

## Opti Canteen

---

*L'IA anti gaspi*



Rédacteur : Quentin Angelot

Titre professionnel « Développeur en intelligence artificielle » de niveau 6 enregistré au RNCP sous le n° 34757

---

---

# Sommaire

## I. Mise en place du projet

- Contexte
- Le client et son besoin
- Analyse du marché et état de l'art
- Besoins et objectifs
- Spécifications fonctionnelles et techniques
- Gestion de projet et planification
- RGPD

## II. Gestion de la donnée

- Acquisition des données
- Consolidation des données

## III. Construction d'un modèle d'IA

- Analyse exploratoire
- Preprocessing et Ingénierie des features
- Sélection des variables et du modèle
- Choix des hyperparamètres
- Analyse des erreurs et interprétation des résultats

## IV. Développement d'une application intégrant l'IA

- Packaging du modèle
- Service du package via une API
- Construction d'une application Web
  - Intégrant un tableau de bord
  - Intégrant une composante prédictive
- Tests et logging

## V. Déploiements

- De l'API
- De l'application Web
- Monitoring

## VI. Difficultés et axes d'amélioration

## Conclusion, Sources et Annexes

---

## I. Mise en place du projet

Dans cette première partie, nous aborderons le point crucial de la connaissance du client et de son besoin, de l'analyse du marché et de l'état de l'art, nous fixerons des objectifs clairs et réalisables, ainsi que les spécifications fonctionnelles et techniques de notre produit. Enfin, nous établirons une gestion de projet adaptée au cadre de la data science et du travail individuel, tout en analysant les contraintes liées à la RGPD.

### A. Contexte

Ce projet s'inscrit dans le contexte du passage du titre professionnel "Développeur en intelligence artificielle" de niveau 6 enregistré au RNCP sous le n°34757. Ce passage conclut une formation de Développeur IA qui se compose de 7 mois de formation intensive suivie de 12 mois d'alternance au siège du Crédit Agricole de Lorraine à Metz où j'ai travaillé au sein du service Pilotage stratégique à un poste de chargé d'études statistiques.

Mon quotidien s'articulait autour de plusieurs missions principales :

- Recueillir les besoins métier et trouver les sources de données pertinentes
- Extraire les données du système source et les valoriser à des fins d'analyse
- Contrôler la qualité de la donnée par des croisements avec les outils métier
- Synthétiser les informations via des tableaux de bord décisionnels
- Écrire des macros VBA afin d'automatiser différents documents Excel en appui aux équipes du contrôle de gestion et du pilotage.

Ainsi, cette alternance a été pour moi très enrichissante d'un point de vue analyse de données mais elle ne contenait pas de mission de développement ou d'IA à proprement parler. Ce projet, parmi les autres que j'ai mené sur mon temps libre, est donc fondamental pour ma montée en compétences.

Concernant le choix du projet, j'étais d'abord parti sur un projet de classification d'images depuis août dernier mais les contraintes matérielles, financières et les temps de latence dans les communications avec le client ne m'ont pas permis de présenter ce projet dans le cadre du passage du titre. Ainsi, j'ai dû changer de projet au mois de mai dernier. Étant particulièrement intéressé par l'open data et les projets d'intérêts publics, je me suis donc remobilisé sur le projet d'IA éthique au service des politiques publiques proposé par la métropole de Nantes. Un autre groupe présente également ce projet, mais ils étaient déjà lancé quand j'ai dû prendre la décision de suspendre mon projet initial. Pour cette raison, je n'ai pu les rejoindre.

---

## B. Le client et son besoin

Ce projet s'inscrit dans le cadre de l'ambition de la Ville de Nantes de favoriser l'innovation en soutenant l'expérimentation de nouveaux usages au service des citoyens et de l'efficience des services publics. L'initiative présentée dans ce rapport consiste à mieux anticiper le nombre de convives dans les restaurants scolaires afin de réduire la surproduction responsable d'une partie du gaspillage alimentaire. Ce défi technique répond à des questions importantes sur le plan environnemental, économique et technique.

La Ville de Nantes sert tous les jours entre 15 000 à 16 000 repas dans ses restaurants scolaires. Comme dans toutes les collectivités, il existe nécessairement une différence entre le nombre de repas anticipés et le nombre de repas produits et consommés. Cependant, là où d'autres collectivités pratiquent une inscription à la semaine, la métropole permet aux parents d'inscrire leurs enfants le matin pour le midi. Cette flexibilité favorise l'accès de tous les enfants à une alimentation équilibrée, mais induit une complexité de l'estimation du nombre de couverts au quotidien.

L'objectif du projet consiste à utiliser les données produites par le service de restauration pour aider à prévoir, plusieurs semaines à l'avance, la fréquentation des cantines. Ceci permettra d'anticiper les commandes aux fournisseurs et adapter le niveau de production de la cuisine centrale. Cette initiative répond à un enjeu en matière de transition écologique et d'alimentation durable pour le plus grand nombre et dès le plus jeune âge. Mais le projet répond également à des considérations économiques, liées à la maîtrise des dépenses de la collectivité quant à l'achat de matières premières.

Pendant l'année 2018-2019 à Nantes, sur environ 16 000 repas produits chaque jour, 1350 repas en moyenne étaient excédentaires. Sachant que le coût indicatif des matières premières est de 1,65€ par repas, les excédents entraînaient des surcoûts journalier et annuel s'élevant respectivement à 2 250 et 350 000€.

---

## C. Analyse du marché et état de l'art

Le gaspillage alimentaire est une préoccupation majeure des citoyens, en témoigne la variété et la diversité d'applications en la matière :

- Phénix et Too Good to Go : supermarchés, boulangeries et autres commerçants à proximité ne sont plus contraints de jeter leurs invendus. Ces applications proposent de les leur acheter à un prix réduit (environ 50%)
- Geev et Hop Hop Food : Geev propose de donner ses restes et de découvrir du même coup les talents gastronomes du voisinage. Cela ayant un double impact écologique et

---

social. La plateforme de dons d'aliments Hop Hop Food organise des collectes alimentaires et des distributions à destination des personnes précaires.

- Meal canteen : le service permet à l'utilisateur de réserver à l'avance un repas dans son restaurant d'entreprise ou scolaire, et donc aux cuisines de ne préparer que les quantités réservées. L'application permet concrètement au personnel de restauration de recevoir les réservations, des alertes d'annulation, de bénéficier d'un calcul des quantités à préparer et d'avoir un retour sur la satisfaction des convives.

Meal canteen est évidemment l'application la plus proche du projet Nantais, néanmoins, elle utilise d'autres techniques que l'IA pour répondre à notre objectif de lutte contre le gaspillage. Il est donc intéressant de travailler sur ce projet afin de voir comment ces deux visions peuvent se compléter afin de minimiser les pertes.

## D. Besoins et Objectifs

### Une IA au service des agents

L'objectif de cette IA n'est pas de remplacer les personnes, mais de porter à leur connaissance des éléments permettant d'éclairer leur jugement. Les algorithmes sont performants pour tenir compte de facteurs mesurés de manière fiable et sur un temps long, et se produisant avec monotonie (vacances, fêtes religieuses...). Mais seuls des agents présents sur le terrain et entretenant des interactions personnelles avec l'ensemble des parties prenantes sont capables de capter des informations plus diffuses (sorties scolaires, événements inattendus...) ou de tenir compte de cas particuliers (allergies, régimes spéciaux...).

Dans des contextes marqués par des crises sanitaires ou des mouvements sociaux, aucun algorithme ne parviendra à prédire la fréquentation des restaurants scolaires avec plus de fiabilité qu'une équipe expérimentée. Mais lorsque les personnes changent ou que leur organisation est perturbée, l'algorithme sert de guide pour limiter le risque d'erreurs et de dysfonctionnements.

En conséquence, le but de cette application est d'améliorer la fiabilité des prévisions réalisées par les agents de terrain et de contribuer au bon fonctionnement du système de restauration collective malgré les aléas. L'outil développé est donc une IA qui vient en soutien des agents sur le terrain et qui peut leur apporter un supplément de confiance quant aux décisions qu'ils sont amenés à prendre au quotidien.

---

## Améliorer les décisions en périodes normales

Deux à trois semaines avant, chaque responsable de restaurant scolaire demande un nombre précis de repas pour le jour en question. La cuisine centrale agrège ces demandes et ajuste les commandes auprès de ses fournisseurs.

Le jour en question, les livreurs de la cuisine centrale déposent les repas dans les écoles. En arrivant à l'école à 8h30, les enfants s'inscrivent, ou non, pour le déjeuner. L'équipe d'encadrement de chaque école fait la somme des convives effectivement inscrits pour le repas du midi et contacte la cuisine centrale si cette somme dépasse le nombre de repas demandés trois semaines auparavant. Chaque livreur dispose de repas supplémentaires dans son camion, qu'il apporte aux écoles où des repas manquent. Les sites qui ont reçu trop de repas appellent la cuisine centrale pour qu'ils soient récupérés. Les chauffeurs viennent alors récupérer les repas excédentaires pour les apporter là où il en manque.

Le besoin porte donc avant tout sur un modèle prédictif pouvant servir d'aide à la décision afin d'ajuster les commandes et la production planifiées deux à trois semaines à l'avance.

Ainsi il convient de concentrer le modèle sur la prédiction des périodes normales car la connaissance d'événements comme les grèves, la météo ou les sorties scolaires non anticipées n'est pas disponible plusieurs semaines à l'avance. De plus, développer un modèle précis par restaurant scolaire est intéressant mais la priorité est clairement de sortir un premier modèle au global car les livreurs parviennent, dans une certaine mesure, à corriger les écarts le jour même.

## **E. Spécifications fonctionnelles et techniques**

### Spécifications fonctionnelles

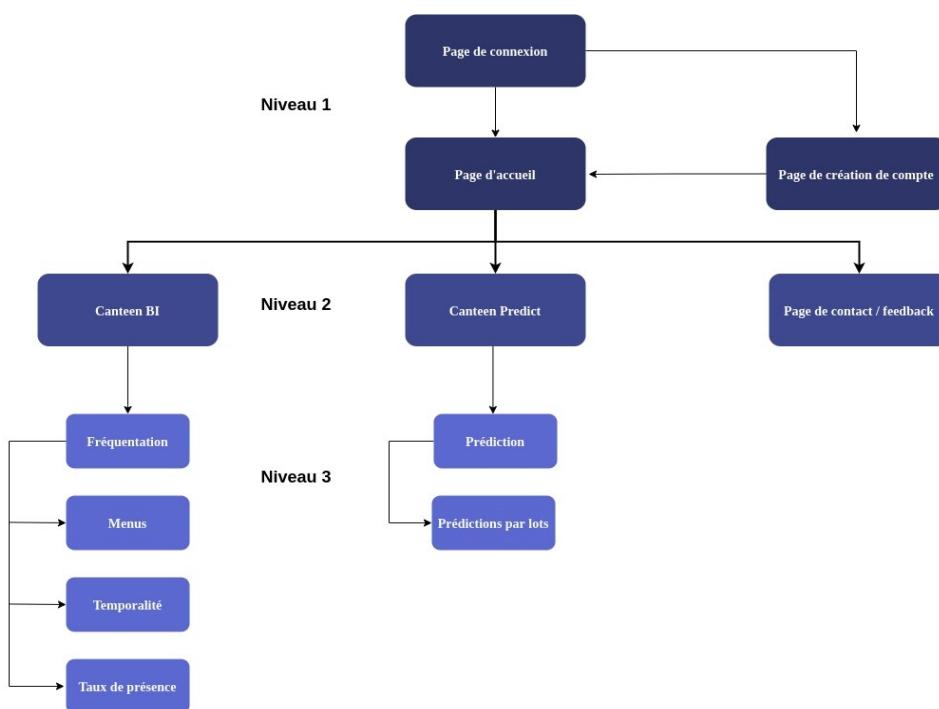
L'application sera mise à disposition des agents techniques de la métropole et sera décomposée en plusieurs vues dans une première version :

- un écran d'accueil permettant de se connecter à l'application ou de s'y inscrire si l'on ne possède pas encore de compte
- une page d'accueil décrivant l'application et les différents outils à disposition
- un premier outil de type tableau de bord afin de visualiser certains points d'intérêt permettant une meilleure compréhension de la fréquentation des cantines

- un second outil de type prédictif permettant d'entrer les données à disposition à l'instant t afin de retourner une prédiction de la fréquentation d'une cantine précise à un horizon de 2 ou 3 semaines

## Arborescence

L'arborescence d'un site est une représentation schématique des différentes pages qui composeront le site, liées entre elles et hiérarchisées par niveaux de profondeur. Cela permet de se représenter la structure du site lors de son développement. Ci-dessous, on peut observer l'arborescence de notre application.



Ici, nous constatons la présence de trois hiérarchies bien définies. Le niveau regroupe les éléments essentiels à l'identification et l'accueil de l'utilisateur. Le niveau 2 regroupe les deux composantes principales de l'application ainsi que la partie feedback. Et enfin le niveau 3 contient l'ensemble des services proposés à l'utilisateur.

## Wireframing (cf Annexes)

Après avoir délimité le périmètre fonctionnel et créé l'arborescence de l'application, on possède tous les éléments nécessaires pour en faire des ébauches. Ces croquis, volontairement simplistes, serviront de document de référence par la suite, en phase de conception. Un wireframing est une technique utilisée pour créer des versions basse fidélité d'un livrable. Des éléments de wireframing concernant les pages principales de l'application sont disponibles en Annexes.

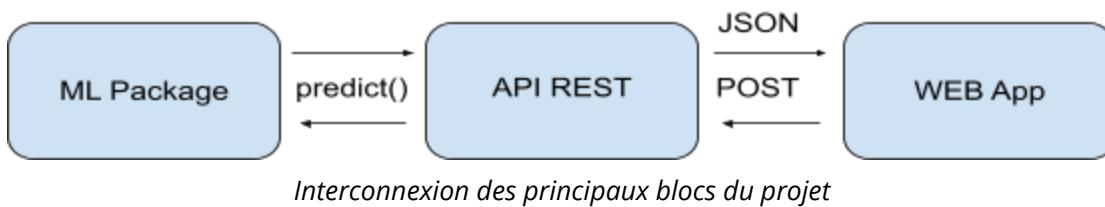
---

## Spécifications techniques

### Architecture du projet

L'architecture retenue fait le choix de la séparation entre le modèle de machine learning packagé, l'API qui "sert" celui-ci et l'application en tant que telle qui le "consomme". Cela permet d'envelopper le modèle dans un service qui peut-être déployé de façon indépendante par rapport à l'application. Le modèle est pré-entraîné et l'application par l'intermédiaire d'une API appelle la fonction "predict" du modèle de machine learning.

Cette architecture est donc flexible et permet également de scaler le serveur ou le modèle de façon séparée selon les besoins. A noter que lors d'un échange avec le client, il a été fait la remarque que la métropole de Nantes ne compte pas dans ses rangs de développeurs Python/IA mais de très talentueux développeurs Javascript et que la mise à disposition d'une API leur permettrait d'exploiter l'IA créée dans d'autres applications dans le futur.



### Stack technique

Le cœur du projet repose sur une API REST et tous les autres éléments (package, web app...) s'articule autour de celle-ci. En dehors du front-end et de la base de données, tout le projet repose sur Python qui est un langage de choix en science des données et qui est le langage maîtrisé par le développeur du projet.

Une autre raison du choix de Python est l'écosystème open source étendu de ce langage.

Ainsi la partie de data science du projet se reposera sur les librairies populaires en la matière telles que :

- Numpy pour le calcul matricielle optimisé
- Pandas pour le nettoyage et le feature engineering
- Scikit-learn, XGBoost et lightGBM pour la modélisation
- Scipy et Statsmodels pour l'étude statistique
- Matplotlib et Seaborn pour l'exploration des données...

La partie développement logiciel tirera aussi aisément parti de l'écosystème Python. En effet, il existe de nombreux frameworks qui peuvent être utilisés pour servir des modèles de ML via une API (Flask, FastAPI...). Le développement Web n'est pas en reste avec Django,

---

généralement utilisé pour les applications à grande échelle, et Flask, la référence pour construire rapidement une interface web en Python.

Pour la partie API, FastAPI est retenu pour plusieurs raisons :

- FastAPI est plus rapide que Flask car il fonctionne en asynchrone en se basant sur la librairie Python asyncio.
- Contrairement à Flask, FastAPI fournit une implémentation plus simple pour la validation des données avec pydantic et les type hints.
- FastAPI génère de façon automatique la documentation de l'API, ce qui est un gain de temps considérable.
- FastAPI permet enfin de gérer efficacement les injections de dépendance dans le code (quand une classe ou une fonction dépend d'une autre pour fonctionner correctement).
- Une montée en compétence du développeur sur une technologie prometteuse.

Concernant l'application Web, Flask est retenu. En effet, Flask permet de générer très rapidement une application fonctionnelle et, contrairement aux idées reçues, ce framework peut permettre de construire des applications web très bien structurées jusqu'à une certaine échelle. Flask met en place les Blueprints à cette fin. Un Blueprint est un package qui encapsule une fonctionnalité spécifique de l'application. Flask permet le recours à une infinité de plugins afin de sécuriser l'application, de faciliter la gestion des formulaires et du statut de l'utilisateur, de simplifier le mailing, l'alimentation et le requêtage de la base de données...

Enfin pour les bases de données, on utilisera la technologie SQL car cela reste la référence pour la gestion des données structurées. Elle se prête très bien à des données de type tabulaire comme nous avons sur ce projet. De plus, pour la partie analytique, la construction d'un datawarehouse est relativement aisée avec des librairies comme Pandas. Pour la gestion des données utilisateurs, nous utiliserons le système de gestion de bases de données relationnelles référence open source MySQL.

Pour finir, la partie front devra être cross platform, s'adapter à tous les supports et tous les navigateurs. Le front-end sera donc développé à l'aide d'un framework responsive HTML/CSS nommé Bootstrap. Il permet d'obtenir un résultat de qualité sans pour autant avoir à maîtriser Javascript. Les graphiques, quant à eux, seront développés grâce à la librairie Plotly. Construit sur la bibliothèque Plotly JavaScript ( plotly.js ), Plotly permet aux utilisateurs de Python de créer des visualisations Web interactives de qualité.

Au sujet du déploiement, il sera effectué sur Heroku pour l'API et sur Azure pour la Web app. Ce choix découpé est motivé par des raisons financières et de montée en compétences sur des technologies alternatives. De plus, Azure permet un monitoring de grande qualité grâce à

la ressource Application Insights et le développeur possède déjà certaines compétences sur la technologie cloud de Microsoft.

## F. Gestion de projet et planification

La gestion de projet est la pratique qui consiste à identifier le problème, créer un plan pour le résoudre, puis exécuter ce plan jusqu'à ce que le problème soit résolu.

Les étapes de réalisation du projet : diagramme de Gantt (cf Annexes)

La chronologie visuelle d'un diagramme de Gantt donne un aperçu détaillé des tâches et des dépendances d'un projet. Il permet de définir une timeline rigoureuse pour le projet. Sur le diagramme complet du projet en Annexes, on peut observer que l'amélioration du modèle qui a eu lieu après la construction du package a nécessité de revenir implémenter les modifications dans celui-ci. On y observe donc la dépendance réelle entre la phase de recherche et la construction du package. De plus, le référentiel du titre et le périmètre de ce projet étant relativement vaste, le diagramme de Gantt permet de bien s'assurer que l'on n'oublie pas un élément important de ceux-ci.

### DIAGRAMME DE GANTT

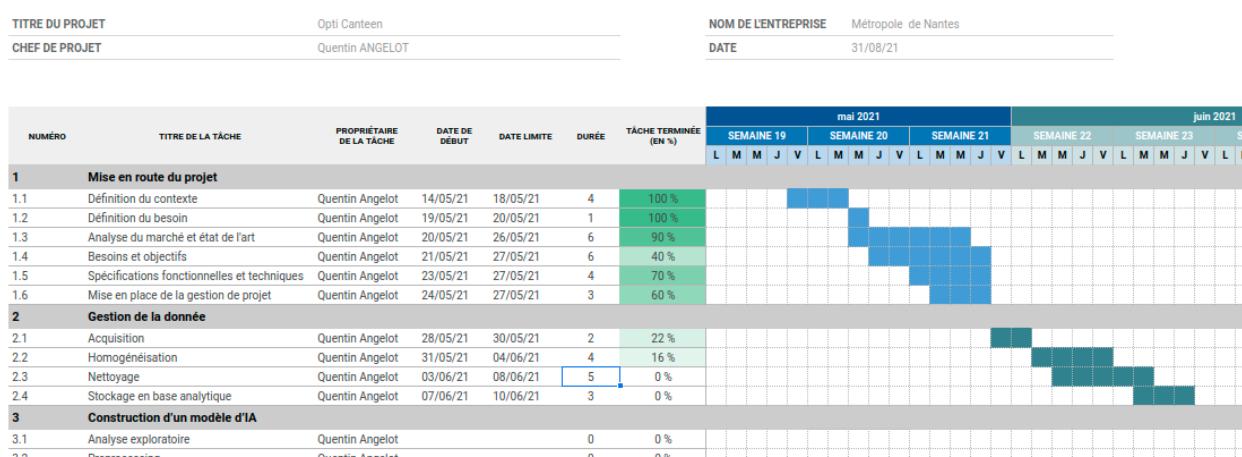


Diagramme de GANTT en début de projet

---

## La méthode et l'outil de suivi : Kanban et Trello

### La gestion de projet Agile en sciences des données

L'essence de la méthode Agile appliquée au machine learning n'est pas foncièrement éloignée de celle appliquée au développement logiciel traditionnel :

- Déployer les produits le plus tôt possible, afin de ne pas se perdre dans une exploration sans fin.
- Veiller à mettre à jour le produit en continu, au lieu de tout enregistrer jusqu'à ce qu'une date limite soit proche.
- Respecter des normes élevées pour tout le code utilisé par le produit, même s'il n'est pas expédié avec le produit.
- Impliquer étroitement les parties prenantes, les collègues et les clients à chaque étape du projet.

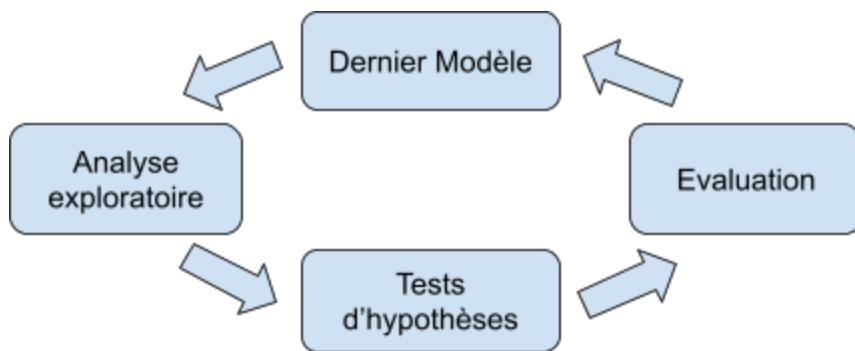
### Kanban, présentation générique

Kanban est beaucoup plus léger que Scrum. Il ne fonctionne pas avec des unités de temps fixes, mais vise à obtenir un flux continu. Tout comme dans Scrum, les tâches à accomplir sont formulées dans un backlog, mais le travail en cours se cantonne à un nombre très limité de tâches à la fois. Chaque fois qu'une tâche est terminée, la plus pertinente ou la plus urgente commence ensuite. Le tableau Kanban est l'élément central de la méthode et se compose au moins des 3 colonnes suivantes : to do, doing et done. La métrique centrale est le temps qu'il faut généralement pour terminer une tâche.

### Tâches linéaires et circulaires

Les tâches du développement logiciel sont parfois considérées comme des tâches linéaires, notamment en Waterfall. Elles proviennent de demandes de fonctionnalités par les parties prenantes ou d'idées au sein de l'équipe elle-même. Le résultat envisagé est capturé dans une user story. L'équipe le traduit en sous-tâches techniques et commence à y travailler.

En revanche, les tâches de science des données sont souvent circulaires, car au début, on ne sait pas quel sera le résultat obtenu. La position de départ est la dernière version du modèle et si l'on n'a pas encore une idée quant à la façon de l'améliorer davantage, nous ferons généralement une analyse exploratoire dans laquelle une hypothèse peut être formée. Cette hypothèse est ensuite testée. Puis, nous évaluons si nous pouvons améliorer le produit sur la base de l'hypothèse testée. Si c'est le cas, nous l'améliorons et passons à l'hypothèse suivante. Sinon, on passe tout de suite à l'hypothèse suivante.



*Tâches circulaires en sciences des données*

Les projets de science des données englobent également des tâches linéaires, telles que la configuration d'un pipeline pour importer les données, exposer les résultats du modèle dans une API ou créer une application web. Cependant, la nature circulaire de l'exploitation des relations entre les variables rend, à mes yeux, la méthode Kanban particulièrement adaptée à ce projet.

### Utiliser Kanban pour la science des données

Scrum comme Kanban sont basés sur la notion de user stories permettant d'établir le bénéfice du produit pour le consommateur final. Cependant en machine learning, ces users stories auront souvent une structure de type : "En tant que ... , je veux obtenir le meilleur modèle possible, afin que ... , soit atteint de la meilleure des manières."

Pour un projet comme celui-ci, je préfère me concentrer sur la notion de tâches qui vont découper le flux de travail.

En Kanban, nous allons formuler le travail en se basant sur deux types de tâches : les hypothèses de recherche d'un côté et le produit logiciel de l'autre. Il y a donc deux "sous-backlogs" différents dans la colonne to do. Grâce à Kanban, si une hypothèse intéressante nous vient, on ne va pas être tenté de la tester directement, en laissant de côté le travail en cours, on va simplement l'incrémenter au backlog et y revenir plus tard. De cette façon, les tâches en cours doivent être terminées dès que possible et après cela, on décide de ce qui est le plus urgent à traiter : une hypothèse ou une implémentation logicielle.

La pratique de Kanban implique souvent une tâche en cours par membre de l'équipe. Cependant afin de garder un peu de polyvalence et de flexibilité et car je suis seul sur le projet, je m'autorise jusqu'à trois tâches en cours. Cela me permet d'éviter l'un des principaux pièges de l'amélioration d'un produit data : l'exploration sans fin d'hypothèses.

---

### Mon implémentation de Kanban avec Trello (cf Annexes)

Pour appliquer Kanban à mon projet je vais utiliser l'outil open-source Trello. Mon tableau Trello comprend les étapes principales d'un projet data basé sur la méthode CRISP-DM (CRoss-Industry Standard Process for Data Mining).

Le processus de CRISP-DM se décompose en 6 blocs : compréhension des besoins et objectifs métier, compréhension des données, préparation des données, modélisation statistique, évaluation de la performance du modèle, déploiement et monitoring.

Chacun des post-its sur mon Trello se rattache à un de ses blocs. Enfin, mon tableau contient deux couleurs différentes pour les étiquettes : le rouge représente les tâches se rattachant à des activités de data science et le vert représente celles qui relèvent davantage du développement logiciel.

### Échanges avec le client : Comptes rendus d'avancement

Dans une gestion de projet Agile, le client est une source d'information incontournable afin de maximiser la valeur du produit. Ainsi pour ce projet, Noura Benhajji remplira le rôle de Product Owner (PO), interface entre le client final et le développeur. En effet, les personnels de la Métropole de Nantes ne sont pas directement disponibles pour faire des retours réguliers sur l'évolution du produit, c'est donc avec Mme Benhajji que la majeure partie de ces échanges auront lieu.

### Premier retour du PO : la validation des maquettes

Après le recueil des besoins, la planification et la formalisation du projet, les spécifications fonctionnelles ont été rédigées. L'arborescence de l'application et son wireframing ont alors été soumis au client afin qu'il donne son avis et apporte quelques amendements à la vision du développeur.

### Deuxième retour du PO : ajout et modification de certains graphiques

Après un premier développement basé sur le wireframing validé précédemment, un premier jet de l'application a été présenté lors d'une démonstration au client. Celle-ci ne contient alors que la page d'accueil et la partie tableau de bord. Le client a souhaité l'ajout de graphiques pertinents concernant les prévisions de fréquentation et le taux de présence dans les différentes cantines.

---

#### Troisième retour du PO : ajout de la fonctionnalité prédictions par lots

Après une seconde phase de développement concernant la partie prédictive de l'application web, le client a évoqué son souhait d'avoir une page permettant la prédition par lots. Les agents importent un fichier excel ou csv correspondant aux différents inputs du modèle, chaque ligne correspondant à une journée à prédire. Cela évitera aux agents de réaliser manuellement de nombreuses prédictions en remplissant le formulaire de mono prédition.

#### Quatrième retour du PO : limiter l'inscription à l'application

Lors de la phase de sécurisation de l'application, le client a fait part de son besoin de limiter l'accès à l'outil aux agents de la métropole. Ainsi, l'accès se limitera à des personnes ayant une adresse mail avec le sous-domaine de la métropole Nantaise (@nantesmetropole.fr).

#### Cinquième retour du PO : test dans le cadre d'un POC

Le client est globalement satisfait du contenu du MVP (minimum viable product). Celui-ci couvre bien les fonctionnalités nécessaires au pilotage quotidien de l'activité des cantines nantaises. L'amélioration du design et du caractère responsive du site sont évoqués. L'ajout de media queries en CSS sera réalisé pour répondre à cela. Des pistes d'améliorations pour des versions futures sont également évoquées.

## **G. RGPD et licence**

#### Obligations de base sous le RGPD

Dans le cadre du règlement européen sur la protection des données personnelles (RGPD), qui est entré en application le 25 mai 2018, les obligations des organisations sont multiples. Celles-ci doivent pouvoir informer les personnes sur qui collectent leurs données personnelles, l'utilisation qui en sera faite et la durée de conservation. Elles doivent également les alerter en cas de fuite ou de détournement de données.

Par ailleurs, les organisations doivent obtenir le consentement de chaque individu dont elles collectent les informations, de manière licite et loyale. Le consentement doit être une action active de la part de la personne concernée. Les DPO (Data Protection Officer) doivent conserver une trace de comment et quand une personne a donné son consentement, laquelle peut s'en soustraire à tout moment.

---

Enfin, les organismes ont l'obligation de garantir la protection des données sensibles, telles que l'orientation sexuelle, l'ethnie, l'état de santé... et d'alerter les autorités de contrôle en cas de défaillance de leur système.

### Données du projet

Après le rappel ci-avant, il est clair que notre projet n'est pas concerné par le RGPD car il ne se base pas sur l'exploitation de données ayant une granularité au niveau de l'individu mais sur des agrégats de fréquentation journalières des différentes cantines. Les données annexes telles que la composition des menus ou les dates d'événements d'intérêt ne sont pas plus des données sensibles de ce point de vue.

### Licence

Conformément à la Loi pour une République numérique, la métropole de Nantes mettra le code source développé par le prestataire sous un régime de licence de logiciel libre, à savoir la Licence MIT. Le développeur s'engage donc à réaliser le logiciel, dans des conditions qui permettent au pouvoir adjudicateur de mettre ce logiciel à disposition de tiers, dans les conditions posées par la licence annexée au présent marché.

## **II. Gestion de la donnée**

### **A. Acquisition des données**

L'élément le plus fondamental en data science ne tient pas en la complexité algorithmique d'un modèle ni aux nombres de couches d'un réseau de neurones, mais à la qualité de la donnée à disposition. En effet, la complétude, la fiabilité et la volatilité des données disponibles pour la variable à prédire déterminent généralement la performance d'un modèle prédictif. Ces dernières années, les pontes de la discipline plaident tous pour que l'IA passe d'une approche centrée sur le modèle à une approche centrée sur la donnée. Dans ce contexte, la métropole de Nantes partage les données complètes et détaillées de fréquentation quotidienne et par établissement sur les neuf dernières années (2010-2019), ainsi que plusieurs jeux de données complémentaires (menus, vacances scolaires, grèves, effectifs des écoles...). On peut retrouver ces données sur le portail open data de la métropole : <https://data.nantesmetropole.fr/pages/home/>

Les jeux de données suivants seront fournis par Nantes Métropole aux personnes acceptant de collaborer au projet :

- 
- Pour chaque cantine et chaque jour ouvré depuis 2011, on dispose du nombre de repas commandés et livrés par la cuisine centrale, ainsi que du nombre de repas effectivement consommés dans les écoles. Le fichier ne contient pas de désagrégation par type de convive.
  - pour chaque journée depuis 2011, les menus des cantines scolaires. Ce fichier contient pour chaque date l'intitulé des plats proposés.
  - le calendrier des vacances scolaires pour l'académie de Nantes construit à partir des archives du calendrier scolaire de l'éducation nationale.
  - le nombre d'enfants inscrits dans chaque école de la ville de Nantes depuis 2011.
  - les jours de grèves : des mouvements de grève peuvent affecter la livraison ou l'organisation des repas dans les cantines.

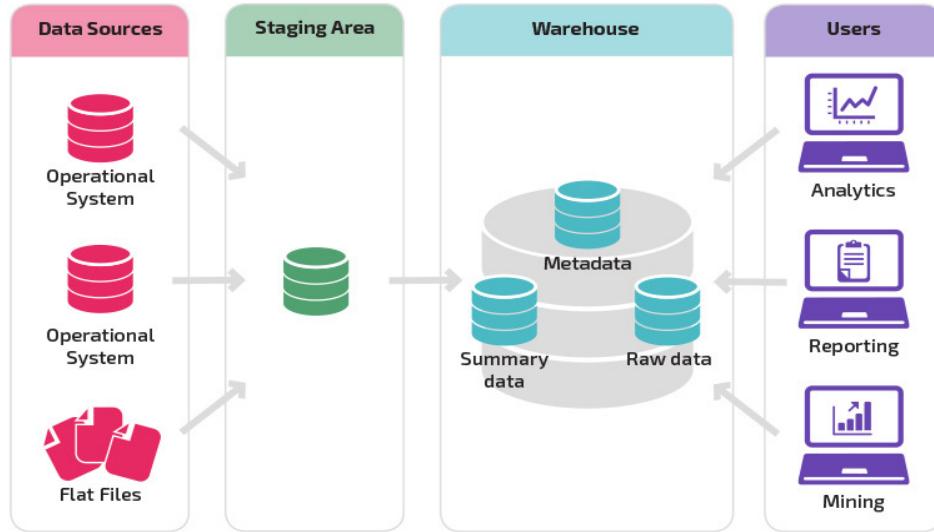
Par ailleurs, j'ai cherché de mon côté à collecter quelques données additionnelles pouvant être pertinentes pour notre problématique :

- le calendrier des jours fériés, disponible sur le portail de données ouvertes de l'Etat français.
- les dates des fêtes religieuses basées sur la liste des principales fêtes recensées par l'administration publique pour les principales religions monothéistes (Christianisme, Islam et Judaïsme).
- les prix moyens au mètre carré de l'immobilier par quartier et quartier détaillé pour les maisons et appartements dans la ville de Nantes. Celui-ci se base sur le jeu de données « Demandes de valeurs foncières » publié par la DGFIP.

## B. Consolidation des données

La consolidation des données élimine les disparités avant l'utilisation des données, ce qui permet de gagner du temps, d'améliorer l'efficacité et d'ajouter de la valeur aux opérations analytiques. Comme nous l'avons vu, la majorité des données utilisées pour alimenter l'outil prédictif sont disponibles sur le site de la métropole de Nantes sous format CSV. Cependant, chaque fois que des données sont collectées à partir de différentes sources, elles sont collectées au format brut, et pour l'analyse, il nous faut effectuer un travail rigoureux d'homogénéisation de ces données.

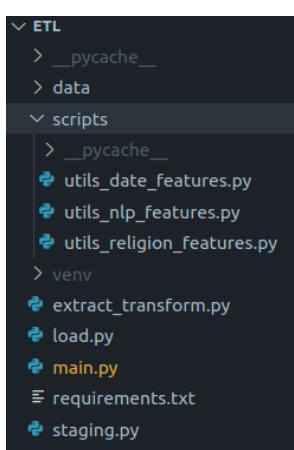
En général, la consolidation des données comprend quatre étapes qui passent par les sources de données, un pipeline ETL, la constitution d'un entrepôt de données et l'analyse ultérieure à l'aide d'outils de business intelligence ou de data mining.



Les 4 étapes de la data consolidation

Nous pouvons donc suivre un schéma ETL classique (cf Annexes et [Github](#)) :

1. **Extract** : conversion des CSV en plusieurs dataframe grâce à la librairie Pandas
2. **Transform** :
  - a. Regroupement et jointure des différents CSV
  - b. Réalisation des opérations de transformations simples des données en fonction de la logique métier (traitement des valeurs aberrantes, feature engineering non sujet à data leakage)
  - c. Persister le dataframe enrichi dans une base de staging (cf Annexes)
3. **Load** : charger les données ainsi traitées dans un entrepôt contenant une table de faits et plusieurs dimensions. A partir de ce datawarehouse, nous pouvons procéder à l'analyse descriptive et prédictive de nos données. Il est possible de créer des datamarts afin de mieux répondre aux besoins (IA, visualisation...).



La partie scripts contient des scripts Python permettant d'extraire certaines features basées sur la date, de nettoyer les données textuelles et de convertir les dates des événements religieux en features booléennes pour leur intégration dans le dataset principal des fréquentations journalières. Ces opérations ne sont pas sujettes à data leakage.

Le module `extract_transform.py` appelle ces fonctions pour joindre et opérer un premier nettoyage sur les données.

Quant à `staging.py`, il permet le stockage du jeu de données ainsi obtenu dans une base de données de staging, ce qui permet de persister les données de manière plus fiable que dans de multiples CSV.

---

Enfin, load.py construit l'entrepôt de données à partir de la base de données de staging afin de répondre aux besoins analytiques.

Le modèle physique de données de ce datawarehouse, disponible en Annexes, se présente comme suit :

- une table de faits détaillée : elle contient la mesure des processus métier ainsi que des clés étrangères pour les tables de dimension. C'est le centre du modèle de données « en étoile », structure multidimensionnelle stockant des données atomiques. Elle contient ici les mesures liées à la fréquentation quotidienne.
- plusieurs dimensions : ce sont les tables situées aux extrémités de l'étoile, elles représentent les axes d'analyse nécessaire au pilotage de l'activité. Ici, nous avons opté pour les dimensions suivantes : site, menu, temporalité et événements.

Le principe d'optimisation de ce modèle en étoile est le suivant : une clé calculée sert de jointure relationnelle entre les tables de dimensions et la table des faits. La requête SQL réalise d'abord sa sélection sur les tables de dimensions (peu volumineuses, ici un peu plus de 1100 lignes au maximum). Et ensuite seulement, à partir des clés sélectionnées, elle procède à la jointure avec la volumineuse table des faits détaillée (ici environ 100000 lignes).

## III. Construction d'un modèle d'IA

### A. Analyse exploratoire

Maintenant que nous avons posé la problématique, récupérer des données fiables pour y répondre et réaliser un ETL sur ces données afin de les stocker de manière pérenne et performante sous un format propice à l'analyse, nous pouvons justement procéder à l'exploration des données afin de décrire le jeu de données et de construire des hypothèses sur celui-ci. Cela pour permettre notamment une modélisation plus performante. Afin de limiter la longueur de ce rapport, tous les graphiques que je commente ici sont disponibles sur [Github](#). Nous allons analyser les éléments suivants : la variable cible, les types de variables, les données manquantes, les variables numériques et les variables catégorielles

#### La cible

La cible, objectif à prévoir défini dans le cahier des charges est la fréquentation réelle des cantines. Attention, il ne s'agit pas d'un jeu de données traditionnel car la granularité est double : jours et cantines. Nous allons donc agréger les données par jour puis par cantines.

De toute évidence, la série temporelle n'est pas continue : les vacances, les jours fériés, certains outliers... viennent y ajouter un caractère erratique. Afin d'en tirer de vraies conclusions, il est nécessaire de la décomposer.

Une série temporelle est communément décomposée en :

- une tendance correspondant à une évolution à long terme de la série (tendance linéaire, quadratique, logarithmique...)
- une saisonnalité correspondant à un phénomène périodique de période identifiée
- une erreur qui est la partie aléatoire de la série (idéalement stationnaire)

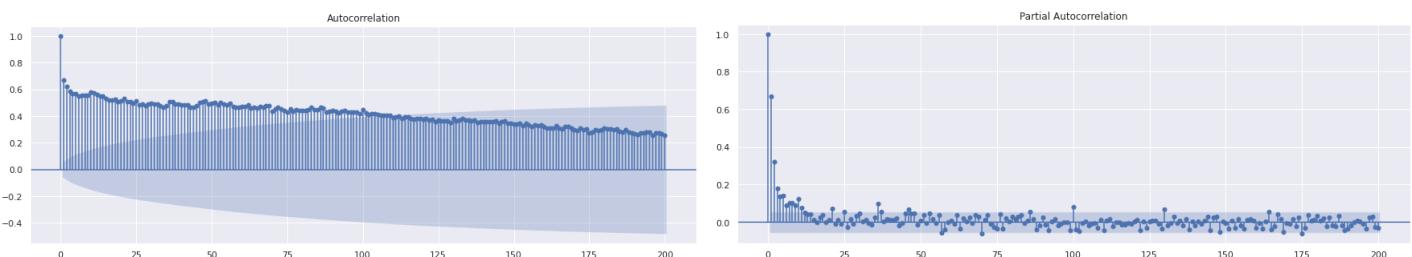
Cette décomposition révèle plusieurs enseignements :

- il y a bel et bien une tendance haussière de fond, notamment entre 2014 et 2016
- d'une année sur l'autre, on retrouve les mêmes fortes variations saisonnières qui correspondent à des vacances ou des jours fériés
- enfin on retrouve un forte concentration des résidus autour de 0, ce qui est rassurant, cependant la présence de quelques résidus importants est bien réelle.

Un point central de l'analyse des séries temporelles est de déterminer le décalage à partir duquel la corrélation est significative et d'utiliser les graphiques ACF (fonction d'autocorrélation) et PACF (fonction d'autocorrélation partielle) :

- L'ACF explique comment la valeur actuelle d'une série chronologique donnée est corrélée avec les valeurs passées (par rapport à  $t-1$ , par rapport à  $t-2$ ...).
- La PACF explique la corrélation des résidus (qui restent après avoir éliminé les effets qui sont déjà expliqués par l'ACF).

Dans notre série, nous avons une bonne corrélation positive jusqu'au 100ème lag, c'est le point où le graphique ACF (à gauche) coupe le seuil de confiance supérieur.



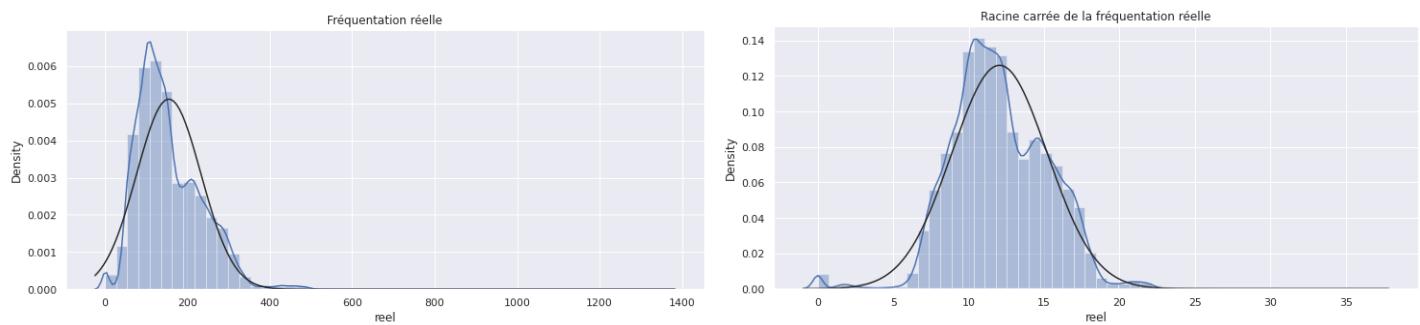
Le graphique PACF (à droite) permet de filtrer et d' observer seulement les lags les plus pertinents...

En effet, les 100 premiers lags supérieurs à l'intervalle de confiance sont donc statistiquement significatifs, mais malgré cela, l'autocorrélation partielle après les 12 premiers lags est très modeste.

---

## Analyse de distribution de la cible

L'histogramme de la fréquentation réelle montre que la cible est continue, et que la distribution est étalée vers la droite. Nous pouvons améliorer celle-ci par une transformation statistique. Ici, la cible comporte des valeurs nulles, nous ne pouvons donc pas utiliser le logarithme. Cependant, on peut utiliser la racine carrée ou le  $\log(1+x)$ . Ici, après expérimentation, nous retiendrons la racine carrée.



## Analyse des variables

Corrélation : après étude de la cible, identifions maintenant les types des variables à disposition. Concernant la corrélation entre les variables et la cible, on observe en toute logique que la variable "prévision" (des agents) est très positivement corrélée à la cible. De façon plus marginale, on observe également une corrélation de la cible avec la latitude et la longitude. Parmi ces variables numériques, il y a en fait des variables de type booléennes (ramadan, grève...), d'autres de types géographiques (longitude, latitude) et enfin des variables de type temporelles (vacances\_dans, depuis\_vacances...). L'utilisation de boxplot ou de scatterplot, nous apprend que ces variables discrètes et continues ne semblent pas avoir, à elles seules, un impact significatif sur la fréquentation, à l'exception notable de la variable "prévision".

Valeurs manquantes : en examinant de plus près les valeurs manquantes, qui peuvent être informatives en soi et requièrent un traitement spécifique avant la modélisation, on remarque que les features géographiques (relatives à la position de la cantines) comptent toutes environ 13% de valeurs manquantes.

Distributions et transformations : la plupart des variables ne sont pas normalement distribuées. Et il y en a quelques-unes en particulier qui sont extrêmement asymétriques comme "prévision". Il est tout à fait logique que les variables temporelles soient asymétriques.

---

Parfois, la transformation des variables pour améliorer la répartition des valeurs accroît les performances du modèle, en particulier dans le cas d'un modèle linéaire. Mais il est peu probable qu'une transformation permette de modifier de façon spectaculaire la distribution des variables asymétriques.

Pour les variables strictement positives, on peut appliquer une transformation logarithmique. Pour les autres, il existe d'autres transformations statistiques applicables. Cependant ici, beaucoup de variables ont des valeurs nulles et le logarithme n'est pas applicable. On peut utiliser la transformation de Yeo-johnson ou  $\log(1+x)$ .

Nous choisissons donc d'appliquer cette transformation de Yeo-johnson. Et nous remarquons qu'elle n'a pas fonctionné pour la variable "prévision". Cependant, elle semble satisfaisante pour les autres variables.

Il reste à voir si cela permet d'améliorer le pouvoir prédictif. Pour déterminer si c'est le cas, nous devrons entraîner un modèle avec les valeurs originales et un autre avec les valeurs transformées, puis déterminer la performance du modèle et l'importance des features. On peut également comparer la relation entre la cible et les variables d'origines ainsi que les variables transformées.

**Cardinalité des variables qualitatives** : La variable "cantine\_nom" a une très haute cardinalité, avec plus de 80 cantines différentes. En conséquence, il faudra garder cela en tête lors de l'encoding. Le type de site ne semble pas avoir d'impact sur la fréquentation, à l'inverse de l'année scolaire et plus généralement de la temporalité et du quartier dans lequel se situe la cantine.

## B. Preprocessing et Ingénierie des features

Le prétraitement des données est une étape essentielle de l'apprentissage automatique, car la qualité des données et les informations utiles qui peuvent en être tirées influent directement sur la capacité d'apprentissage de notre modèle. Il est donc extrêmement important de prétraiter nos données avant de les intégrer à notre modèle.

### Transformation de la cible

Une bonne partie de ce prétraitement découle directement de l'analyse des données, aussi par exemple, en application des conclusions de celle-ci, nous procédons à la transformation de la cible avec la racine carrée.

---

## Traitement des outliers

Le cas des outliers lui est plus complexe. Après une inspection approfondie de l'ensemble de données, la plupart des valeurs aberrantes semblent apparaître proches d'événements de grève ou simplement dues à une erreur humaine. La plupart de ces événements se sont produits avec un court préavis, c'est-à-dire que nous n'aurons pas cette information au moment de la prédiction. En un mot, ils ne sont pas significatifs de la distribution des données, ils n'ont pas de sens particulier. Pour rappel, la demande du client consiste à produire un MVP qui prédit les périodes normales avec la meilleure précision possible et cela deux à trois semaines à l'avance.

Ici, j'utilise la méthode du Z-score et considère comme aberrant tout point de donnée qui s'éloigne de plus de trois écart-types de la moyenne. Après ce premier écrémage, j'élimine tous les points qui correspondent à des grèves et/ou la valeur de la cible est égale à zéro. En effet, la plupart des zéros sont dus à des erreurs et ne sont pas des valeurs significatives. Le traitement des outliers est toujours un point délicat et l'important est avant tout de bien justifier ces choix.

## Création de features dérivées de la date

Comme nous l'avons constaté lors de l'exploration, nos données ont par nature une composante temporelle très importante. Aussi, il est important d'insister sur ce point et d'orienter le modèle dans cette direction. En effet, la seule date n'a que peu de valeur pour un modèle de machine learning. Ainsi, nous pouvons dériver des features à partir de cette date : l'année, le mois, le jour, le jour de la semaine, le jour de l'année, le trimestre, la semaine, la semaine du mois...

En général, les algorithmes peuvent être trompés par ce type de caractéristiques de date et d'heure. En fait, disons que nous codons lundi avec 0 et dimanche avec 6. Cette différence d'échelle ne signifie pas que le dimanche soit plus important que le lundi, cependant c'est ce que le modèle peut amener à comprendre. Une méthode ingénue pour coder les données cycliques consiste à transformer les données en deux dimensions à l'aide d'une transformation en sinus et en cosinus. Il est important de coder les caractéristiques correctement pour l'algorithme d'apprentissage automatique spécifique utilisé. D'autres algorithmes d'apprentissage automatique peuvent être robustes vis-à-vis des caractéristiques cycliques brutes, notamment les approches basées sur les arbres. Cependant, les réseaux neuronaux profonds ont tout à gagner de la stratégie d'encodage décrite ci-dessus, notamment en terme de vitesse de convergence du réseau.

Ici, puisque nous voulons essayer plusieurs types d'algorithmes d'apprentissage, nous utiliserons cet encodage.

---

## Création de features liées aux menus

L'idée est d'utiliser différentes techniques de NLP afin d'obtenir des features intéressantes à partir du texte contenu dans les menus :

- l'exploration des données, qui permet de dériver des features manuellement sans se reposer sur un quelconque algorithme.
- le Bag of Words, qui permet d'obtenir les mots et les séquences les plus associés à chaque niveau de fréquentation. Ensuite, nous pouvons dériver des caractéristiques des mots les plus discriminants.
- l'algorithme Latent Dirichlet Allocation, qui permet d'extraire des thèmes à partir des différents menus. On fixe un nombre K de thèmes et on cherche à apprendre les thèmes représentés dans chaque document et les mots associés à ces thèmes. Ensuite, nous pouvons dériver des caractéristiques sur les thèmes les plus discriminants.
- le word embedding, qui est une représentation vectorielle des mots où ceux qui ont la même signification ont une représentation similaire.

## Création de variables statistiques

Ces variables sont très sujettes à ce que l'on nomme le data leakage : de l'information du test set se propage dans le train set et conduit le modèle à des prédictions trop optimistes. Afin d'éviter cela, il faut calculer la statistique sur le train set et l'imputer au train et au test set par la suite. Une technique fréquemment rencontrée consiste à agréger les données de manière significative (ici, au niveau de la cantine, de l'année et de la semaine) et calculer des statistiques sur ces sous-groupes. Puis, on fait une moyenne sur les années, pour avoir un seul chiffre par cantine et par semaine, et ainsi pouvoir l'étendre à l'ensemble de test.

## Séparation en jeux d'entraînement, de validation et de test

Cela permet de développer des modèles pertinents pour réaliser des prédictions sur des données futures, et pas seulement pour les données sur lesquelles le modèle a été formé. En formant les modèles, en les validant et en les testant sur l'ensemble des données retenues, on a une idée réelle de la précision de ceux-ci, ce qui permet de prendre de meilleures décisions et d'avoir davantage confiance dans les prédictions obtenues. Le jeu d'entraînement sert à obtenir les meilleurs paramètres pour le modèle, le jeu de validation sert à obtenir les meilleurs hyperparamètres et enfin le jeu de test sert de juge de paix quant à la puissance prédictive réelle du modèle.

---

## Imputations des valeurs manquantes

Suivant la variable étudiée, la technique d'imputation doit être différente. Ainsi, pour imputer la variable "prévision" hautement prédictive, j'ai choisi d'utiliser une technique bien adaptée au contexte de séries temporelles : l'interpolation temporelle, qui fonctionne sur des données quotidiennes pour interpoler une longueur d'intervalle donnée. Les autres valeurs manquantes ont été imputées, plus classiquement, avec la médiane.

## Encodage des variables catégorielles

Pour ce faire, j'ai procédé à un encodage appelé "target encoding", où il s'agit d'encoder chaque modalité d'une variable qualitative selon la moyenne de la valeur de la cible pour celle-ci. Tout en ajoutant un peu de bruit pour éviter le sur-apprentissage.

## **C. Sélection des features et du modèle**

### Sélection des features

Maintenant que nous avons dérivé une multitude de features potentiellement informatives grâce à l'EDA, notre dataset est trop fourni. Il y a un risque important de sur-apprentissage. C'est le moment d'opérer une sélection de features. Pour cela, nous utilisons l'algorithme Boruta, qui est pour la plupart des datasets un des meilleurs outils actuels.

L'algorithme Boruta est une enveloppe construite autour de l'algorithme de la forêt aléatoire. Pour sélectionner les caractéristiques intéressantes , il procède comme suit :

- Tout d'abord, il duplique l'ensemble de données, et mélange les valeurs dans chaque colonne. Ces valeurs sont appelées caractéristiques fantômes. Ensuite, il entraîne un modèle, tel qu'un Random Forest, sur l'ensemble de données.
- À chaque itération, l'algorithme compare les Z-scores des caractéristiques fantômes et des caractéristiques d'origine pour voir si ces dernières sont plus performantes que les premières. Si c'est le cas, l'algorithme marque la caractéristique comme importante. Essentiellement, l'algorithme essaie de valider l'importance de la caractéristique en la comparant à des copies aléatoires, ce qui augmente la robustesse. Pour ce faire, il suffit de comparer le nombre de fois où une caractéristique est meilleure que les caractéristiques fantômes en utilisant une distribution binomiale.
- Si une caractéristique n'a pas été enregistrée comme un succès en 15 itérations, par exemple, elle est rejetée et supprimée également de la matrice d'origine. Après un certain nombre d'itérations - ou si toutes les caractéristiques ont été confirmées ou rejetées, Boruta s'arrête.

---

Pour le projet, j'ai laissé Boruta réaliser 50 itérations, ce qui a duré environ une dizaine d'heures. Il a retenu une vingtaine de variables sur plus d'une centaine dérivées précédemment.

## Sélection du modèle

Maintenant que nous avons créé des features grâce à l'étude du domaine métier et à l'EDA, et que nous les avons sélectionnées grâce à Boruta, nous pouvons passer à la phase de sélection du ou des modèles les plus adaptés à notre dataset. L'EDA soulève quelques interrogations à ce sujet mais le meilleur moyen reste de tester une variété de modèles avec quelques combinaisons de leurs hyperparamètres.

Pour procéder de façon méthodique à la sélection du modèle, j'ai construit un package disponible sur [Github](#). Ce package teste la plupart des modèles de régression proposés par Scikit-learn avec une combinaison restreinte d'hyperparamètres afin de voir quel type de modèle (linéaire, à base d'arbres, à noyaux, réseau de neurones...) performe le mieux. Concernant le choix de ces hyperparamètres, pour le learning rate ou la régularisation, ajouter 0.1 à 0.01 à un effet important sur le comportement du modèle, en revanche ajouter 0.1 à 10 n'a pratiquement aucune conséquence. Ainsi, pour ce type de cas, nous devrions privilégier une distribution logarithmique de l'hyperparamètre.

Ce package de sélection a rendu ses conclusions. Encore une fois, celui-ci n'est pas omniscient et est encore perfectible. En effet, il est tout simplement impossible de créer un package qui balaye de façon exhaustive tous les espaces d'hyperparamètres et ce pour chaque jeu de données. Cependant, c'est un outil riche d'enseignement quant au type de modèle à utiliser pour tel ou tel dataset. Ici très clairement les modèles à base d'arbres sont les grands gagnants et cela n'est pas si surprenant car comme on l'a constaté lors de l'EDA, les relations entre les prédicteurs et la variable indépendante ne sont, la plupart du temps, pas linéaires mais bien plus complexes.

---

| Model                      | neg_mean_squared_error | Time/clf (s) |
|----------------------------|------------------------|--------------|
| LinearRegression           | -0.619                 | 0.054        |
| Ridge                      | -0.609                 | 0.024        |
| Lasso                      | -0.595                 | 0.027        |
| ElasticNet                 | -0.591                 | 0.063        |
| Lars                       | -0.637                 | 0.03         |
| LassoLars                  | -0.595                 | 0.013        |
| OrthogonalMatchingPursuit  | -0.612                 | 0.035        |
| BayesianRidge              | -0.619                 | 0.144        |
| SGDRegressor               | -inf                   | inf          |
| PassiveAggressiveRegressor | -inf                   | inf          |
| HuberRegressor             | -0.646                 | 0.744        |
| KernelRidge                | -inf                   | inf          |

The winner is: ElasticNet with score -0.591.  
(ElasticNet(alpha=1e-05, l1\_ratio=0.9, normalize=True),

*Exemple d'output de ce package concernant les modèles linéaires*

---

Afin d'aller au bout des choses, il serait intéressant de retenir les principaux modèles sélectionnés et de les optimiser avec une autre technique que Grid Search : l'optimisation Bayésienne. De plus, il serait utile de tenter des méthodes d'agrégation de modèles afin de voir si l'on peut encore progresser en terme de performance.

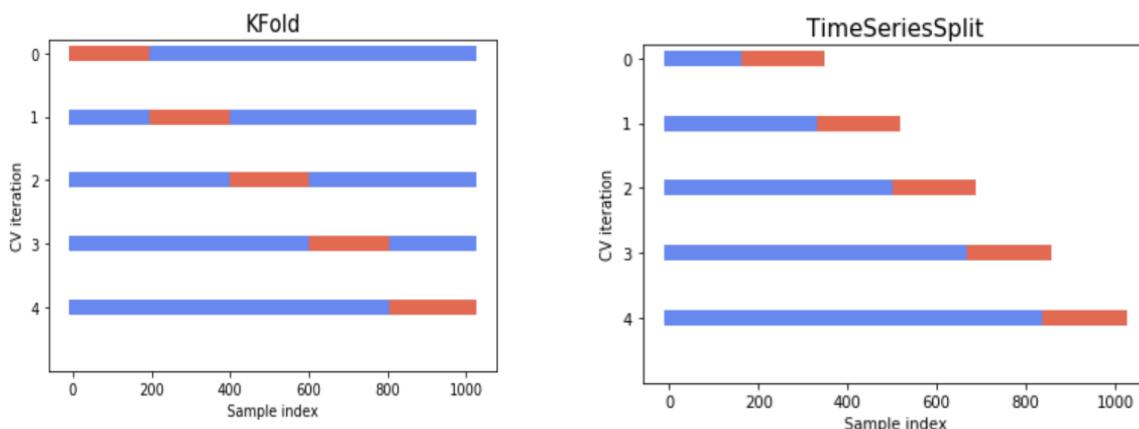
Suite à ce travail de sélection, les modèles retenus pour poursuivre l'exploration sont les suivants : ElasticNet et LassoLars, Multilayer Perceptron, Random Forest, XGBoost et LightGBM.

## D. Choix des hyperparamètres

Comme nous l'avons vu, il est nécessaire de diviser le jeu de données à disposition en plusieurs sous-ensemble pour l'entraînement, la validation et l'évaluation. Le jeu de validation permet de sélectionner les meilleurs hyperparamètres pour le modèle choisi. Un algorithme de type Grid Search ou d'optimisation Bayésienne peut alors former le modèle sur un jeu d'entraînement et le tester sur le jeu de validation, avant de recommencer et ce jusqu'à trouver la meilleure combinaison possible.

Dans ce contexte, la méthode de référence est la validation croisée. Pour chaque combinaison d'hyperparamètres testée, on entraîne et évalue le modèle plusieurs fois. Cela permet d'avoir plus de certitudes quant aux conclusions sur les performances de la combinaison testée.

Cependant, nous travaillons ici dans un contexte de séries temporelles. Cela signifie que l'ordre de nos données a son importance. Il existe une variante de la validation croisée dans le contexte des séries temporelles. Il s'agit de la validation croisée sur une base mobile. On commence par un petit sous-ensemble de données à des fins d'entraînement, on prédit les points de données ultérieurs et on vérifie la précision de ces prédictions. Les points de données prévus sont ensuite inclus dans l'ensemble de données de formation suivant et ainsi de suite.



---

## Techniques d'optimisation

Les plus communes de ces techniques actuellement sont Grid Search, Randomized Search et plus récemment, l'optimisation Bayésienne.

Grid Search n'est pas souvent utilisée dans la pratique, car le nombre de modèles à entraîner croît de manière exponentielle lorsque le nombre d'hyperparamètres à entraîner augmente. Cela peut être très inefficace, tant en termes de puissance de calcul que de temps. C'est la méthode que mon outil de recherche utilise, mais avec un nombre de combinaisons limité.

Randomized Search diffère en ce sens que nous ne fournissons plus un ensemble explicite de valeurs pour chaque hyperparamètre, mais une distribution statistique à partir de laquelle les valeurs sont échantillonnées. Essentiellement, nous définissons une distribution d'échantillonnage pour chaque hyperparamètre afin d'effectuer une recherche aléatoire (exponentielle, lognormal, normal, uniform, loguniform...). C'est la méthode que j'ai choisie pour tester davantage les combinaisons sur les modèles présélectionnés par mon package.

Dans les deux méthodes précédentes, nous avons réalisé des expériences individuelles en construisant plusieurs modèles avec différentes valeurs des hyperparamètres. Toutes ces expériences étaient indépendantes les unes des autres. Comme chaque expérience a été réalisée indépendamment, nous ne sommes pas en mesure d'utiliser les informations d'une expérience pour améliorer l'expérience suivante. A contrario, l'optimisation bayésienne est un algorithme séquentiel basé sur un modèle qui utilise les résultats de l'itération précédente pour décider des prochaines valeurs candidates des hyperparamètres. Concrètement, cela est réalisé en construisant un modèle de probabilité de la fonction objectif qui associe les valeurs d'entrée à une probabilité de perte :  $p(\text{perte} | \text{valeurs d'entrée})$ . Le modèle de probabilité, également appelé surrogate, est plus facile et rapide à optimiser que la fonction objectif initiale.

Dans le cadre du projet, j'ai choisi d'utiliser le bien nommé package BayesianOptimization pour la qualité de son implémentation et sa simplicité d'utilisation. Il requiert la définition de quatre éléments :

- une fonction objectif : elle prend une entrée et renvoie une perte à minimiser
- un espace du domaine : la gamme des valeurs d'entrée à évaluer
- un algorithme d'optimisation : la méthode utilisée pour construire la fonction surrogate et choisir les prochaines valeurs à évaluer
- des résultats : les paires de valeurs que l'algorithme utilise pour construire le modèle.

| iter | target | eta     | gamma   | max_depth | n_estimators |
|------|--------|---------|---------|-----------|--------------|
| 1    | 0.956  | 0.2614  | 0.06776 | 4.863     | 368.8        |
| 2    | 0.9637 | 0.06828 | 0.5474  | 4.333     | 195.4        |
| 3    | 0.9633 | 0.1491  | 0.8501  | 3.045     | 226.4        |
| 4    | 0.9634 | 0.2251  | 0.2347  | 3.065     | 78.04        |
| 5    | 0.964  | 0.08854 | 0.8799  | 3.064     | 136.3        |
| 6    | 0.9567 | 0.2495  | 0.7831  | 6.992     | 153.4        |
| 7    | 0.9499 | 0.2693  | 0.1507  | 5.551     | 468.3        |
| 8    | 0.9609 | 0.2052  | 0.4348  | 4.065     | 225.3        |
| 9    | 0.961  | 0.2613  | 0.486   | 3.084     | 229.8        |
| 10   | 0.9633 | 0.1178  | 0.9844  | 4.545     | 133.8        |
| 11   | 0.9621 | 0.06833 | 0.4562  | 6.117     | 137.7        |
| 12   | 0.9588 | 0.1339  | 0.03721 | 6.207     | 192.0        |

*Exemple d'output de la méthode maximize() de la classe BayesianOptimization appliquée à un XGBRegressor*

## Résultats

Ici, les résultats sont globalement identiques par rapport à la Randomized Search. La technique est cependant plus complexe à prendre en main et davantage de réglages sont requis. Par la suite, nous continuerons donc à privilégier la randomized search. Plus de détails sur cette phase peuvent être trouvés sur ce [Notebook](#).

Pour conclure, le modèle qui sort gagnant de cette phase est le LightGBM de Microsoft, légèrement devant une autre implémentation d'arbres de décision boostés : XGboost. La RandomForest et le Multilayer Perceptron obtiennent également des résultats intéressants. Du fait de la similitude des prédictions réalisées par ces différents modèles, les techniques d'ensembles n'ont pas apporté de gain significatif.

```
LGBMRegressor(max_depth=5, n_estimators=150, num_leaves=8)

Root Mean Square Error      = 22.094410468949704
Mean Absolute Error        = 15.99914128977054
Median Absolute Error      = 12.415289455391928
```

*LGBM, le meilleur modèle à l'issue de cette phase de recherche*

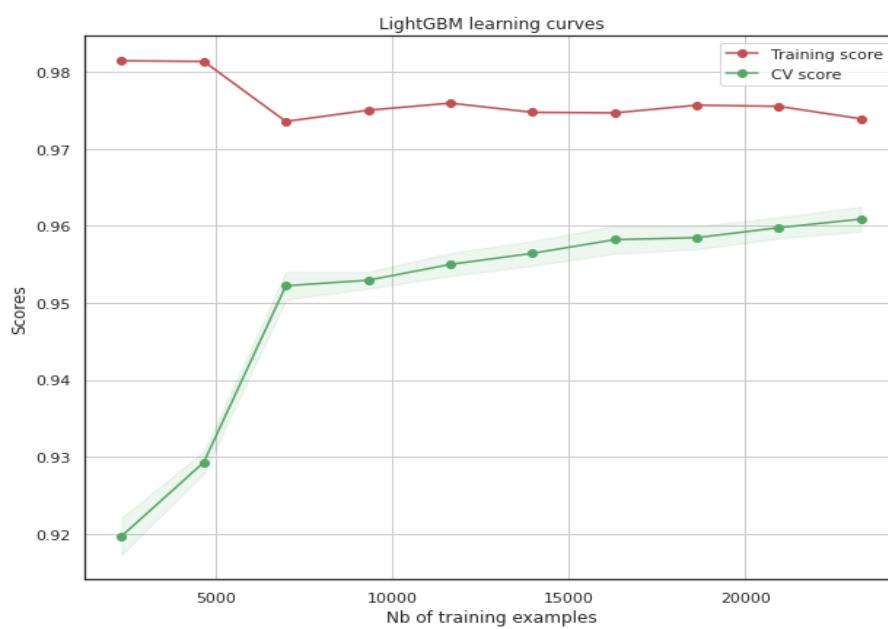
A noter que j'ai également testé la librairie d'auto-ML Auto-Sklearn, sans pour autant parvenir à battre ces métriques ([Notebook](#)). Cela dit, il peut être intéressant d'utiliser cet outil sur un dataset vierge de tout processing afin de mesurer la force de l'auto ML par rapport à un processing humain. Une chose est sûre, l'auto-ML permet un gain de temps certain et suivant l'urgence de la demande, cela peut-être un outil à considérer.

## E. Analyse des erreurs et interprétation des résultats

L'analyse des erreurs est la clé pour améliorer son modèle de machine learning. Si on obtient un modèle avec une précision de 85%, il faut se poser la question d'où viennent les 15% de classifications incorrectes. Le notebook sur le sujet se trouve [ici](#).

### Analyse des courbes d'apprentissage

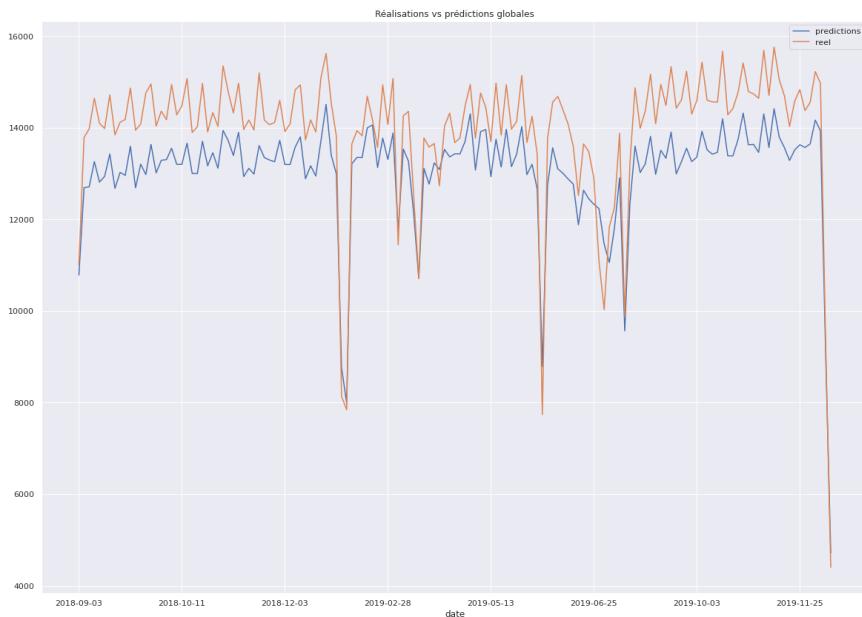
Avant de passer à l'analyse des erreurs au sens strict, il est intéressant d'observer ces courbes. Elles constituent un outil de diagnostic largement utilisé en apprentissage automatique pour les algorithmes qui apprennent de manière incrémentielle à partir d'un ensemble de données. Le modèle peut être formé et évalué sur l'ensemble de données d'apprentissage et de validation après chaque mise à jour pendant l'apprentissage et des graphiques des performances mesurées peuvent être créés pour montrer les courbes d'apprentissage. L'examen des courbes d'apprentissage des modèles pendant l'entraînement peut être utilisé pour diagnostiquer les problèmes d'apprentissage, tels qu'un modèle sous-entraîné ou sur-entraîné, ainsi que pour déterminer si les ensembles de données de formation et de validation sont suffisamment représentatifs.



Les courbes d'apprentissage de notre modèle semblent satisfaisantes. En effet, on observe une convergence entre les courbes des jeux d'entraînement et de validation et cela à un bon niveau de précision. Cela signifie qu'il n'y a ni sous-apprentissage, ni sur-apprentissage et qu'injecter davantage de données lors de

l'entraînement ne serait pas forcément bénéfique.

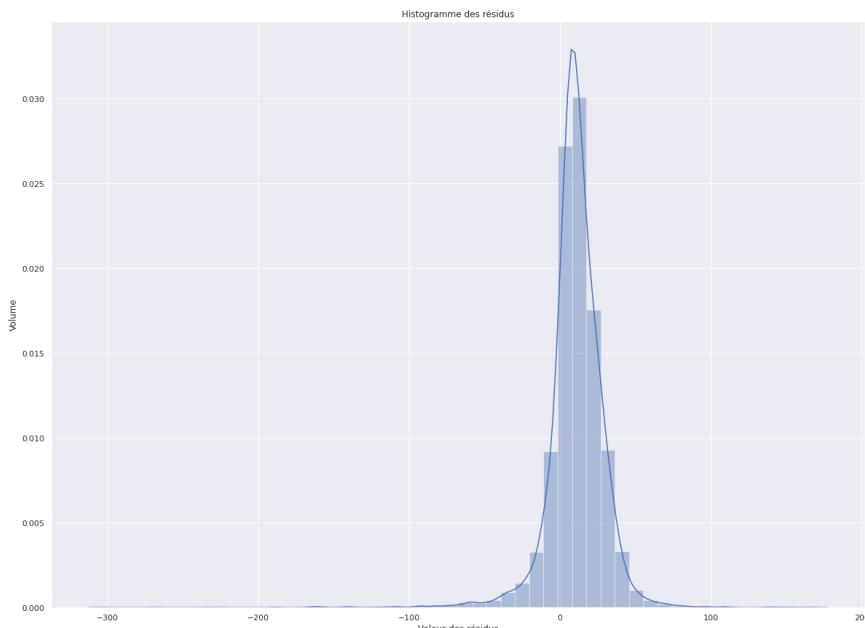
## Analyse des prédictions du modèle



Ici, ce premier visuel nous montre clairement que malgré des métriques et des résultats corrects, notre modèle sous-évalue de façon chronique la fréquentation des cantines. Notre client ne peut absolument pas se permettre de ne pas nourrir un enfant à cause d'une sous-évaluation. Ce serait pire que le gaspillage !

## Analyse des erreurs du modèles

Pour approfondir l'évaluation de la performance du modèle, il convient d'analyser la distribution des résidus. Si cette distribution ne suit pas une loi normale, cela signifie qu'il reste de l'information non capturée par le modèle dans ces résidus.



Cet histogramme montre qu'il y a une distorsion notable dans la distribution. La différence entre  $y_{test}$  et  $y_{pred}$  est souvent positive, cela signifie que notre modèle sous-évalue assez régulièrement la fréquentation des cantines. On note aussi la présence de quelques outliers qui tirent la RMSE vers le haut.

---

En réalisant d'autres graphiques, comme un diagramme Quantile-Quantile, il apparaît que les résidus sont hétéroscélestiques. Cela signifie que la variance de l'erreur n'est pas constante pour les différents niveaux de la fréquentation. En effet, les résidus sont globalement positifs (ie.  $y_{test} > y_{pred}$ ) jusqu'à 400-500 puis ensuite, on observe la présence de quelques importants résidus négatifs. Par conséquent, la signification statistique des variables indépendantes peut être surestimée ou sous-estimée.

De cette analyse, il apparaît clairement qu'il faudrait se passer de quelques outliers supplémentaires dans notre fonction basée sur le Z-score. De plus, il faut se pencher sur un moyen de régler cette sous-évaluation chronique avant des niveaux de fréquentation de 500 et plus. Ainsi, un modèle entraîné pour chaque cantine permettrait certainement d'améliorer la qualité des prédictions. La solution basée sur le Z-score est intéressante et retenue pour le modèle mis en production (encore une fois, il s'agit de prédire la période normale). La solution basée sur l'entraînement d'un modèle par cantine est intéressante et pourra être explorée pour une prochaine version du produit. Il paraît plus simple ici de se pencher sur la régression quantile pour notre MVP.

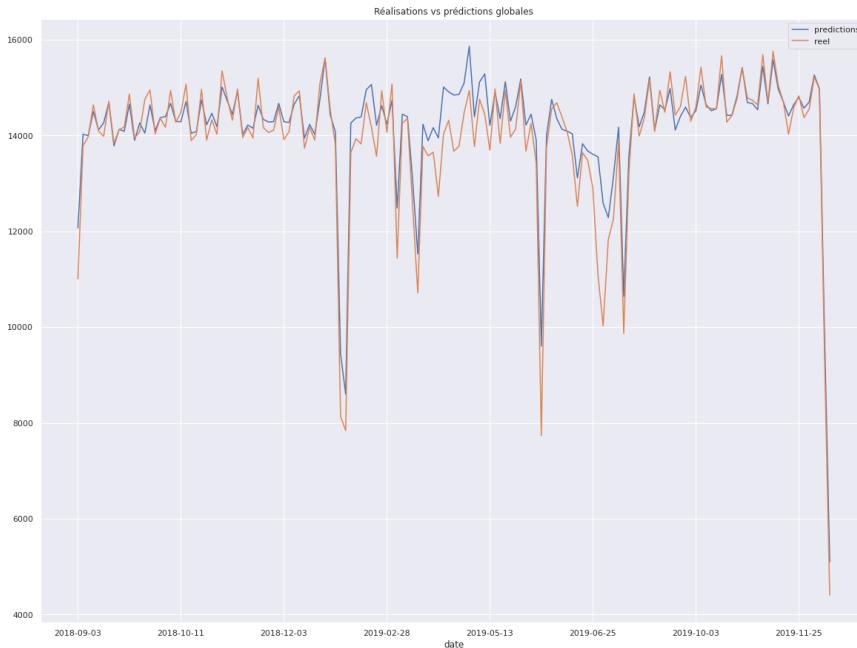
### La régression quantile

Jusqu'ici, nous avons utilisé la méthode des moindres carrés qui estime la moyenne conditionnelle de la cible pour les valeurs des prédicteurs, la régression quantile, elle, estime la médiane conditionnelle (ou d'autres quantiles) de la variable de réponse.

La différence majeure de la régression quantile par rapport à la régression classique réside dans la fonction de perte, qui est appelée perte pinball ou perte quantile. **Pour une prédiction à quantile élevé, une erreur plus élevée est moins "punie", ceci est logique dans la mesure où la fonction de perte encourage une valeur de prédiction plus élevée, et inversement pour une prédiction à quantile bas.** Dans une régression classique, la perte est punie équitablement qu'elle soit positive ou négative et quelque soit son amplitude.

Dans le cas des prévisions de fréquentation des cantines nantaises, nous pouvons constater que nos prédictions par la méthode des moindres carrés sont quasi systématiquement sous-évaluées par rapport à la fréquentation réelle. Ainsi, prédire en se basant sur une régression à quantile plus élevé paraît être une bonne idée. Cela est réalisable avec la classe LGBMRegressor en choisissant de se baser sur la fonction de perte quantile.

## Apprendre des nos erreurs



Avec une régression à un quantile de 0.90 nous obtenons un résultat bien meilleur qu'avec la méthode des moindres carrées. Pénaliser davantage les erreurs positives a porté ces fruits et orienté notre modèle dans la bonne direction. De plus, remplissons bien mieux le cahier des charges de Nantes Métropole en évitant à tout prix les sous-évaluations.

### Résultats de la régression quantile 0.90 :

- Root Mean Square Error = 20.48030046462323
- Mean Absolute Error = 12.008848289765066
- Median Absolute Error = 7.58742563216888

Quelque soit la métrique, l'erreur est bien en dessous de celle du LGBM initial, tout en corrigeant le tir concernant les sous évaluations. Nous retenons donc ce modèle pour la mise en production.

### Interprétation des résultats

Obtenir un modèle performant est bien évidemment la base en machine learning, cependant notre discipline se doit également de guider le métier de façon plus subtil et cela en justifiant les choix du modèle qui, pour un néophyte, peuvent paraître parfois bien arbitraires. Par exemple, si un modèle indique qu'une banque ne devrait pas prêter de l'argent à quelqu'un, la banque est légalement tenue d'expliquer le fondement de chaque refus de prêt. Dans notre projet, nous avons retenu un modèle qui se base sur les arbres de décision boostés. Ce modèle représente l'état de l'art en machine learning sur des données structurées. Mais, à part l'importance des caractéristiques, ce modèle ne permet pas d'obtenir des coefficients statistiquement significatifs comme une régression de Lasso le permettrait, par exemple. C'est un modèle de type boîte noire. Cependant, il est possible d'expliciter davantage ces résultats grâce à quelques librairies python. Ce [notebook](#) permet d'approfondir le sujet.

## Permutation importance

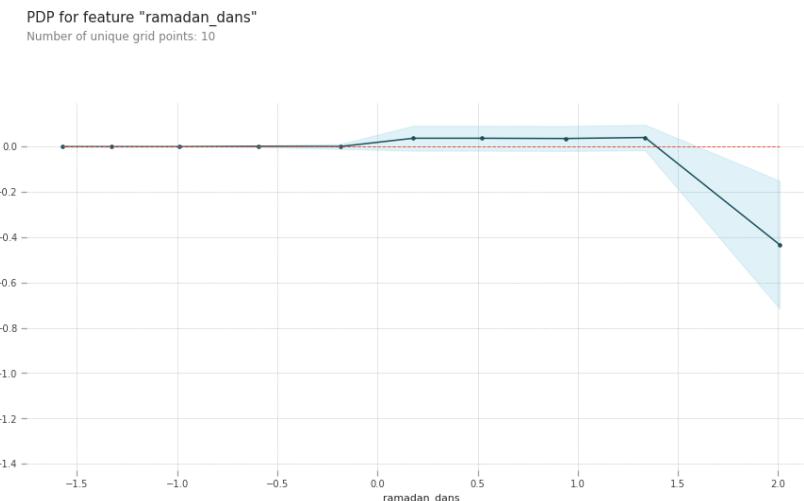
Cette technique consiste à obtenir un modèle entraîné, puis à mélanger les valeurs pour un seul prédicteur, faire des prédictions en utilisant l'ensemble de données résultant, utiliser ces prédictions et les vraies valeurs cibles pour calculer dans quelle mesure la fonction de perte a souffert du brassage. Cette détérioration des performances mesure l'importance de la variable que l'on vient de mélanger. On répète ce procédé jusqu'à ce qu'à avoir calculé l'importance de chaque variable.

| Weight              | Feature                   |
|---------------------|---------------------------|
| $1.8350 \pm 0.0131$ | prevision                 |
| $0.0046 \pm 0.0001$ | annee_scolaire            |
| $0.0034 \pm 0.0002$ | effectif                  |
| $0.0024 \pm 0.0003$ | cantine_nom               |
| $0.0012 \pm 0.0001$ | day_of_week_sin           |
| $0.0011 \pm 0.0001$ | day_of_year_cos           |
| $0.0011 \pm 0.0000$ | depuis_ramadan            |
| $0.0007 \pm 0.0001$ | prix_moyen_m2_appartement |
| $0.0006 \pm 0.0000$ | ramadan_dans              |

Cette technique permet d'estimer un poids pour chaque feature ainsi qu'un écart type associé à la dispersion des différents runs de l'algorithme. On observe comme prévu que la variable "prévision" est prépondérante. Suivie par l'année scolaire, reflétant le caractère temporel de la série, et par l'effectif. De même "cantine\_nom" souligne l'unicité de chaque cantine.

## Partial dependence plots

Alors que l'importance des caractéristiques montre quelles variables affectent le plus les prédictions, les graphiques de dépendance partielle montrent comment une caractéristique affecte les prédictions. En d'autres termes, quelle serait la valeur de la cible, toutes choses égales par ailleurs, en ne modifiant qu'une variable bien précise ? Un peu comme pour les modèles de régression linéaire, les diagrammes de dépendance partielle peuvent être interprétés de la même manière que les coefficients. Toutefois, les diagrammes de dépendance partielle de modèles sophistiqués peuvent rendre compte de schémas plus complexes que les coefficients de modèles linéaires simples.



L'axe des y représente le changement de la prédiction par rapport à ce qu'elle serait à la valeur de base. Une zone ombrée en bleu indique le niveau de confiance. Sur ce graphique en particulier, on constate qu'à l'approche du Ramadan la fréquentation des cantines baisse.

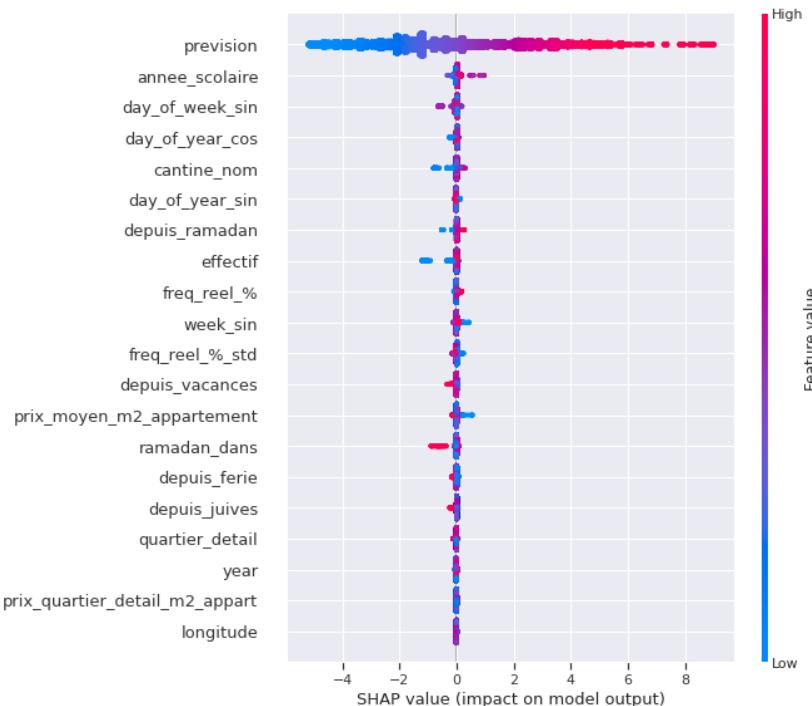
---

## Les valeurs SHAP

Les valeurs SHAP (SHapley Additive exPlanations) décomposent une prédiction spécifique pour montrer l'impact de chaque variable sur celle-ci. On ne se positionne plus au niveau de la variable mais au niveau de la prédiction.

Le graphique est constitué de tous les points des données du jeu d'entraînement. Il montre les informations suivantes :

- les variables sont classées par ordre décroissant en termes d'importance,
- l'emplacement horizontal montre si l'effet de cette valeur est associé à une prédiction plus élevée ou plus faible et dans quelle mesure,
- la couleur quant à elle, indique si cette variable est élevée (en rouge) ou faible (en bleu) pour cette observation.



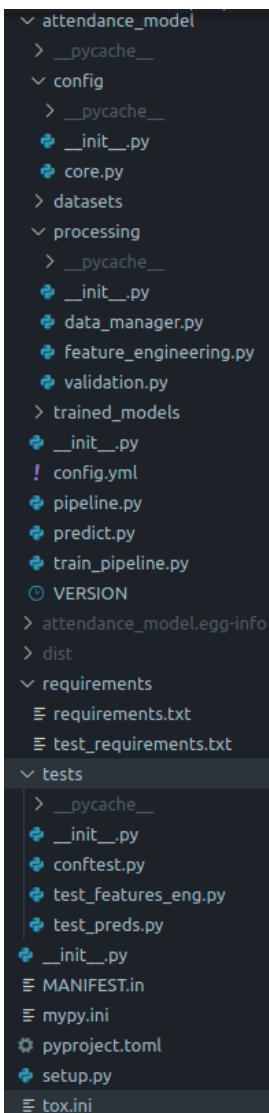
Ici, on observe qu'un niveau élevé de la valeur de l'année scolaire a un impact élevé et positif sur la fréquentation. De même, un effectif bas implique un bas niveau de fréquentation. Le prix moyen est corrélé négativement avec la cible. Attention, les valeurs SHAP n'identifient pas la causalité, qui est mieux identifiée par la conception expérimentale ou des approches similaires.

Maintenant que nous avons en notre possession un modèle performant et intelligible, nous pouvons réaliser le packaging de celui-ci.

## IV. Développement d'une application intégrant l'IA

### A. Construction du package de machine learning

Jusqu'à présent, nous avons uniquement travaillé sur du code de recherche sans nous soucier de quelque convention que ce soit. Nous devons maintenant mettre en production notre modèle. Cela consiste à tester et maintenir notre code, en améliorer sa performance, à suivre des conventions de rédaction et à le documenter de façon explicite. Ici, nous allons déployer notre modèle sous la forme d'un package Python. Ce package pourra ensuite être utilisé dans une application ou via une API. Tout d'abord, pour bien comprendre en quoi consiste cette étape clé, analysons l'architecture du package que nous allons créer. Le code du package peut se consulter [ici](#).



A la racine du package, nous trouvons des fichiers de configuration : MANIFEST.in, pyproject.toml, mypy.ini, setup.py et tox.ini.

Ce sont des éléments utilisés pour le packaging, la configuration, le linting, le type checking...

Tox est un outil de virtualisation d'environnement qui présente des avantages majeurs : éliminer les problèmes liés au PYTHONPATH lors de l'exécution de scripts/tests, éliminer la confusion liée à la configuration de virtualenv, rationaliser les étapes telles que la formation et la publication des modèles...

requirements.txt regroupe la liste des dépendances nécessaires au bon fonctionnement et au testing du package.

Dans le répertoire de tests, on trouve des tests concernant la pipeline de feature engineering et les prédictions du modèle.

Toute la logique se trouve dans le répertoire attendance\_model :

- pipeline.py exécute la pipeline de machine learning grâce à la classe Pipeline de Scikit-learn, et cela du preprocessing en passant par le scaling et l'entraînement du modèle

- train\_pipeline.py accède au jeu de données et réalise les splits appropriés, puis gère les outliers et enfin appelle la pipeline définie dans pipeline.py. Un modèle y est entraîné et stocké dans le dossier trained\_model.

- predict.py contient la fonction make\_prediction() qui utilise la pipeline entraînée pour faire des prédictions.

Dans ce répertoire, on trouve également des fichiers "helpers" :

- core.py appelle la configuration définie dans le fichier .yml et rend disponible les constantes ainsi définies sous la forme d'attributs de classe. Il y a notamment une classe AppConfig, qui regroupe les attributs

---

de configuration du package et une classe ModelConfig, qui regroupe ceux du modèle. Ici, on procède au typage des différentes constantes nécessaires à la validation à l'aide de pydantic.

- Dans le sous répertoire processing, on trouve le data\_manager.py qui contient les méthodes pour charger le dataset depuis le datawarehouse et pour sauvegarder et charger les pipelines entraînés.

On trouve aussi feature\_engineering.py qui lui se charge de définir les estimateurs et imputeurs nécessaires à pipeline.py. Toutes ces étapes de la pipeline sont des classes qui héritent des BaseEstimator et TransformerMixin de Scikit-Learn. Cela permet d'accéder à des méthodes héritées et d'utiliser la classe Pipeline de Scikit-learn.

Enfin, validation.py contient des méthodes et un schéma de validation afin de valider les entrants du modèle. Ainsi, drop\_na\_inputs() va simplement drop les lignes contenant des N/A dans des variables qui n'en contenaient pas à l'entraînement et que la Pipeline ne sait donc pas imputer. La méthode validate\_inputs() fait appel au schéma de validation des inputs défini dans le même fichier pour contrôler que les entrants ont bien le format attendu par le modèle en utilisant pydantic.

## Publication du package

Une fois le package construit et testé de façon satisfaisante, c'est le moment de le déployer. Conformément à la Loi pour une République numérique, le code ne souffre d'aucune contrainte de confidentialité et chacun peut y accéder librement. Il n'est donc pas nécessaire de déployer le code du modèle sur un dépôt hébergé pour paquets privés comme Gemfury, une simple publication sur PyPI suffira. La publication est un processus relativement simple si le paquet est bien configuré :

- créer un compte sur PyPI
- installer les outils twine et build
- créer les archives de distribution : python3 -m build
- publier le package : twine upload dist/\*

## **B. Servir le modèle packagé via une API REST**

Il y de nombreuses façons d'appréhender l'architecture d'un projet de machine learning, mais en voici deux qui semblent crédible dans le cadre de ce projet :

- Modèle intégré : le modèle est vu comme une simple dépendance de notre application. Ceci est d'une grande simplicité mais si l'on veut réaliser une mise à jour du modèle, il faut redéployer l'intégralité de l'application.
- Modèle servi par une API : notre application réalise ensuite des appels sur cette API dédiée. Ici, le compromis est inversé : il est plus complexe de maintenir deux services

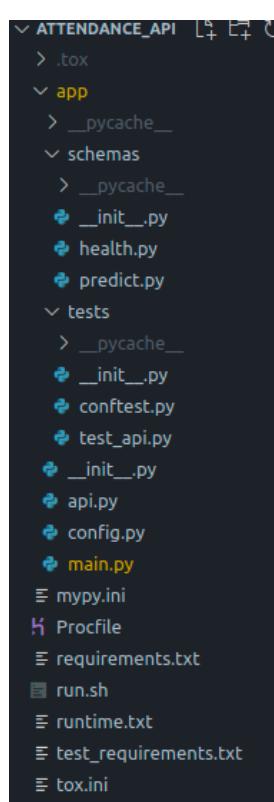
---

qu'un seul, mais désormais le déploiement du modèle est séparé du déploiement de l'application. Un autre avantage est que l'on peut utiliser l'API pour servir d'autres applications et lors d'un échange avec le client, il a été fait la demande de la mise à disposition d'une API qui permettrait aux équipes de la Métropole d'exploiter l'IA créée dans d'autres projets dans le futur.

Suite à cette réflexion nous choisissons l'option de séparer le modèle de l'application et de le servir via une API REST réalisée avec FastAPI pour les raisons évoquées dans les spécifications techniques.

Le point de terminaison principal de l'API est le endpoint predict. Il prend en entrée un ou plusieurs inputs dont le format est validé par pydantic, puis il appelle le package attendance\_model et réalise une ou plusieurs prédictions. Ce qu'il faut noter concernant la validation des inputs, c'est que l'on importe le schéma de validation des données défini dans le package attendance\_model. Ainsi, le moindre changement de ce schéma dans celui-ci sera répercuté dans l'API pour peu que l'on incrémente la version du modèle dans le fichier requirements.txt.

Afin de mieux comprendre le concept, voyons plus en détails l'architecture de l'API dont le code est disponible [ici](#).



A la racine du code, nous retrouvons, comme précédemment, des fichiers de configuration : mypy.ini, Procfile, tox.ini, requirements.txt et un script bash. Procfile est un fichier qui sera nécessaire au déploiement de l'API.

Dans le répertoire tests, on trouve des tests concernant l'appel du endpoint predict et l'analyse de la réponse qu'il retourne suite à l'envoi d'un payload au format JSON.

Dans le répertoire schemas, on retrouve notre validation implémentée grâce à pydantic mais surtout héritée du package via la classe DataInputSchema. predict.py contient également un exemple d'inputs pour alimenter la documentation de l'API.

Dans config.py, je procède à la configuration du logger de l'API sur un niveau INFO et j'utilise la classe InterceptHandler pour configurer le handler principal.

Il faut également configurer le format du endpoint de l'API et le Cross Origin Resource Sharing. Pour des raisons de sécurité, une application web qui utilise l'API peut uniquement émettre des requêtes vers la même origine que celle à partir de laquelle l'application a été chargée, sauf si des en-tête CORS sont utilisés.

Dans main.py, on initialise le logging, puis on crée l'API et une route '/'.

---

Dans api.py, on crée une route '/health' pour vérifier le statut de l'API et une route '/predict' pour y envoyer les payloads JSON contenant les inputs à prédire via une requête POST.

## C. Construire le MVP : l'application Web

### Conception de l'interface web

En ce qui concerne l'outil de développement, j'ai choisi Flask. C'est un framework web écrit en Python connu pour simplifier la création de sites web et d'API, notamment avec son architecture Model-View-Controller (MVC). Cette architecture met l'accent sur la séparation entre la logique métier et l'affichage du logiciel. Cette séparation des "préoccupations" permet une meilleure répartition du travail et une maintenance optimisée :

- un modèle (Model) contient les données à afficher,
- une vue (View) contient la présentation de l'interface graphique,
- un contrôleur (Controller) contient la logique concernant les actions effectuées par l'utilisateur.

Cette organisation a largement accru ma productivité et la qualité de suivi de mon travail. Ce framework dispose d'une documentation riche et d'une base d'utilisateurs très active. De plus Flask possède un ORM (Object-Relational Mapper), qui permet d'interagir avec la base de données et de transmettre des données entre une base relationnelle et un modèle d'application. Enfin, il possède bon nombres de plugins qui facilitent la gestion des utilisateurs ou la construction des formulaires, entre autres. Le code de la web app est disponible [ici](#).

### Structure du projet web : les Blueprints

Le schéma de base est donc un MVC, mais pour des projets plus structurés, il convient de diviser celui-ci en plusieurs packages à l'aide des Flask Blueprints. Un Blueprint est un package qui encapsule une fonctionnalité spécifique de l'application. Chaque Blueprint possède au moins un module qui contient les routes de celui-ci, un répertoire stockant les différents templates HTML et un autre pour les fichiers de statiques spécifiques de ce Blueprint (images, CSS, JS...).

Ainsi, au lieu que les itinéraires aient un décorateur @app, ils seront décorés du nom du plan auquel ils appartiennent, par exemple @home\_bp.route('/login'). D'autres fichiers viennent s'ajouter à ces éléments indispensables et notamment à la racine de l'application. Ils sont détaillés ci-après.

```

> tests
> venv
< webapp
  > __pycache__
  < canteen_bi
    > __pycache__
    > templates
    & __init__.py
    & bi_routes.py
  < canteen_predict
    > __pycache__
    > templates
    & __init__.py
    & predict_routes.py
  > data
  < home
    > __pycache__
    > templates
    & __init__.py
    & routes.py
  < scripts
    > __pycache__
    & data_load.py
    & data_prep_freq.py
    & data_prep_geo.py
    & data_prep_menus.py
    & data_prep_tempo.py
  > static
  > templates
  & __init__.py
  & Forms.py
  & models.py
  & .env
  & .gitignore
  & config.py
  & LICENSE
  & Procfile
  & readme.md
  & requirements.txt
  & webapp.log
  & wsgy.py

```

À la racine, on retrouve comme à l'accoutumé, un répertoire de test, un fichier de configuration, un fichier de logs et le module wsgy.py où l'application est créée. Nous reviendrons plus en détails sur la configuration, le testing et le logging de l'application par la suite.

La fonction `create_app()` est définie dans le fichier `__init__.py` à la racine de `webapp`. C'est ce que l'on nomme une fonction App Factory. L'utilisation de la méthode Factory Pattern permet de choisir la configuration de l'application avant de l'instancier. Ainsi, la fonction renvoie une instance d'application entièrement configurée sur-mesure.

Dans cette fonction, on trouve :

- l'instanciation de l'application Flask
- le choix de la configuration de l'application
- l'instanciation et utilisation des plugins Flask
- l'enregistrement des Blueprints
- la création des modèles de données
- la gestion d'erreurs et logging

Toujours à la racine de `webapp`, on trouve `forms.py`, dont le rôle est de configurer et sécuriser les différents formulaires de l'application. Ici chaque formulaire est une classe qui hérite de la classe `FlaskForm` du plugin `flask_wtf`. Les attributs de ces classes sont les champs des formulaires et chaque champ est soumis à diverses contraintes de formatage.

`models.py`, quant à lui, définit une classe `User` qui hérite de la classe `UserMixin` de `flask_login` et de `db.Model` instancié lors de la création de l'application. Cette classe définit les différents champs et les contraintes associés à notre modèle utilisateur. Elle contient également les méthodes `set_password` et `check_password` qui assure que le mot de passe utilisateur n'est pas stocké en clair dans la base de données. Dans `set_password()`, le mot de passe est haché avec l'algorithme SHA256. `check_password()` sert à le décrypter par la suite et à vérifier que l'utilisateur entre le bon mot de passe quand il tente de se connecter.

Concernant les Blueprints de notre application, ils sont au nombre de trois :

- `home` gère la partie inscription, connexion, suppression du compte et contient la page d'accueil qui présente l'application
- `canteen_bi` gère la partie tableau de bord de l'application, elle contient les graphiques et les outils de pilotage nécessaires au client pour appréhender visuellement son activité. A noter que cette partie fait appel au répertoire `scripts`, qui contient le code

---

nécessaire au requêtage du datawarehouse et à la création des différents graphiques Plotly.js.

- canteen\_predict gère la partie IA de l'application et contient une page pour réaliser des prédictions individuelles et une page pour réaliser des prédictions par lots.

## Gestion de la configuration

La configuration d'une application est le processus de spécification de la valeur des paramètres clés utilisés pour contrôler le comportement de l'instance d'application et de ses plugins. Une méthode de configuration qui s'adapte bien et permet de spécifier différents paramètres pour différents environnements consiste à utiliser un ensemble de classes python.

Dans le fichier config.py situé dans le répertoire racine du projet , on définit 4 classes :

- la classe Config, qui est la classe de configuration de base qui contient les paramètres de configuration qui s'appliquent à tous les environnements. Les 3 autres classes héritent de cette classe.
- la classe DevConfig(Config), qui contient les paramètres de configuration de l'environnement de développement.
- la classe TestConfig(Config), qui contient les paramètres de configuration de l'environnement de test.
- la classe ProdConfig(Config), qui contient les paramètres de configuration de l'environnement de production.

En pratique, il existe certains paramètres tels que la clé secrète, l'adresse et le mot de passe de l'administrateur, le nom d'utilisateur et le mot de passe du serveur de messagerie... qu'il ne faut absolument pas coder en clair dans config.py pour des raisons de sécurité. Ainsi, les paramètres sensibles sont définis dans l'environnement (le fichier .env) et sont ensuite importés dans le fichier config.py à l'aide de la méthode os.environ.get() après avoir appelé la méthode load\_dotenv(). Attention, il faut impérativement ajouter.env au .gitignore pour ne pas que ce fichier sensible se retrouve sur Github.

## Gestion des erreurs

HTTP définit un ensemble de méthodes de requête qui indiquent l'action que l'on souhaite réaliser sur la ressource indiquée. Lorsqu'un utilisateur envoie une requête HTTP à notre application, le message de réponse inclut un code d'état HTTP. C'est un nombre à 3 chiffres qui indique le résultat du traitement de la demande. Les codes d'état sont regroupés en 5 catégories en fonction du premier chiffre, et chaque catégorie représente un type de réponse (200 = succès, 300 = redirection vers une URL, 400 = erreur côté client et 500 = erreur côté serveur).

---

Une bonne pratique est de créer des pages personnalisées. Cela permet de garder une cohérence graphique sur tout le site même lors de la gestion des erreurs et surtout, cela permet d'informer le client de façon plus explicite qu'un déroutant "500: Internal server error".

Dans le fichier où se trouve la méthode App Factory, toutes les fonctions de gestion d'erreurs sont définies dans une fonction mère. Cette fonction est appelée dans la fonction App factory et enregistre tous les gestionnaires d'erreurs à la fois. En effet, il est préférable d'enregistrer les gestionnaires d'erreurs avec l'instance d'application afin qu'ils puissent être utilisés par toutes les fonctions d'affichage dans tous les Blueprints.

### Sécurisation de la Web App

Le Cross-Site Request Forgery est une attaque qui utilise les informations d'identification de la victime pour effectuer des actions indésirables au nom de la victime. Cette vulnérabilité devient grave si elle est enchaînée avec une injection XSS.

Par défaut, le framework Flask n'a pas de protection CSRF, mais nous pouvons utiliser l'extension Flask-WTF pour activer celle-ci. Dans `_init__.py`, lorsque l'on instancie l'application, on lui passe le plugin CSRF. Cela permettra la protection CSRF à l'échelle de l'application. Chaque formulaire aura un champ de saisie caché contenant notre jeton de protection CSRF généré aléatoirement par le Flask-WTF. Puis dans les templates, on protège nos formulaires en insérant ce jeton dans le code HTML.

## **D. Testing et logging**

### Le testing

Le testing est une pratique de test logiciel qui consiste à tester des sites ou des applications web pour bogues potentiels :

- Les tests basiques regroupent l'ensemble des vérifications sur les différentes pages et leurs accessibilités. Par exemple, pour tester la fonctionnalité de login de l'utilisateur, il faudra reprendre la fonction implémentée dans l'application puis injecter des données pour ensuite vérifier que le résultat est exactement celui attendu. Ici, le code « 200 » signifie que la requête s'est bien déroulée.
- Les tests des modèles regroupent l'ensemble des vérifications des modèles. En effet, ces tests permettent de savoir si les modèles de l'application ont été créés correctement et peuvent recevoir les données envoyées par les pages.

---

Certains tests sont dits unitaires car ils ne testent qu'une fonctionnalité, qu'une méthode du code et d'autres sont dits tests d'intégration car ils s'attachent à vérifier la manière dont les différents éléments du code s'exécutent en harmonie.

Pour réaliser ce testing, j'utilise le module pytest qui apporte de nombreuses features utiles par rapport à unittest. Une feature très utile est la fixture, elle rend la maintenance du code aisée. Les fonctions de fixtures pytest sont réutilisables et peuvent être utilisées aussi bien pour des tests unitaires simples que pour des tests de scénarios plus complexes. Ici une fixture se charge de créer un nouvel utilisateur qui sera utilisé pour tester la page de connexion. Une autre fixture va créer une instance de l'application à des fins de test. Cette application viendra en entrée de la plupart des fonctions de test créées : test\_home\_get(test\_client), test\_home\_post(test\_client), test\_valid\_login\_logout(test\_client), test\_valid\_prediction(test\_client)...

A noter que le package et l'API sont également testés et le code est consultable sur Github.

## Le logging

Il est utilisé pour enregistrer les événements au fur et à mesure qu'ils se produisent et constitue un excellent outil pour déboguer tout problème et obtenir un aperçu du fonctionnement de l'application. On peut enregistrer les événements liés à l'utilisateur (connexion, déconnexion...), événements spécifiques à l'application (ici, la réalisation d'une ou plusieurs prédictions, la consultation d'un graphique pour telle cantine, sur telle période...) et bien-sûr toutes les erreurs spécifiques à l'application.

Le module standard de logging de python contient 4 éléments principaux :

- les loggers : objets qui créent les messages de log et les impriment dans la console, par défaut. On peut désactiver le logger par défaut et en configurer un nouveau pour enregistrer un certain niveau critique et stocker l'output dans un fichier.
- les handlers : ce sont eux qui permettent justement de choisir la destination des logs (mails, fichier texte, base de données, Azure App Insights...)
- les filters : ils laissent le choix au développeur d'ajouter certaines informations à capturer (comme les adresses IP)
- les formatters : ils permettent de définir le format des informations captées par le logger (par défaut, il y aura %(asctime)s, %(levelname)s et %(message)s).

Comme vu précédemment, on configure le logging dans la fonction App factory : on instancie l'objet handler, on définit le niveau de logging, on définit un formatter et on l'ajoute au logger et enfin, on désactive le logger par défaut. Pour notre application web, j'ai choisi de logger l'activité des utilisateurs quant à l'utilisation des différents graphiques et de l'outil prédictif. J'ai aussi procédé au logging des activités traditionnelles de l'utilisateur. Par exemple, à chaque

---

fois qu'un utilisateur réalise une requête POST via le formulaire de prédiction, je capture les données entrées ainsi que la réponse de l'API, avec deux lignes de code :

- app.logger.info(f"Réalisation d'une prédiction sur les inputs suivant : {DATA}")
- app.logger.info(f"Résultat de la prédiction : {json\_format['predictions'][0]}")

Les logs sont ensuite sauvegardés dans un fichier .txt pour une exploitation future. La classe RotatingFileHandler du module logging permet de limiter la taille d'un fichier de log avant d'en créer un nouveau et également de limiter le nombre de fichiers conservés en historique.

## V. Déploiement des différentes composantes

### A. Déploiement de l'API

Pour déployer cette API, il nous faut un serveur Web dédié capable de gérer une certaine charge et un certain nombre de requêtes entrantes. Une spécificité de FastAPI par rapport à Flask ou Django tient justement à cela. Avec l'introduction de la librairie python asyncio, l'Asynchronous Server Gateway Interface (ASGI) est née et remplace la WSGI de Flask et Django. Uvicorn est une implémentation de celle-ci.

Pour l'hébergement à proprement parler, nous allons utiliser Heroku, un PAAS (Platform As A Service) qui simplifie grandement le processus car on n'a pas à configurer le Hardware, l'OS ou le Middleware. Cela nous permet de nous focaliser sur notre cœur de métier : le machine learning et le développement applicatif. Enfin, Heroku est simple, gratuit pour une application et très fiable, car c'est une référence sur le marché et qu'il garantit 99.999% d'uptime.

Pour déployer sur Heroku, il suffit de créer un compte sur le site et d'activer l'authentification multifacteur. Ensuite, il faut télécharger le CLI d'Heroku et utiliser les commandes suivantes :

- heroku create (créer l'application hébergée par Heroku)
- heroku git:remote -a immense-harbor-59241 (c'est un nom donné aléatoirement par Heroku à l'application déployée, il est possible de le changer).
- git subtree push --prefix attendance\_api heroku main (pousse le sous répertoire git local qui contient l'API, sur le répertoire distant Heroku défini ci-dessus)

Une fois l'API déployée, on peut la requêter grâce à l'exemple intégré à la documentation ou par une commande curl dans le terminal, cela permet de s'assurer que le service fonctionne avant d'intégrer le point de terminaison dans l'application web. A noter que la commande heroku logs --tail affiche les derniers logs de notre API.

## B. Déploiement de la web application

### Migrer la base MySQL locale sur Azure

Pour se faire, il est nécessaire de créer une ressource Azure Database for MySQL sur le portail Azure. Ensuite, il faut autoriser notre IP à se connecter à la base nouvellement créée.

The screenshot shows the MySQL Workbench Migration Wizard Report. On the left, there's a sidebar with navigation links for SOURCE & TARGET, OBJECT MIGRATION, DATA MIGRATION, and REPORT. The REPORT section is currently selected, showing a "Migration Report" button. The main pane displays the migration report details:

- MySQL Workbench Migration Wizard Report**
- Date: Thu Aug 5 11:23:30 2021
- Source: MySQL 5.7.34
- Target: MySQL 5.7.32
- I. Migration**
  - 1. Summary**  
Number of migrated schemas: 1
  - 1. users**  
Source Schema: users
    - Tables: 3
    - Triggers: 0
    - Views: 0
    - Stored Procedures: 0
    - Functions: 0
  - 2. Migration Issues**
  - 3. Object Creation Issues**

Pour la migration, j'utilise le logiciel Workbench qui dispose d'un excellent outil pour cela. Je précise ma source, avec mes identifiants MySQL locaux et la base que je veux migrer. Et je précise la cible, la base Azure nouvellement créée. Je sélectionne la table myusers et Workbench va rédiger un script de migration pour celle-ci. Je peux choisir de transférer ou non mes données utilisateur, ce que je fais dans un souci de réalisme.

### Déploiement de la Web app

Il existe de nombreuses façons de déployer sur Azure : via Azure Devops et une pipeline de CI/CD, via un répertoire Github, en passant par le portail Azure ou le CLI... Pour ma part et afin de manipuler une technologie très demandée sur le marché de l'emploi, j'ai choisi de passer par un container hébergé sur le Docker Hub. Avec la technologie Docker, on peut traiter les conteneurs comme des machines virtuelles très légères et modulables. En outre, ces conteneurs offrent une grande flexibilité : création, déploiement, copie et déplacement du contenu d'un environnement à un autre. Cela est particulièrement efficace pour optimiser nos applications pour le cloud.

Chaque conteneur Docker débute avec un Dockerfile. Il comporte les instructions de création d'une image Docker. Un Dockerfile précise le système d'exploitation sur lequel sera basé le conteneur, le langage utilisé, les variables d'environnement, les emplacements de fichiers, les ports réseaux... Une image Docker est composée de plusieurs couches empaquetant toutes les installations, dépendances, bibliothèques, processus et codes d'application nécessaires

---

pour un environnement de conteneur pleinement opérationnel. Après avoir écrit le Dockerfile, on utilise l'utilitaire build pour créer une image basée sur ce fichier.

A noter à la fin du Dockerfile, la présence de la commande : ENTRYPOINT ["init.sh"]. Celle-ci est fondamentale car elle indique à Docker comment lancer notre application une fois l'image construite. init.sh est un script bash qui va lancer un serveur SSH sécurisé puis exécuter l'application sur un serveur web de production. Nous utilisons ici le serveur Gunicorn car une fois l'application déployée, il sera nécessaire de traiter nombre de requêtes et d'envoyer les réponses associées. Le serveur de développement de Flask en serait incapable.

Une fois l'image docker construite, il convient de la tester localement à l'aide de la commande docker run <image\_name>. Si tout fonctionne correctement nous pouvons maintenant "pousser" l'image sur un registre privée sur le Docker Hub. Pour des raisons de sécurité, Azure favorise l'utilisation de registres privés dans le cadre de l'utilisation de ses App Services. Pour se faire, il faut se connecter via le terminal avec docker login, puis commiter les changements éventuels par docker commit et ensuite "pousser" l'image sur le Hub avec docker push.

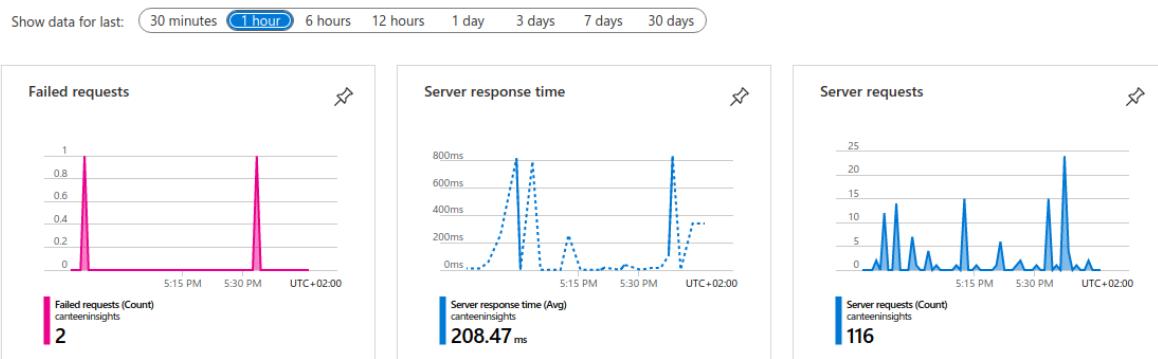
Pour des raisons de sécurité, j'ai ajouté un fichier nommé sshd\_config à la racine du projet. Ce fichier active un serveur ssh et le client peut alors avoir une connexion sécurisée avec le conteneur.

Une fois la construction et l'hébergement du conteneur réalisés, nous pouvons créer notre ressource Azure App Service sur le portail Azure. Il faut préciser un déploiement par conteneur Docker et le registre de ce conteneur sur le Docker Hub en précisant l'image et le tag de celui-ci. Ensuite, il faut lier l'application à un groupe de ressources et créer un App Service Plan.

Quelques points de blocages potentiels sont à évoquer. Comme le fichier .env n'a pas été poussé dans le registre Docker pour des raisons de sécurité, il est nécessaire d'ajouter les variables d'environnement soit dans le Dockerfile, soit sous l'onglet Configuration dans notre App Service (connexion à l'API, à la base de données Azure...). De plus, la base de données Azure que nous avons créé précédemment doit être configurée afin d'autoriser l'accès aux services Azure, sans quoi notre application ne pourra s'y connecter. Enfin, le serveur SMTP de l'application qui fonctionne parfaitement en local est bloqué par Azure, en conséquence il est nécessaire d'utiliser la ressource Sendgrid de Twilio disponible sur le Marketplace de Microsoft. L'implémentation dans notre application Flask est exactement la même.

## C. Monitoring de l'application

Un autre avantage au déploiement de l'application Web sur Azure est la possibilité de créer une ressource Application Insights spécialement dédiée. Cela permet de montrer les différentes requêtes et connexions entrantes, les échecs, les temps de réponses, les pays qui consultent le plus l'application, les pages les plus consultées, les pages les plus lentes...



*Application Insights Overview page*

| OPERATION NAME      | DURATION (AVG) | COUNT |
|---------------------|----------------|-------|
| Overall             | 196 ms         | 130   |
| GET /freq           | 1.83 sec       | 3     |
| POST /freq          | 1.71 sec       | 2     |
| GET /geo            | 1.62 sec       | 3     |
| POST /menus         | 1.59 sec       | 2     |
| GET /menus          | 1.58 sec       | 2     |
| GET /tempo          | 636 ms         | 3     |
| POST /multi_predict | 525 ms         | 2     |

*Application Insights Operations Performance page*

Et bien-sûr, il est possible de récupérer les logs spécifiques à notre application que nous avons manuellement configurer dans notre code python et ainsi, nous obtenons des informations sur les utilisateurs créés, les différentes prédictions réalisées, les inputs postés et les graphiques consultés...

**Completed.** Showing results from the last 24 hours.

## *Application Insights Logs/Traces page*

Enfin, Application Insights nous permet de définir des règles d'alertes de journal avec le portail Azure. On peut utiliser des alertes par défaut proposées par Azure ou configurer ses propres alertes sur-mesure. Ces alertes peuvent déclencher plusieurs actions : notifications ou automatisation, e-mail, SMS, webhook, etc. Ici dans l'onglet Monitoring de Application Insight, je configure une alerte où je reçois un email récapitulatif concernant les requêtes traitées, uniquement si celles-ci dépassent un certain seuil sur une certaine période.

Run Time range : Last 3 days Save Share + New alert rule Export ...

```
1 // Operations performance
2 // Calculate request count and duration by operations.
3 // To create an alert for this query, click '+ New alert rule'
4 requests
5 | summarize RequestsCount=sum(itemCount), AverageDuration=avg(duration),
percentiles(duration, 50, 95, 99) by operation_Name // you can replace
'operation_Name' with another value to segment by a different property
6 | order by RequestsCount desc // order from highest to lower (descending)
```

...

Results Chart | Columns Display time (UTC+00:00) Group columns

Completed. Showing results from the last 3 days. 00:00.2 17 records

| operation_Name                    | RequestsCount | AverageDuration | percentile_duration_ |
|-----------------------------------|---------------|-----------------|----------------------|
| › GET /static/img/nantes_nobg.png | 24            | 5.833           | 1                    |
| › GET /                           | 9             | 39.778          | 4                    |
| › GET /login                      | 6             | 172.667         | 14                   |
| › GET /static/img/contact.jpg     | 4             | 15.5            | 14                   |
| › GET /predict                    | 4             | 12              | 12                   |

*Exemple d'alerte configurable via l'onglet Monitoring d'App Insights (Récapitulatif des requêtes traitées par l'application sur les 72 dernières heures)*

Pour activer Application Insights pour notre application, il faut créer une ressource Application Insights dans Azure, puis la mapper sur l'application à montrer. Ensuite, lors de la création, une chaîne de connexion est générée. Dans le code de l'application, il faut remplacer la classe `RotatingFileHandler` que l'on a configurée initialement et qui stockait les logs sous format .txt, par la classe `AzureLogHandler` issu de la librairie Opencensus qui lui permet de stocker les logs directement dans Application Insights.

---

# Conclusion

## Conclusions et axes d'amélioration

Il est possible d'apporter de nombreuses améliorations à ce projet. Mais ce MVP répond à la demande initiale de la métropole de Nantes. En effet, il était demandé de délivrer un modèle prédictif pouvant servir d'aide à la décision afin d'ajuster les commandes et la production planifiées deux à trois semaines à l'avance. Ce modèle ayant la charge de réaliser des prédictions sur des périodes normales. Enfin, il fallait mettre cet outil à disposition via une application Web comprenant également des éléments de data visualisation. C'est à ce jour le produit proposé à la Métropole et à ses agents.

Concernant les points qui pourraient faire l'objet d'une seconde version :

- réaliser des recommandations personnalisées par rapport aux prévisions des agents et à la façon dont ils devraient les corriger et des recommandations par rapport aux menus à succès, non pas en terme de gaspillage imputable aux erreurs de prévision, mais en terme de gaspillage dans les assiettes,
- ce gaspillage par assiette pourrait être estimé par de la computer vision ou plus simplement en pesant les déchets alimentaires à chaque fin de service,
- entraîner un modèle pour chaque cantine, ce qui permettrait potentiellement d'améliorer la qualité des prédictions de ceux-ci,
- créer une solution unifiée sur Azure, ce qui permettrait d'avoir tous les services regroupés au même endroit, des expériences de data science, jusqu'à l'API en passant par le modèle de machine learning,
- améliorer le design du site en intégrant des composants React, ce qui permettrait de fédérer encore davantage les agents à l'utilisation de l'outil.

## Difficultés et apprentissages

Ce projet a démarré pour moi en mai 2021, suite à de trop lourdes contraintes pesant sur mon projet initial (liées à l'obtention de données fiables et labellisées, aux coûts de la puissance de calcul nécessaire et au manque de disponibilité du client). Au vu de ce contexte d'urgence, j'ai mesuré l'ampleur de la tâche à venir et son caractère chronophage. Ceci fut la première difficulté majeure que j'ai rencontrée car après une journée en entreprise, il est parfois compliqué de se remobiliser le soir pour travailler. Cela dit, le sujet de l'IA au service de l'intérêt général m'a tout de suite captivé et mes ambitions pour la suite m'ont poussé à relever le défi.

---

Un autre écueil que j'ai rencontré tient au fait de travailler seul sur ce projet. Dans mon quotidien en entreprise, j'aime échanger et challenger les idées de mes collègues. N'étant pas issu du monde de l'informatique et n'ayant pas un entourage à même de m'aider sur ces problématiques, j'ai passé de nombreuses heures sur les forums et sites d'entraide afin d'éclairer mes pérégrinations programmatiques. Cette situation a été insécurité car je n'étais jamais sûr de la direction que j'empruntais et si j'allais perdre quinze jours sur une fausse piste ou non. J'ai donc usé et abusé de la plus belle invention de la fin du 20ème siècle qu'est Internet et je suis passé ceinture noire en "Google-fu"!

Mais ces difficultés ont boosté ma motivation et ont contribué à une acquisition active des savoirs. C'était un véritable challenge qui m'a permis d'acquérir des compétences en matière de gestion de projet, de bases de données, de data science mais également en terme de programmation python et SQL, de développement web avec Flask et Bootstrap, mais aussi toute la partie configuration et environnements virtuels, le déploiement à l'aide de conteneurs et de la plateforme de cloud Azure, jusqu'aux tests, au logging et au monitoring une fois l'outil déployé.

Travailler seul sur ce projet m'a obligé à sortir de ma zone de confort et à m'obstiner encore davantage à chaque échec. Ce projet n'est qu'un commencement, mais il valide le choix que j'ai fait en rejoignant Simplon et me donne de la confiance pour la suite de mon parcours professionnel.

Je tiens à remercier les personnes qui m'ont apporté un soutien moral indéfectible sur ce projet et vous également pour avoir pris le temps de me lire. J'attends avec intérêt vos retours et conseils quant à mon travail.

---

## Sources

### Livres :

- Hands-On Machine Learning with Scikit-Learn and TensorFlow par Aurélien Géron
- Machine Learning Engineering par Andriy Burkov
- Applied Machine Learning par David Forsyth
- Applied Predictive Modeling de Kjell Johnson

### MOOCS :

- Coursera : Deep learning specialization, Machine learning par Andrew NG, Tensorflow developer certificate, Natural language processing specialization, Python 3 programming specialization
- EDX : Querying data with transact-SQL, Principles of machine learning
- Datacamp : Time Series Analysis in Python

### Sites web :

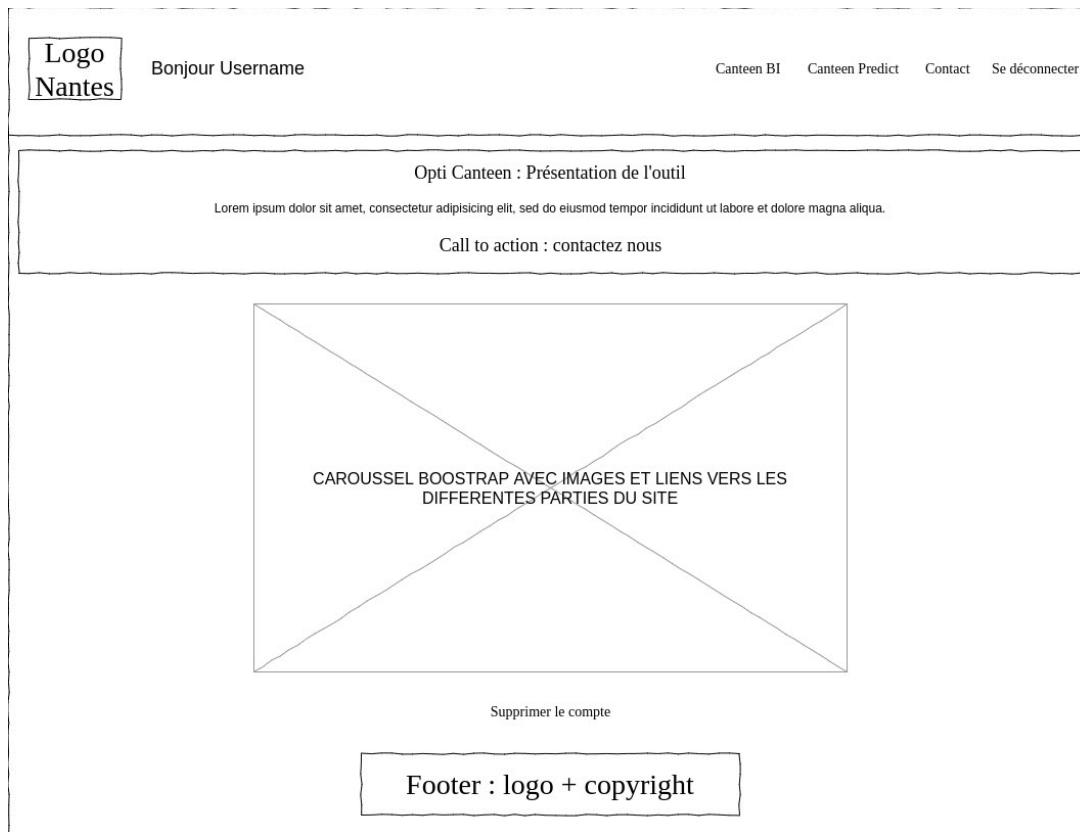
- [stackoverflow.com](https://stackoverflow.com)
- [stats.stackexchange.com](https://stats.stackexchange.com)
- [medium.com](https://medium.com)
- [towardsdatascience.com](https://towardsdatascience.com)
- [kaggle.com](https://kaggle.com)
- [scikit-learn.org](https://scikit-learn.org)
- [kdnuggets.com](https://kdnuggets.com)
- [machinelearningmastery.com](https://machinelearningmastery.com)
- [analyticsvidhya.com](https://analyticsvidhya.com)
- [realpython.com](https://realpython.com)
- et plus...

---

## Annexes

- A. Wireframing
- B. Diagramme de GANTT
- C. Snapshots mensuels Trello
- D. Extrait de code de l'ETL
- E. MPD de la database analytique
- F. Visuels de l'application finale

## Eléments du wireframing



**Logo Nantes**

Opti Cantines BI

Fréquentation    Menus    Temporalité    Taux de présence    Se déconnecter

Tableau de suivi de l'activité des cantines nantaise

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Filtrer par cantine ▾    Date de début    Date de fin    Submit

Two line charts and two bar charts. The top row shows a line chart with a light blue line and a bar chart with blue and grey bars. The bottom row shows another line chart with a light blue line and a horizontal bar chart with three blue bars of increasing length.

Footer : logo + copyright

**Logo Nantes**

Opti Cantines BI

Prédiction    Prédiction par lots    Se déconnecter

Outil de prédition de la fréquentation

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Règles de gestion

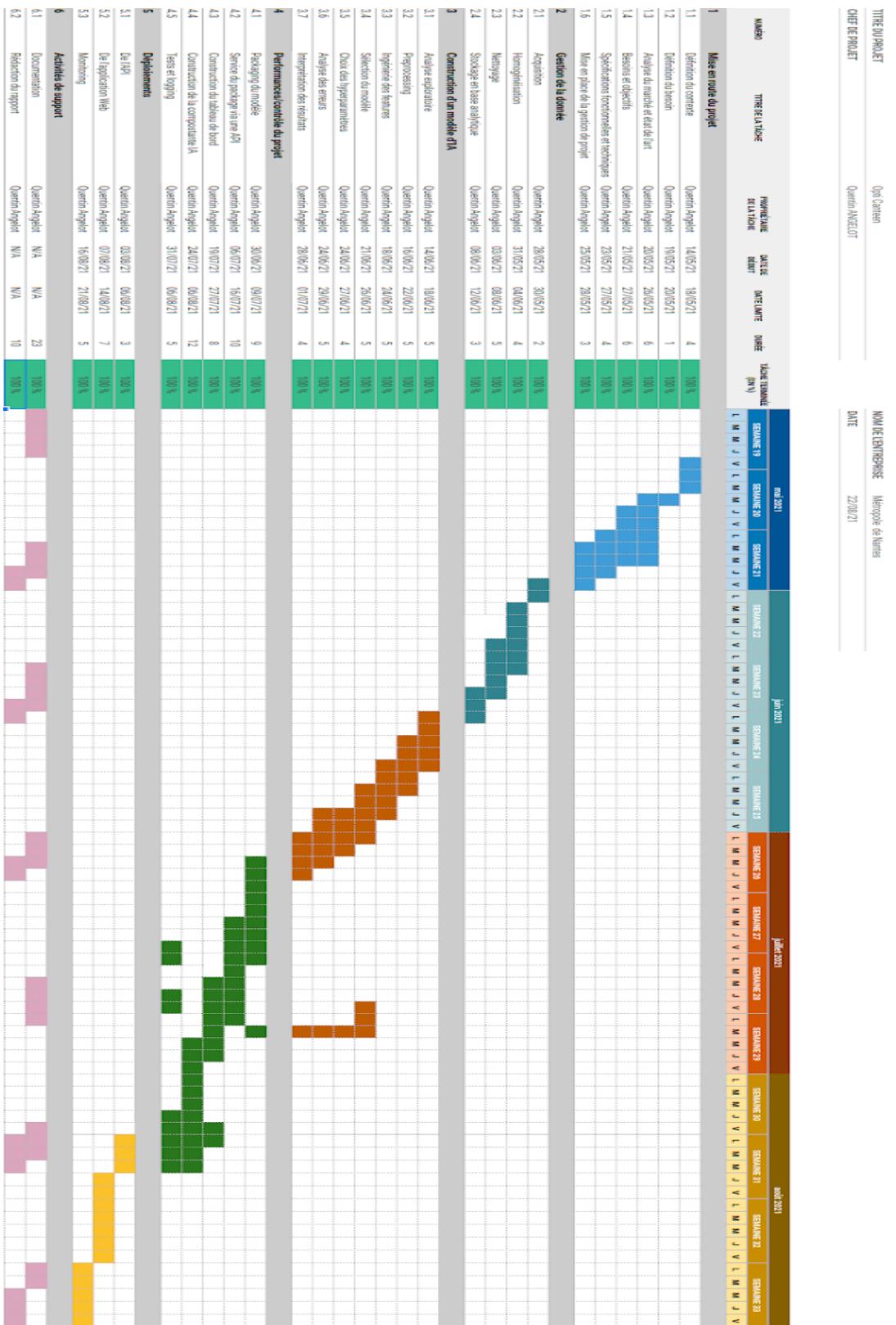
Réaliser une prédition

Input1    Input2  
Input3    Input4  
Input5    Input6  
Input7    Input8  
Input9    Input10  
Input11    Input12

Predict

Footer : logo + copyright

# DIAGRAMME DE GANTT



## Exemples de cartes Trello

**Recherche et documentation**

Dans la liste Done Software

**ÉTIQUETTES**

 +

**Description** Modifier

Autoformation, lecture de blogs, livres, cours en ligne afin de mieux apprendre les différentes étapes du développement applicatif dans le contexte d'un projet de machine learning

**Checklist**

22%

- comment bien organiser le contrôle de versions
- comment structurer le code d'un projet de ML
- comment tirer parti de l'open source afin de gagner en efficacité
- comment packager un modèle de ML
- comment implémenter les tests dans un projet de ML
- comment construire une API avec FastAPI
- comment déployer un modèle de ML (Heroku, docker...)
- comment mettre en place le monitoring du modèle une fois en production
- comment créer et structurer une web app avec Flask

**Masquer les tâches cochées** **Supprimer**

**Activité**

Q Écrivez un commentaire...

Afficher les détails

**Compréhension des données**

Dans la liste Done Data science

**ÉTIQUETTES**

 +

**Description** Modifier

Décrire les données, en explorer les propriétés statistiques et en vérifier la qualité

**EDА approfondie**

0%

- Analyse de Forme : head, shape, qualitatives versus quantitatives, analyse des valeurs manquantes...
- Visualisation de la cible et de sa distribution
- Visualisation des variables continues
- Visualisation des variables qualitatives
- Relation entre la cible et les variables (plots, cross tables...)
- Relation entre les variables (pairplot, clustermap, Implot, correlation heatmap, cross tables...)
- Tests statistiques pour valider la solidité des découvertes visuelles

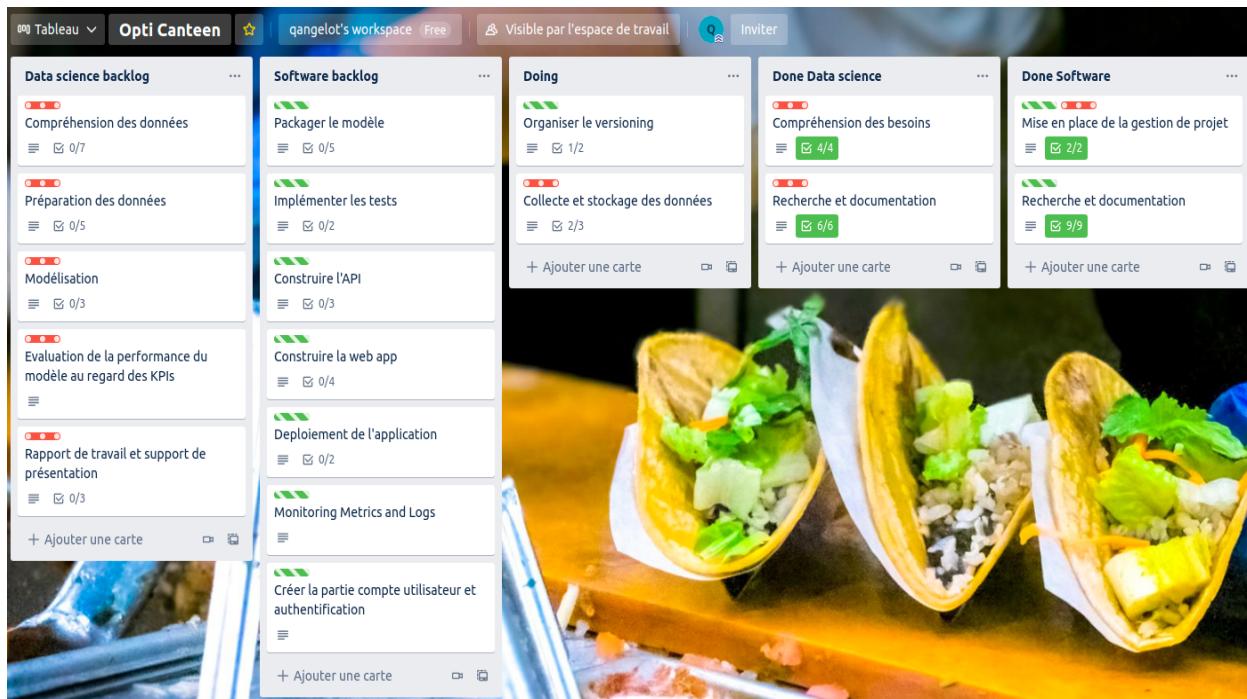
**Supprimer**

**Activité**

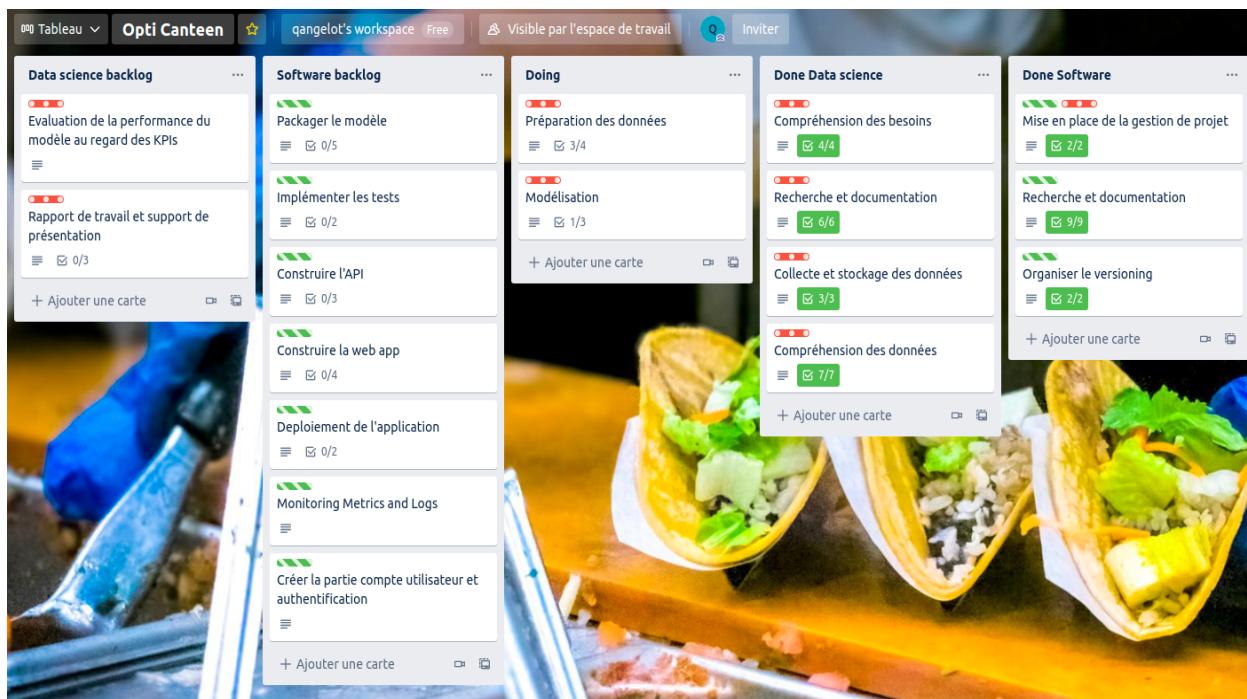
Q Écrivez un commentaire...

Afficher les détails

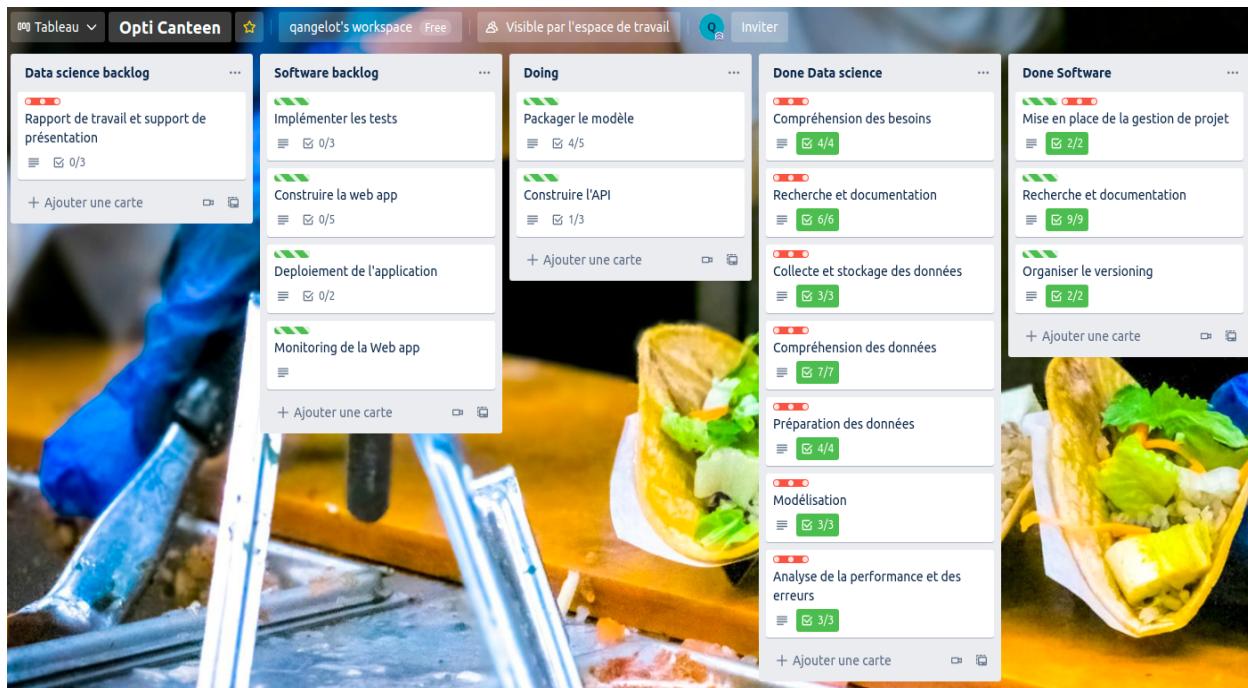
## Snapshot Trello mi Mai (début de projet)



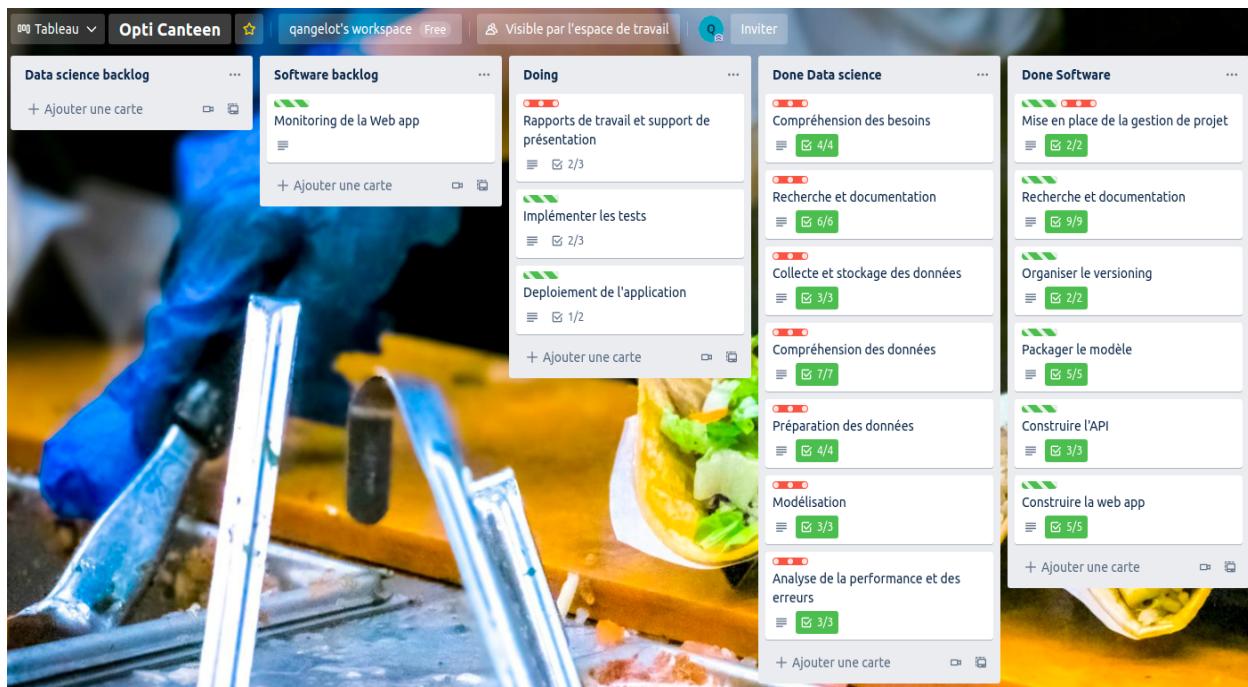
## Snapshot Trello mi Juin (partie Data science)



## Snapshot Trello mi Juillet (partie Développement)



## Snapshot Trello mi Août (partie Développement suite)



---

## Extraits d'une partie du code de l'ETL (de extract\_transform.py et load.py, respectivement)

```
# read data from main CSVs
effectifEcoles = pd.read_csv('../data/effectifs_ecolesnantes.csv', header=0, sep=';')
effectifEcoles.drop(['Début année scolaire'], axis=1, inplace=True)

appariement = pd.read_csv('../data/appariement_ecoles_cantines.csv', header=0, sep=',')
appariement.rename(columns={"ecole": "Ecole"}, inplace=True)

freqJ = pd.read_csv('../data/frequentation_cantines_v2.csv', header=0, sep=',', low_memory=False)
freqJ["date"] = pd.to_datetime(freqJ["date"])

# drop useless duplicate columns
freqJ.drop(['site_nom_sal', 'site_id', 'prevision_s', 'reel_s'], axis=1, inplace=True)
freqJ.rename(columns={'site_nom': 'cantine_nom'}, inplace=True)
freqJ.sort_values(by='date', inplace=True, ascending=True)
freqJ.reset_index(inplace=True, drop=True)

# get aggregated headcounts by canteen
effectifEcoles = pd.merge(effectifEcoles, appariement[['cantine_nom', 'Ecole']], on='Ecole')
effectifEcoles.rename(columns={'Année scolaire': 'annee_scolaire'}, inplace=True)
effectifCantines = effectifEcoles.groupby(['annee_scolaire', 'cantine_nom'], as_index=False).sum()

# join main df with the school year
freqJ = get_school_year(freqJ, 'date', '../data/annees_scolaires.csv')

# join main df with canteen headcounts
freqJ = pd.merge(freqJ, effectifCantines[['annee_scolaire', 'cantine_nom', 'Effectif']], on=['annee_scolaire', 'cantine_nom'])

## adding variable of interest based on insights from canteen employees
# often parents often withdraw their children a few days before the holidays or do not return until a few days later
freqJ = get_distance_holidays(freqJ, 'date', '../data/vacances.csv')

# same logic goes for public holidays
freqJ = get_distance_public(freqJ, 'date', '../data/jours_feries.csv')
```

```
# Create fact table (setting up the FKs and reference to dimensions)
cursor.execute('''CREATE TABLE IF NOT EXISTS `Frequentation_quotidienne` (
  `jour_site_id` INTEGER PRIMARY KEY AUTOINCREMENT,
  `date` DATE NOT NULL,
  `reel` INT NULL,
  `prevision` INT NULL,
  `jour_id` INT NOT NULL,
  `site_id` INT NOT NULL,
  CONSTRAINT `jour_id`
    FOREIGN KEY (`jour_id`)
    REFERENCES `Dim_temporelle` (`jour_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `site_id`
    FOREIGN KEY (`site_id`)
    REFERENCES `Dim_site` (`site_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Frequentation_quotidienne_Dim_menu1`
    FOREIGN KEY (`jour_id`)
    REFERENCES `Dim_menu` (`jour_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Frequentation_quotidienne_Dim_evenement1`
    FOREIGN KEY (`jour_id`)
    REFERENCES `Dim_evenement` (`jour_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION);
  ''')

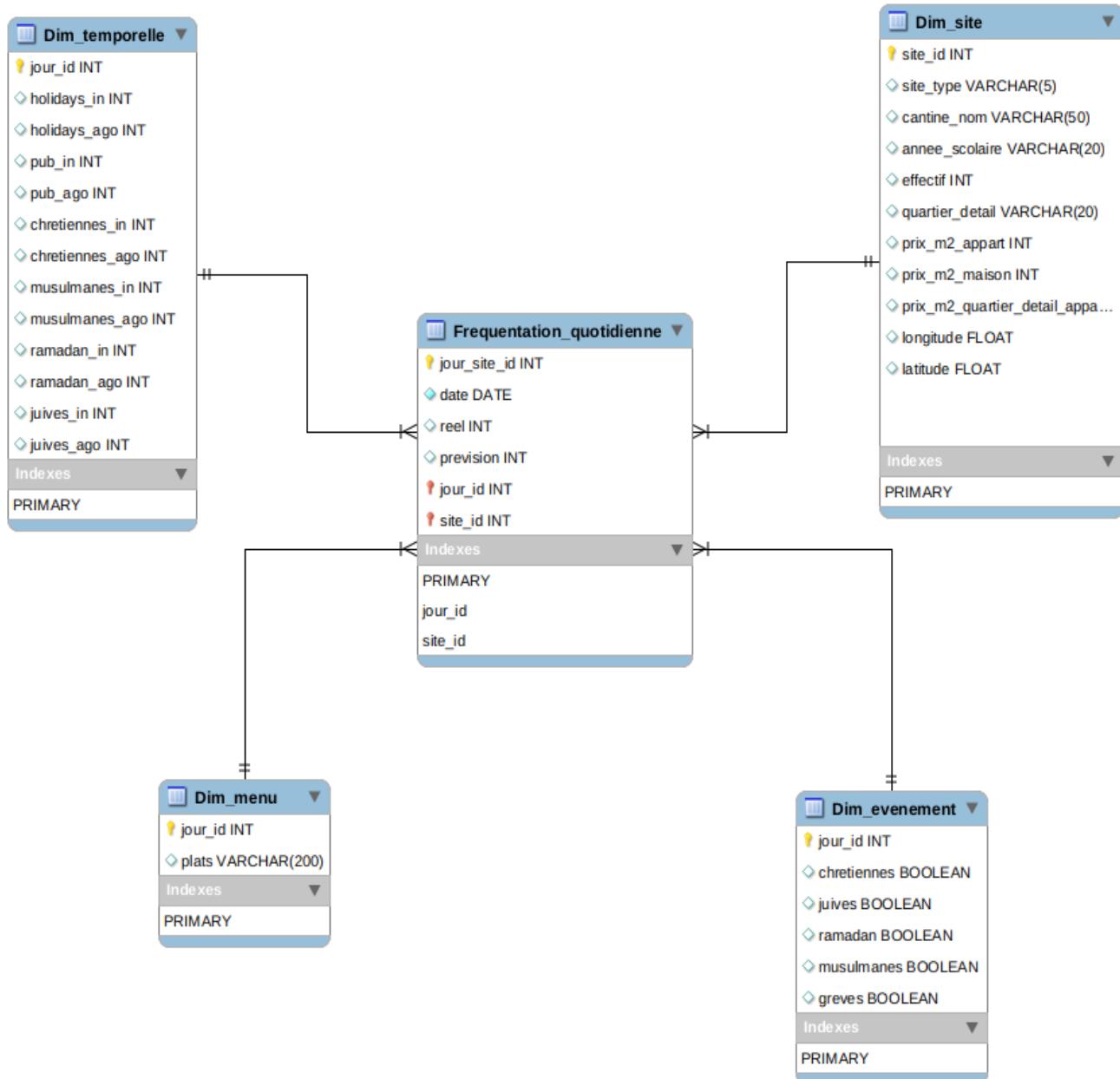
# create index on date for faster retrieval
cursor.execute(''' CREATE INDEX date
ON Frequentation_quotidienne(date); ''')
conn.commit()
```

Ci-dessus, le code permettant la lecture et la jointure des différents CSV grâce à l'appel de fonctions helpers.

Ci-contre, le code SQL permettant la création de la table de faits détaillée du datawarehouse, avec la création des clés étrangères permettant la connexion avec les différentes dimensions. A noter également, la création d'un Index sur la date afin d'améliorer les performances de requête du datawarehouse.

## MPD du datawarehouse

Construit avec le logiciel MySQL Workbench



## Visuels de l'application finale

### Page de connexion

Opti Canteen

Canteen BI Canteen Predict Contact

Opti Canteen permet un suivi à 360 degrés de votre activité.

Bienvenue sur votre nouvel outil de pilotage et d'intelligence artificielle.  
En se basant sur l'historique de données à disposition et les retours des équipes sur le terrain, cet outil doit permettre de résoudre les problématiques de gaspillage alimentaire et d'approvisionnement auxquelles vous êtes confronté au quotidien. Pour se faire, Opti Canteen mêle analyses descriptive, prédictive et prescriptive.

N'hésitez pas à nous rendre dans la partie Contact afin de nous faire part de vos idées!

**Se connecter**

Email :

Mot de passe :

[Créer un compte : S'inscrire.](#)

  
Copyright © 2021 Opti Canteen

### Page d'accueil

Bonjour Angelot Quentin

Canteen BI Canteen Predict Contact Se déconnecter

N'hésitez pas à nous rendre dans la partie Contact afin de nous faire part de vos idées!



Opti Canteen BI

## Page Fréquentation de la sous-partie Tableau de bord

Opti Cantines BI

Fréquentation Menus Temporalité Taux de présence Se déconnecter

### Tableau de suivi de l'activité des cantines nantaise

Analyse des facteurs de la fréquentation des cantines de la métropole Nantaise au regard des prévisions réalisées par les agents depuis 2010.

Filtrer par cantine ▾ Date de début DD/MM/YYYY Date de fin DD/MM/YYYY Submit

#### Données de fréquentation de la cantine AGENETS

##### Fréquentation réelle et prévisions

#### Décomposition de la série temporelle globale

Fréquentation globale réelle

Tendance globale

Saisonalité globale

Résidus globaux

Opti Cantines BI

## Page de Prédition unitaire de la sous partie IA

Opti Cantines Predict      Prédiction    Prédiction par lots    Se déconnecter

### Outil de prédition de la fréquentation

Prédiction de la fréquentation des cantines de la métropole Nantaise à l'aide d'un algorithme d'IA, de l'expérience des équipes métier et d'un historique de données de 10 ans.  
Cet algorithme n'a pas pour but de se substituer à votre expérience métier mais au contraire de vous faciliter la prise de décision au quotidien.

De plus, il se contente de réaliser des prédictions précises sur les périodes dites normales (hors mouvements sociaux ou autres imprévus). Ainsi, lorsque ces événements surviennent, votre expérience est irremplaçable.

### Réaliser une prédition

|                                       |                                 |
|---------------------------------------|---------------------------------|
| Date                                  | 200                             |
| DD/MM/YYYY                            |                                 |
| Cantine                               | Année scolaire                  |
| MAISDON PAJOT                         | 2018-2019                       |
| Effectif                              | Quartier                        |
| 200                                   | Zola                            |
| Prix/m <sup>2</sup> quartier détaillé | Prix/m <sup>2</sup> du quartier |
| 3500                                  | 3550                            |
| Prix/m <sup>2</sup> des maisons       | Longitude                       |
| 4500                                  | 1.5848                          |
| Latitude                              | Depuis les vacances             |

## Page de Prédictions par lots

Opti Cantines Predict      Prédiction    Prédiction par lots    Se déconnecter

### Outil de prédition de la fréquentation

Prédiction de la fréquentation des cantines de la métropole Nantaise à l'aide d'un algorithme d'IA, de l'expérience des équipes métier et d'un historique de données de 10 ans.  
Cet algorithme n'a pas pour but de se substituer à votre expérience métier mais au contraire de vous faciliter la prise de décision au quotidien.

De plus, il se contente de réaliser des prédictions précises sur les périodes dites normales (hors mouvements sociaux ou autres imprévus). Ainsi, lorsque ces événements surviennent, votre expérience est irremplaçable.

### Réaliser plusieurs préditions

Insérez votre fichier au format CSV :

inputs\_test.csv

  
Copyright © 2021 Opti Canteen