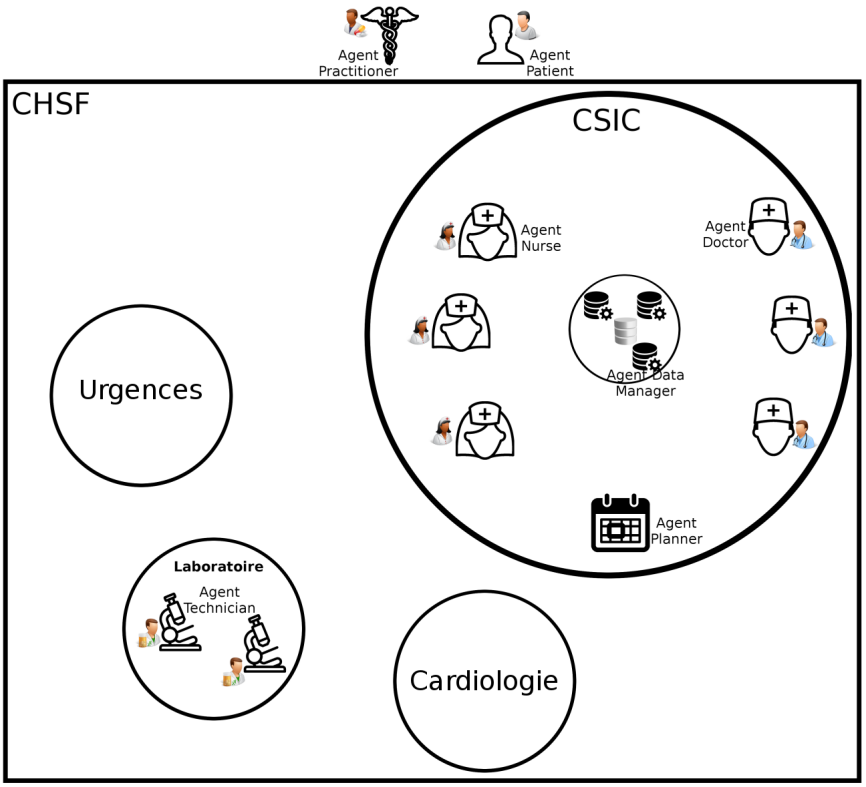


FIGURE 3.4 – Intégration du système multi-agents au sein de l'écosystème de la *CSIC*.

l'utilisation de communication humain à humain semble une solution facile à mettre en place. Ce choix resterait cependant sujet à changement, selon les observations de l'impact sur les performances humaines.

Dans la figure 3.3, nous avons modélisé l'écosystème entourant la *CSIC* et les différents acteurs y coexistant. Cet écosystème se divise en deux environnements, celui des acteurs appartenant au *CHSF* et celui des acteurs n'appartenant pas au *CHSF*.

Ces acteurs se trouvent respectivement à l'intérieur et à l'extérieur du carré délimitant le *CHSF*. Pour cet exemple et dans un soucis de simplicité, seul le médecin généraliste du malade a été conservé parmi les externes hors malade. Ce choix n'influe pas sur la modélisation puisque les externes ne possèdent qu'un rôle passif dans le processus de la consultation semi-urgente. Le *CHSF* lui-même est constitué de divers services tels que les urgences, la cardiologie, les laboratoires et bien sûr le projet de la *CSIC*. Les services représentés ici ne constituent bien sûr pas une liste exhaustive des services de l'hôpital. Chacun de ces services est constitué d'un certain nombre de membres possédant un statut parmi infirmier, docteur et laborantin. Le projet de la *CSIC* dispose d'un élément supplémentaire qui est la base de données dans laquelle seront stockés les données des formulaires téléphoniques et de consultation. Á partir de cette situation, nous avons représentés les différents acteurs et leurs agents dans la figure 3.4. Ici l'agent *External* associé au médecin généraliste a été nommé *Practitioner* afin de pouvoir identifier rapidement l'acteur humain associé. L'agent ne diffère cependant pas d'un *External* classique. On retrouve donc chaque acteur associé à son agent mais on peut également observer l'apparition des agents *DM* et *Planner*. Conformément à nos explications, plusieurs agents *DM* sont associés à la base de données afin d'en assurer l'accès à tout moment. Finalement la figure 3.5 met en évidence les relations entre les différents agents du système. Ces relations mettent en

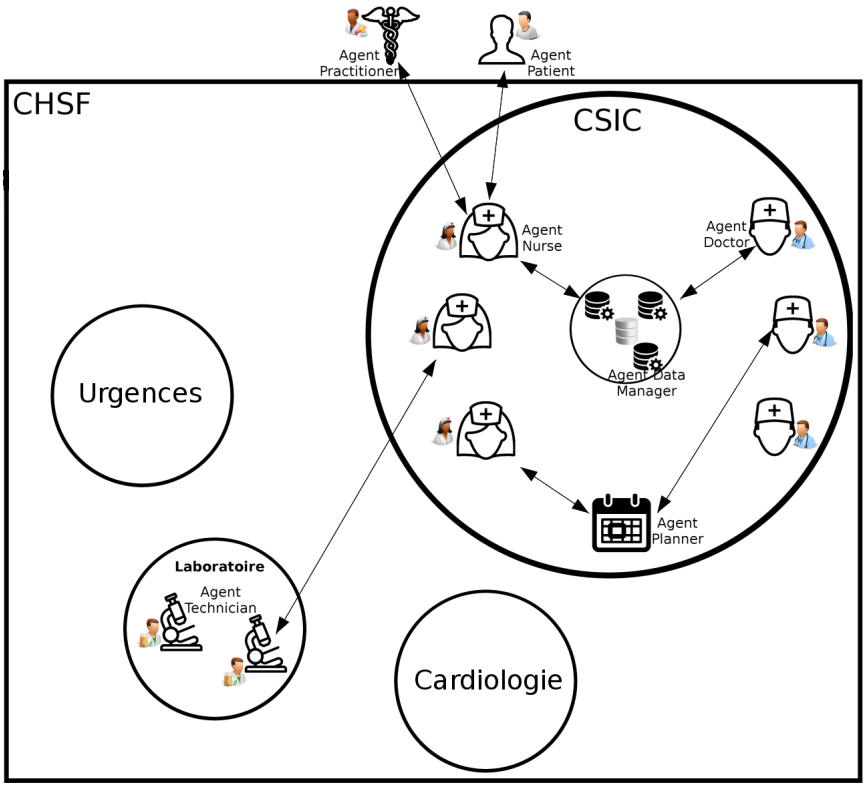


FIGURE 3.5 – Représentation des différentes interactions possibles entre les agents de la *CSIC*.

évidence que l'infirmier et son agent représentent un point focal du système puisqu'ils interagissent avec l'ensemble des autres acteurs de l'écosystème. Cette observation concorde avec la place de l'infirmier dans le processus de soin et de sa prise en charge des appels téléphoniques. Pour ce qui est des *DM*s, on peut voir qu'un *DM* donné communique avec un seul des deux profils médicaux. Cette exclusivité s'explique bien sûr par la spécialisation des *DM*s pour un type donné de classifieur.

Nous achevons donc cette section sur ces figures. Nous avons pu présenter et expliquer les choix nous ayant mené à construire de la sorte cette architecture multi-agents. La prochaine partie consistera en la description de l'implémentation effectuée à partir de cette architecture où nous y aborderons des choix comme le langage utilisé.

3.5 Implémentation

Pour l'implémentation de notre architecture, nous avons examiné différents langages ou *frameworks* se spécialisant dans la modélisation de systèmes multi-agents. Parmi les choix s'offrant à nous, notre intérêt se porta sur le *framework JAVA Agent DEvelopment* plus communément connu sous le nom de *JADE* [56, 57]. *JADE* est un *middleware* originellement développé en 2000 par la branche recherche et développement de *Telecom Italia Group* et devenu depuis un projet communautaire *open source*. La communauté l'entourant se révèle toujours active, le perfectionne et s'assure de sa comptabilité avec les dernières technologies. Pour preuve de ce suivi, la mise à jour la plus récente remonte à Juin 2017. Ce point constitue un des facteurs attractifs de *JADE* et se révèle assez rare pour être souligné. Aussi, ce *framework* possède l'avantage d'intégrer les normes de la

Foundation for Intelligent Physical Agents ou *FIPA*. Cette dernière est une organisation de standardisation affiliée à l'*IEEE* et cherchant à promouvoir l'émergence du paradigme multi-agents et de l'interopérabilité entre les divers systèmes existants. Pour ce faire, un certain nombre de spécifications ont été définis afin de faciliter la communication et coopération inter-agents. Nous décrirons plus en détail ces normes plus tard dans cette section. Aussi, comme son nom l'indique, *JADE* a été entièrement été écris en *JAVA* et permet donc de jouir des nombreuses fonctionnalités mises à disposition du langage. Finalement, le langage *JAVA* est connu pour sa grande portabilité ce qui en fait un argument de poids d'un point de vue mobilité et interopérabilité.

Les langages choisis pour la partie apprentissage automatique et la partie multi-agents étant respectivement *Python* et *JAVA*, il nous fallait trouver une manière de faire la liaison entre les classifieurs et les futures agents. Une solution aurait pu être de simplement sélectionner un langage et abandonner l'autre dans un soucis de comptabilité. Nous n'avons pas opté pour cette approche puisqu'aucun *framework* multi-agents *Python* ne se relevait être encore maintenu. À l'inverse, se reporter sur le langage *JAVA* signifiait de devoir abandonner les nombreuses bibliothèques spécialisés en apprentissage automatique de *Python*. Bien que techniquement moins compliqué que la création d'un paradigme multi-agents, le codage des différents algorithmes d'apprentissage automatique représentaient un effort conséquent en terme de temps. Une des premières solutions que nous envisageâmes fut l'emploi de la bibliothèque *Jython* qui propose une implémentation du *Python* en *JAVA*. Cependant cette approche se révéla être une impasse puisque des bibliothèques comme *Tensorflow* convertissent certaines parties du code en C pour accélérer l'exécution du programme. Ce point empêchait donc tout simplement la compilation de nos algorithmes d'apprentissage. De plus, la dernière mise à jour de Jython datait de Mai 2015 ce qui laissait à penser que la bibliothèque n'était plus maintenue. Après de plus amples recherches, c'est finalement la bibliothèque *JPMML* développée par Villu Ruusmann de l'université Tartu, Estonie qui semblait correspondre à nos critères. *JPMML* offre la possibilité de convertir et d'enregistrer les modèles développés en *Python* en format *PMML* (*Predictive Model Markup Language*). Basé sur le *XML*, le *PMML* a été développé en 1998 et se spécialise dans la description de modèles prédictifs. Grâce à ce format, il nous est donc possible d'entraîner nos modèles en *Python* pour ensuite récupérer et conserver les caractéristiques des modèles. Les méthodes implémentées dans *JPMML* nous permettent ensuite de récupérer, d'interpréter et de reconstruire les classifieurs ainsi enregistrés dans un environnement *JAVA*.

Cette présentation des différentes technologies étant terminée, nous allons pouvoir nous concentrer plus avant sur certains détails comme les conventions de communication *FIPA*, la plateforme multi-agents proposée par *JADE* ou encore le format *PMML*.

Les standards de communication *FIPA* se basent sur la théorie des actes de langage. Cette dernière énonce que toute parole peut se définir comme l'accomplissement de plusieurs actes de langage, appelés *performatives* dans *JADE*. Ces actes expriment les intentions véhiculés par les messages du locuteur comme informer, demander ou refuser. Le nombre d'actes de langage mis à disposition des agents pour leur communication s'élève à 22. Une propriété attendue de tous les agents est la capacité à pouvoir traiter n'importe quelle requête dont la *performative* appartient à cette liste et d'être capable de répondre avec un acte de langage "Non compris" le cas échéant. Les normes *FIPA* peuvent se subdiviser en 7 couches différentes. La première couche définit le protocole de transport utilisé pour véhiculer les messages de l'agent. Un exemple de protocole de communication est *HTTP*. La couche suivante permet de choisir la manière dont seront encodés les mes-

sages ; *FIPA* permettant de structurer les messages sous la forme de String ou de *XML*. Tous les paramètres nécessaires à l'envoi du message sont contenus dans la couche 3 et on y retrouve l'adresse de l'envoyeur, du receveur ou encore de la *deadline* pour l'envoi d'une réponse. La quatrième couche définit l'ontologie nécessaire à la compréhension des messages. Une ontologie définit une sorte de "grammaire" permettant d'extraire de manière pragmatique les informations contenues dans le corps du dit message. Ce dernier est contenu dans la couche suivante et peut prendre n'importe quelle forme. Les normes *FIPA* définissent toutefois un certain nombre de formules logiques comme "ou", "et" etc. qui sont mises à la disposition des agents. L'acte de langage est quant à lui défini dans la couche supérieure et contient une des 22 *performatives* standards de *FIPA*. Enfin, la couche finale indique dans le cadre de quel protocole s'inscrit le message envoyé. Par protocole, on signifie une séquence d'interactions liées les unes aux autres. L'acte d'émettre une requête peut être considéré comme un protocole puisque l'on attend une réponse et il existe une éventualité de négociation. *FIPA* définit un certain nombre de protocoles tels que le protocole d'interaction *Contract Net* décrit dans la figure 3.6. Ce protocole décrit un agent initiateur qui souhaite qu'une action soit accomplie par d'autres agents. Ce *workflow* permet de suivre et comprendre les actions que suivront les deux partis dans le cadre de cette interaction. Les normes *FIPA* se révèlent donc être des approches efficaces afin de mettre en place des communications entre nos agents.

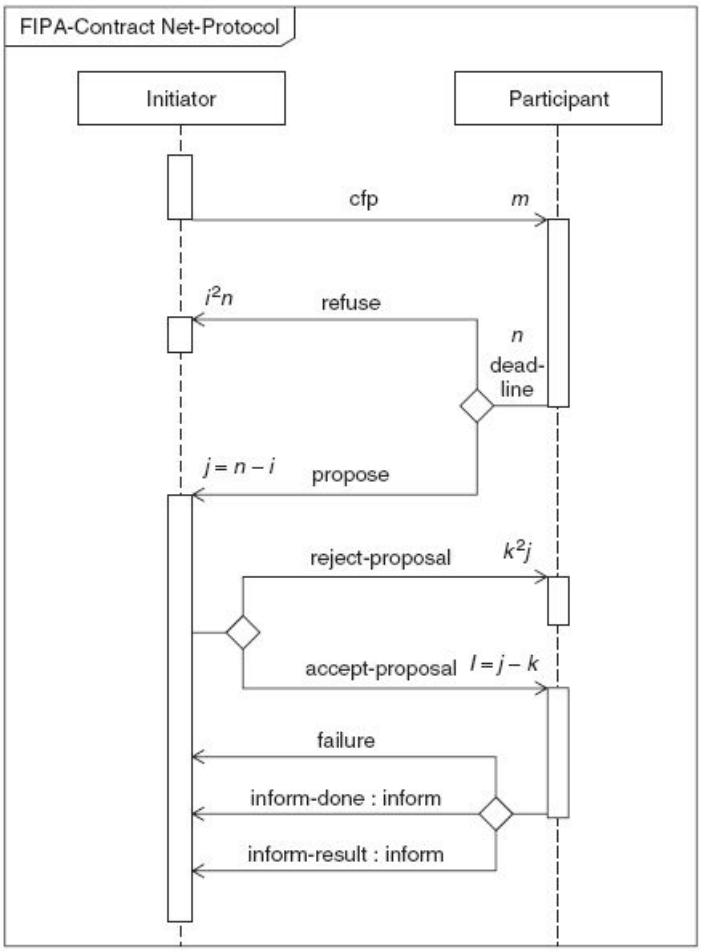


FIGURE 3.6 – Protocole d'interaction *Contract Net* [56]

En plus de ces conventions, *JADE* implémente un certain nombre de concepts en-

tourant les systèmes multi-agents et leur gestion. Ainsi, dans *JADE*, les agents vivent et meurent au sein d'un environnement connu sous le nom de *Agent Plateform* (*AP*). Les agents ne sont cependant cloisonnés à cette plateforme et peuvent éventuellement migrer d'une plateforme à une autre si cela est nécessaire. Cette plateforme encapsule les machines, les systèmes d'exploitation ainsi que les différents éléments nécessaires à la gestion du système. L'*Agent Management System* (*AMS*) fait partie de ces éléments et se charge de toutes les tâches relatives à la prise en charge des agents. Il tient de part cette position un rôle important au sein du système. L'*AMS* supervise toutes les opérations relatives à la création des agents ou leur suppression. C'est également à lui que doivent s'adresser les agents afin d'obtenir leurs *AID*s, les identifiants uniques utilisés pour leur distinction au sein de la plateforme. Finalement la gestion des migrations à partir de et vers la plateforme échoie également à ce système. Le *Directory Facilitator* (*DF*) est un système servant de pages jaunes pour l'ensemble des agents. Il regroupe l'ensemble des services proposés par chacun des agents et leurs permet de rechercher rapidement l'adresse des agents fournissant un service d'intérêt. Les agents sont libres de s'enregistrer ou de se dés-enregistrer du *DF* à leur gré. Contrairement à l'*AMS* qui n'existe qu'en un unique exemplaire par plateforme, il peut exister n'importe quel nombre de *DF*s sur une même plateforme. Enfin le *Message Transport Service* (*MTS*) est le service fourni aux agents de la plateforme pour véhiculer leurs messages (suivant les normes *FIPA*) les uns aux autres. Le système proposé par *JADE* peut ainsi être modélisé par la figure 3.7 ci-dessous.

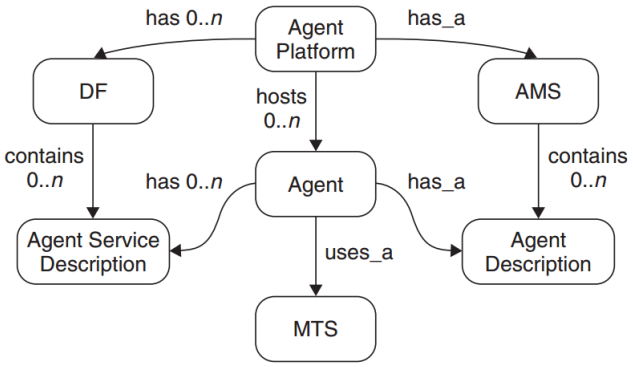


FIGURE 3.7 – Ontologie du système de gestion des agents [56]

Le format PMML décrit un modèle en utilisant un certain nombre d'éléments dont le début et la fin sont marqués par des balises portant le nom des dits éléments. Le *Header* contient les informations générales sur le modèle comme l'*application name* faisant référence au nom du fichier *PMML* et le *timestamp* indiquant sa date de création. Le *Data Dictionnary* contient l'ensemble des champs existants ainsi que leurs caractéristiques. Ces caractéristiques sont le nom, le type (Int,Float ...) et la nature (continu, catégorique ...) du champ. La distinction entre les variables d'entrées et de sorties n'est effectuée que plus tard dans la fichier. Il existe une balise *Data Transformation* qui permet de définir des opérations de normalisation ou de discrétisation des données par exemple. Nous n'avons pas rencontré cette dernière dans le cadre de nos travaux puisque les tâches de prétraitement furent toutes effectuées en amont de la rencontre avec les modèles. Finalement le modèle est décrit dans le *Mining Schema*. Ce dernier contient des informations sur le type d'algorithme auquel appartient le modèle, la nature de la prédiction (classification, régression ..) et des caractéristiques propres à chaque algorithme. Dans le cas de l'arbre de décision, on peut trouver le critère de sélection des nœuds et pour le *SVM*, la méthode

de classification (*OneAgainstOne..*). Ensuite est décrit le *mining schema*. Il nous permet de déterminer les champs réellement utilisés en entrée du modèle et les champs de sortie. Ces derniers sont identifiés par la présence de "*UsageType=target*" à côté de leurs noms. Finalement, le dernier élément du fichier *PMML* est constitué de la description de la structure et des caractéristiques du modèle. Cette description varie en fonction des différents algorithmes. Pour les arbres de décision, les nœuds sont marqués par des *nodes* marqués par le champs inspecté et la condition appliquée pour la décision. Ces *nodes* sont indentés afin de déterminer les nœuds pères et fils. C'est donc ainsi que nous transcrirons nos modèles afin de les recréer en *JAVA* par l'intermédiaire de *JPMML*.

Le développement d'un système multi-agents se révèle être une tâche complexe, nécessitant beaucoup de temps et de ressources. C'est d'autant plus vrai lorsque l'on recherche une véritable capacité de réflexion de la part des agents. Cela nécessite non seulement de définir leurs caractéristiques mais également de traduire leurs besoins et leurs buts. Sans cela, ils ne seront pas capables de raisonner et de s'adapter pour atteindre leurs objectifs, c'est à dire ici suppléer au mieux leurs partenaires humains. Afin d'obtenir cette forme d'intelligence, il est nécessaire de définir des ontologies et des protocoles qui permettront aux agents de pouvoir communiquer les uns avec les autres. Cette tâche ne nous semblait pas envisageable dans le laps de temps dont nous disposions. Ainsi, notre objectif fut de créer une plateforme et d'implémenter de agents basés sur les profils décrits précédemment afin de s'en servir pour valider des cas d'usages. Il nous était toutefois possible d'observer la pertinence de notre architecture sur des cas classiques du processus de consultation semi-urgente. Nous voulions vérifier si il n'y avait pas de propriétés, de comportements ou d'actions nécessaires à accomplissement des tâches qui auraient été négligées. Les agents que nous avons implémentés sont donc régis par des réflexes, dont les réponses sont conditionnées par l'état de l'agent et le stimuli extérieur. Leurs différents comportements peuvent être modélisés sous la forme d' automates.

La création de nos agents s'est donc basée sur les comportements standards décrits auparavant. Nos agents disposent d'un certain degré d'indépendance puisque le choix de l'interlocuteur est effectué par l'agent et ce de manière dynamique. Ainsi, le déroulement des interactions ne prend pas en compte d'interlocuteurs données et le choix des interlocuteurs repose sur les épaules de l'agent ainsi que du contexte dans lequel il évolue. Pour ce faire, nous avons exploiter la présence du *DF*. À la création de nos agents, ils enregistrent leur identité et les services qu'ils peuvent procurer auprès de lui. Pour les services dont ils auront besoin, il ne leur reste plus qu'à contacter le *DF* le moment venu. Ainsi, l'agent *Planner* de la plateforme est capable de trouver automatiquement les différents médecins dans le "service" et de les associer avec une journée de travail. À l'inverse, le *Planner* s'étant enregistré auprès du *DF*, une *Nurse* est capable de faire des requêtes pour lui demander une vacation possible pour une consultation. Il est à noter que le patient est dans le cadre de cette simulation muni d'un comportement semi-aléatoire. Lorsqu'on lui propose un horaire de rendez-vous, il choisit équiprobablement si il accepte ou refuse. Deux profils ne sont pas systématiquement inclus dans la simulation, le *Technician* et l'*External*. En effet, le premier ne possède pas d'influence sur le déroulement des interactions puisqu'il ne fait que transmettre un message tandis que le second se trouve en fin de processus et ne constitue pas un intervenant direct. Une fois la mise en place de nos différents types d'agents effectuée, nous nous sommes appuyés sur le *GUI* mis à disposition par *JADE* afin d'observer le comportement de nos agents. Ce *GUI* permet en premier lieu à l'utilisateur d'accéder à l'*AMS*. Il nous est alors possible gérer de la création et la suppression des agents de la plateforme. Un exemple de *GUI* est présenté sur la figure

3.8. Dans cette figure, on peut observer des agents appelés *Sniffer*s. Les *Sniffers* sont des outils nous permettant d'observer les messages envoyés et reçus par un ou des agents donnés. On peut alors obtenir une trace nous permettant d'observer le déroulement du processus de consultation semi-urgente. Deux de ces traces seront présentées plus loin, l'une décrivant la redirection du malade vers un autre service et l'autre la prise de rendez-vous pour une consultation semi-urgente. Le *GUI* étant parfois capricieux, il nous a été impossible de capturer le message d'initiation envoyé par le patient et qui déclenche le début des interactions.

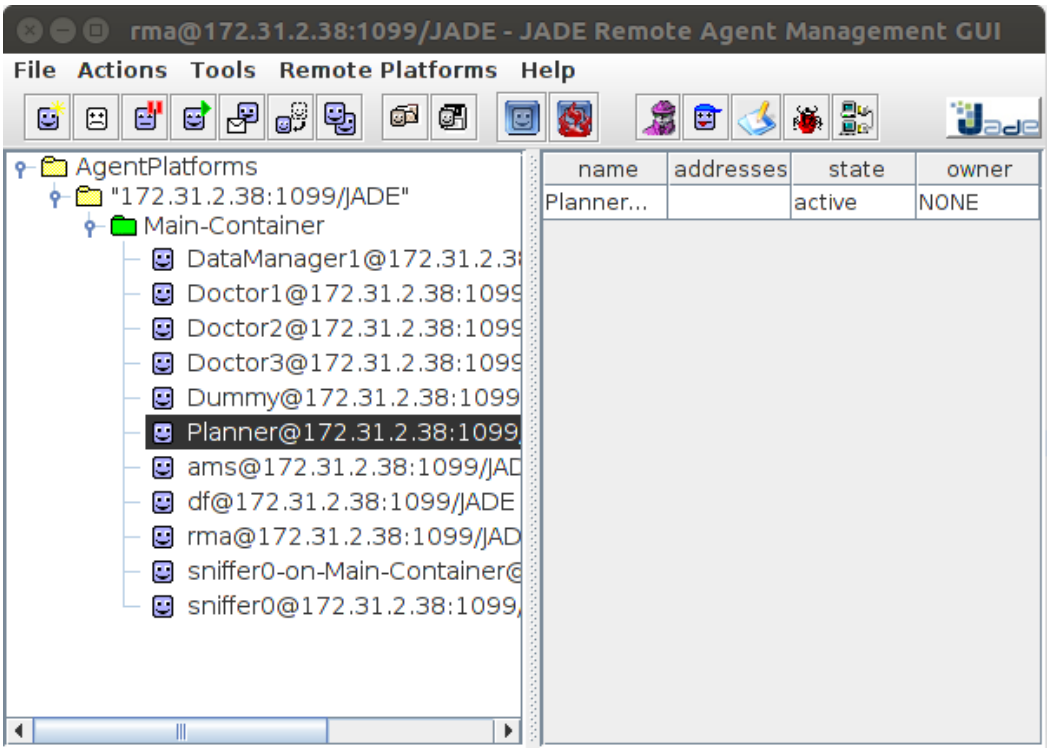


FIGURE 3.8 – Exemple de *GUI* proposé par *JADE*

La figure 3.9 nous montre le déroulement des échanges des différents acteurs du système. On suppose que l'agent du patient est rentré en contact avec l'agent de l'infirmier suite à sa réponse à l'appel téléphonique. L'agent *Nurse* confirme sa disponibilité à l'agent du patient en 1. Le patient transmet donc les informations qu'il a sa disposition en 2 et l'infirmier pose les questions des critères manquants dans le formulaire. Ici, on peut voir que le *Nurse* envoie un message *REQUEST* en 3 suivi d'un nouveau message d'informations de la part du *Patient*. Cette réaction est générée par un temps de réponse considéré comme trop long par l'agent et qui essaie de relancer son interlocuteur. Il n'est malheureusement pas possible de visualiser ce qu'il se passe entre le message 4 et le message 5 puisque le *GUI* ne permet pas d'accéder aux processus internes de l'agent. Durant cette période, les données transmises par le patient sont soumis au classifieur de l'agent *Nurse*. Ici, le résultat de la classification est la redirection du patient vers son médecin traitant. Le message est transmis à l'agent de ce médecin, un *external* que nous avons ici appelé *GeneralP*, puis l'infirmier et son agent quitte le processus. L'absence de message 6 pour informer l'agent *Patient* de cette décision est un choix délibéré. Nous sommes parti du principe que l'agent du médecin traitant enverrait un message et prendrait directement la relève concernant la prise en charge du patient. Ce choix n'est cependant pas irrévocable et sa finalité dépendra beaucoup des conditions observées dans les situations réelles.

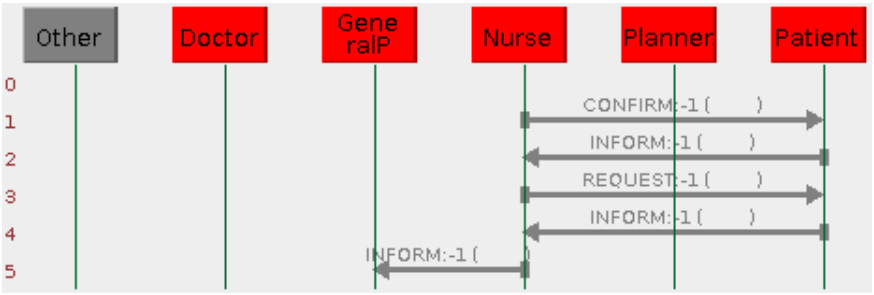


FIGURE 3.9 – Exemple de *GUI* proposé par *JADE*

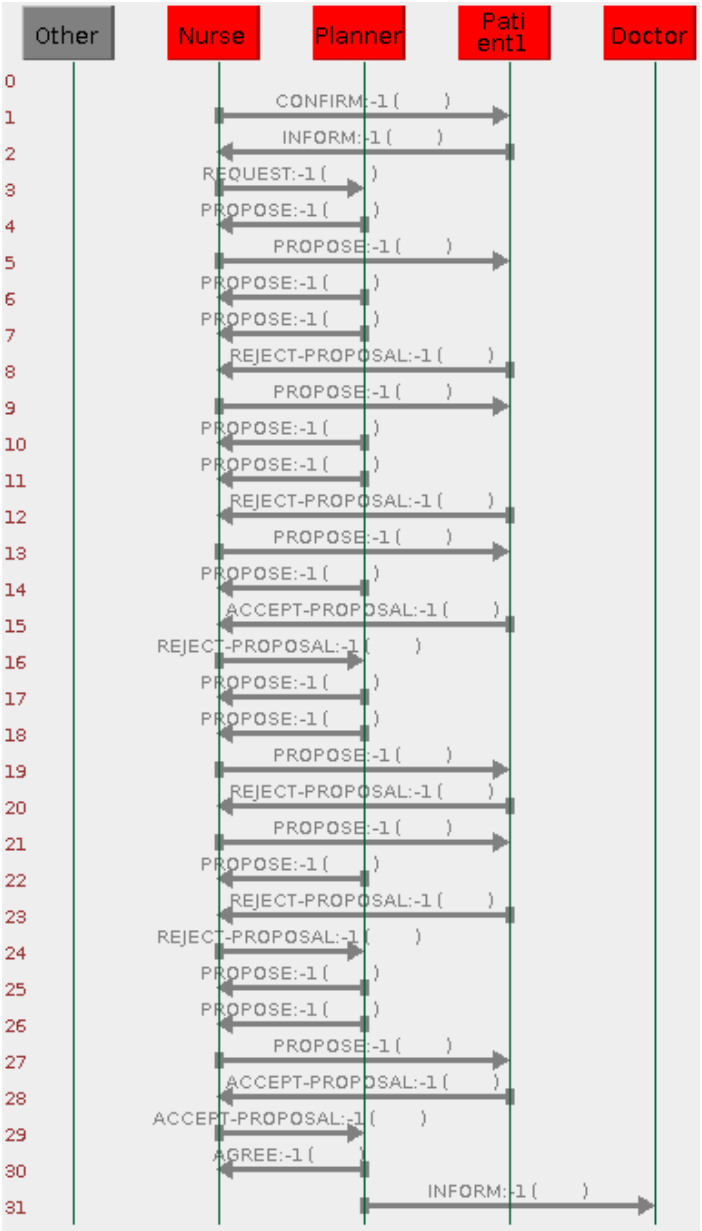


FIGURE 3.10 – Exemple de *GUI* proposé par *JADE*

La trace proposée par la figure 3.10 commence de manière similaire à celle de la figure précédente. Ici, l'envoi d'informations a été assez rapide pour que la *Nurse* n'ait

pas à envoyer de message de relance et il a également jugé le patient comme admissible. Malheureusement pour nos observations, lorsque le *GUI* entre en jeu dans ce protocole, l'envoi des messages semble particulièrement perturbé. Ainsi, ces interactions prennent quelques secondes à se terminer lorsque l'on dessine une trace tandis qu'elles s'effectuent en quelques millièmes de seconde sans. Afin de confirmer le résultat de ces échanges, nous avons donc dû effectuer des vérifications sur les variables et valeurs des messages afin de s'assurer de la validité du protocole. Toutefois, la figure nous permet d'observer le motif d'interaction désiré à savoir une requête effectué auprès de l'agent *Planner* qui renvoie une proposition. Cette proposition est ensuite relégué au *Patient* qui décide ou non d'accepter la date indiquée. Un cas de refus prend la forme de l'enchaînement [23,24,25] où une nouvelle proposition est effectuée. Un cas d'acceptation prend la forme de l'enchaînement [28,29,30,31] où le *Doctor* est automatiquement informé par le *Planner* de la consultation programmée.

Cette présentation conclut cette partie Implémentation et nous a permis de valider deux cas d'usage de la consultation semi-urgente. La conception d'un système multi-agents capable de s'adapter à une plus large gamme de cas d'usage nécessite une modélisation plus générique au niveau de l'automate associé à chaque agent. Enfin, une fois l'implémentation d'un tel système terminé, il sera possible d'évaluer son apport niveau applicatif pour le *CHSF*. Nous pouvons désormais nous tourner vers les discussions et les conclusions de ce mémoire.

Chapitre 4

Perspectives et conclusion

Comme nous avons pu l'expliquer précédemment, ce stage se situe dans le cadre d'un projet global cherchant à concevoir un écosystème permettant de simuler le fonctionnement d'un environnement hospitalier. L'objectif est double puisque l'on cherche à améliorer le parcours de soins des patients mais également la qualité des conditions de travail du personnel. Par conséquent, l'architecture que nous proposons dans ce mémoire, bien que constituant un solide squelette, sera amenée à évoluer avec le temps et avec le projet de la *CSIC*. Ainsi notre agent *Doctor* possède deux facettes sujettes à changement. La première concerne son classifieur. Aujourd'hui, l'agent tel que décrit dans notre architecture n'est pas implémenté faute de données. Il ne peut donc effectuer de prédiction sur l'hospitalisation tant que nous sommes touchés par cette pénurie d'informations. L'imposition d'un standard de remplissage de formulaire se révélera aussi nécessaire afin de faciliter leur future exploitation. Cependant, les performances des modèles de l'agent *Nurse* laisse présager des scores similaires et donc respectables. Bien que n'ayant pas encore abouti, il est dans nos projets d'utiliser les historiques des patients conservés par l'hôpital. Ainsi l'on pourra enrichir la phase d'apprentissage ainsi qu'envisager un sujet de prédiction plus poussé, le diagnostic. Ce point constitue donc la seconde facette dont nous parlions. Notre *Doctor* s'intéressera à déterminer la nécessité d'une hospitalisation du patient mais ne pourra fournir que difficilement des justifications pour sa décision. C'est ici un problème inhérent à beaucoup d'algorithmes d'apprentissage automatique. Nous pensons pouvoir contourner ce problème en orientant le sujet de prédiction vers le diagnostic médical.

D'une part, cela permettrait d'augmenter la valeur de l'agent aux yeux du praticien en lui attribuant un sujet de prédiction plus attrayant. Le médecin pourrait être guidé dans ses choix s'il connaît les signes ou maladies repérés par l'agent. Ces informations apparaissent plus enrichissantes que le sujet de prédiction initial. Cependant, ce dernier ne sera pas abandonné mais bien renforcé par ce nouvel aspect. Pour le médecin, si le patient souffre de tel ou tel mal, cela oriente forcément son choix pour le pronostic du patient. D'une manière similaire, notre agent s'appuiera sur les résultats de ses diagnostics et les caractéristiques du patient pour appuyer son verdict final concernant l'hospitalisation du patient. Dans l'éventualité d'une extension du système à d'autres services, une question est soulevée. Est-il plus avantageux que les agents disposent d'un système de diagnostics génériques leur permettant de prédire tout un panel de maladies ou plutôt d'en faire des experts selon le service dans lequel ils existent ? Dans le cas d'experts, il serait intéressant que les agents ne soient que des squelettes génériques et qu'ils puissent assimiler le rôle d'archétype. Ainsi, en s'enregistrant en tant que cardiologue et pédiatre, un agent disposerait de l'ensemble des modèles, des ontologies et des protocoles mis à

disposition à chacun de ces deux profils. Cela demanderait toutefois un travail conséquent sur la manière de joindre les comportements de plusieurs profils de manière cohérente. La prédiction des traitements et de leurs réajustements consiste également en une piste intéressante puisque le changement de médication est souvent utilisé dans le but de stabiliser les patients insuffisants cardiaques.

Puisque nous avons travaillé sur la détection de vraies et de fausses alarmes cardiaques, le rôle de l'agent *Patient* lui-même pourrait se voir étendu à l'observation des signes vitaux de son utilisateur. Associé à un cardiologue du *CHSF*, cela permettrait de faciliter le suivi des patients et la télémédecine. Concernant la mort d'un agent *DM* et l'impossibilité pour un nouvel agent d'obtenir un modèle, un mécanisme que nous avons envisagé serait la capacité à demander à ses pairs de fournir leurs modèles. Cela s'incorporerait parfaitement dans le cadre d'une collaboration plus étroite entre les différents agents qui, pour le moment, coexistent et échangent simplement des requêtes au sein d'un même environnement. Cette coopération ne semble cependant pertinente que lorsque les agents expliqués ne partagent pas toutes les mêmes connaissances. Ainsi dans un système où il n'existe pas d'experts, l'utilité de communiquer avec des agents en dehors de protocoles semble faible. Pour ce qui est de la sécurité des informations, l'usage de messages semble être risqué, en particulier si les agents sont sur des plateformes mobiles. Une solution pourrait être d'isoler la base de données et les agents *DM*s dans un environnement différent du projet de la *CSIC*. Les agents voulant accéder aux informations devraient alors migrer dans cet environnement où les *DM*s feraient offices de "douaniers" contrôlant le flux d'accès. Les données ne se révéleraient disponibles à l'agent et à son utilisateur qu'au sein de cet environnement, ce qui éviterait des risques de fuite dans l'environnement de travail moins sécurisé. Les pistes de développement que nous avons énoncé ici ne constituent bien sûr pas une liste exhaustive.

Nous avons pu dans ce mémoire présenter une architecture multi-agents modélisant le projet de la *CSIC*. Cette architecture emploie bien sûr des mécanismes propres au domaine multi-agents mais intègre également des algorithmes d'apprentissage automatique pour créer un outil de soutien et d'aide à la décision. Cette architecture intègre sous forme d'agents l'ensemble des acteurs majeurs de l'écosystème hospitalier. D'autres agents ont également été imaginés afin d'automatiser et/ou simplifier des actions comme la prise de rendez-vous. Parmi ces agents, les agents *Doctor* et *Nurse* possèdent un statut à part puisqu'ils intègrent des classifieurs pour aider les décisions de leurs partenaires humains. Ces algorithmes ont été testés et évalués sur les données mises à notre disposition par le *CHSF*. Ils obtiennent des performances respectables qui se verront sûrement améliorées par un enrichissement des bases d'entraînement et des paramétrages plus pointus. Une implémentation du système a été effectuée afin de répondre à deux cas d'usages de la consultation semi-urgente. Les agents se basent en partie sur des automates afin d'agir et intègrent grâce à la bibliothèque *JPMML* les modèles développés en amont. Les interactions entre les agents et l'accomplissement de leurs objectifs se sont révélées conformes à nos attentes. La continuité directe de nos travaux consisterait à se plonger plus avant dans la complexité des deux domaines employés afin de créer une implémentation disposant d'ontologies pour atteindre un niveau d'indépendance et d'adaptabilité plus avancées.

Bibliographie

[1] A. P. Ambrosy, G. C. Fonarow, J. Butler, O. Chioncel, S. J. Greene, M. Vaduganathan, S. Nodari, C. S. Lam, N. Sato, A. N. Shah, *et al.*, "The global health and economic burden of hospitalizations for heart failure : lessons learned from hospitalized heart failure registries," *Journal of the American College of Cardiology*, vol. 63, no. 12, pp. 1123–1133, 2014.

[2] M. R. Cowie, S. D. Anker, J. G. Cleland, G. M. Felker, G. Filippatos, T. Jaarsma, P. Jourdain, E. Knight, B. Massie, P. Ponikowski, *et al.*, "Improving care for patients with acute heart failure : before, during and after hospitalization," *ESC Heart Failure*, vol. 1, no. 2, pp. 110–145, 2014.

[3] J. Butler, E. Braunwald, and M. Gheorghiade, "Recognizing worsening chronic heart failure as an entity and an end point in clinical trials," *Jama*, vol. 312, no. 8, pp. 789–790, 2014.

[4] C. M. O'connor, A. B. Miller, J. E. Blair, M. A. Konstam, P. Wedge, M. C. Bahit, P. Carson, M. Haass, P. J. Hauptman, M. Metra, *et al.*, "Causes of death and rehospitalization in patients hospitalized with worsening heart failure and reduced left ventricular ejection fraction : results from efficacy of vasopressin antagonism in heart failure outcome study with tolvaptan (everest) program," *American heart journal*, vol. 159, no. 5, pp. 841–849, 2010.

[5] Y. LeCun, "The mnist database of handwritten digits," *http ://yann. lecun. com/exdb/mnist/*, 1998.

[6] S. R. Safavian and D. Landgrebe, "A survey of decision tree classifier methodology," *IEEE transactions on systems, man, and cybernetics*, vol. 21, no. 3, pp. 660–674, 1991.

[7] L. Rokach and O. Maimon, *Data mining with decision trees : theory and applications*. World scientific, 2014.

[8] M. A. Hearst, S. T. Dumais, E. Osuna, J. Platt, and B. Scholkopf, "Support vector machines," *IEEE Intelligent Systems and their applications*, vol. 13, no. 4, pp. 18–28, 1998.

[9] I. Steinwart and A. Christmann, *Support vector machines*. Springer Science & Business Media, 2008.

[10] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[11] J. Schmidhuber, "Deep learning in neural networks : An overview," *Neural networks*, vol. 61, pp. 85–117, 2015.

[12] M. Wooldridge, "Intelligent agents : The key concepts," *Multi-Agent-Systems and Applications*, vol. 2322, pp. 3–43, 2001.

[13] V. R. Lesser, "Cooperative multiagent systems : A personal view of the state of the art," *IEEE Transactions on knowledge and data engineering*, vol. 11, no. 1, pp. 133–142, 1999.

[14] S. Kraus, "Negotiation and cooperation in multi-agent environments," *Artificial intelligence*, vol. 94, no. 1-2, pp. 79–97, 1997.

[15] E. David and E. Manisterski, "Strategy proof mechanism for complex task allocations in prior consent for subtasks completion environment," in *Proceedings of the 2013 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)-Volume 02*, pp. 209–215, IEEE Computer Society, 2013.

[16] T. Finin, R. Fritzson, D. McKay, and R. McEntire, "Kqml as an agent communication language," in *Proceedings of the third international conference on Information and knowledge management*, pp. 456–463, ACM, 1994.

[17] A. Fipa, "Fipa acl message structure specification," *Foundation for Intelligent Physical Agents, http ://www. fipa. org/specs/fipa00061/SC00061G. html (30.6. 2004)*, 2002.

[18] J. G. Anderson, "Surgical training, error," *Jama*, vol. 291, no. 14, pp. 1775–1776, 2004.

[19] A. Brown and D. A. Patterson, "To err is human," in *Proceedings of the First Workshop on evaluating and architecting system dependability (EASY'01)*, 2001.

[20] L. K. Barger, N. T. Ayas, B. E. Cade, J. W. Cronin, B. Rosner, F. E. Speizer, and C. A. Czeisler, "Impact of extended-duration shifts on medical errors, adverse events, and attentional failures," *PLoS medicine*, vol. 3, no. 12, p. e487, 2006.

[21] G. Neale, M. Woloshynowych, and C. Vincent, "Exploring the causes of adverse events in nhs hospital practice," *Journal of the Royal Society of Medicine*, vol. 94, no. 7, pp. 322–330, 2001.

[22] N. Menachemi and T. H. Collum, "Benefits and drawbacks of electronic health record systems," *Risk management and healthcare policy*, vol. 4, p. 47, 2011.

[23] S. Palaniappan and R. Awang, "Intelligent heart disease prediction system using data mining techniques," in *Computer Systems and Applications, 2008. AICCSA 2008. IEEE/ACS International Conference on*, pp. 108–115, IEEE, 2008.

[24] I. Kononenko, "Machine learning for medical diagnosis : history, state of the art and perspective," *Artificial Intelligence in medicine*, vol. 23, no. 1, pp. 89–109, 2001.

[25] R. Bellazzi and B. Zupan, "Predictive data mining in clinical medicine : current issues and guidelines," *International journal of medical informatics*, vol. 77, no. 2, pp. 81–97, 2008.

[26] R. Das, I. Turkoglu, and A. Sengur, "Effective diagnosis of heart disease through neural networks ensembles," *Expert systems with applications*, vol. 36, no. 4, pp. 7675–7680, 2009.

[27] P. C. Austin, J. V. Tu, J. E. Ho, D. Levy, and D. S. Lee, "Using methods from the data-mining and machine-learning literature for disease classification and prediction : a case study examining classification of heart failure subtypes," *Journal of clinical epidemiology*, vol. 66, no. 4, pp. 398–407, 2013.

[28] Z. C. Lipton, D. C. Kale, C. Elkan, and R. Wetzell, "Learning to diagnose with lstm recurrent neural networks," *arXiv preprint arXiv :1511.03677*, 2015.

[29] T. Pham, T. Tran, D. Phung, and S. Venkatesh, "Deepcare : A deep dynamic memory model for predictive medicine," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pp. 30–41, Springer, 2016.

[30] E. Choi, M. T. Bahadori, A. Schuetz, W. F. Stewart, and J. Sun, "Doctor ai : Predicting clinical events via recurrent neural networks," in *Machine Learning for Healthcare Conference*, pp. 301–318, 2016.

[31] R. Miotto, L. Li, B. A. Kidd, and J. T. Dudley, "Deep patient : An unsupervised representation to predict the future of patients from the electronic health records," *Scientific reports*, vol. 6, p. 26094, 2016.

[32] P. Thodoroff, J. Pineau, and A. Lim, "Learning robust features using deep learning for automatic seizure detection," in *Machine Learning for Healthcare Conference*, pp. 178–190, 2016.

[33] S. Nemati, M. M. Ghassemi, and G. D. Clifford, "Optimal medication dosing from suboptimal clinical examples : A deep reinforcement learning approach," in *Engineering in Medicine and Biology Society (EMBC), 2016 IEEE 38th Annual International Conference of the*, pp. 2978–2981, IEEE, 2016.

[34] F. Cheng and Z. Zhao, "Machine learning-based prediction of drug–drug interactions by integrating drug phenotypic, therapeutic, chemical, and genomic properties," *Journal of the American Medical Informatics Association*, vol. 21, no. e2, pp. e278–e286, 2014.

[35] D. C. Kale, Z. Che, M. T. Bahadori, W. Li, Y. Liu, and R. Wetzel, "Causal phenotype discovery via deep networks," in *AMIA Annual Symposium Proceedings*, vol. 2015, p. 677, American Medical Informatics Association, 2015.

[36] T. A. Lasko, J. C. Denny, and M. A. Levy, "Computational phenotype discovery using unsupervised feature learning over noisy, sparse, and irregular clinical data," *PloS one*, vol. 8, no. 6, p. e66341, 2013.

[37] J. J. Oresko, Z. Jin, J. Cheng, S. Huang, Y. Sun, H. Duschl, and A. C. Cheng, "A wearable smartphone-based platform for real-time cardiovascular disease detection via electrocardiogram processing," *IEEE Transactions on Information Technology in Biomedicine*, vol. 14, no. 3, pp. 734–740, 2010.

[38] D. Isern and A. Moreno, "A systematic literature review of agents applied in healthcare," *Journal of medical systems*, vol. 40, no. 2, p. 43, 2016.

[39] R. Cruz-Correia, P. Vieira-Marques, P. Costa, A. Ferreira, E. Oliveira-Palhares, F. Araújo, and A. Costa-Pereira, "Integration of hospital data using agent technologies–a case study," *Ai Communications*, vol. 18, no. 3, pp. 191–200, 2005.

[40] M. D. Rodríguez, J. Favela, A. Preciado, and A. Vizcaíno, "Agent-based ambient intelligence for healthcare," *Ai Communications*, vol. 18, no. 3, pp. 201–216, 2005.

[41] B. G. Silverman, N. Hanrahan, G. Bharathy, K. Gordon, and D. Johnson, "A systems approach to healthcare : agent-based modeling, community mental health, and population well-being," *Artificial intelligence in medicine*, vol. 63, no. 2, pp. 61–71, 2015.

[42] M. Laskowski, B. C. Demianyk, J. Witt, S. N. Mukhi, M. R. Friesen, and R. D. McLeod, "Agent-based modeling of the spread of influenza-like illness in an emergency department : a simulation study," *IEEE Transactions on Information Technology in Biomedicine*, vol. 15, no. 6, pp. 877–889, 2011.

[43] Ö. Kafalı, S. Bromuri, M. Sindlar, T. van der Weide, E. Aguilar Pelaez, U. Schaechtle, B. Alves, D. Zufferey, E. Rodriguez-Villegas, M. I. Schumacher, *et al.*, "Commodity12 : A smart e-health environment for diabetes management," *Journal of Ambient Intelligence and Smart Environments*, vol. 5, no. 5, pp. 479–502, 2013.

[44] C.-J. Su and T.-W. Chu, "A mobile multi-agent information system for ubiquitous fetal monitoring," *International journal of environmental research and public health*, vol. 11, no. 1, pp. 600–625, 2014.

[45] S. P. Ferrando and E. Onaindia, "Context-aware multi-agent planning in intelligent environments," *Information Sciences*, vol. 227, pp. 22–42, 2013.

[46] S. Singh, B. I. Ismail, F. Haron, and C. H. Yong, "Architecture of agent-based healthcare intelligent assistant on grid environment.," in *PDCAT*, pp. 58–61, Springer, 2004.

[47] H. González-Vélez, M. Mier, M. Julià-Sapé, T. N. Arvanitis, J. M. García-Gómez, M. Robles, P. H. Lewis, S. Dasmahapatra, D. Dupplaw, A. Peet, *et al.*, "Healthagents : distributed multi-agent brain tumor diagnosis and prognosis," *Applied Intelligence*, vol. 30, no. 3, pp. 191–202, 2009.

[48] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, *et al.*, "Scikit-learn : Machine learning in python," *Journal of Machine Learning Research*, vol. 12, no. Oct, pp. 2825–2830, 2011.

[49] S. Ansari, A. Belle, and K. Najarian, "Multi-modal integrated approach towards reducing false arrhythmia alarms during continuous patient monitoring : the physionet challenge 2015," in *Computing in Cardiology Conference (CinC), 2015*, pp. 1181–1184, IEEE, 2015.

[50] F. Chollet *et al.*, "Keras," 2015.

[51] J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio, "Theano : A cpu and gpu math compiler in python," in *Proc. 9th Python in Science Conf*, pp. 1–7, 2010.

[52] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, *et al.*, "Tensorflow : Large-scale machine learning on heterogeneous distributed systems," *arXiv preprint arXiv :1603.04467*, 2016.

[53] T. Tieleman and G. Hinton, "Lecture 6.5-rmsprop, coursera : Neural networks for machine learning," *University of Toronto, Tech. Rep*, 2012.

[54] D. Kingma and J. Ba, "Adam : A method for stochastic optimization," *arXiv preprint arXiv :1412.6980*, 2014.

[55] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pp. 315–323, 2011.

[56] F. L. Bellifemine, G. Caire, and D. Greenwood, *Developing multi-agent systems with JADE*, vol. 7. John Wiley & Sons, 2007.

[57] F. Bellifemine, G. Caire, A. Poggi, and G. Rimassa, "Jade : A software framework for developing multi-agent applications. lessons learned," *Information and Software Technology*, vol. 50, no. 1, pp. 10–21, 2008.

Appendices

Annexe A

Organigramme de la consultation semi-urgente

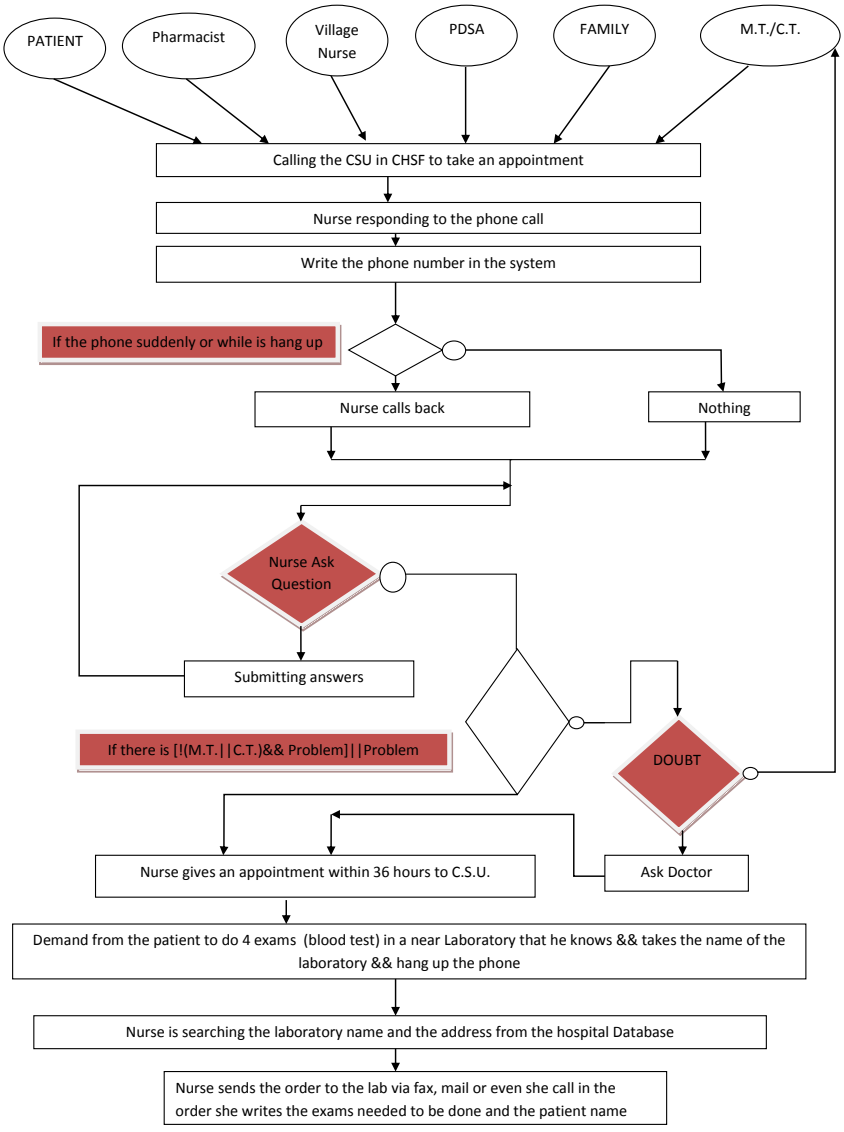


FIGURE A.1 – Page 1 de l'organigramme de la consultation semi-urgente

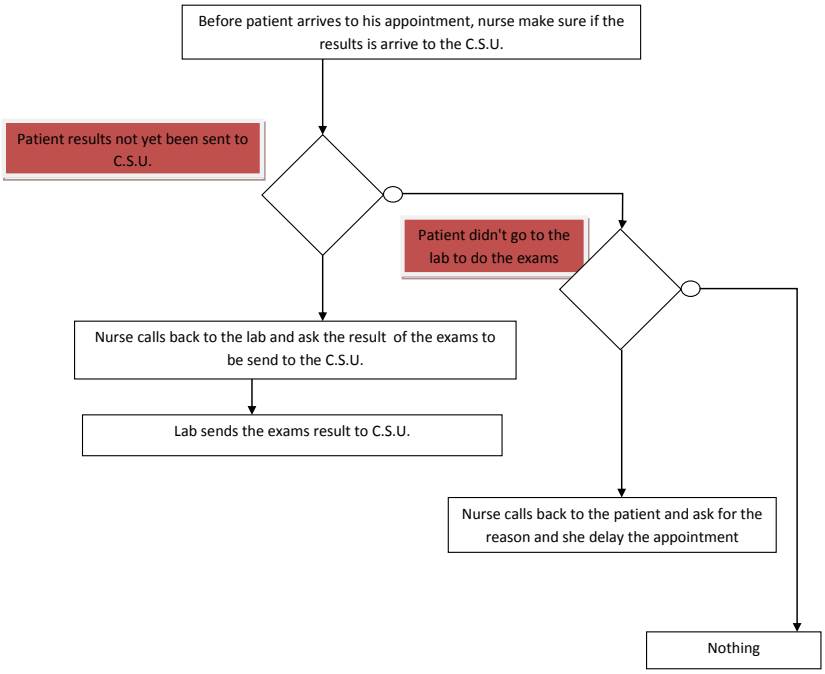
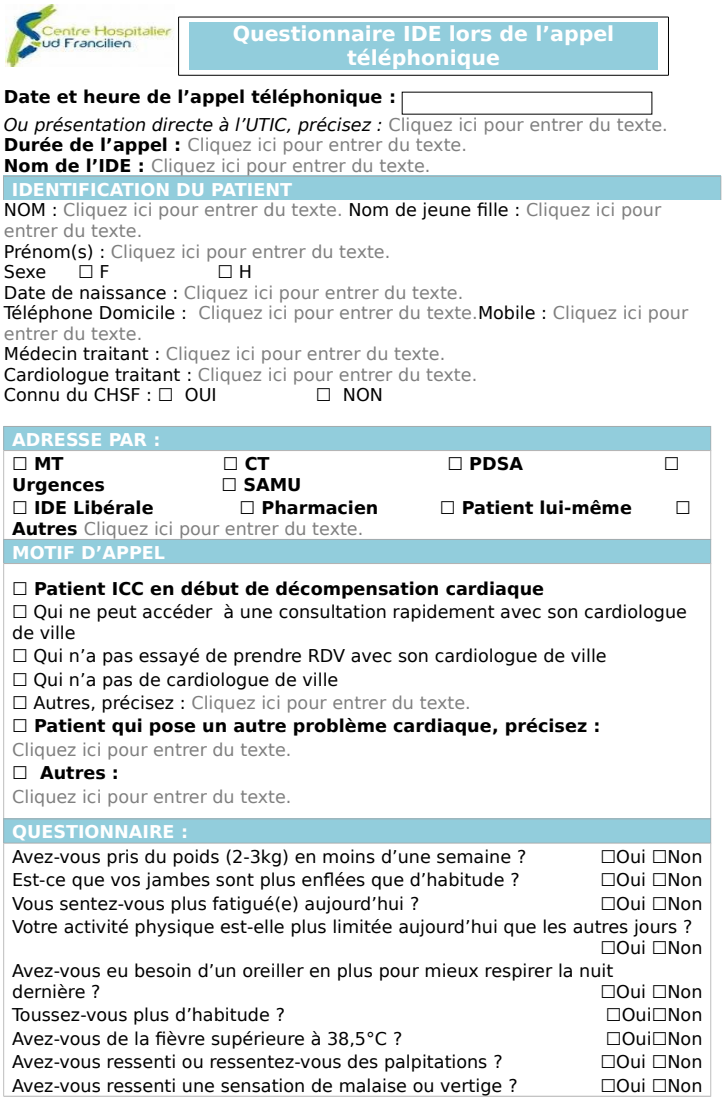


FIGURE A.2 – Page 2 de l'organigramme de la consultation semi-urgente

Annexe B

Questionnaire lors de l'appel téléphonique



Centre Hospitalier Sud Francilien

Questionnaire IDE lors de l'appel téléphonique

Date et heure de l'appel téléphonique :

Ou présentation directe à l'UTIC, précisez : Cliquez ici pour entrer du texte.
Durée de l'appel : Cliquez ici pour entrer du texte.
Nom de l'IDE : Cliquez ici pour entrer du texte.

IDENTIFICATION DU PATIENT

NOM : Cliquez ici pour entrer du texte. Nom de jeune fille : Cliquez ici pour entrer du texte.
Prénom(s) : Cliquez ici pour entrer du texte.
Sexe ☐ F ☐ H
Date de naissance : Cliquez ici pour entrer du texte.
Téléphone Domicile : Cliquez ici pour entrer du texte.Mobile : Cliquez ici pour entrer du texte.
Médecin traitant : Cliquez ici pour entrer du texte.
Cardiologue traitant : Cliquez ici pour entrer du texte.
Connu du CHSF : ☐ OUI ☐ NON

ADRESSE PAR :

☐ **MT** ☐ **CT** ☐ **PDSA** ☐ **Urgences** ☐ **SAMU**
☐ **IDE Libérale** ☐ **Pharmacien** ☐ **Patient lui-même** ☐ **Autres** Cliquez ici pour entrer du texte.

MOTIF D'APPEL

☐ **Patient ICC en début de décompensation cardiaque**
☐ Qui ne peut accéder à une consultation rapidement avec son cardiologue de ville
☐ Qui n'a pas essayé de prendre RDV avec son cardiologue de ville
☐ Qui n'a pas de cardiologue de ville
☐ Autres, précisez : Cliquez ici pour entrer du texte.
☐ **Patient qui pose un autre problème cardiaque, précisez :**
Cliquez ici pour entrer du texte.
☐ **Autres :**
Cliquez ici pour entrer du texte.

QUESTIONNAIRE :

Avez-vous pris du poids (2-3kg) en moins d'une semaine ?	☐Oui ☐Non
Est-ce que vos jambes sont plus enflées que d'habitude ?	☐Oui ☐Non
Vous sentez-vous plus fatigué(e) aujourd'hui ?	☐Oui ☐Non
Votre activité physique est-elle plus limitée aujourd'hui que les autres jours ?	☐Oui ☐Non
Avez-vous eu besoin d'un oreiller en plus pour mieux respirer la nuit dernière ?	☐Oui ☐Non
Toussez-vous plus d'habitude ?	☐Oui☐Non
Avez-vous de la fièvre supérieure à 38,5°C ?	☐Oui☐Non
Avez-vous ressenti ou ressentez-vous des palpitations ?	☐Oui ☐Non
Avez-vous ressenti une sensation de malaise ou vertige ?	☐Oui ☐Non

FIGURE B.1 – Page 1 du questionnaire téléphonique

Centre Hospitalier Sud Francilien

Avez-vous eu la sensation d'avoir un voile noir devant les yeux ? ☐Oui ☐Non

Autres symptômes :

Cliquez ici pour entrer du texte.

PRINCIPAUX ANTECEDENTS

Cliquez ici pour entrer du texte.

NIVEAU D'URGENCE

L'état de santé du patient est compatible avec la CSIC (délai de 36h) Oui ☐

Si oui, date et heure du rdv proposé : Cliquez ici pour entrer du texte.

Si rdv proposé dans un délai supérieur à 36h, justifiez Cliquez ici pour entrer du texte.

Non ☐

Si non, précisez le motif :

☐ CSIC non justifiée ☐ Pas de possibilités d'inscription dans les 36h

Autres motifs, précisez : Cliquez ici pour entrer du texte.

ET Réorientation :

☐ **MT** ☐ **CT** ☐ **Urgences**

☐ **SAMU**

☐ **SOS Médecin** ☐ **Autres** Cliquez ici pour entrer du texte.

PRISE DE DECISION

Avis d'un cardiologue pour la prise de décision Oui ☐Non ☐

Si oui, précisez le nom du cardiologue : Cliquez ici pour entrer du texte.

BIOLOGIE

Laboratoire où est adressée l'ordonnance de prélèvement sanguin :

Nom du Laboratoire : Cliquez ici pour entrer du texte.

Adresse : Cliquez ici pour entrer du texte.

Téléphone : Cliquez ici pour entrer du texte. Fax : Cliquez ici pour entrer du texte.

Ordonnance faxée le : Cliquez ici pour entrer du texte.

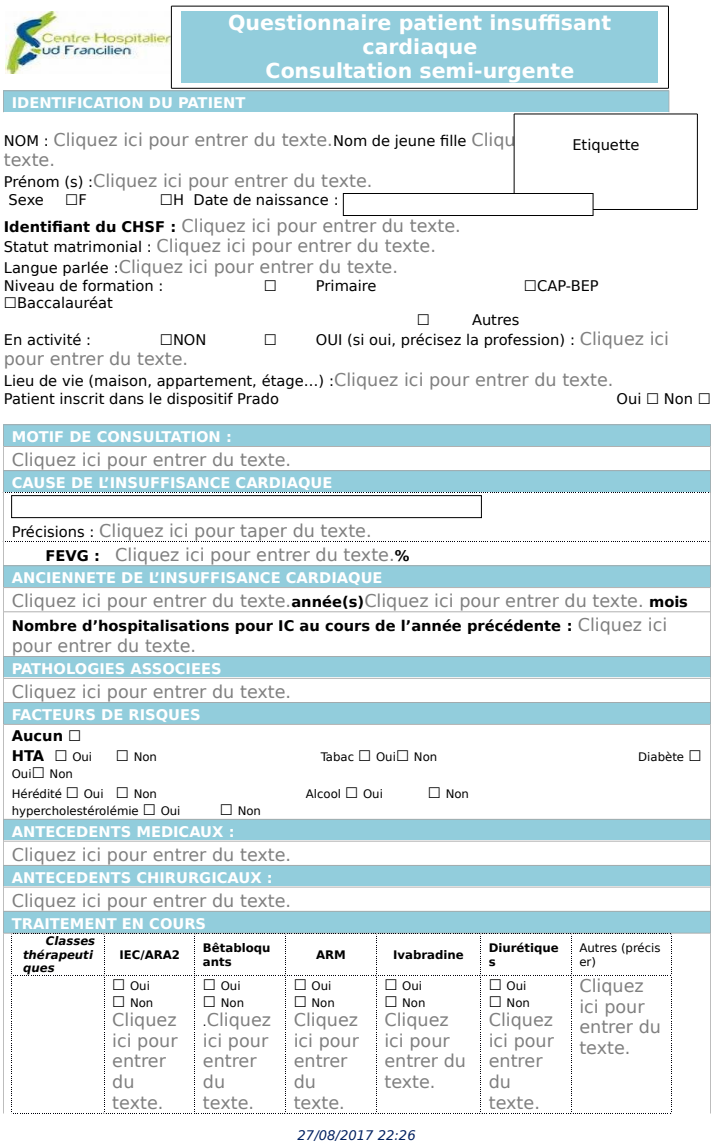
Dossier CHSF

Dossier demandé aux archives Oui ☐Non ☐

FIGURE B.2 – Page 2 du questionnaire téléphonique

Annexe C

Questionnaire lors de la consultation semi-urgente



Centre Hospitalier Sud Francilien

Questionnaire patient insuffisant cardiaque
Consultation semi-urgente

IDENTIFICATION DU PATIENT

NOM : Cliquez ici pour entrer du texte. Nom de jeune fille Cliquez ici pour entrer du texte.

Etiquette

Prénom (s) : Cliquez ici pour entrer du texte.

Sexe ☐F ☐H Date de naissance :

Identifiant du CHSF : Cliquez ici pour entrer du texte.

Statut matrimonial : Cliquez ici pour entrer du texte.

Langue parlée : Cliquez ici pour entrer du texte.

Niveau de formation : ☐ Primaire ☐CAP-BEP ☐Baccalauréat ☐ Autres

En activité : ☐NON ☐ OUI (si oui, précisez la profession) : Cliquez ici pour entrer du texte.

Lieu de vie (maison, appartement, étage...) : Cliquez ici pour entrer du texte.

Patient inscrit dans le dispositif Prado Oui ☐ Non ☐

MOTIF DE CONSULTATION :

Cliquez ici pour entrer du texte.

CAUSE DE L'INSUFFISANCE CARDIAQUE

Précisions : Cliquez ici pour taper du texte.

FEVG : Cliquez ici pour entrer du texte.**%**

ANCIENNETE DE L'INSUFFISANCE CARDIAQUE

Cliquez ici pour entrer du texte.**année(s)** Cliquez ici pour entrer du texte. **mois**

Nombre d'hospitalisations pour IC au cours de l'année précédente : Cliquez ici pour entrer du texte.

PATHOLOGIES ASSOCIEES

Cliquez ici pour entrer du texte.

FACTEURS DE RISQUES

Aucun ☐

HTA ☐ Oui ☐ Non Tabac ☐ Oui ☐ Non Diabète ☐ Oui ☐ Non

Hérédité ☐ Oui ☐ Non Alcool ☐ Oui ☐ Non

hypercholestérolémie ☐ Oui ☐ Non

ANTECEDENTS MEDICAUX :

Cliquez ici pour entrer du texte.

ANTECEDENTS CHIRURGICAUX :

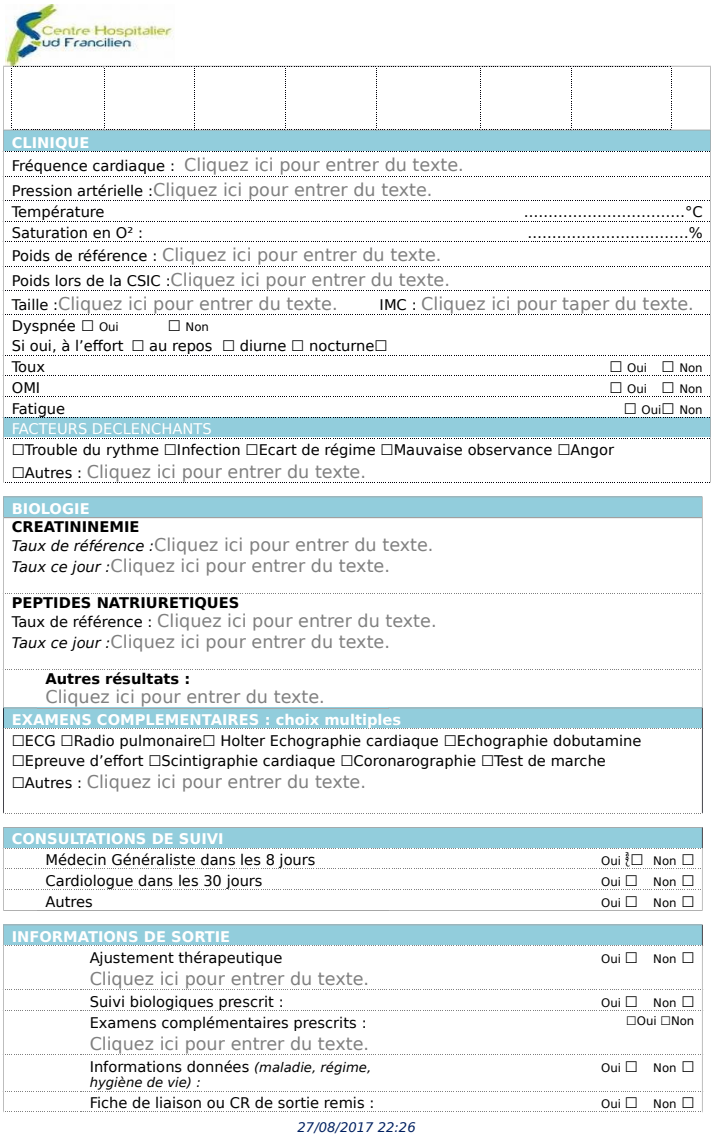
Cliquez ici pour entrer du texte.

TRAITEMENT EN COURS

Classes thérapeutiques	IEC/ARA2	Bêtabloquants	ARM	Ivabradine	Diurétiques	Autres (préciser)
	☐ Oui ☐ Non Cliquez ici pour entrer du texte.	☐ Oui ☐ Non .Cliquez ici pour entrer du texte.	☐ Oui ☐ Non Cliquez ici pour entrer du texte.	☐ Oui ☐ Non Cliquez ici pour entrer du texte.	☐ Oui ☐ Non Cliquez ici pour entrer du texte.	Cliquez ici pour entrer du texte.

27/08/2017 22:26

FIGURE C.1 – Page 1 du questionnaire de consultation semi-urgente



CLINIQUE

Fréquence cardiaque : Cliquez ici pour entrer du texte.

Pression artérielle :Cliquez ici pour entrer du texte.

Température .....°C

Saturation en O^2 : .....%

Poids de référence : Cliquez ici pour entrer du texte.

Poids lors de la CSIC :Cliquez ici pour entrer du texte.

Taille :Cliquez ici pour entrer du texte. IMC : Cliquez ici pour taper du texte.

Dyspnée ☐ Oui ☐ Non

Si oui, à l'effort ☐ au repos ☐ diurne ☐ nocturne☐

Toux ☐ Oui ☐ Non

OMI ☐ Oui ☐ Non

Fatigue ☐ Oui☐ Non

FACTEURS DECLENCHANTS

☐Trouble du rythme ☐Infection ☐Ecart de régime ☐Mauvaise observance ☐Angor

☐Autres : Cliquez ici pour entrer du texte.

BIOLOGIE

CREATININEMIE

*Taux de référence :*Cliquez ici pour entrer du texte.

*Taux ce jour :*Cliquez ici pour entrer du texte.

PEPTIDES NATRIURETIQUES

Taux de référence : Cliquez ici pour entrer du texte.

*Taux ce jour :*Cliquez ici pour entrer du texte.

Autres résultats :

Cliquez ici pour entrer du texte.

EXAMENS COMPLEMENTAIRES : choix multiples

☐ECG ☐Radio pulmonaire☐ Holter Echographie cardiaque ☐Echographie dobutamine

☐Epreuve d'effort ☐Scintigraphie cardiaque ☐Coronarographie ☐Test de marche

☐Autres : Cliquez ici pour entrer du texte.

CONSULTATIONS DE SUIVI

Médecin Généraliste dans les 8 jours Oui ☐ Non ☐

Cardiologue dans les 30 jours Oui ☐ Non ☐

Autres Oui ☐ Non ☐

INFORMATIONS DE SORTIE

Ajustement thérapeutique Oui ☐ Non ☐

Cliquez ici pour entrer du texte.

Suivi biologiques prescrit : Oui ☐ Non ☐

Examens complémentaires prescrits : ☐Oui ☐Non

Cliquez ici pour entrer du texte.

Informations données *(maladie, régime, hygiène de vie) :* Oui ☐ Non ☐

Fiche de liaison ou CR de sortie remis : Oui ☐ Non ☐

27/08/2017 22:26

FIGURE C.2 – Page 2 du questionnaire de consultation semi-urgente

DEVENIR	
☐Retour à domicile avec ☐retour vers médecin traitant ☐programmation HDJ ☐programmation d'une consultation hospitalière	
Education thérapeutique proposée :	Oui ☐ Non ☐ Si oui, date du rdv proposé :
Passage aux urgences évité par la CSIC	Oui ☐ Non ☐ Non évaluable ☐
SYNTHESE	
Cliquez ici pour entrer du texte.	

Signature de l'IDE :

Signature du médecin :

27/08/2017 22:26

FIGURE C.3 – Page 3 du questionnaire de consultation semi-urgente