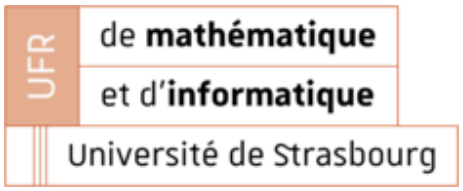
UNIVERSITÉ DE STRASBOURG



CSMI : Calcul Scientifique et Mathématiques de l'Innovation

Rapport de Stage de Fin d'étude

Tuteurs ACM :
Christophe GERALDES PEREIRA

Auteur :
Komi Joseph Béni ASSIGBE

Université de Strasbourg

Remerciements

Ce stage de fin d'études marque une étape importante de mon parcours, tant sur le plan académique que personnel. Il a été l'occasion de mettre en pratique les connaissances acquises tout au long de ma formation, de découvrir le monde professionnel au sein d'un grand groupe, et d'évoluer dans un environnement stimulant.

À cette occasion, je souhaite adresser mes plus sincères remerciements à toutes les personnes qui ont contribué au bon déroulement de cette expérience. Je tiens à remercier Mr Christophe PRUD'HOMME, Responsable du Master et l'équipe pédagogique du Master CSMI (Calcul Scientifique et Mathématiques de l'Innovation), pour leur accompagnement, leur disponibilité et la qualité de l'enseignement dispensé tout au long de ces deux années. Grâce à leur encadrement, j'ai pu acquérir des compétences solides, tant théoriques que pratiques, qui m'ont permis d'aborder ce stage avec confiance et professionnalisme.

Je tiens également à exprimer toute ma gratitude à mon tuteur de stage, Mr Christophe GERALDES PEREIRA, pour son encadrement attentif, sa bienveillance et la confiance qu'il m'a témoignée dès les premiers jours. Son accompagnement m'a permis de monter progressivement en compétences et de m'investir pleinement dans les missions qui m'ont été confiées. Sa disponibilité, ses conseils pertinents et sa pédagogie, ses points de suivie ont joué un rôle essentiel dans la réussite de ce stage.

Je souhaite aussi remercier chaleureusement mes collègues Paul BECAT, Christophe MAETZ, Yann BARBIER et Sureya KULAHCI, avec qui j'ai eu la chance de collaborer au quotidien. Leur accueil chaleureux, leur soutien constant, leur patience et leurs encouragements ont rendu cette expérience à la fois enrichissante et humaine. J'ai beaucoup appris à leurs côtés, non seulement sur le plan technique et professionnel, mais aussi sur le plan relationnel et organisationnel.

Je remercie également l'ensemble des collaborateurs des Assurances Crédit Mutuel, qui m'ont permis de m'intégrer rapidement au sein de l'équipe et de participer à des projets concrets et valorisants. Intégrer une structure aussi dynamique et engagée a été une réelle opportunité.

Assurances Crédit Mutuel

Je garderai de ce stage un souvenir très positif, fait de rencontres enrichissantes, d'apprentissages concrets et d'un réel sentiment d'évolution. Merci à toutes celles et ceux qui ont contribué à faire de cette période un moment fort de mon parcours.

Résumé

Ce rapport de stage rend compte du stage de fin d'études réalisé au sein des ASSURANCES DU CREDIT MUTUEL, dans le cadre de mon Master en Calcul Scientifique et Mathématiques de l'Innovation à l'Université de Strasbourg.

Mon stage s'est déroulé au sein du service Pilotage, Études et Statistiques de la Direction des Indemnisations et des Prestations. L'objectif principal était de contribuer à l'optimisation de la planification des activités métiers. Les missions qui m'ont été confiées consistaient à comprendre le travail existant et à identifier les principales contraintes opérationnelles définies par le métier, puis à traduire mathématiquement le modèle, et enfin à assurer le suivi du projet, de la collecte des besoins à la mise en production. Il s'agissait ainsi d'utiliser la modélisation mathématique pour optimiser et générer la planification tout en veillant à son adaptation aux réalités du terrain ; cette démarche s'inscrit dans une logique d'amélioration continue, en lien étroit avec les besoins du service et les objectifs stratégiques définis. Ce procédé est crucial car il permet aux encadrants des services de gestion de prendre des décisions offrant un meilleur traitement aux assurés.

Ce stage m'a permis d'acquérir, de mettre en œuvre et d'approfondir des compétences en data science et en optimisation. Cette expérience enrichissante m'a permis de découvrir les spécificités du secteur de l'assurance, les enjeux liés à la relation client, l'impact des décisions sur l'activité ainsi que la transformation digitale du métier et la manière dont la modélisation mathématique se met au service des entreprises.

Table des matières

Remerciements **1**
Résumé **3**
1 Introduction **6**
2 Présentation de l'Entreprise **7**
2.1 Le Groupe Crédit Mutuel Alliance Fédérale 7
2.2 Les secteurs d'activités 8
2.3 Engagement et valeurs 9
2.4 Les Assurances du Crédit Mutuel - ACM 9
2.5 Quelques chiffres clés de l'année 2O24 10
2.6 La Direction des Indemnisations 12
3 Présentation du Sujet et de l'Equipe **14**
3.1 Contexte du projet 14
3.2 Présentation de L'équipe 14
3.3 Rôle au sein de L'équipe 15
4 Outils **16**
4.1 Le Référentiel de Ressources 16
4.2 SAS 16
4.2.1 Les modules et composants de SAS 18
4.2.2 SAS/OR 19
4.3 Datalab 21
4.3.1 Espace de travail 22
4.3.2 Fonctionnalités du Datalab 22
4.3.3 Pyomo 23
5 Missions et tâches **24**
6 La planification **27**
6.1 Tables de paramétrage 27
6.2 Les contraintes métiers 29
6.3 Modélisation mathématique 30

6.4 Solveurs 35
7 Démarches et Résultats 36
7.1 Implémentation 37
7.2 Résultats 44
8 Application du Machine Learning pour la prévision du temps d'attente 47
8.1 Le jeu de données 47
8.2 Préparation et exploration des données 47
8.3 Modélisation 48
8.4 Évaluation des modèles 49
8.5 Application 50
9 Difficultés rencontrées 52
10 Développement futur 52
11 Conclusion 53

1 Introduction

La planification du travail joue un rôle crucial dans le bon fonctionnement d'une entreprise au quotidien. Historiquement, la gestion des plannings reposait souvent sur des méthodes basées sur l'expérience et l'intuition des responsables : ajustements manuels, gestion au cas par cas des demandes individuelles. Bien que ces méthodes aient pu répondre aux besoins dans certains contextes, elles étaient fréquemment sources de déséquilibres : surcharge ou sous-utilisation de certaines équipes, manque d'équité dans la répartition des tâches, perte de productivité et démotivation des collaborateurs.

Aujourd'hui, face à la complexité croissante des organisations et à l'exigence d'agilité, l'optimisation des plannings s'impose comme un levier incontournable pour répondre aux défis d'efficacité et de flexibilité. Gérer efficacement les plannings permet non seulement d'optimiser la productivité, mais aussi d'améliorer la satisfaction des collaborateurs et collaboratrices en leur offrant des horaires mieux adaptés à leurs besoins.

Dans le secteur des assurances et principalement au Crédit Mutuel, cette problématique prend une dimension particulière : la diversité des métiers et la sensibilité des contraintes opérationnelles rendent la planification d'autant plus complexe et stratégique. Les méthodes traditionnelles montrent rapidement leurs limites et la nécessité de solutions plus robustes et automatisées devient évidente.

Grâce aux avancées de la recherche opérationnelle et des outils d'optimisation, il est désormais possible de proposer des solutions innovantes et personnalisées, capables de concilier les objectifs de l'entreprise et le bien-être des équipes. Ces approches permettent d'anticiper les besoins, de s'adapter aux évolutions du secteur et de renforcer la performance globale.

C'est dans ce contexte que s'inscrit mon stage : concevoir et développer une solution de planification optimisée, adaptée aux spécificités du métier et aux attentes des équipes. Ce rapport retrace les différentes étapes du projet, les défis rencontrés, les solutions apportées, ainsi que les perspectives d'évolution pour une gestion des plannings toujours plus performante, agile et tournée vers l'avenir.

2 Présentation de l'Entreprise

2.1 Le Groupe Crédit Mutuel Alliance Fédérale

Le groupe Crédit Mutuel Alliance Féderale est une banque française coopérative et mutualiste, fondé en 1882 en Alsace sous l'impulsion de Friedrich Guillaume Raiffeisen. Elle est présente sur l'ensemble du territoire national, depuis son rapprochement avec le CIC en 1998, suite à cela le groupe Crédit Mutuel poursuit son dévéloppement, se diversifie, s'affirme comme leader des nouvelles technologies et s'ouvre à l'international.

FIGURE 1 – Logo officiel de Crédit Mutuel Alliance fédérale

Depuis plusieurs années, le Groupe mène une stratégie de transformation numérique ambitieuse, concrétisée par son Plan Stratégique 2024-2027 intitulé « Ensemble – Performant – Solidaire », qui met l'accent sur la digitalisation des parcours clients en préservant les relations humaines. Pour renforcer sa performance, le Crédit Mutuel s'est résolument engagé dans l'adoption des innovations technologiques. Dans cette dynamique, il prévoit de poursuivre, dans les années à venir, le déploiement d'outils cognitifs, de solutions de reconnaissance optique de caractères (OCR) et de reconnaissance vocale, tout en investissant massivement dans l'intelligence artificielle générative. Cette stratégie lui a valu d'être récompensé par le prix du Meilleur Groupe Bancaire Français de l'année 2025.

FIGURE 2 – Récompense Meilleur Groupe Bancaire Français

Assurances Crédit Mutuel

2.2 Les secteurs d'activités

Le Groupe Crédit Mutuel Alliance fédérale, en plus de ses activités bancaires et financières, diversifie ses domaines de prestations :

— **Assurance** : représente le deuxième métier du groupe, avec **38 millions de contrats** et possède des filiales d'assurances qui sont au service de plus de 13,6 millions d'assurés, ce qui leur vaut de se classer au 5e bancassureur sur le marché en france selon L'argus de l'assurance [1]

— **Technologies** : Le groupe propose des solutions de paiement en ligne, des applications mobiles et des innovations fintech, en s'appuyant sur sa filiale technologique "Euro-Information" pour améliorer l'expérience client. Parmi ses solutions de paiement, on peut citer Lyf Pay. Le groupe est également le premier acteur français de la télésurveillance résidentielle grâce à ses solutions partenaires Homiris et Arkea Sécurité.

— **Métiers spécialisés :** Le groupe propose également des solutions de leasing automobile, notamment la location longue durée, permettant à ses clients de financer l'usage d'un véhicule sans en devenir propriétaire. Ces offres sont proposées à travers ses filiales Crédit Mutuel Factoring, Crédit Mutuel Leasing et Real Estate Lease.

— **Immobilier et le Crédit à la Consommation :** Pour répondre à l'ensemble des besoins des clients, le groupe propose toutes les expertises des métiers de l'immobilier en prenant toutes les charges du cycle immobilier. Le groupe propose également des offre de crédit à la consommation par sa filiale Cofidis à travers l'offre FINANCO

— **L'international :** Le groupe poursuit son expansion à l'international, notamment en Allemagne et dans d'autres pays.

1. Source : Rapport d'activité du Crédit Mutuel 2024, et https://www.argusdelassurance.com/classements/classement-des-bancassureurs-2024.232292

FIGURE 3 – Secteur d'activités du Groupe

2.3 Engagement et valeurs

Depuis sa création, le Crédit Mutuel développe des initiatives fondées sur la solidarité. Libre de tout actionnariat, Le Groupe Coopératif possède comme valeurs :

- **Liberté & Ouverture** : Chacun est libre de pouvoir y devenir sociétaire, ce qui favorise une participation aux décisions.
- **Égalité démocratique** : Les représentants sont élus par les sociétaires-clients, selon le principe démocratique : « Une personne = Une voix »
- **Égalité vertueuse** : Le groupe non coté en bourse a des réserves impartageables qui ne bénéficient pas à un actionnariat. Les dividendes soutiennent le développement économique local et régional.
- **Solidarité** : Une solidarité entre les caisses locales au niveau régional et les Fédérations au niveau national.
- **Autonomie & Responsabilité** : Chaque entité a une autonomie, ce qui favorise plus de réactivité dans la prise des décisions, plus de proximité.

2.4 Les Assurances du Crédit Mutuel - ACM

Lors de mon stage, j'ai eu le privilège d'être accueilli au sein des Assurances du Crédit Mutuel. L'assurance est le deuxième métier du groupe. Acteur majeur de l'assurance et de la protection sociale en France, le Groupe des Assurances du Crédit Mutuel (GACM) est né en 1971. Fort d'une expérience de la bancassurance de près de 50 ans, les activités portées par le

Assurances Crédit Mutuel

GACM sont pleinement intégrées sur le plan commercial et technologique de l'Alliance fédérale. La distribution s'effectue par les réseaux bancaires Crédit Mutuel et CIC, ainsi que via les enseignes telles que Cofidis et Monabanq. Les Assurances du Crédit Mutuel couvrent trois grands domaines de l'assurance :

- **L'assurance de Biens**
- **L'assurance de personnes**
- **L'assurance-vie & Épargne Retraite**

} Particuliers et familles

- **Protection des Biens et de l'Activité**
- **Protection et Valorisation du Capital Humain**
- **Protection des Dirigeants**

Professionnels, Entreprises, Agriculteurs et Associations

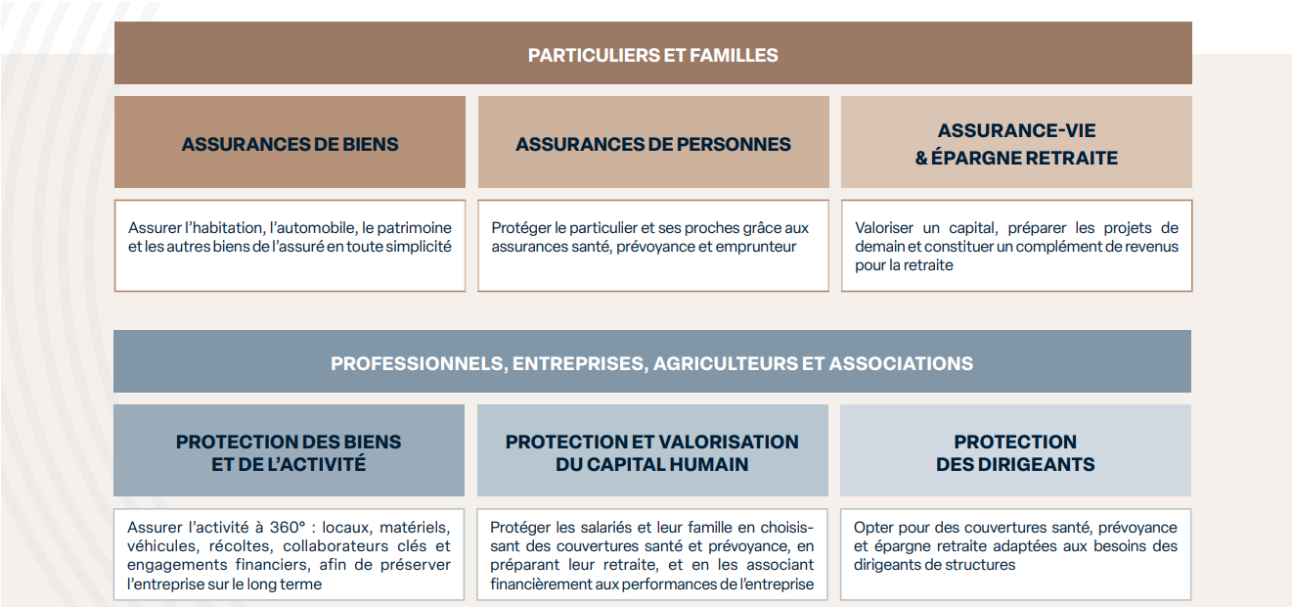


FIGURE 4 – Gamme d'offres assurance

2.5 Quelques chiffres clés de l'année 2O24

Le chiffre d'affaires du Groupe des Assurances Crédit Mutuel a progressé de 11,3 % par rapport à la fin 2023 pour atteindre 15,2 milliards d'euros avec une hausse de 2,6 % pour arriver à 38 millions de Contrats en portefeuille en 2024 comme nous pouvons le constater sur la figure suivante :

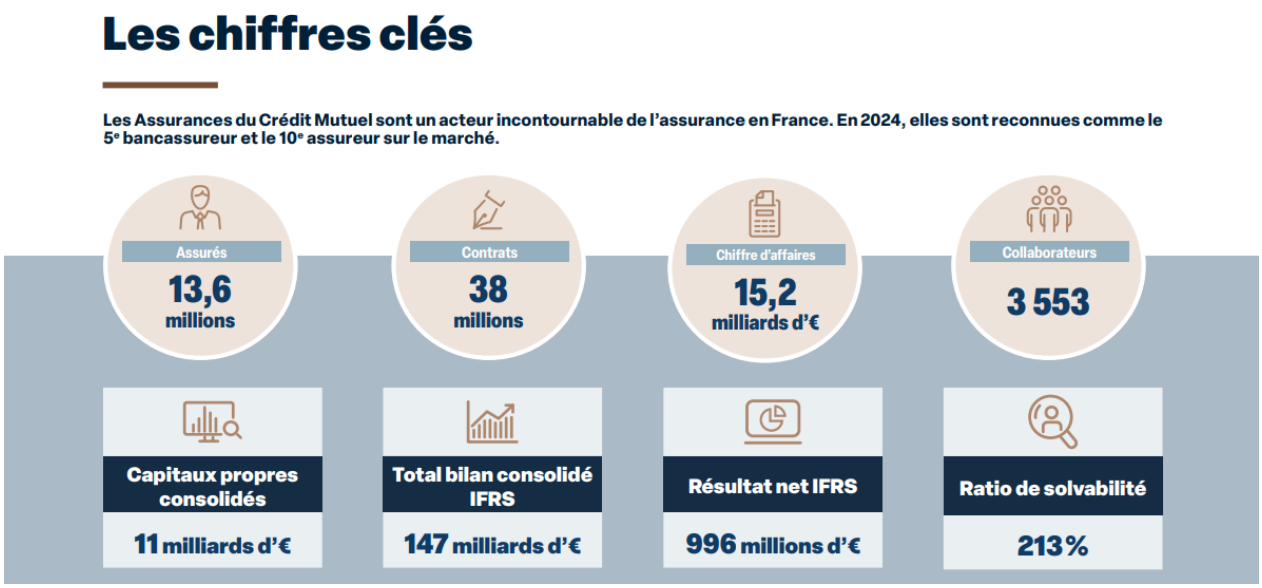


FIGURE 5 – Chiffres Clés

La repartition en pourcentage du chiffre d'affaires se repartit comme suit :

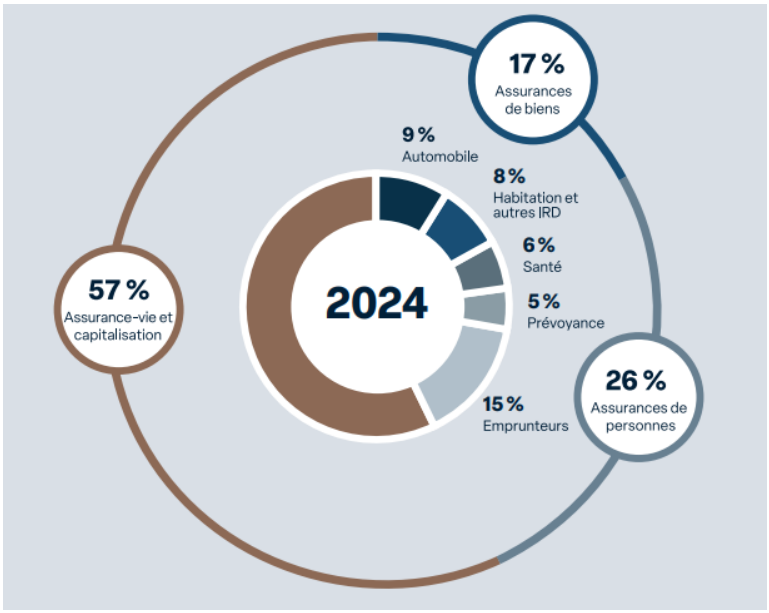


FIGURE 6 – Repartition du Chiffre d'affaires

Par secteur d'activité,

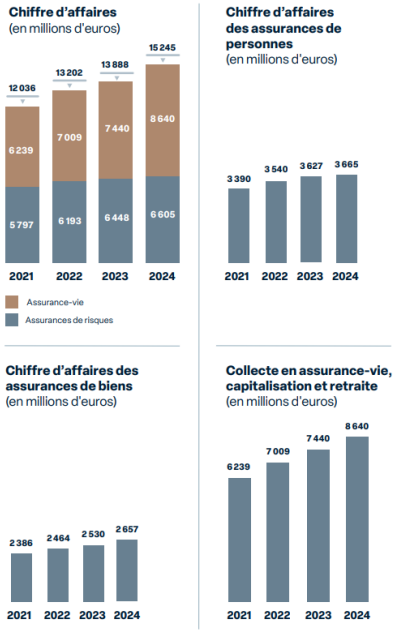


FIGURE 7 – Chiffre d'affaires par activités

2.6 La Direction des Indemnisations

La Direction des Indemnisations des Assurances du Crédit Mutuel, est responsable de la gestion des sinistres et du versement des indemnisations aux assurés, aux victimes, conformément à leurs contrats. Elle est également en charge d'exercer les recours contre les responsables des accidents, et assurer la maîtrise des risques que peuvent générer la société ou le Groupe. Cette direction joue un rôle essentiel dans la satisfaction client en garantissant un traitement rapide et transparent des dossiers. Le service d'indemnisation est organisé de la manière suivante :

- Indemnisation des Biens
- Prestations aux Personnes,
- appuyés par un service Juridique et Contentieux ainsi qu'une cellule projet et pilotage dite Stratégie Transformation Innovation

Elle pilote des équipes sur l'ensemble du territoire français, capables de répondre aux besoins de ses clients, en collaboration avec d'autres services du Groupe des Assurances du Crédit Mutuel.

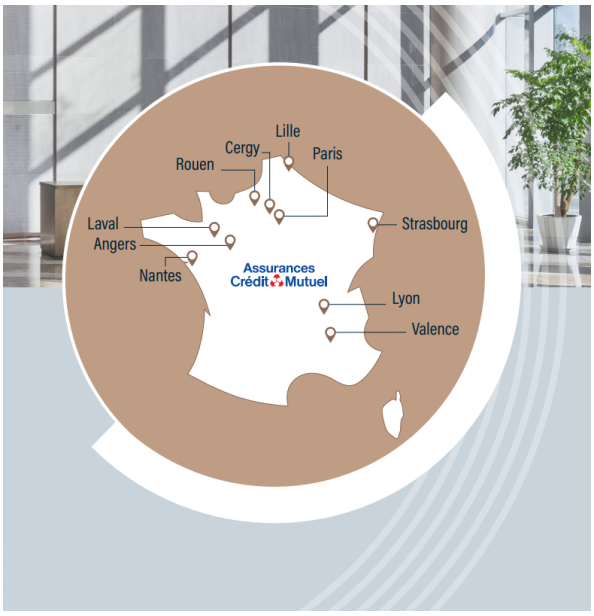


FIGURE 8 – Répartition des sites

3 Présentation du Sujet et de l'Equipe

3.1 Contexte du projet

Possédant un grand nombre de clients fidélisés, grâce à ses valeurs mutualistes, Les Assurances du Crédit Mutuel, notamment la direction des Indemnisations, fait face à un volume important d'appels et de demandes clients, dépendant de plusieurs facteurs contrôlés ou non. Parmi ces facteurs, on peut citer les aléas climatiques, les accidents, les stocks. Il revient donc d'assurer une meilleure organisation de ses équipes, afin de garantir une réponse rapide et efficace. Cela s'inscrit aussi dans une volonté des ACM d'aller vers une digitalisation des processus métiers et une automatisation des tâches chronophages.

Cette démarche prend en compte de pouvoir s'imprégner des réalités métiers afin d'élaborer des règles et des contraintes qui satisfont les besoins métiers et répondent aux exigences et engagements des ACM, de pouvoir anticiper les flux à traiter : dossiers reçus, appels sollicités et échanges informatisées avec les acteurs marchés (sécurité sociale, experts, garagistes....).

Arriver à répondre à ces exigences fait des ACM une entreprise dont la satisfaction des clients et le confort de ses collaborateurs sont au centre de son développement.

C'est dans ce sens que s'inscrit mon stage au sein de la Direction des Indemnisations plus précisément au département Stratégie Transformation Innovation, pour participer à la modélisation des contraintes et la transformation des métiers.

3.2 Présentation de L'équipe

Au cours de 6 derniers mois, j'ai eu l'occasion de travailler auprès de collègues issus de differents services et ayant diverses compétences, qui m'ont permis de réaliser à bien cette mission.
J'ai été acceuilli au service **PES : Pilotage et Études Statistiques**, rattaché au département Transformation Innovation, composé de Data Analyst ayant plusieurs casquettes. Sous la Supervision du Responsable GERALDES PEREIRA Christophe, cette équipe se situe sur deux sites (Strasbourg et Lyon). En ce qui me concerne, mon stage s'est déroulé sur le site de Strasbourg.

Le PES est en charge du contrôle interne, du reporting et des analyses transverses venant en aide aux métiers. Cette équipe joue un rôle clé dans la collecte, le traitement et l'exploitation de données stratégiques, destinés à accompagner et à aider dans la prise de décisions, puis participe à la transformation des métiers. Le périmètre du service PES s'étend sur :

— Des suivis d'activité et de Gestion
— De l'accompagnement Projets, Transformation métiers et outils
— Des suivis financiers, conventionnels, et de prestataires
— Des contrôles de qualité notamment sur les flux informatisés et les données financières servant au bilan
— Des suivis de processus automatisés (Ouvertures, prolongations, classements, règlements ...)
— de la planification d'activité

Leur expertise sert à d'autres services tel que la direction générale, la réassurance, l'actuariat, les équipes de modélisation de risques de contrat.

3.3 Rôle au sein de L'équipe

Mon implication dans l'équipe s'est principalement portée sur la planification des activités au service des indemnisations des biens (DDI) et service de prestations aux Personnes (DPP).
Les services de gestion s'occupent des demandes des assurés plus précisement :

— **Gestion des déclarations de sinistres** : l'ouverture et le traitement des dossiers lorsque les assurés déclarent un sinistre (incendie, vol, dégât des eaux catastrophes naturelles, bris de matériel etc..) ou demande de remboursement pour les frais de santé.
— **Evaluation des dommages** : Analyse du montant à rembourser conformément à leurs contrats d'assurance et garanties
— **Suivi de dossiers** : Le suivi administratif et financier des dossiers de sinistres jusqu'à leur clôture, ainsi que la gestion des réclamations.

Au vue de ces activités il est important de pouvoir concevoir une planification optimale permettant aux gestionnaires de réaliser les différentes tâches qui leur sont confiées et de satisfaire les besoins de ses clients. Pour y arriver nous avons utilisé des outils nous permettant de pouvoir traduire efficacement des règles définies en problème puis proposer une solution.

4 Outils

4.1 Le Référentiel de Ressources

Le réferentiel de ressources est un outil développé au sein de l'équipe, principalement en VBA (Visual Basic for Application), qui permet de pouvoir récolter et/ou consigner toutes les données nécessaires pour une planifcation. Parmi ces données on peut citer :

- les absences
- les horaires de travail sur site et de Télétravail
- Le statut des gestionnaires, et le périmètre opérationnel

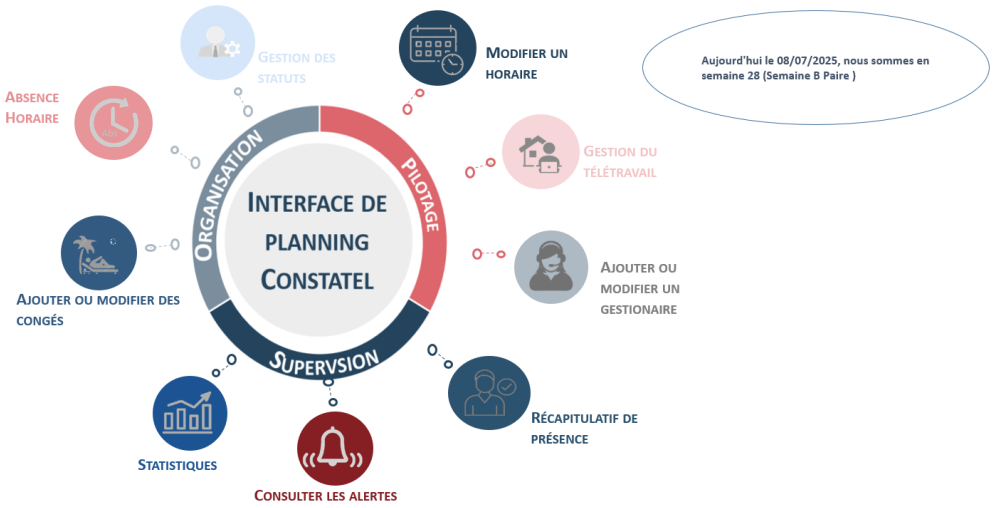


Figure 9 – Référentiel de Ressources

Cet outil permet aux encadrants des équipes de gestion de pouvoir saisir les horaires de leurs gestionnaires, en fonction de leur de temps de présence et de toutes autres informations utiles (comme les réunions d'équipe, etc ..). Ces informations sont alors traitées afin de disposer un Ensemble sur lequel on définira les règles pour la planification.

4.2 SAS

SAS (Statistical Analysis System) est un logiciel dévéloppé par SAS Institute, spécialisée dans l'analyse statistique, la gestion des données et l'informatique décisionnel. Fondé en 1976, SAS [2] est largement utilisé dans divers secteurs tels que la santé, la finance, le marketing etc, pour sa capacité à

2. Wikipédia, SAS institute Website

traiter de grands volumes de données et à fournir des analyses avancées.

FIGURE 10 – Logo de SAS

Le logiciel SAS utilise un langage de programmation propriétaire de quatrième génération, et se compose d'un ensemble de modules permettant de répondre aux besoins par la programmation :

- La création et la gestion de bases de données,
- Traitement analytiques des bases de données,
- Création et diffusion de rapports de synthèse et de listing.

Le langage SAS se base sur deux parties :

- les étapes Data : Elles permettent de lire, créer, transformer et gérer les tables de données (qui peuvent être appelées Datasets ou tables SAS) Cette étape permet de définir la structure des tables, effectuer les calculs sur les variables et filtrer les observations.
- Les procédures (PROC) : Les procédures se basent et travaillent sur les tables créées par les étapes DATA. La définition de chaque processus a un objectif précis. Elles permettent de pouvoir trier, analyser statistiquement, produire les rapports (proc print), gérer les formats et aussi l'export des tables, sans forcément à avoir à réécrire le code de base pour chaque opération. Elles sont la "Boîte à Outils" de l'utilisateur SAS pour traiter et exploiter les données.

```
proc sql;
     create table semaine24b as
     select a.*, b.callback
     from sem&sem_select.
 as a left join planif.statut as b
     on a.memoid=b.memoid
     where a.'Statut Gestionnaire'n ne 'BM0'
     order by num_sem, memoid, jour_sem;
 quit;

data creneau;
    input crenom $ @@;
    t = _N_;
    datalines;
 08:00 08:30 09:00 09:30 10:00 10:30 11:00 11:30 12:00 12:30 13:00 13:30 14:00 14:30 15:00
 15:30 16:00 16:30 17:00 17:30 18:00 18:30
 ;
```

FIGURE 11 – Exemple Etape data & proc

4.2.1 Les modules et composants de SAS

Les modules de SAS Logiciel, se classent en deux parties, la plateforme classique (SAS 9) et la plateforme moderne SAS Viya. La plateforme SAS 9 comprends des modules de base comme :

- SAS/BASE : est le langage de base propriétaire à SAS INSTITUTE, et permet de pouvoir exécuter d'autres langages.
- SAS/GRAPH : module d'analyse graphique
- SAS/ETS : Outils de modélisation et de simulation
- SAS/SHARE : Module de traitement en partage / ressources
- SAS/OR : est le module qui est en charge de la Recherche Opérationnelle, permettant de faire de l'optimisation, la simulation et la Planification.
- SAS/connect : C'est le module de connexion entre différentes systèmes permettant de faire fonctionner SAS en mode Client / Server.
- ...

Pour ne citer que ceux-ci, les modules SAS sont généralement utilisés pour répondre à un besoin précis.
La plateforme SAS Viya est un module moderne intégré, fonctionnant sur CAS (Cloud Analytics Services) qui est une plate-forme unifiée cloud-native, couvrant toute la chaîne allant de la Data vers l'IA, dont les modules les plus importants sont :

— Exploration & Visualisation des données : SAS Visual Analytics et Statistics

— Modélisation : Machine Learning, Forecasting, Optimisation, Econometrics, Text Analytics.

— Gestion & déploiement : SAS Model Manager pour l'enregistrement, comparaison, surveillance et le déploiement des modèles.

— ...

4.2.2 SAS/OR

Au cours de mon stage, j'ai utilisé et travaillé avec le module SAS/OR installé en 2023 sein de l'entreprise. Les solutions d'optimisation et de planification étaient principalement réalisées à l'aide de ce module. Les premiers développements du projet ont été réalisés à l'aide du module SAS/OR, dédié à l'exploration des modèles de distribution de réseau, de la production système, aux problèmes d'allocation de ressources, et aux problèmes de planification en utilisant les ressources de la recherche opérationnelle. Ce module est utilisable via l'interface SAS Enterprise Guide, un environnement graphique sous Windows qui permet d'accéder aux fonctionnalités avancées de SAS, dont SAS/OR, de manière conviviale et sans forcément nécessiter de programmation en ligne de commande.

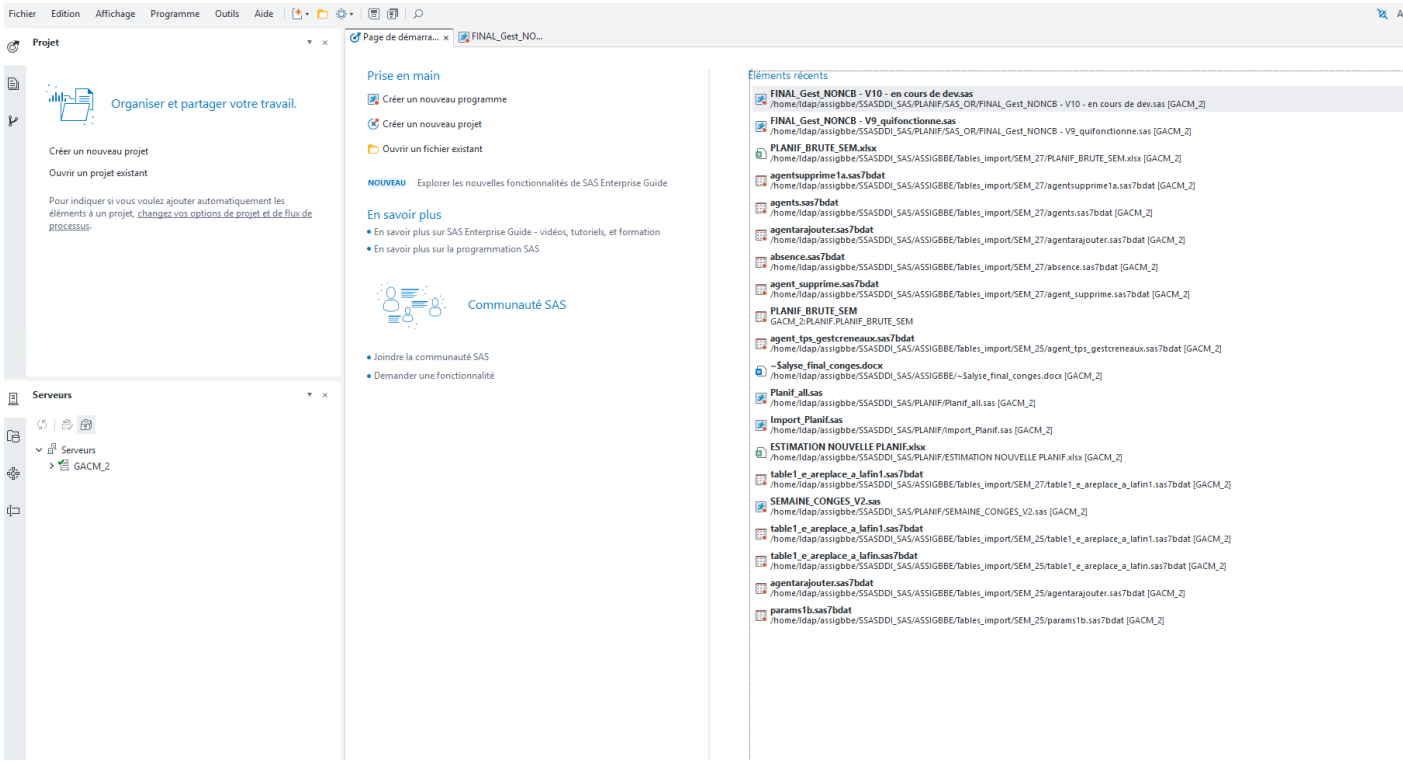


FIGURE 12 – Interface de SAS Entreprise Guide

Le module SAS/OR permet de résoudre les problèmes qui peuvent inclure :

— Problèmes de mix-produit : consistent à déterminer la combinaison de produits qui génère le meilleur rendement, en tenant compte de la concurrence entre plusieurs produits pour des ressources limitées.

— Problèmes de mélange (blending) : s'agissant de trouver la combinaison optimale d'ingrédients à utiliser dans un produit, de façon à satisfaire des normes minimales tout en minimisant les coûts.

— Problèmes à objectifs multiples : problèmes qui impliquent plusieurs objectifs, parfois contradictoires. En général, les objectifs sont hiérarchisés et les problèmes sont résolus successivement selon cet ordre de priorité.

— Problèmes d'ordonnancement (scheduling) : visent à attribuer des personnes à des horaires, des lieux ou des tâches, afin d'optimiser leurs préférences ou leurs performances tout en respectant les contraintes du planning.

— ...

En fonction du type programmation mathématique, SAS/OR propose une classification générale de procédure permettant de les résoudre. Parmi ces procédures, on peut citer [3]

- **La procédure OPTMILP**
- **La Procedure OPTMODEL**

} Programmation linéaire mixte nombre entière

- **La procédure OPTLP**
- **La Procedure OPTMODEL**

} Programmation lineaire

- **la procédure OPTQP**
- **La Procedure OPTMODEL**

} Programmation quadratique

- Problèmes non linéaires

La procédure PROC OPTMODEL est repartie en trois catégories :

— l'instruction PROC : permet d'invoquer la procédure et de définir les options initiales,

— Les instructions de Déclaration : servent à déclarer les composants du modèle d'optimisation,

3. SAS/OR Documentation

— les instructions de programmation : permettent de lire, d'écrire les données, d'appeler le solveur et d'afficher les résultats.

La PROC OPTMODEL est une procédure commune à ces types de problèmes.

4.3 Datalab

Le Datalab est un outil faisant partie intégrante de la plateforme analytique du Groupe permettant une exploration des données et d'effectuer des modélisations, c'est-à-dire des modèles de machine learning. Il s'agit d'un outil Data Science orienté, vers la programmation et offrant un environnement de travail à ses utilisateurs. Un outil interne et maintenu au sein du groupe, il offre un environnment de développement à l'instar de code server.
Le Datalab est un outil dont les langages de programmation supportés sont Python et R, il n'héberge pas de données c'est un portail, un environnement sécurisé permettant d'accéder aux données hébergées sur des systèmes de la plateforme analytique. Il est accessible et sécurisé à travers un navigateur

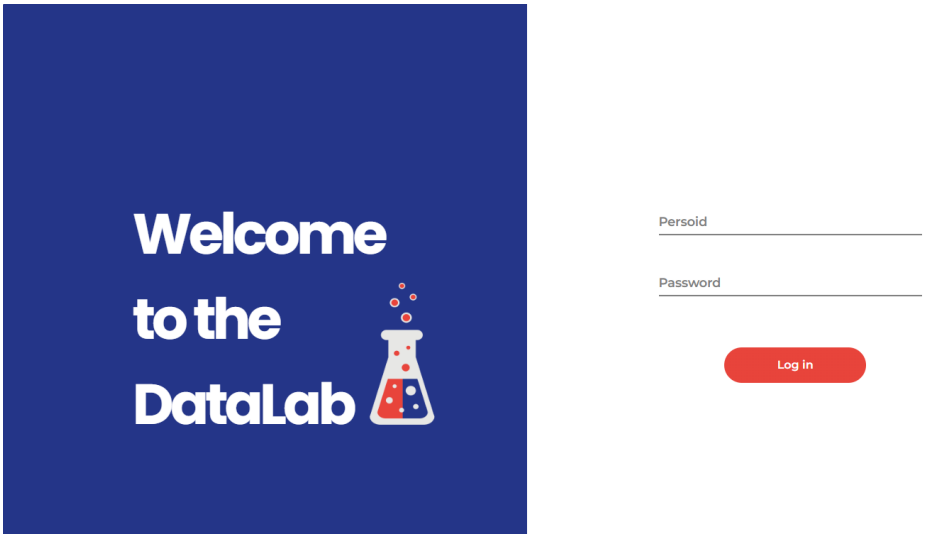


FIGURE 13 – Interface Datalab

Les accès au Datalab sont donnés par des Relais data. Le relai data est par défaut un « manager » (au sens « reviseur ») et « propriétaire » des lignes de diffusion, réprésentant des espaces. Un espace de travail appartient à une équipe Entité Data.

4.3.1 Espace de travail

Un espace de travail est composé des éléments suivants :

- un pool de ressources CPU/RAM (éventuellement partagé avec d'autres espaces de travail)
- un espace de librairies
- répertoire de données dédié à cet espace de travail

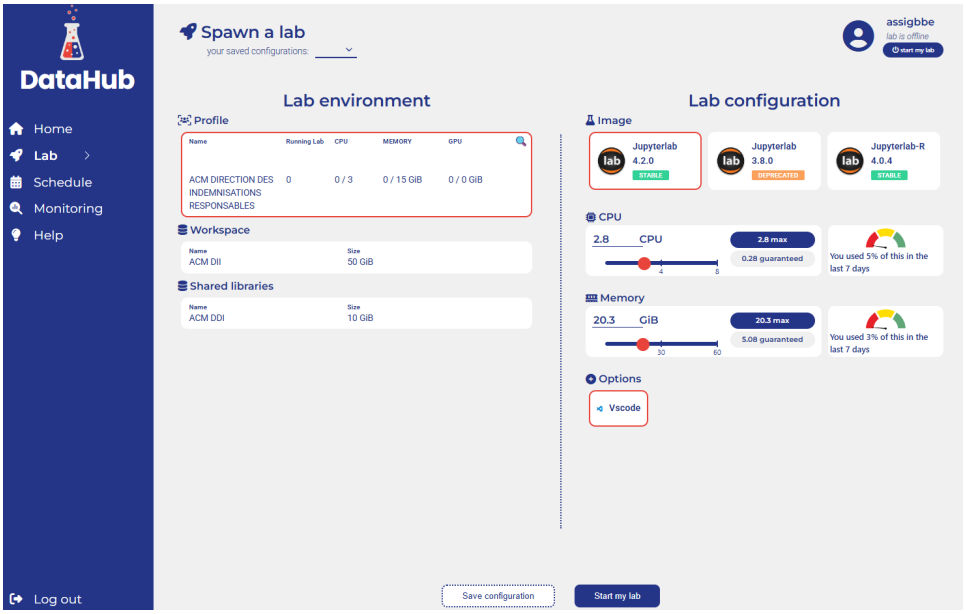


FIGURE 14 – Configuration du Lab

4.3.2 Fonctionnalités du Datalab

Le Datalab dispose de multiples fonctionnalités, mais nous nous concentrerons sur ceux qui nous ont été utiles lors de mon stage.
On dispose d'un environnement basé sur JupyterLab qui permet de construire des notebooks à travers le langage Python et de disposer d'un environnement RStudio afin d'exploiter le langage R.
Le Datalab s'appuie sur un ensemble de librairies Python et R validées et préinstallées. Au cours de mon stage, j'ai identifié le besoin d'utiliser certaines librairies qui n'étaient pas présentes initialement. J'ai donc sollicité leur installation auprès de l'équipe technique, qui a validé et ajouté ces librairies à l'environnement, après vérification de leur sécurité. De plus, une image Code Server a été introduite récemment à ma demande, permettant d'accéder à

Visual Studio Code via l'environnement, fonctionnalité qui n'existait pas auparavant. JupyterLab est principalement utilisé pour les notebooks et VS Code pour le scripting.

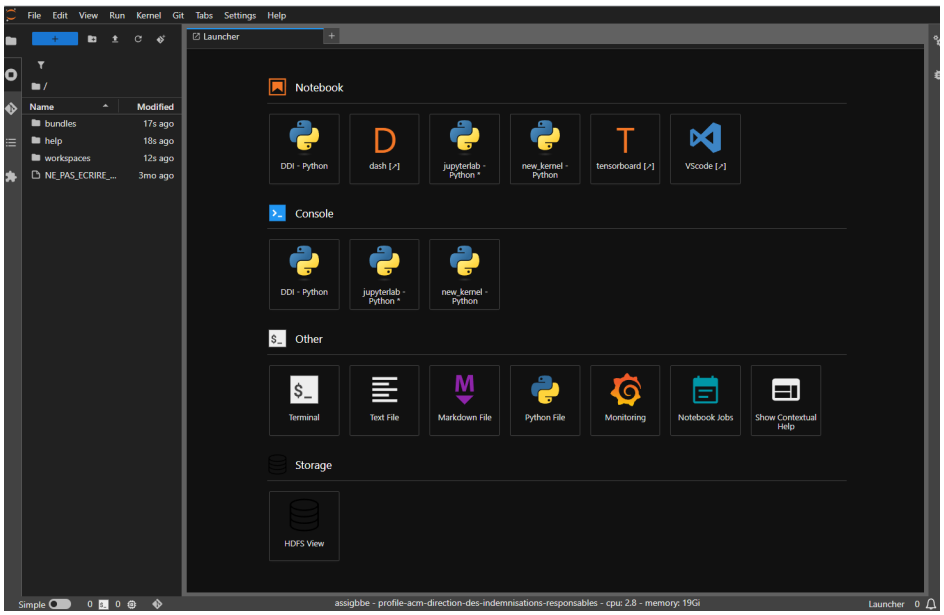


FIGURE 15 – Interface du Datalab

4.3.3 Pyomo

Pyomo[4] (Python Optimization Modeling Objects) est une bibliothèque Python open-source permettant de formuler des modèles d'optimisation mathématique, tout en restant agnostique du solveur utilisé pour la résolution. Elle a été principalement utilisée au cours de mon stage à travers le Datalab.

FIGURE 16 – Logo de Pyomo

Pyomo supporte plusieurs types de programmation : la programmation **linéaire, quadratique, non linéaire, linéaire en nombres entiers mixtes, quadratiques en nombres entiers mixtes, etc ...**. Pyomo offre :

4. Pyomo Presentation

— Puissance de Modélisation : Pyomo permet de décrire des contraintes complexes de manière lisible et modulaire.

— Compatibilité avec des solveurs perfomants de résolution en fonction du problème.

— Flexibilité.

5 Missions et tâches

Durant le premier Semestre 2025, les Assurances du Crédit Mutuel ont déjà servi **2,3 millions d'appels** de ses assurés et réseaux, qui ne constituent qu'une part des activités servies des ACM. En particulier le service des indemnisations de biens à lui seul, au cours des trois derniers mois, a reçu **549 129 appels**, avec un taux moyen d'appels servis de **60,22 %**.

	Semaine 10	Semaine 11	Semaine 12	Semaine 13	Semaine 14	Semaine 15	Semaine 16	Semaine 17	Semaine 18	Semaine 19	Semaine 20	Semaine 21	Semaine 22	Semaine 23	Semaine 24	Semaine 25	Semaine 26	Semaine 27
Nombre d'appels reçus	31 035	34 370	33 261	33 603	28 562	29 929	29 603	27 382	25 040	31 846	39 164	36 285	26 000	37 127	32 245	40 437	41 712	36 528
Taux d'appels servis	68%	59%	60%	54%	78%	74%	65%	66%	67%	51%	49%	56%	61%	58%	52%	54%	51%	61%
Entrées en stock	23 975	23 622	24 113	24 385	23 373	23 811	21 162	18 488	17 712	18 778	26 530	30 300	22 051	23 344	30 598	28 345	30 282	30 293
Stock fin de semaine	20 851	19 684	18 030	17 590	17 151	17 379	19 512	19 053	18 556	19 902	19 276	20 003	21 265	16 920	21 422	26 047	26 563	23 263

TABLE 1 – Evolution du Nombres d'appels du service Indemnisations de Biens

Pour répondre à ces besoins, le service Indemnisation dispose des chargés de sinistres qui analysent les dossiers, évaluent les dommages, et organisent l'indemnisation des clients.

Pour traiter les différentes missions et tâches, l'activité des gestionnaires est découpée en différents temps :

— un temps pour prendre les appels entrants des clients (T)

— un temps pour gérer les documents envoyés par les clients (Back-office)

— un temps pour traiter les flux digitaux issus de différentes applications à disposition des clients et prestataires (E)

— un temps pour traiter des appels sortants sur rendez-vous Call-back (CB)

Dans un second temps, je me suis focalisé à répertorier les contraintes métiers, les comprendre et de prendre en main les développements existants afin d'apporter des améliorations pertinentes et de contribuer efficacement à l'évolution des outils en place.
Ma mission consistait à proposer une variante sous un développement Python en utilisant les outils open Source et contribuer à l'écriture de nouvelles contraintes et la correction des problèmes rencontrés. Elle s'est inscrite dans un contexte de transformation des métiers de gestion, ce qui l'a rendue encore plus intéressante.

La transformation consistait à la réorganisation des services, avec un nouveau découpage des tâches, l'ajout de spécialités de gestion, qui a rendu quasi impossible la réalisation de la planification manuelle, d'où la volonté des services de disposer d'outils de planification automatisés et performants.
Tout au long de mon stage, j'ai participé à des réunions régulières avec les équipes métier. Ces échanges constants m'ont permis de mieux comprendre leurs attentes, de recueillir leurs besoins spécifiques et d'adapter en continu les solutions proposées.

Cette collaboration étroite a été essentielle pour garantir la pertinence des développements réalisés et assurer une bonne adéquation entre les outils informatiques et les processus métier. Grâce à cette immersion, j'ai développé plusieurs compétences clés :

- **Capacité d'écoute et d'analyse :** cela m'a permis de pouvoir receuillir les besoins, poser les bonnes questions et reformuler pour s'assurer une compréhension commune

- **Adaptabilité :** ajuster mon travail en fonction des retours du métier et évolutions des priorités.
- **Communication :** rendre compte de l'avancement notamment des points de suivie d'activité par semaine.
- **Gestion de projet :** organiser mon travail, planifier les tâches.
- **Compétences techniques :** développer mes compétences en modélisation mathématique et en programmation python, La prise en main de nouveaux outils comme Pyomo, SAS, etc ..

6 La planification

Les éléments rentrant dans la planification des activités sont les contraintes ou règles du métiers, les tables de paramétrages, des cibles à appliquer en fonction des activités et créneaux. Dans le processus, on entend par créneau : une plage horaire, un intervalle de temps précis définit par une heure de début et une heure de fin pendant lequel une tâche ou une activité doit être réalisée. La planification des activités se fait une semaine à l'avance. On définit une convention de nommage des activités de la manière suivante :

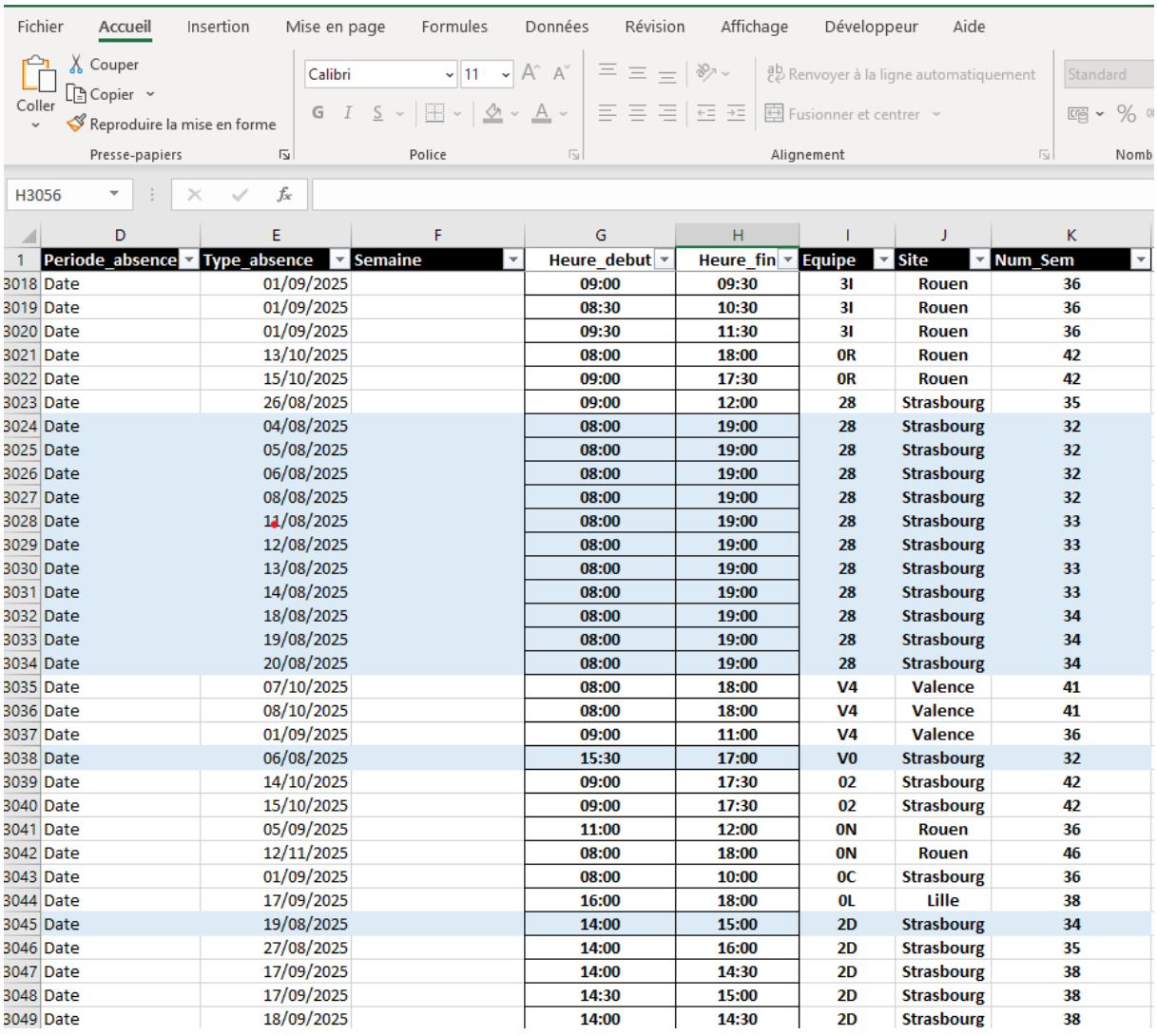
- Téléphonie équivalent à l'activité **1**
- E-déclaration équivalent à l'activité **3**
- Gestion équivalent à l'activité **4**

On attribue un nombre de créneaux minimum et maximum pour chaque activité notamment :

- pour la Téléphonie, 2 créneaux minimum et 5 créneaux maximum
- pour l'E-décla, 2 créneaux minimum et 3 créneaux maximum
- pour la Gestion, 2 créneaux minimum et 4 créneaux maximum

6.1 Tables de paramétrage

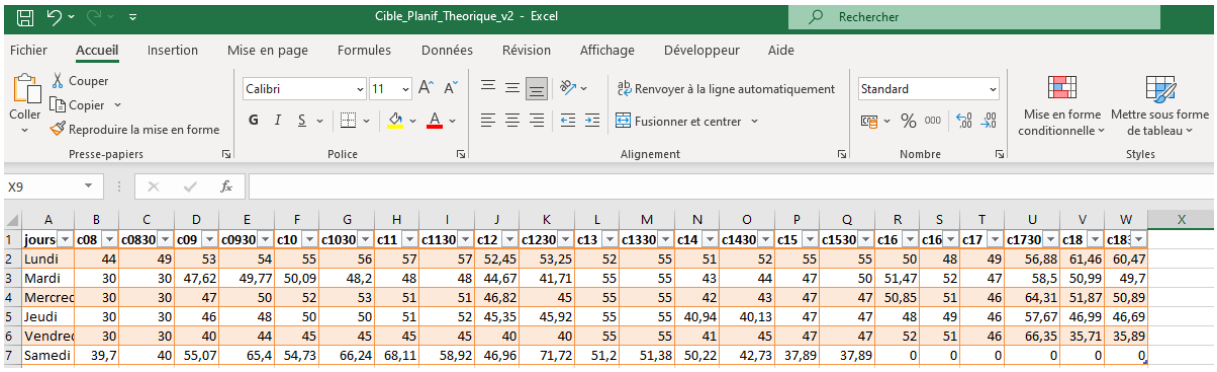
Les Tables de Paramétrages sont des tables qui servent à stocker des paramètres (des cibles) bien définies dans notre cas pour chaque activité. Après la saisie des informations par les responsables d'équipe dans le Référentiel de Ressources, on récupère un fichier Excel qui fait une restitution suivant le jour, les créneaux présents, les absences horaires (maladie, congés, Zone de communication, etc ..) des gestionnaires.



	D	E	F	G	H	I	J	K
1	Periode_absence	Type_absence	Semaine	Heure_debut	Heure_fin	Equipe	Site	Num_Sem
3018	Date	01/09/2025		09:00	09:30	3I	Rouen	36
3019	Date	01/09/2025		08:30	10:30	3I	Rouen	36
3020	Date	01/09/2025		09:30	11:30	3I	Rouen	36
3021	Date	13/10/2025		08:00	18:00	0R	Rouen	42
3022	Date	15/10/2025		09:00	17:30	0R	Rouen	42
3023	Date	26/08/2025		09:00	12:00	28	Strasbourg	35
3024	Date	04/08/2025		08:00	19:00	28	Strasbourg	32
3025	Date	05/08/2025		08:00	19:00	28	Strasbourg	32
3026	Date	06/08/2025		08:00	19:00	28	Strasbourg	32
3027	Date	08/08/2025		08:00	19:00	28	Strasbourg	32
3028	Date	11/08/2025		08:00	19:00	28	Strasbourg	33
3029	Date	12/08/2025		08:00	19:00	28	Strasbourg	33
3030	Date	13/08/2025		08:00	19:00	28	Strasbourg	33
3031	Date	14/08/2025		08:00	19:00	28	Strasbourg	33
3032	Date	18/08/2025		08:00	19:00	28	Strasbourg	34
3033	Date	19/08/2025		08:00	19:00	28	Strasbourg	34
3034	Date	20/08/2025		08:00	19:00	28	Strasbourg	34
3035	Date	07/10/2025		08:00	18:00	V4	Valence	41
3036	Date	08/10/2025		08:00	18:00	V4	Valence	41
3037	Date	01/09/2025		09:00	11:00	V4	Valence	36
3038	Date	06/08/2025		15:30	17:00	V0	Strasbourg	32
3039	Date	14/10/2025		09:00	17:30	02	Strasbourg	42
3040	Date	15/10/2025		09:00	17:30	02	Strasbourg	42
3041	Date	05/09/2025		11:00	12:00	0N	Rouen	36
3042	Date	12/11/2025		08:00	18:00	0N	Rouen	46
3043	Date	01/09/2025		08:00	10:00	0C	Strasbourg	36
3044	Date	17/09/2025		16:00	18:00	0L	Lille	38
3045	Date	19/08/2025		14:00	15:00	2D	Strasbourg	34
3046	Date	27/08/2025		14:00	16:00	2D	Strasbourg	35
3047	Date	17/09/2025		14:00	14:30	2D	Strasbourg	38
3048	Date	17/09/2025		14:30	15:00	2D	Strasbourg	38
3049	Date	18/09/2025		14:00	14:30	2D	Strasbourg	38

FIGURE 17 – Restitution des absences horaires

Pour les activités de Gestion et d'Edecla, on définit des tables de paramétrages qui contiennent des taux à appliquer en fonction du temps de présence hebdomadaire du gestionnaire, pour respecter une équité. Ces tables sont utiles et permettent facilement de s'adapter en fonction du stock des demandes à traiter. On fait passer ces données en entrée dans un programme SAS qui permet de pré-traiter, préparer pour la résolution. Pour la Téléphonie, on joue un programme SAS qui permet de récupérer le nombre d'appels reçus par semaine, en fonction des tendances des semaines passées pour prédire le nombre d'appels à recevoir la semaine suivante en fonction des tendances des semaines antérieures. Ainsi, on arrive à déterminer le pourcentage d'appels décrochés par créneaux.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
1	jours	c08	c0830	c09	c0930	c10	c1030	c11	c1130	c12	c1230	c13	c1330	c14	c1430	c15	c1530	c16	c16	c17	c1730	c18	c18	
2	Lundi	44	49	53	54	55	56	57	57	52,45	53,25	52	55	51	52	55	55	50	48	49	56,88	61,46	60,47	
3	Mardi	30	30	47,62	49,77	50,09	48,2	48	48	44,67	41,71	55	55	43	44	47	50	51,47	52	47	58,5	50,99	49,7	
4	Mercrec	30	30	47	50	52	53	51	51	46,82	45	55	55	42	43	47	47	50,85	51	46	64,31	51,87	50,89	
5	Jeudi	30	30	46	48	50	50	51	52	45,35	45,92	55	55	40,94	40,13	47	47	48	49	46	57,67	46,99	46,69	
6	Vendrec	30	30	40	44	45	45	45	45	40	40	55	55	41	45	47	47	52	51	46	66,35	35,71	35,89	
7	Samedi	39,7	40	55,07	65,4	54,73	66,24	68,11	58,92	46,96	71,72	51,2	51,38	50,22	42,73	37,89	37,89	0	0	0	0	0	0	

FIGURE 18 – Taux de décroches d'appels

6.2 Les contraintes métiers

Les contraintes métiers sont des règles qui ont été définies par le service en fonction des activités principales du service, pour assurer une planification optimale en fonction des besoins du métiers et des contraintes liées au gestionnaires eux-mêmes. Les contraintes sont élaborées par les chefs d'équipe et sont susceptibles de changer. Elles sont rédigées en tenant compte des stocks et éventuellement des aléas climatiques. Elles sont définies sur les jours de la semaine, excepté le dimanche, et dépendent des activités.
Les contraintes sont regroupées par niveaux selon les activités et des règles générales commune à tous. Celles recensées sont :

- Une seule activité par créneaux pour toutes les activités.
- Pas de paires consécutives d'activités
- Téléphonie :
  - Disposer d'un nombre minimum d'agents au téléphone par créneau, soit un nombre de chargés de sinistre au téléphone correspondant aux cibles pré-calculées
  - Un minimum de téléphone par jour et par chargé(e), reparti suivant une plage de téléphone la matinée et une autre dans l'après-midi, soit 4 ou 5 créneaux de téléphonie repartis sur les deux plages.
- Gestion :
  - Une équité de gestion par semaine et par chargé(e) de sinistre
  - Un nombre maximal d'heure de gestion qu'un chargé peut effectuer par demi-journée
  - Un nombre minimum d'heure de gestion qu'un chargé peut effectuer chaque par demi-journée

— Pour les chargés de sinistres ayant qu'une demi-journée : suivant la demi-journée qui n'a pas plus de 12 créneaux, si il s'agit de la matinée ils n'ont pas plus de 4 créneaux et si il s'agit de l'après-midi pas plus que 3.
— sur le samedi, le nombre d'activités de Gestion est limité à 2 créneaux par gestionnaire

- E-DECLA :
  — Une Équité d'e-decla par semaine et par chargé de gestionnaire
  — Un nombre minimum de 2 créneaux par demi-journée.
  — Un nombre maximum de 3 créneaux demi-journée.

Une fois la recette des contraintes métiers, je suis passé à la modélisation mathématique des contraintes.

6.3 Modélisation mathématique

Pour effectivement résoudre notre problème, nous transformons nos contraintes en formulation mathématique qui nous permettra d'utiliser des outils et techniques appropriés pour rechercher une solution efficace.
Le problème traité est un problème d'optimisation binaire (de programmation linéaire en nombres entiers binaires).
On note :

- Variables de Décision :
  — P : Variable binaire qui vaut 1 si une activité est affecté à un Agent pour les jours de lundi au vendredi, 0 sinon.
  — Z : Variable binaire qui vaut 1 si une activité est affectée à un agent pour le jour (d = samedi) uniquement, 0 sinon.
- Sous ensembles :
  — AGENT_DAY_CRENEAUX_IN_SEM : est un sous-ensemble d'indexation, c'est l'ensemble des triplets (a, d, t) défini en semaine soit lundi - vendredi avec :
  a : le mémo-id identifiant le gestionnaire
  d : le jour de travail
  t : le créneau où il travaille
  — AGENT_DAY_IN_SEM : sous-ensemble contenant toutes les combinaisons des agents et les jours ouvrés de la semaine.

— AGENT_DAY_CRE_MATIN : sous-ensemble contenant les combinaisons d'agents, des jours d'une semaine ouvrée en prenant uniquement leurs créneaux matinaux.

— AGENT_DAY_CRE_APRESMIDI : sous-ensemble contenant les combinaisons d'agents, des jours d'une semaine ouvrée en prenant uniquement leurs créneaux d'après midi.

— AGENT_DAY_CRE_IN_SEM_a_d : sous-ensemble contenant, pour chaque agent et chacun de leurs jours de présence, l'ensemble des créneaux de la matinée uniquement pour les jours ouvrés.

— AGENT_DAY_CRE_SAMEDI_MATIN : sous-ensemble contenant les combinaisons d'agents pour le samedi matin.

— AGENT_DAY_ACTIVITY_S_L_IN_SEM : sous-ensemble de tuples (a, d,i,s,l) utilisé pour sommer sur les affectations possibles d'activités, pour chaque agent, jour ouvré de la semaine , activité, créneau de début et la longueur possible du créneau.

— AGENT_DAY_ACTIVITY_S_L_IN_SAM : sous-ensemble de tuples (a,d,i,s,l) utilisé pour sommer sur les affectations possibles d'activités, pour chaque agent, activité, créneau de début et la longueur possible du créneau sur le samedi uniquement.

- Contraintes modélisées :

  - Unique activité si présence les jours ouvrés en semaine et le samedi (prendre en compte le sous-ensemble d'index et de sommation pour le samedi) :

    $\forall (a, d, t) \in$ AGENT_DAY_CRE_IN_SEM,

$$\sum_{(a,d,i,s,l) \in \text{AGENT\_DAY\_ACTIVITY\_S\_L\_IN\_SEM} t \in [s, s+l-1]} P_{a,d,i,s,l} = 1$$

  - Pas d'activités consécutives simultanées pour les jours ouvrés et le samedi en veillant sur les ensembles d'indexation :

$$\text{CONSECUTIVE\_PAIRS} = \Big\{(a,d,i,s_1,l_1,s_2,l_2) \;\Big|\; (a,d,i,s_1,l_1) \in \text{AGENT\_DAY\_ACTIVITY\_S\_L\_IN\_SEM},\ (a,d,i,s_2,l_2) \in \text{AGENT\_DAY\_ACTIVITY\_S\_L\_IN\_SEM},\ s_2 = s_1 + l_1\Big\}$$

$$\boxed{\forall (a,d,i,s_1,l_1,s_2,l_2) \in \text{CONSECUTIVE\_PAIRS}, P_{a,d,i,s_1,l_1} + P_{a,d,i,s_2,l_2} \leq 1}$$

- Les contraintes de Gestion et d'Edecla par agent et semaine :
  $\forall a \in$ AGENTS,

$$\boxed{\begin{aligned} \sum_{\substack{(a,d,4,s,l) \in \\ \text{AGENT\_DAY\_ACTIVITY\_S\_L\_IN\_SEM}}} l \cdot P_{a,d,4,s,l} &= \text{Creneau\_G\_Applique2}(a) \\ \sum_{\substack{(a,d,3,s,l) \in \\ \text{AGENT\_DAY\_ACTIVITY\_S\_L\_IN\_SEM}}} l \cdot P_{a,d,3,s,l} &= \text{Creneau\_E\_Applique2}(a) \end{aligned}}$$

  Avec, Creneau_G_Applique2 et Creneau_E_Applique2, le nombre de créneaux des activités G ou E à appliquer.
- Contrainte de maximum d'activité Jour :
  $\forall (a,d) \in$ AGENT_DAY_IN_SEM :

$$\boxed{\sum_{\substack{i,s,l \\ (a,d,i,s,l) \in \text{AGENT\_DAY\_ACTIVITY\_S\_L\_IN\_SEM}}} P_{a,d,i,s,l} \leq 6}$$

- Contrainte sur la Téléphonie :
  $\forall d \in$ DAYS_IN_SEM, $\forall t \in$ CRENEAU :

$$\boxed{\sum_{\substack{(a,d,1,s,l) \in \text{AGENT\_DAY\_ACTIVITY\_S\_L\_IN\_SEM} \\ t \in [s, s+l-1]}} P_{a,d,1,s,l} \geq \text{activity1\_minpercent}_{d,t} \times \text{cible} \times \sum_{\substack{a,i,s,l \\ (a,d,i,s,l) \in \text{AGENT\_DAY\_ACTIVITY\_S\_L\_IN\_SEM} \\ t \in [s, s+l-1]}} P_{a,d,i,s,l}}$$

activity1_minpercent$_{d,t}$ représentant le pourcentage de taux décroche de la téléphonie par créneaux horaires et jour.
Cible représentant le taux d'activité de la téléphonie par rapport aux autres activités.

- Les contraintes de Gestion et d'Edecla par Agent et par Jour i-e les minimums et les maximums d'activité :
  - La Contrainte Maximum Gestion Jour
    $\forall(a,d) \in$ AGENT_DAY_IN_SEM :

$$\sum_{\substack{s,l \\ (a,d,4,s,l)\in \text{AGENT\_DAY\_ACTIVITY\_S\_L\_IN\_SEM}}} l \cdot P_{a,d,4,s,l} \leq \begin{cases} 2 & \text{si Taux\_Gestion\_theorique2}(a) \times n_{a,d} < 2 \text{ et} \\ & \text{Taux\_Gestion\_theorique2}(a) > 0 \\ \lceil \text{Taux\_Gestion\_theorique2}(a) \times \text{poids} \times n_{a,d} \rceil & \text{, sinon} \end{cases}$$

    avec Taux_Gestion_theorique2(a) : taux théorique de gestion pour l'agent a.
    $n_{a,d}$: le nombre total des heures des gestionnaires par jour et créneaux.

  - La Contrainte de minimum Gestion Jour
    $\forall(a,d) \in$ AGENT_DAY_CRE_IN_SEM_a_d :

$$\sum_{\substack{s,l \\ (a,d,4,s,l)\in \text{AGENT\_DAY\_ACTIVITY\_S\_L\_IN\_SEM}}} l \cdot P_{a,d,4,s,l} \geq \left\lfloor \frac{\text{Taux\_Gestion\_theorique2}(a)}{2} \times n_{a,d} \right\rfloor$$

    En fonction des spécificités des heures des gestionnaires, on a des contraintes limitant le nombre de créneaux de gestion la matinée et pour ceux ayant une longue après-midi.

  - La Contrainte du nombre maximum de gestion la matinée :
    $\forall(a,d,t) \in$ AGENT_DAY_CRE_MATIN :

$$\sum_{\substack{s,\,l \\ (a,d,4,s,l)\in \text{AGENT\_DAY\_ACTIVITY\_S\_L\_IN\_SEM} \\ t\in[s,s+l-1]}} l \cdot P_{a,d,4,s,l} \;\leq\; 4$$

– La Contrainte du nombre maximum de gestion pour une longue après-midi :
$\forall (a,d,t) \in$ AGENT_DAY_CRE_APRESMIDI :

$$\sum_{\substack{s,\,l \\ (a,d,4,s,l)\in \text{AGENT\_DAY\_ACTIVITY\_S\_L\_IN\_SEM} \\ t\in[s,s+l-1]}} l \cdot P_{a,d,4,s,l} \;\geq\; \begin{cases} 3 & \text{si } n_{a,d} \geq 12 \\ 0 & \text{sinon} \end{cases}$$

avec

$$n_{a,d} = |\{c \mid (a,d,c) \in \text{AGENT\_DAY\_CRE\_APRESMIDI}\}|$$

∘ La contrainte sur le minimum d'Edecla
$\forall (a,d) \in$ AGENT_DAY_CRE_IN_SEM_a_d :

$$\sum_{\substack{s,\,l \\ (a,d,3,s,l)\in \text{AGENT\_DAY\_ACTIVITY\_S\_L\_IN\_SEM}}} l \cdot P_{a,d,3,s,l} \;\geq\; \begin{cases} 2 & \text{si } n_{a,d} > 10 \\ 0 & \text{sinon} \end{cases}$$

∘ La contrainte sur l'Edecla entre les créneaux de la pause :

$\forall d \in$ DAYS_IN_SEM, $\forall t \in$ CRENEAU tels que $11 \leq t \leq 12$:

$$\sum_{\substack{a,\,s,\,l \\ (a,d,3,s,l)\in \text{AGENT\_DAY\_ACTIVITY\_S\_L\_IN\_SEM} \\ t\in[s,s+l-1]}} P_{a,d,3,s,l} \leq 8$$

- Contraintes de la Gestion et de L'Edecla sur le Samedi

$$\forall (a,d,t) \in \text{AGENT\_DAY\_CRE\_SAMEDI\_MATIN} :$$

$$\sum_{\substack{s,l \\ (a,d,3,s,l) \in \text{AGENT\_DAY\_ACTIVITY\_S\_L\_IN\_SAM}}} Z_{a,d,3,s,l} = 1$$

$$\forall (a,d,t) \in \text{AGENT\_DAY\_CRE\_SAMEDI\_MATIN} :$$

$$\sum_{\substack{s,l \\ (a,d,4,s,l) \in \text{AGENT\_DAY\_ACTIVITY\_S\_L\_IN\_SAM}}} Z_{a,d,4,s,l} = 1$$

6.4 Solveurs

Pour la résolution de notre problème d'optimisation binaire, nous avons utilisé le langage Python avec l'outil de modélisation Pyomo. Plusieurs solveurs ont été testés afin d'évaluer leur efficacité et leurs limites dans ce contexte :

— **GLPK (GNU Linear Programming Kit)** : il s'agit d'un solveur open-source adapté à la programmation linéaire en nombres entiers. Il est facile à utiliser et bien intégré à Pyomo. Cependant, il a montré rapidement ses limites pour les problèmes de grande taille ou comportant de nombreuses variables binaires, avec des temps de résolution qui peuvent devenir très longs, voire des échecs à trouver une solution optimale.

— **CBC (COIN-OR Branch and Cut)** : Ce solveur également open-source est spécialisé dans la programmation linéaire en nombres entiers. CBC est généralement plus performant que GLPK sur les problèmes binaires, mais rencontre des difficultés à converger sur de gros modèles, notamment en termes de mémoire et de temps de calcul sur des instances complexes.

— **SCIP** : Ce solveur est réputé pour ses performances sur les problèmes combinatoires et d'optimisation binaire. Toutefois, la version open-source de SCIP est limitée en fonctionnalités avancées et en vitesse par rapport à la version commerciale. De plus, son intégration avec l'environnement Python demande plus de configuration.

— **Gurobi** : Ce solveur commercial très performant, reconnu pour sa rapidité, a été efficace et tient sa robustesse sur les problèmes d'optimisation

binaire de grande taille. Il propose de nombreuses fonctionnalités avancées comme le pré-processing, les heuristiques et la parallélisation. La principale limite de Gurobi reste sa licence, qui est payante, mais étant étudiant, j'ai eu la capacité de pouvoir l'utiliser pour tester mon modèle.

7 Démarches et Résultats

Je présente les démarches et les résultats à la résolution du problème : Dans un premier temps, on importe dans les dataframes nécessaires contenant les informations dont nous avons besoin pour implémenter nos contraintes, on upload dans le Datalab :

- La table des Agents
- La table des Absences
- La table de paramétrages des taux de téléphonie et ceux de la gestion et de l'édecla

Une fois ces données disponibles dans le Datalab, nous pouvons créer les sous-ensembles nécessaires, puis traduire les contraintes en des fonctions qui sont facilement reprises par le module Pyomo.

7.1 Implémentation

```python
def MaxActivityJourT(model, a, d ) :
    # je recupère l'array des valeurs de AGENT_DAY_CRE_IN_SEM
    t  = np.array(list(model.AGENT_DAY_CRE_IN_SEM))
    # je verifie que l'agent appartient en faite au set de  AGENT_DAY_CRE_IN_SEM
    sous_ensemble =  [(a_, d_) for  (a_, d_, t) in model.AGENT_DAY_CRE_IN_SEM]
    if (a, d ) in sous_ensemble :
        # on calcul la somme
        somme  = sum (
            model.P[a_, d_, i_, s_, l_ ]
            for (a_, d_, i_, s_, l_) in model.AGENT_DAY_ACTIVITY_S_L_IN_SEM
            if (a_ == a and d_ == d  and i_ == 1 )
        )
        return  somme  <=  2
    else  :
        return Constraint.Skip


"""Je defins maintenant les fonctions de la contrainte de la Téléphonie, les contriantes spécifiques
à la téléphonie"""



def TelephonieCon_rule(model, d, t, activity2_minpercent):

    # Conversion de activity1_minpercent en dictionnaire
    activity2_minpercent['day'] = activity2_minpercent['day'].astype(str)
    activity1_minpercent_dict = activity2_minpercent.set_index(['day', 'Creneau'])['value'].to_dict()

    # j'arrondis les valeurs pour avoir les resultats voulues, il arrondit
    # juste les virgules.
    activity1_minpercent_round = {
        key: round(value, 3)
        for key, value in activity1_minpercent_dict.items()
    }
    min_percent = activity1_minpercent_round.get((str(d), t))


    # Somme des activités de téléphonie (i == 1) pour le jour `d` et créneau `t`
    telephonie_sum = sum(
        model.P[a_, d_, i_, s_, l_]
        for (a_, d_, i_ , s_, l_) in model.AGENT_DAY_ACTIVITY_S_L_IN_SEM
        if  t in range(s_, s_ + l_) and i_  == 1 and d_ == d
    )
    # Somme totale des activités pour le jour `d` et créneau `t`
    total_sum = sum(
        model.P[a_, d_, i_, s_, l_]
        for (a_, d_, i_, s_, l_) in model.AGENT_DAY_ACTIVITY_S_L_IN_SEM
        if t in range(s_, s_ + l_) and d_ == d
    )

    expr = (telephonie_sum >= min_percent * 1 * total_sum)
    # Si l'expression est triviale (True), on retourne Constraint.Feasible
    if not isinstance(expr, bool):
        return expr
    else:
        return Constraint.Skip
```

FIGURE 19 – Implementation des contraintes

```python

def NoConsecutiveWithSameActivity(model, a, d, i, s1, l1, s2, l2) :
    # j'implémente la contrainte de la contrainte consecutive,
    #  en gros il ne doit pas avoir des activités consécutives après un long creneaux
    # d'actvités.
    if (a, d, i, s1, l1) not in model.P or (a, d, i, s2, l2) not in model.P:
        print("skip")
        return Constraint.Skip
    return  model.P[a, d, i, s1, l1] + model.P[a, d, i ,s2, l2] <= 1


#  GESTION:
# Equité de Gestion de 31 % par semaine propre à chaque Gestionnaire.
#
def index_Gestion(model, Creneaux_G_Applique):
    """
    Cette contrainte définie l'equité propre à chaque Gestionnaire pour la semaine, par
    rapport à son creneaux pré-calculer
    """
    # je recupère les agents premièrement
    agents = Creneaux_G_Applique['MEMOID']
    # je convertis en numpy array
    agents = np.array(agents)
    # je cree le model pyomo de agents
    model.AGENTS = Set(dimen  = 1,
                       initialize = agents
                       )
    return  model.AGENTS


def equity_Gestion(model, a, Creneaux_G_Applique):
    """
    Cette contrainte définit l'équité de gestion pour chaque agent.
    Elle vérifie que le nombre total de créneaux attribués à un agent
    est égal à la valeur spécifiée dans Creneaux_G_Applique_dict.
    """
    creneaux_dict = Creneaux_G_Applique.set_index('MEMOID')['Creneau_G_Applique'].to_dict()

    return sum (
        l * model.P[a_, d, 4, s, l]
        for (a_, d, i, s, l) in model.AGENT_DAY_ACTIVITY_S_L_IN_SEM
        if (i == 4 and a_ == a)
    ) ==  creneaux_dict[a]


# je definis le Max de Gestion Journalier

# Pour débuger afficher des prints
def MaxGestionJour(model, a, d, AGENT_TAUX_G_NEC) :

    # je convertis AGENT_TAUX_G_NEC en dictionnaire
    AGENT_TAUX_G_NEC_dict = AGENT_TAUX_G_NEC.set_index('MEMOID')['Taux_Gestion_theorique'].to_dict()
    # on calcul le nombre de créneaux disponible pour l'agent et le jour
    #  on fixe l'agent et on compte par agent et jour
    t  = np.array(list(model.AGENT_DAY_CRE_IN_SEM))  #str, str, str
    # on va proceder ainsi, quand je recois (a, d) en entrée
    # je vais le chercher dans model.AGENT_DAY_CRE_IN_SEM
    # je recupère le sous ensemble agent , jour
    sous_ensemble = [(a, d) for (a, d, t) in model.AGENT_DAY_CRE_IN_SEM]
    if (a, d) in sous_ensemble:  #str, int
        # je compte les créneaux de l'agent pour ce jour
        filter = t[(t[:, 0] == a) & (t[:, 1] == str(d))]
        # je compte le nombre de créneaux
        # print(filter)
        cardi_cren = len(filter)
        # print("cardi : " , cardi_cren)
        # j'implémente maintenant le formalisme de la limite pour la contriante
        if (AGENT_TAUX_G_NEC_dict[a] * cardi_cren < 2 ) and (AGENT_TAUX_G_NEC_dict[a] > 0) :
            limite  =  2
        else :
            limite = ceil(AGENT_TAUX_G_NEC_dict[a] * 1.3 * cardi_cren)
        somme  =  sum(
            l_ * model.P[a_, d_, i_, s_, l_]
            for (a_, d_, i_ , s_, l_) in model.AGENT_DAY_ACTIVITY_S_L_IN_SEM # str, str, str, str, str
            if (a_ == a and d_ == d and  i_ == 4)
            )
        return somme <= limite
    else :
        # il est absent
        return Constraint.Skip

```

FIGURE 20 – Implementation des contraintes

```python
def MinGestionJour(model, a, d, AGENT_TAUX_G_NEC) :
    # je convertis AGENT_TAUX_G_NEC en dictionnaire
    AGENT_TAUX_G_NEC_dict = AGENT_TAUX_G_NEC.set_index('MEMOID')['Taux_Gestion_theorique'].to_dict()
    # on calcul le nombre de créneaux disponible pour l'agent et le jour
    #  on fixe l'agent et on compte par agent et jour
    t  = np.array(list(model.AGENT_DAY_CRE_IN_SEM))  #str, str, str
    # on va proceder ainsi, quand je recois (a, d) en entrée
    # je vais le chercher dans model.AGENT_DAY_CRE_IN_SEM
    # je recupère le sous ensemble agent , jour
    sous_ensemble = [(a, d) for (a, d, t) in model.AGENT_DAY_CRE_IN_SEM]
    if (a, d) in sous_ensemble:  #str, int
        # je compte les créneaux de l'agent pour ce jour
        filter = t[(t[:, 0] == a) & (t[:, 1] == str(d))]
        # je compte le nombre de créneaux
        # print(filter)
        cardi_cren = len(filter)
        # print("cardi : " , cardi_cren)
        # j'implémente maintenant le formalisme de la limite pour la contriante
        limite = floor( ((AGENT_TAUX_G_NEC_dict[a]) / 2 ) * cardi_cren)
        somme  =  sum(
            l_ * model.P[a_, d_, i_, s_, l_]
            for (a_, d_, i_ , s_, l_) in model.AGENT_DAY_ACTIVITY_S_L_IN_SEM # str, str, str, str, str
            if (a_ == a and d_ == d and  i_ == 4)
            )
        return somme >= limite
    else :
        # il est absent
        return Constraint.Skip


# Je definis maintenant les activités de Gestion par demi-Journée,
# la matinée comme l'après midi

# même chose pour l'après midi
def  CrenauxGestion(model, a, d, t, filtred) :
    #    je verifie bien que le gard appartient à l'élement
    # sinon je skip

    if (a, d, t) not in filtred :
        return Constraint.Skip
    else :

        return sum (
            l_ * model.P[a_, d_, i_, s_, l_]
            for (a_, d_, i_ , s_, l_) in filtred[(a, d, t)]
            if (i_ == 4)
        ) <= 4

# # Pour ceux qui ont de longue après midi
def CrenauxGestionLongueApresmidi(model, a, d, t, filtred) :
    #    je verifie bien que le gard appartient à l'élement
    t_array  = np.array(list(model.AGENT_DAY_CRE_APRESMIDI))
    # sinon je skip
    sous_ensemble = [(a_, d_) for (a_, d_, t) in model.AGENT_DAY_CRE_APRESMIDI]
    if (a, d) in sous_ensemble:  #str, int
        # je compte les créneaux de l'agent pour ce jour
        filter = t_array[(t_array[:, 0] == a) & (t_array[:, 1] == str(d))]
        print(filter[:6])
        if len(filter) >= 12 :
            limite  =  3
        else :
            limite = 0
        return sum (
            l_ * model.P[a_, d_, i_, s_, l_]
            for (a_, d_, i_ , s_, l_) in filtred[(a, d, t)]
            if (i_ == 4)
        ) <= limite
    else :
        # il est absent
        return Constraint.Skip

```

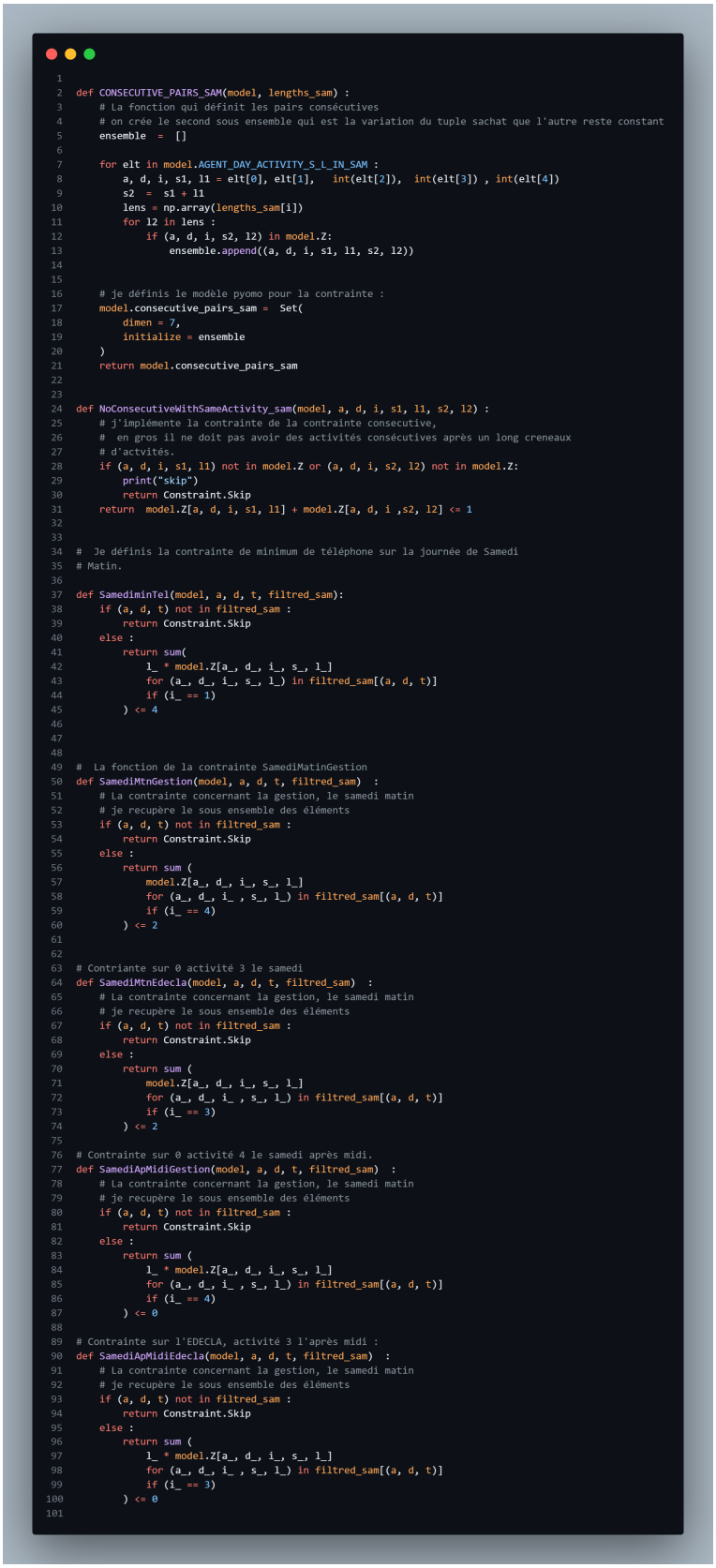
FIGURE 21 – Implementation des contraintes

```python

"""CONTRAINTES EDECLA"""
# la contriante des Agents par semaine

def equity_EDECLA(model, a, Creneaux_E_Applique):
    """
    Cette contrainte définit l'équité de gestion pour chaque agent.
    Elle vérifie que le nombre total de créneaux attribués à un agent
    est égal à la valeur spécifiée dans Creneaux_G_Applique_dict.
    """
    creneaux_dict = Creneaux_E_Applique.set_index('MEMOID')['Creneau_E_Applique'].to_dict()

    return sum (
        l * model.P[a_, d, i, s, l]
        for (a_, d, i, s, l) in model.AGENT_DAY_ACTIVITY_S_L_IN_SEM
        if (i == 3 and a_ == a)
    ) ==  creneaux_dict[a]




# Pour débuger afficher des prints
def MinEdeclaJour(model, a, d) :
    t  = np.array(list(model.AGENT_DAY_CRE_IN_SEM))  #str, str, str
    # on va proceder ainsi, quand je recois (a, d) en entrée
    # je vais le chercher dans model.AGENT_DAY_CRE_IN_SEM
    # je recupère le sous ensemble agent , jour
    sous_ensemble = [(a, d) for (a, d, t) in model.AGENT_DAY_CRE_IN_SEM]
    if (a, d) in sous_ensemble:  #str, int
        # je compte les créneaux de l'agent pour ce jour
        filter = t[(t[:, 0] == a) & (t[:, 1] == str(d))]
        # je compte le nombre de créneaux
        # print(filter)
        cardi_cren = len(filter)
        if (cardi_cren > 10 ) :
            limite  = 2
        else  :
            limite =  0

        somme  =  sum(
            l_ * model.P[a_, d_, i_, s_, l_]
            for (a_, d_, i_ , s_, l_) in model.AGENT_DAY_ACTIVITY_S_L_IN_SEM # str, str, str, str, str
            if (a_ == a and d_ == d and  i_ == 3)
            )
        return somme >= limite
    else :
        # il est absent
        return Constraint.Skip


# j'implémente l'activité dd'Edecla entre les créneaux de 12 et 13
def ActivityEdecla(model, d, t) :
    if t >= 11 and t <= 12 :
        somme = sum(
            model.P[a, d, i, s, l]
            for (a, d_, i, s, l) in model.AGENT_DAY_ACTIVITY_S_L_IN_SEM
            if d_ == d and i == 3 and t in range(s , s+l)
            )
        return somme <= 8
    else :
        return Constraint.Skip
```

FIGURE 22 – Implementation des contraintes

```python
def minTelJour (model, a, d)  :

    t  = np.array(list(model.AGENT_DAY_CRE_IN_SEM))  #str, str, str
    # on va proceder ainsi, quand je recois (a, d) en entrée
    # je vais le chercher dans model.AGENT_DAY_CRE_IN_SEM
    # je recupère le sous ensemble agent , jour
    sous_ensemble = [(a, d) for (a, d, t) in model.AGENT_DAY_CRE_IN_SEM]
    if (a, d) in sous_ensemble:  #(str, int)
        # je compte les créneaux de l'agent pour ce jour
        filter = t[(t[:, 0] == a) & (t[:, 1] == str(d))]
        # je compte le nombre de créneaux
        # print(filter)
        cardi_cren = len(filter)
        if (cardi_cren > 7 ) :
            limite  = 4
        somme  =  sum(
            l_ * model.P[a_, d_, i_, s_, l_]
            for (a_, d_, i_ , s_, l_) in model.AGENT_DAY_ACTIVITY_S_L_IN_SEM # str, str, str, str, str
            if (a_ == a and d_ == d and  i_ == 1)
            )
        return somme >= limite
    else :
        # il est absent
        return Constraint.Skip


"""Définition des contraintes de Max-activités par jour pour les différents activités """
# je definis la fonction qui permet d'instancier la contrainte de Maxactivity
def MaxactivityJourcon(model, a, d ) :
    # je recupère le sous_ensemble
    sous_ensemble = [(a_, d_) for (a_, d_, t) in model.AGENT_DAY_CRE_IN_SEM]
    if (a, d) in sous_ensemble :
        somme  =  sum(
            model.P[a_, d_, i_, s_, l_]
            for (a_, d_, i_ , s_, l_) in model.AGENT_DAY_ACTIVITY_S_L_IN_SEM # str, str, str, str, str
            if (a_ == a and d_ == d )
        )
        return somme <= 6
    else :
        return Constraint.Skip

# je definis la de Maxactivite Jour de Gestion sans la télephonie
# En gros c'est le switch d'activité outre celui de la Téléphonie.


def MaxActivityJourG(model, a, d ) :
    # je recupère l'array des valeurs de AGENT_DAY_CRE_IN_SEM
    t  = np.array(list(model.AGENT_DAY_CRE_IN_SEM))
    # je verifie que l'agent appartient en faite au set de  AGENT_DAY_CRE_IN_SEM
    sous_ensemble =  [(a_, d_) for  (a_, d_, t) in model.AGENT_DAY_CRE_IN_SEM]
    # maintenant, je verifie si le couple donné en haut est d'autant valable pour
    if (a, d) in sous_ensemble :
        # je vais filter pour avoir la présence de chaque gestionnaire  pour voir le
        # temps de présence :
        filter = t[(t[:, 0] == a) & (t[:, 1] == str(d))]
        # je compte le nombre de créneaux
        # print(filter)
        cardi_cren = len(filter)
        # je definis la variable limite
        limite = 6 if cardi_cren > 10 else 4
        somme  = sum (
            model.P[a_, d_, i_, s_, l_ ]
            for (a_, d_, i_, s_, l_) in model.AGENT_DAY_ACTIVITY_S_L_IN_SEM
            if (a_ == a and d_ == d  and i_ !=1 )
        )
        return  somme  <= limite
    else :
        return Constraint.Skip

```

FIGURE 23 – Implementation des contraintes



```python

def CONSECUTIVE_PAIRS_SAM(model, lengths_sam) :
    # La fonction qui définit les pairs consécutives
    # on crée le second sous ensemble qui est la variation du tuple sachat que l'autre reste constant
    ensemble  =  []

    for elt in model.AGENT_DAY_ACTIVITY_S_L_IN_SAM :
        a, d, i, s1, l1 = elt[0], elt[1],   int(elt[2]),  int(elt[3]) , int(elt[4])
        s2  =  s1 + l1
        lens = np.array(lengths_sam[i])
        for l2 in lens :
            if (a, d, i, s2, l2) in model.Z:
                ensemble.append((a, d, i, s1, l1, s2, l2))


    # je définis le modèle pyomo pour la contrainte :
    model.consecutive_pairs_sam =  Set(
        dimen = 7,
        initialize = ensemble
    )
    return model.consecutive_pairs_sam


def NoConsecutiveWithSameActivity_sam(model, a, d, i, s1, l1, s2, l2) :
    # j'implémente la contrainte de la contrainte consecutive,
    #  en gros il ne doit pas avoir des activités consécutives après un long creneaux
    # d'actvités.
    if (a, d, i, s1, l1) not in model.Z or (a, d, i, s2, l2) not in model.Z:
        print("skip")
        return Constraint.Skip
    return  model.Z[a, d, i, s1, l1] + model.Z[a, d, i ,s2, l2] <= 1


#  Je définis la contrainte de minimum de téléphone sur la journée de Samedi
# Matin.

def SamediminTel(model, a, d, t, filtred_sam):
    if (a, d, t) not in filtred_sam :
        return Constraint.Skip
    else :
        return sum(
            l_ * model.Z[a_, d_, i_, s_, l_]
            for (a_, d_, i_, s_, l_) in filtred_sam[(a, d, t)]
            if (i_ == 1)
        ) <= 4



#  La fonction de la contrainte SamediMatinGestion
def SamediMtnGestion(model, a, d, t, filtred_sam)  :
    # La contrainte concernant la gestion, le samedi matin
    # je recupère le sous ensemble des éléments
    if (a, d, t) not in filtred_sam :
        return Constraint.Skip
    else :
        return sum (
            model.Z[a_, d_, i_, s_, l_]
            for (a_, d_, i_ , s_, l_) in filtred_sam[(a, d, t)]
            if (i_ == 4)
        ) <= 2


# Contriante sur 0 activité 3 le samedi
def SamediMtnEdecla(model, a, d, t, filtred_sam)  :
    # La contrainte concernant la gestion, le samedi matin
    # je recupère le sous ensemble des éléments
    if (a, d, t) not in filtred_sam :
        return Constraint.Skip
    else :
        return sum (
            model.Z[a_, d_, i_, s_, l_]
            for (a_, d_, i_ , s_, l_) in filtred_sam[(a, d, t)]
            if (i_ == 3)
        ) <= 2

# Contrainte sur 0 activité 4 le samedi après midi.
def SamediApMidiGestion(model, a, d, t, filtred_sam)  :
    # La contrainte concernant la gestion, le samedi matin
    # je recupère le sous ensemble des éléments
    if (a, d, t) not in filtred_sam :
        return Constraint.Skip
    else :
        return sum (
            l_ * model.Z[a_, d_, i_, s_, l_]
            for (a_, d_, i_ , s_, l_) in filtred_sam[(a, d, t)]
            if (i_ == 4)
        ) <= 0

# Contrainte sur l'EDECLA, activité 3 l'après midi :
def SamediApMidiEdecla(model, a, d, t, filtred_sam)  :
    # La contrainte concernant la gestion, le samedi matin
    # je recupère le sous ensemble des éléments
    if (a, d, t) not in filtred_sam :
        return Constraint.Skip
    else :
        return sum (
            l_ * model.Z[a_, d_, i_, s_, l_]
            for (a_, d_, i_ , s_, l_) in filtred_sam[(a, d, t)]
            if (i_ == 3)
        ) <= 0

```

FIGURE 24 – Implementation des contraintes

```python
def MaxActivityJourT(model, a, d ) :
    # je recupère l'array des valeurs de AGENT_DAY_CRE_IN_SEM
    t  = np.array(list(model.AGENT_DAY_CRE_IN_SEM))
    # je verifie que l'agent appartient en faite au set de  AGENT_DAY_CRE_IN_SEM
    sous_ensemble =  [(a_, d_) for  (a_, d_, t) in model.AGENT_DAY_CRE_IN_SEM]
    if (a, d ) in sous_ensemble :
        # on calcul la somme
        somme  = sum (
            model.P[a_, d_, i_, s_, l_ ]
            for (a_, d_, i_, s_, l_) in model.AGENT_DAY_ACTIVITY_S_L_IN_SEM
            if (a_ == a and d_ == d  and i_ == 1 )
        )
        return  somme  <=  2
    else  :
        return Constraint.Skip


"""Je defins maintenant les fonctions de la contrainte de la Téléphonie, les contriantes spécifiques
à la téléphonie"""



def TelephonieCon_rule(model, d, t, activity2_minpercent):

    # Conversion de activity1_minpercent en dictionnaire
    activity2_minpercent['day'] = activity2_minpercent['day'].astype(str)
    activity1_minpercent_dict = activity2_minpercent.set_index(['day', 'Creneau'])['value'].to_dict()

    # j'arrondis les valeurs pour avoir les resultats voulues, il arrondit
    # juste les virgules.
    activity1_minpercent_round = {
        key: round(value, 3)
        for key, value in activity1_minpercent_dict.items()
    }
    min_percent = activity1_minpercent_round.get((str(d), t))


    # Somme des activités de téléphonie (i == 1) pour le jour `d` et créneau `t`
    telephonie_sum = sum(
        model.P[a_, d_, i_, s_, l_]
        for (a_, d_, i_ , s_, l_) in model.AGENT_DAY_ACTIVITY_S_L_IN_SEM
        if  t in range(s_, s_ + l_) and i_  == 1 and d_ == d
    )
    # Somme totale des activités pour le jour `d` et créneau `t`
    total_sum = sum(
        model.P[a_, d_, i_, s_, l_]
        for (a_, d_, i_, s_, l_) in model.AGENT_DAY_ACTIVITY_S_L_IN_SEM
        if t in range(s_, s_ + l_) and d_ == d
    )

    expr = (telephonie_sum >= min_percent * 1 * total_sum)
    # Si l'expression est triviale (True), on retourne Constraint.Feasible
    if not isinstance(expr, bool):
        return expr
    else:
        return Constraint.Skip
```

FIGURE 25 – Implementation des contraintes

Pour résoudre ce problème, Pyomo offre deux choix de modèle :

- Concrete Model : C'est un modèle où les données entrantes (ensembles,

paramètres, ..) sont directement définies dans le script, on l'utilise lorsqu'on dispose de la totalité des données au début du problème d'optimisation..

- Abstract Model : Ce modèle contrairement à l'autre est défini sans spécifier les données dans le script, elles sont fournies par un fichier *.dat* qui peut être soit un dictionnaire ou un set, lors de la création ou de la résolution du modèle. Ce modèle offre la flexibilité et réutilisable.

Dans le cadre de notre problème, nous avons opté pour **Concrete Model**, je l'ai défini ainsi :

```
model = ConcreteModel()
```

Le choix du solveur pour résoudre notre problème se fait à travers le module *SolverFactory* suivant cette démarche :

```
solver = SolverFactory('solveur')
results = solver.solve(model, tee=True)
```

la solution est stockée dans ***results*** et le paramètre ***tee*** nous permet d'avoir l'arborescente (l'arbre) de résolution.

7.2 Résultats

Le résultat est formaté en un fichier Excel qui est la solution brute puis ensuite, on fait passer une macro VBA qui permet aux chargés de sinsitres de pouvoir consulter leurs plannings. On dispose de trois sortes de Plannings :

- Planning Global : contenant tous les chargés planifiés par jour et par créneaux montrant le nombre de personnes planifiées et le nombre de personnes attendu.

Semaine A		8h		09h		10h		11h		12h		13h		14h		15h		16h		17h		18h	
		0	30	0	30	0	30	0	30	0	30	0	30	0	30	0	30	0	30	0	30	0	30
Equipes		42	62	83	87	91	92	94	92	52	43	35	49	84	86	91	91	72	58	47	47	0	0
		31	38	46	39	40	41	39	42	27	23	12	15	49	50	38	41	41	42	30	23	0	0
		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
		17	22	22	28	28	25	25	21	16	12	17	21	24	22	30	27	25	15	15	9	0	0
		0	0	1	1	1	1	1	1	1	1	0	0	1	1	1	1	1	1	0	0	0	0
		40	60	80	83	87	88	90	88	50	42	33	47	80	82	87	87	69	55	45	45	0	0
2X		C	C	C	C	C	C	C	C	C	C			C	C	C	C	C	C				
2P																							
0R																							
V7																							
2X		G	G	E	E	T	T	T	T			T	T	T	T	T	G	G					
2U		E	E	T	T	T	T	T			G	G	G	T	T	T	T						
2F			M	M	M	M	M	M	M	M	M			M	M	M	M	M	M				
2A				E	E	E	T	T	T	T	T			G	G	G	T	T	T	T	T		
V7					E	E	T	T	T	T				G	G	G	T	T	T	T	T		
1U																							
0B		T	T	T	T	T	G	G	E	E			E	E	G	G	T	T	T	T	T		
2U		T	T	T	T	E	E	G	G			G	G	T	T	T	T						
2U		E	E	G	G	T	T	T	T			T	T	T	T	T	E	E					
1B																							
0L																							
1U																							
0R		T	T	T	G	G	G	E	E			T	T	T	T	T	G	G	G	G			
2R			G	G	G	G	T	T	T	T	T			T	T	T	T	T	E	E			
2U																							
2U		T	T	T	G	G	T	T	T	T	T			G	G	G	E	E	G	G	G		
2A																							
0B			T	T	T	T	E	E			E	E	T	T	T	T	T	G	G	G			
2H																							
2S		E	E	G	G	G	G	T	T	T	T			G	G	G	T	T	T	T	T		
1H		T	T	T	T	E	E	G	G	G	G			G	G	T	T	T					
2H		G	G	E	E	G	G	T	T	T	T			G	G	G	T	T	T	T	T		
0R																							
2P		T	T	T	T	T	G	G	G			E	E	T	T	T	T						
2A		G	G	T	T	T	T	T	E	E			E	E	T	T	T	T	T	G	G		
2E																							

FIGURE 26 – Aperçu du Planning Global pour la journée du Lundi

- Planning Équipe : La répartition par équipe permettant au chef d'équipe d'avoir un visuel sur leur équipe.

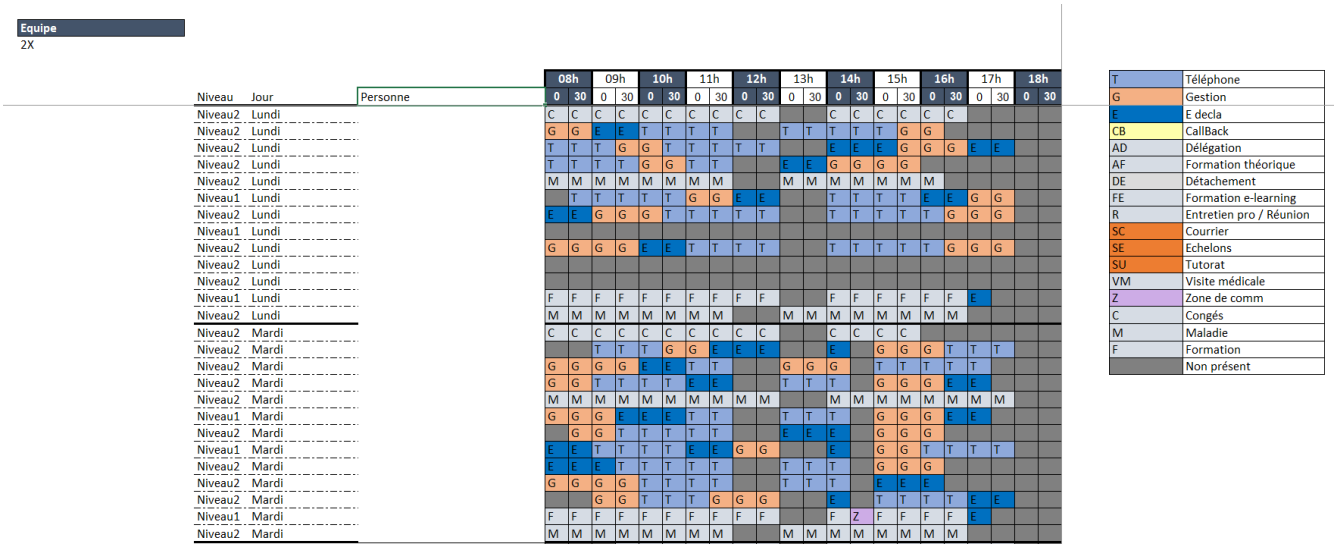


FIGURE 27 – Aperçu du Planning par Equipe

- Planning Individuel : Ce sont les plannings personnels à chaque gestionnaire, consultable à volonté.

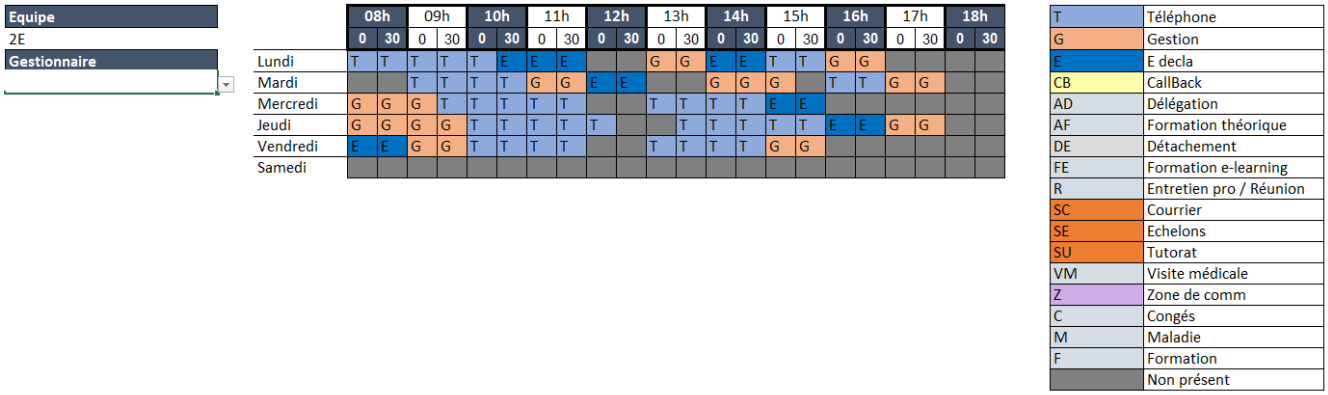


FIGURE 28 – Aperçu du Planning Individuel

Une fois les planning générés, ils sont envoyés aux responsables d'équipe pour une dernière vérification et la prise en compte des absences de dernière minute.
En plus de la qualité des résultats obtenus, il est important de souligner que le développement sous python offre une solution de secours fiable, permettant d'assurer la continuité des travaux en cas de problèmes de l'outil principal.

8 Application du Machine Learning pour la prévision du temps d'attente

Dans la seconde partie de mon stage, j'ai travaillé sur un sujet de modélisation et de prédiction de temps d'attente moyen des assurés avant d'être pris en charge. L'enjeu consiste à anticiper les périodes de fortes affluences et d'optimiser la gestion des chargés des sinistres, mais aussi de permettre aux chefs d'équipe d'adapter une stratégie adéquate pour le traitement des dossiers des assurés.

8.1 Le jeu de données

Le jeu de données utilisé, provient des historiques des appels des chargés de sinistres avec les assurés. Il contient plusieurs informations permettant d'analyser et prédire le temps d'attente moyen des assurées. Les données analysées remontent à l'année 2023.
Les principales variables présentes sont :

Nom de la variable	Description
`nbapp`	Nombre d'appels reçus
`nbrep`	Nombre d'appels répondus
`taux_decroche`	Taux de décroche (proportion d'appels pris en charge)
`delai_PEC_EDECLA`	Délai de prise en charge des e-déclarations
`taux_reappel`	Taux de rappel des clients
`taux_nonrappel`	Taux de non-rappel
`temps_att_moy`	Temps d'attente moyen (variable cible)
`jour_semaine`	Jour de la semaine
`semaine`	Numéro de la semaine
`annee`	Année

TABLE 2 – Description des principales variables du jeu de données

8.2 Préparation et exploration des données

La préparation et l'exploration des données constituent une étape cruciale dans tout projet d'apprentissage automatique, car elles permettent de mieux comprendre la structure et les caractéristiques du jeu de données.

Dans un premier temps, j'ai traité les valeurs manquantes en adoptant une double approche : d'une part, en remplaçant les valeurs manquantes par la

médiane des colonnes concernées, et d'autre part, en supprimant les lignes comportant des valeurs manquantes pour certaines analyses spécifiques. J'ai également vérifié la présence de doublons dans le jeu de données et, le cas échéant, je les ai supprimés afin de garantir la fiabilité des analyses.

Par la suite, j'ai étudié les corrélations entre les différentes variables explicatives et la variable cible, afin d'identifier les facteurs ayant le plus d'influence sur le temps d'attente moyen.

8.3 Modélisation

J'ai principalement utilisé deux modèles de machines learning pour effectuer la prédiction. il s'agit de la régression linéaire et la méthode de Random Forest Regressor.
La **régression linéaire** est une technique statistique utilisée pour déterminer la relation entre les variables. Dans notre contexte, la régression linéaire permet de trouver la relation entre les caractéristiques (variables explicatives) et une étiquette(variable cible).
Le **Random Forest Regressor**, quant à lui, est un modèle d'ensemble basé sur la combinaison de plusieurs arbres de décision. Il est capable de capturer des relations complexes et non linéaires entre les variables, et il est généralement plus robuste face aux variations des données. Dans notre cas, les caractéristiques prises en compte, sont : **le nombre d'appels reçus, le taux de décroche et le taux de reappel.**
Mathématiquement :
Pour le random forest regressor, la prédiction du temps d'attente moyen $\hat{y}$ pour une observation $x = (x_1, x_2, x_3)$, où :

- x_1 = nombre d'appels reçus (`nbapp`),
- x_2 = taux de décroche (`taux_decroche`),
- x_3 = taux de réappel (`taux_reappel`),

est donnée par la moyenne des prédictions de M arbres de décision :

$$\hat{y} = \frac{1}{M} \sum_{m=1}^{M} h_m(x_1, x_2, x_3)$$

où :

- M est le nombre total d'arbres dans la forêt,
- $h_m(x_1, x_2, x_3)$ est la prédiction du m-ième arbre pour les valeurs (x_1, x_2, x_3),

- $\hat{y}$ est la prédiction finale du temps d'attente moyen.

Pour la régression linéaire, la prédiction s'écrit :

$$\hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3$$

où x_1, x_2, x_3 sont respectivement le nombre d'appels reçus, le taux de décroche et le taux de rappel, et β_i les coefficients appris par le modèle.
Mon jeu de données a été divisé en deux ensembles distincts : un ensemble d'entraînement (80%) pour entraîner le modèle et un ensemble de test (20%) pour évaluer le modèle sur les données jamais vues.

8.4 Évaluation des modèles

Pour l'évaluation de mon modèle, j'ai utilisé deux métriques :

- **RMSE (Root Mean Squared Error)** : elle mesure la différence moyenne entre les valeurs prédites et les valeurs observées. Plus la RMSE est faible, plus le modèle est précis. Mathématiquement, elle s'écrit :

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

où y_i est la valeur réelle, $\hat{y}_i$ la valeur prédite et n le nombre d'observations.

- **Score R^2** : elle indique la proportion de la variance de la variable cible expliquée par le modèle. Le score R^2 est compris entre 0 et 1 : plus il est proche de 1, plus la prédiction est précise. S'il est proche de 0, le modèle n'explique pas la variabilité des données. Sa formule est :

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

où $\bar{y}$ est la moyenne des valeurs réelles.
Je présente le tableau comparatif des deux modèles sous la base des deux métriques :

Modèle	**RMSE (test)**	**R^2 (test)**
Random Forest Regressor	0.9256	0.9549
Régression linéaire	0.7971	0.9683

TABLE 3 – Comparaison des performances des modèles sur le jeu de test

On constate bien que les deux modèles choisis arrivent bien à capter notre jeu de données, la différences entre eux est minime.

On peut aussi représenter par un graphique les valeurs prédites par les deux modèles et les variables réelles, on observe alors une dégradation du modèle de Random forest lors des pics.

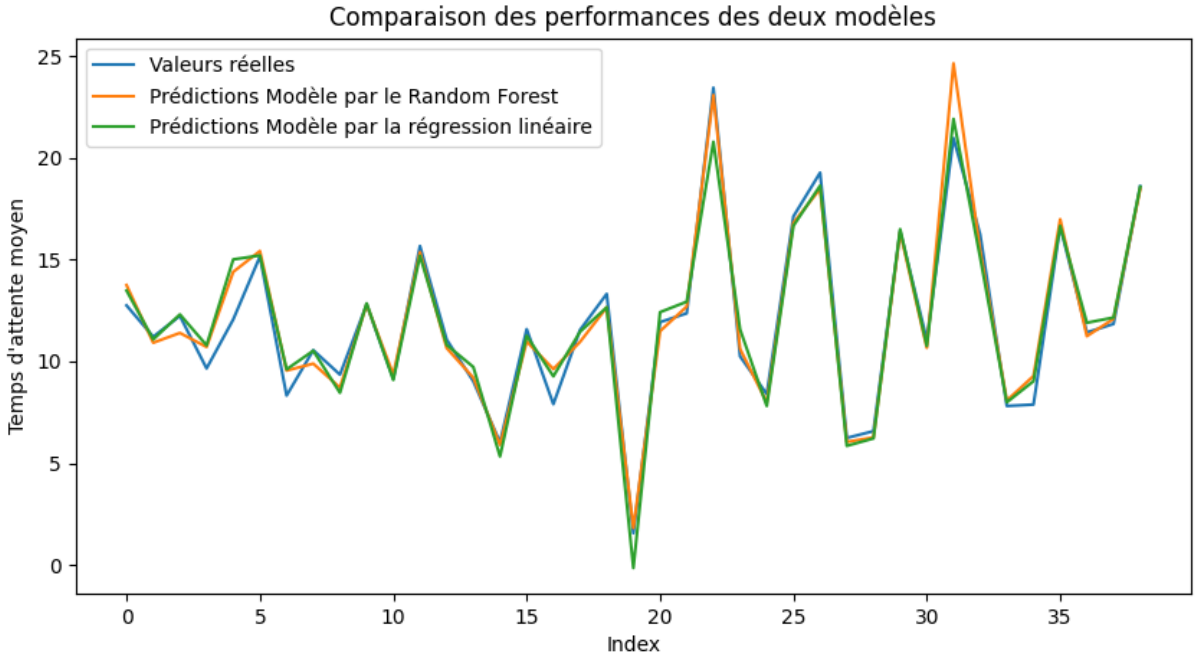


FIGURE 29 – Comparaison des valeurs réelles et des prédictions des deux modèles.

8.5 Application

Ce travail permet aux responsables d'équipe d'estimer le temps d'attente moyen afin de prendre des décisions stratégiques qui amélioreront la prise en charge des assurés. Pour cela, le modèle a été déployé et est utilisable en lui fournissant trois variables explicatives, à savoir : **le nombre d'appels reçus, le taux de décroche et le taux de réappel**. Le modèle prédit alors le temps d'attente moyen des appels du service.

J'ai donc déployé le modèle sous forme d'une fonction où l'on peut préciser le modèle à utiliser et entrer les variables explicatives.

```python
def predict_with_multiple_features(model, input_data):

    """
    Prédiction du temps d'attente moyen des jours de la semaine.
    Parameters:
    model  : le model à utiliser pour la prédiction (RandomForestRegressor ou LinearRegression)
    input data : DataFrame contenant les données d'entrée avec 5 lignes et 3 colonnes.
    Returns:
    DataFrame contenant les prédictions du temps d'attente moyen pour chaque jour de la semaine.
    """



    # Verfication des dimensions de input_data
    if input_data.shape[0] != 5 or input_data.shape[1] != 3:
        raise ValueError("Input data must have 5 rows and 3 columns.")


    # Vérification des shapes
    # print(f"Shape of input_data: {input_data.shape}")

    # la prédiction
    predictions = model.predict(input_data)

    #Conversion des prédictions en DataFrame avec une colonne nommée "Temps d'attente moyen"

    jours = ['Lundi', 'Mardi', 'Mercredi', 'Jeudi', 'Vendredi']
    predictions = pd.DataFrame(predictions, columns=["Temps d'attente moyen"])
    predictions.index = jours[:len(predictions)]
    predictions.index.name = 'jour_semaine'

    # vérifions le shape des prédictions
    print(f"Shape of predictions: {predictions.shape} et du type : {predictions.dtypes}")


    # retour les prédictions
    return predictions
```

FIGURE 30 – Implémentation de la fonction predict

Par exemple, pour les données suivantes, on a

Jour	nbapp	taux_decroche	taux_reappel	Tps RF	Tps RL
Lundi	5591	0,61	0,23	12.74	12.97
Mardi	5857	0,67	0,20	10.04	10.29
Mercredi	5419	0,70	0,20	8.99	9.21
Jeudi	5368	0,59	0,23	13.60	13.80
Vendredi	0	0,00	0,00	0	0

TABLE 4 – Exemple de données d'entrée et temps d'attente moyen prédit par les deux modèles

9 Difficultés rencontrées

Au cours de ces six mois passionnants au sein des Assurances du Crédit Mutuel , j'ai eu l'opportunité d'acquérir de nombreuses compétences techniques liées à la recherche opérationnelle, ainsi que de développer des compétences relationnelles. Cependant, cette aventure n'a pas été sans difficultés.

L'un des principaux défis rencontrés durant mon stage fut mon adaptation au métier : il a fallu comprendre en profondeur le processus métier. Les contraintes métiers sont particulièrement sensibles et doivent être adaptées aux besoins spécifiques du secteur. La difficulté résidait dans le fait de saisir précisément les attentes du métier, de les traduire en formulations mathématiques, puis de les implémenter afin de proposer une solution opérationnelle. Pour surmonter cette difficulté, j'ai multiplié les échanges et les collaborations avec les collègues. Des réunions régulières et suivis constants m'ont permis de mieux cerner les attentes, d'ajuster la formulation des contraintes et de garantir la pertinence des solutions proposées.

Outre l'adaptation au métier, un autre défi majeur est apparu au fil du développement : la disponibilité et l'intégration des solveurs pour effectuer les différents tests. Il a parfois été complexe de trouver le bon équilibre entre les exigences du modèle, la performance des solveurs et leur capacité à répondre aux besoins du projet. Afin de dépasser cet obstacle, j'ai mené une veille documentaire et réalisé des tests comparatifs sur différents solveurs. Cette démarche m'a permis d'identifier les outils les plus adaptés, d'optimiser l'implémentation du modèle et d'assurer des résultats fiables et rapides. L'auto-formation sur les outils d'optimisation et la documentation régulière de chaque étape m'ont également aidé à progresser plus efficacement.

Par ailleurs, l'optimisation binaire, notamment la planification des tâches, est un sujet particulièrement complexe pour lequel il n'existe pas toujours de solution exacte. Lorsque le modèle ne fonctionnait pas comme prévu, cela m'a permis de développer de la patience et de la persévérance, notamment dans la phase de débogage et d'amélioration du modèle.

10 Développement futur

Au vu des travaux réalisés, une première version fonctionnelle a pu être mise en place grâce aux outils open source disponibles en Python et ses librairies. Toutefois, l'un des axes majeurs qui reste à explorer concerne l'explicitation des contraintes rendant le modèle infaisable. En effet, dans

l'état actuel, la solution développée ne permet pas d'identifier directement la contrainte problématique lorsque le modèle est infaisable. Un développement futur consisterait à intégrer un mécanisme permettant de détecter et de calculer le seuil de faisabilité du modèle. Cela réduirait considérablement le temps de débogage en facilitant l'identification des contraintes à ajuster ou à reformuler.

Un autre axe d'amélioration serait de travailler à concilier le confort d'utilisation pour le gestionnaire avec l'atteinte des objectifs fixés par le métier. L'objectif serait de proposer une solution ergonomique, adaptée aux besoins opérationnels tout en garantissant la performance du modèle.

Enfin, travailler sur la robustesse de la solution afin de permettre son extension à d'autres départements ou domaines d'activité. Cela passerait par une généralisation du modèle et une adaptation aux spécificités des différents secteurs concernés.

11 Conclusion

Ce projet de planification au sein des assurances m'a permis d'acquérir de solides compétences techniques en optimisation et en recherche opérationnelle, tout en développant des qualités relationnelles essentielles à la réussite d'un projet collaboratif. Les défis rencontrés, qu'ils soient liés à la compréhension du métier, à la gestion des contraintes ou à l'intégration des outils informatiques, ont été autant d'opportunités d'apprentissage et de progression.

La solution développée constitue une première étape vers une organisation du travail plus efficace et plus adaptée aux besoins des équipes. Les axes d'amélioration identifiés, notamment l'explicitation des contraintes problématiques et la robustesse du modèle, ouvriront la voie à de futurs développements permettant d'étendre et de renforcer cet outil au sein de l'entreprise. Ce stage a ainsi été une expérience enrichissante, tant sur le plan professionnel que personnel, et pose les bases d'une optimisation continue au service de la performance et du bien-être au travail.

Références

[1] COIN-OR. Cbc user guide. URL : https://coin-or.github.io/Cbc/intro.html.

[2] Axys Consultants. Data science et service de planification sous contraintes. URL : https://www.axys-consultants.com/publications/articles/data-science-service-de-planification-contraintes-2/.

[3] DataCamp. Qu'est-ce que la rmse ? URL : https://www.datacamp.com/fr/tutorial/rmse.

[4] DataCamp. Qu'est-ce que le coefficient de détermination (r^2) ? URL : https://www.datacamp.com/fr/tutorial/coefficient-of-determination.

[5] L'Argus de l'Assurance. Classement des bancassureurs 2024. URL : https://www.argusdelassurance.com/classements/classement-des-bancassureurs-2024.232292.

[6] Google Developers. Cours accéléré d'apprentissage automatique : Régression linéaire. URL : https://developers.google.com/machine-learning/crash-course/linear-regression?hl=fr.

[7] GACM : Assurances du Crédit Mutuel. Rapport-d-activite-2024-des-acm. Technical report. URL : https://www.acm.fr/fr/document/Rapport-d-activite-2024-des-ACM.pdf.

[8] Python Software Foundation. Python documentation. URL : https://docs.python.org/3/.

[9] SAS Institute. Sas/or 13.2 user's guide : Mathematical programming. URL : https://documentation.sas.com/doc/en/pgmsascdc/9.4_3.5/ormpug/titlepage.htm.

[10] Gurobi Optimization. Gurobi optimizer. URL : https://www.gurobi.com/solutions/gurobi-optimizer/

?_gl=1*npg5h3*_up*MQ..*_gs*MQ..&gclid=EAIaIQobChMIk5yxi8D7jgMVFDYGAB0SgCX2EAAYASAAEgJ8AvD_BwE.

[11] GNU Project. Glpk (gnu linear programming kit). URL : https://www.gnu.org/software/glpk/.

[12] Pyomo. Pyomo presentation. URL : https://www.pyomo.org/about.

[13] SCIP Optimization Suite. Scip - solving constraint integer programs. URL : https://scipopt.org/.

[14] Wikipedia. Random forest. URL : https://en.wikipedia.org/wiki/Random_forest.