UNIVERSITÉ DE PARIS
PARIS DESCARTES

Master 2 : Ingénierie Mathématiques et Biostatistiques

Traitement et analyse de données de consommation énergétique de bâtiments résidentiels et tertiaires issues de réseaux de capteurs

---

James Kelson LOUIS

Entreprise : École Supérieure d'Ingénieur en Électrotechnique et Électronique (ESIEE)

Laboratoire : Laboratoire d'électronique, systèmes de communication et microsystèmes (ESYCOM)

Adresse : (2 boulevard Blaise Pascal Cité Descartes BP 99 93162 Noisy-le-Grand Cedex)

Encadré par :

M. Elyes NEFZAOUI	Tuteur de stage	elyes.nefzaoui@esiee.fr
M. Vittorio PERDUCA	Enseignant référent	vittorio.perduca@parisdescartes.fr

Table des matières

1 Remerciements

La réalisation de ce rapport de stage a été possible grâce au concours de plusieurs personnes à qui je voudrais témoigner toute ma gratitude.

Je souhaite tout d'abord remercier toute l'équipe pédagogique de l'université de Paris et les intervenants professionnels responsables de ma formation, pour l'enseignement de qualité dispensé par le Master Ingénierie Mathématiques et Biostatistiques et pour avoir contribué à la réussite de mes études universitaires.

Je tiens à remercier vivement mon tuteur de stage, Mr Elyes NEFZAOUI, pour sa confiance placée en moi, son accueil, sa gentillesse, le temps passé ensemble et le partage de son expertise. Son exigence m'a grandement stimulé.

Je souhaite également remercier Mathieu BOURDEAU et Amine BOUZIDI qui m'ont encadré et soutenu durant ces six mois. Grâce à leurs expériences professionnelles et leurs expertises, j'ai pu découvrir la diversité du métier data analyst.

Je voudrais exprimer ma reconnaissance envers les amis et collègues qui m'ont apporté leur soutien moral et intellectuel tout au long de ma démarche.

Je tiens à remercier très sincèrement tous les membres de ma famille, spécialement ma femme Angeline Pierre LOUIS et mon fils Jason Kelvin LOUIS pour leur soutien et leurs encouragements constants durant ces deux années.

Je n'oublie bien évidemment pas la ville de Paris qui, par sa position géographique, son histoire, m'a permis d'élargir ma culture. En plus, grâce à mon séjour en France, j'arrive à explorer d'autres grandes villes européennes. Cette expérience a été pour moi très enrichissante.

2 Introduction

De nos jours, l'analyse de données est considérée comme étant une discipline essentielle et incontournable pour les entreprises, tant dans le public que dans le privé. Elle permet entres autres de prendre des décisions calculées grâce au traitement et à l'interpétation de données. Dans un article publié le 10-11-2020 sur le site futura-sciences, il est écrit : *IBM estime ainsi à 2,5 quintillions de bytes le nombre de données générées chaque jour dans le monde. Le cabinet IDC évoque quant à lui 1,7 Mo de données produites par personne à chaque seconde dans le monde* [1].[2]

Tenant compte du besoin et de l'importance de la discipline, plusieurs formations ont été créées pour former des cadres dans ce domaine. Cette formation en Ingénierie Mathématiques et Biostatistiques dispensée à l'Université de Paris en est un exemple.

Dans le cadre d'un projet de recherche financé par l'I-SITE FUTURE impliquant plusieurs laboratoires et établissement de l'est francilien sur la consommation et les usages énergétiques dans les bâtiments résidentiels, un réseau de capteurs comportant plus de 250 capteurs est utilisé pour mesurer différentes grandeurs énergétiques et liées aux usages des occupants à un pas de temps fin allant de quelques secondes à une heure sur une durée de plusieurs années. Le réseau de capteurs produit une importante quantité de données qu'il convient de collecter, nettoyer, stocker et préparer à différents usages allant de la simple visualisation à l'intégration dans des outils de simulation énergétique. L'objectif de mon stage était de nettoyer, stocker et visualiser les données collectées en utilisant le langage Python.

Pour valider mon Master, j'avais souhaité réaliser mon stage de fin d'étude dans le domaine de la data science. Lorsque j'ai vu l'offre de stage, cela a tout de suite attiré mon attention car j'avais particulièrement une appréciation pour tout ce qui est : l'analyse de données, la programmation en Python et l'apprentissage automatique. Ainsi j'ai décidé de postuler et heureusement ma candidature a été retenue.

Ce stage a été réalisé au sein du laboratoire ESYCOM pendant une période de six mois. Les principales missions sont : prétraitement des données, visualisation, analyse de donnée approfondie, application de visualisation de données. Durant ce stage j'ai été encadré par Elyes NEFZAOUI, Mathieu BOURDEAU et Amine BOUZIDI.

Dans ce rapport, nous allons dans un premier temps décrire l'entreprise et le laboratoire d'accueil, puis nous présenterons mes missions lors de ce stage avant de dresser un bilan de celui-ci.

1. https ://www.futura-sciences.com/sciences/questions-reponses/ecoles-formations-formations-devenir-data-analyst-14573/

3 Présentation de l'entreprise d'accueil et du laboratoire

ESIEE : École Supérieure d'Ingénieurs en Électrotechnique et Électronique, membre fondateur de l'Université Gustave Eiffel créée au 1er janvier 2020. C'est une école de la chambre de commerce et d'industrie de région paris Île-de-France placée sous la tutelle du ministère de l'économie et des finances et, depuis janvier 2021, du ministère de l'enseignement supérieur et de la recherche dans le cadre de l'Université Gustave Eiffel. Elle forme des ingénieurs dans les domaines du numérique tels que : internet des objets, systèmes embarqués, cybersécurité, intelligence artificielle mais également dans les domaines de l'usine du futur, des biotechnologies, de la e-santé et des énergies nouvelles. les travaux de recherches des enseignants chercheurs s'associent à trois laboratoires :

1. **ESYCOM :** Laboratoire, Électronique, Systèmes de Communications et Microsystèmes - UMR CNRS 9007
2. **LIGM :** Laboratoire d'Informatique Gaspard Monge - UMR CNRS 8049
3. **LISIS :** Laboratoire Interdisciplinaire Sciences, Innovations, Sociétés - UMR CNRS 9003

Le laboratoire **ESYCOM** s'inscrit dans les domaines suivants :
— Les systèmes de communication
— Les micro-capteurs
— Les micro-sources d'énergie

Les enseignants-chercheurs qui forment cette unité sont issus des établissements suivants : l'Université de Paris-Est Marne-la-Vallée (UPEM), l'école d'ingénieur ESIEE-Paris et le Conservatoire National des Arts et Métiers (le CNAM).

4 Présentation du stage

4.1 Sujet

Traitement et analyse de données de consommation énergétique de bâtiments résidentiels et tertiaires issues de réseaux de capteurs.

4.2 Contexte du stage

Le sujet proposé s'inscrit dans le cadre d'un projet de recherche de l'I-SITE FUTURE (Université Gustave Eiffel) et mené au sein de ESIEE Paris et du laboratoire ESYCOM sur l'efficacité énergétique des bâtiments. Ce projet de recherche vise l'amélioration des modèles de simulation et prédiction de consommations énergétiques des bâtiments résidentiels et tertiaires, notamment en contexte de rénovation énergétique, par l'assimilation dans les modèles de données expérimentales provenant de réseaux de capteurs déployés sur les sites d'intérêt. Les données sont collectées sur des bâtiments existants et occupés par un réseau de plus de 250 capteurs déployés sur place.

4.3 Missions de stage

Le stage s'intéresse au pré-traitement et à l'analyse des données collectées par le réseau de capteurs. À l'heure actuelle, différentes solutions de remontée de données sont utilisées. Les données sont collectées sous différents formats et avec différents niveaux d'agrégation. Une première mission consiste donc à proposer et mettre en place une solution pour harmoniser l'accessibilité de l'ensemble des données collectées en vue d'une utilisation optimale pour leur analyse.

Une seconde mission était d'analyser les données après le pré-traitement à partir de la méthode ci-dessous :

- Faire l'inventaire des données collectées (par type de données, type de capteurs, par bâtiment, etc.) ;
- Évaluer la qualité des données collectées (valeurs aberrantes, données manquantes, etc.) ;
- Description des données disponibles en vue de dégager des tendances et relations entre les différents paramètres physiques ;
- Visualisation des données collectées ;
- Mise en place d'une solution de visualisation adaptable à différents types de données collectées.

4.4 Méthodes de travail

Dans tous les secteurs d'activités, le covid-19 a eu de conséquences assez importantes, en particulier sur le stage, notamment sur le temps de présence sur site qui est passé de cinq à trois jours en présentiel. Cependant, on a pu s'adapter à la nouvelle situation pour au final présenter un travail de qualité. Des réunions hebdomadaires étaient organisées une ou deux fois par semaine pour échanger avec mes tuteurs concernant les avancements de mon travail sur les différentes missions. J'ai aussi participé à quelques réunions avec toute l'équipe qui travaille sur le projet.

4.5 Déroulement du stage

Le stage s'est effectué en deux parties :
La première consistait à faire du prétraitement et de l'analyse de données. C'est une partie très importante, car elle m'a surtout aidé à me familiariser avec les données et à comprendre l'objectif du projet.
La seconde partie était dédiée au développement d'une application de visualisation de données.
Lors de ce stage j'ai bénéficié d'un véritable encadrement pédagogique de la part de mes tuteurs, ce qui a favorisé un climat agréable à la réalisation de mon travail. Mathieu BOURDEAU était là pour tout ce qui qui se rapporte à la Physique et l'analyse de données, car c'est lui qui avait installé tous les capteurs, il avait la responsibilité de faire les mises à jour des données et du tableau de référencement des capteurs. Mis à part les réunions hebdomadaires

avec les trois encadreurs, on avait toujours des échanges sur l'état d'avancement de mon travail par rapport aux missions, il m'assistait durant toute la période du stage. Amine BOUZIDI m'encadrait surtout dans le domaine de la programmation, dans la seconde partie qui consistait au développement de l'application, il m'avait fait aussi une introduction sur la librairie **dash** et sur une application de visualisation qu'il avait dévelopée dans le cadre du projet.

5 Présentation des outils et des données utilisés

5.1 Logiciels et langage(s) de programmation utilisés

Durant ce stage, on a utilisé le langage Python pour écrire les codes, pour le partage des scripts on a utilisé des notebooks jupyter et colab qui permet de partager, de faire des mises à jour d'un script Python en ligne par un ou plusieurs utilisateurs. Pour les tableaux de données on les partage sur des fichiers excels.

5.2 Description des bibliothèques utilisées

i) **Numpy**
Version :1.19.2
C'est une bibliothèque libre et open source, elle sert à manipuler des matrices ou tableaux multidimensionnels ainsi que des fonctions Mathématiques opérant sur ces tableaux.

ii) **Pandas**
Version :1.2.3
Pandas est un outil d'analyse et de manipulation de données open source rapide, puissant, flexible et facile à utiliser.

iii) **Math**
C'est une bibliothèque libre et open source, elle permet d'effectuer des taches mathématiques, elle contient un ensemble de fonctions Mathématiques intégrées.

iv) **Matplotlib**
Version :3.3.4
C'est une bibliothèque libre et open source destinée à tracer et visualiser des données sous forme de graphiques.

v) **Seaborn**
Version :0.11.1
C'est une bibliothèque libre et open source destinée à tracer et visualiser des données sous forme de graphiques, elle est basée sur matplotlib.

vi) **Datetime**
C'est une bibliothèque libre et open source destinée à manipuler les dates et les heures.

vii) **Plotly**
Version :4.14.3
C'est une bibliothèque libre et open source qui permet de créer des gra-

phiques interactifs.

5.3 Description des données

i) **Lot 1**
Dans le cadre de ce projet, on a installé des capteurs à des endroits bien spécifiques dans trois bâtiments. Ces capteurs étaient destinés à recueillir des informations telles que la température d'un endroit ou d'un appareil, l'ouverture des fenêtres etc...

Description des variables

- ID number : Identifiant du capteur
  Chaque capteur est différencié par un code héxadécimal
  ex : $70B3D580A0109505$
- date and time : Date et heure de récupération des données
  Lorsque le capteur enregistre une information, il enregistre en même temps la date et l'heure, cette information est capitale car elle permet de voir l'évolution des données en fonction du temps.
- type of measure : Type de mesures
  Ce jeu de données contient trois types de mesures :
  1) Température
  2) Ouverture de fenêtre
  3) Niveau de la batterie
- values : Valeur de la mesure captée
  Pour les données de température, ce sont des valeurs numériques comprises entre 0 à 100 degrés celcius. Pour les données d'ouverture de fenêtre, ce sont des données binaires :
  1= **Ouvert** et 0=**Fermé**.
  Le niveau de la batterie est un nombre réel qui varie entre 0 à 100 (Pourcentage), elle indique le pourcentage de charge de la batterie à chaque instant.

ii) **Lot 2**

Description des variables

- SYS : Identifiant du capteur
- TS : Date et heure de récupération des données
- metric : Type de mesures (température, puissance électrique, intensité, etc...)
- value : Valeur de la mesure captée

iii) **Tableau de référencement des capteurs**
On dispose d'un tableau Excel qui contient toutes les informations sur les capteurs, telles que : date d'installation des capteurs, date d'enlèvement pour les travaux, identifiant, type de capteurs etc...

Ce tableau est régulièrement mis à jour en fonction des installations, il est utilisé pour les analyses de données ainsi que les visualisations.

Description des variables

- devEUI : Identifiant du capteur
  Chaque capteur contient un identifiant, cet identifiant se trouve à la fois dans les jeux de données ainsi que le tableau de référencement de capteurs.
- Bâtiment :
  Le bâtiment auquel le capteur est installé.
- Etage :
  L'étage auquel le capteur est installé.
- Date installation :
  Date à laquelle le capteur a été installé.
- Date enlèvement travaux :
  Certains capteurs ont été enlevés pour des raisons de travaux, on enregistre les dates d'enlèvement.
- Date réinstallation :
  Pour les capteurs qui ont été enlevés et réinstallés, on enregistre la date de réinstallation de ces capteurs.
- Type de capteurs :
  Chaque capteur fait une mesure différente dépendemment de la grandeur physique observée, ce qui définit le type du capteur.
- Pas de temps acquisition :
  Les capteurs enregistrent les données de façon périodique, cette durée est stockée dans cette variable
- Nombre de mesures :
  Certains capteurs enregistrent plusieurs types de mesures, cette variable donne le nombre de mesures enregistrés par capteur.
- Zone installation :
  Zone d'installation du capteur.

Autre outil

- **Une application de visualisation de données**
  C'est une application de visualisation de données développée par l'un des tuteurs, elle a servi de base pour créer une nouvelle application pour visualiser les données.

6 Prétraitement des données

Le réseau est constitué de plus de 250 capteurs qui captent des informations tous les jours, ces informations sont stockées dans plusieurs fichiers et elles n'ont été soumises à aucun traitement, ce sont des fichiers brutes. En plus, tous les capteurs ont des caractéristiques différentes et ne fonctionnent pas tous de la

même manière. Pour pouvoir exploiter les données il est nécessaire de faire le pré-traitement des données.

6.1 Format

— Les données faisant partie du **Lot 1** arrivent sous forme de fichiers textes auxquels on trouve les données en format geojson et en xml. La figure suivante permet de voir ce qu'il y a à l'intérieur des fichiers textes.

```
--e7289537-C--
<?xml version="1.0" encoding="UTF-8" standalone="yes"?><measure href="http://slbase.sensing-
labs/com.sl.application.server/V1.0/70B3D580A01033D5/SenlabD/digital_state"><desc id="digital_state" name="digital
state" unit="is opened?" is="BOOL"/><values><value timestamp="2021-05-04T22:06:00Z">true</value></values></measure>
--e7289537-E--

--e7289537-Z--

--6a8b8002-A--
[05/May/2021:00:06:17 +0200] YJHFWSgNAROfsFTiHg7pGwAAAFQ 80.65.250.126 9918 192.168.0.10 8123
--6a8b8002-B--
POST / HTTP/1.1
Content-Type: application/json
User-Agent: Java/1.7.0_25
Host: andreproject.ddns.net:8123
Accept: text/html, image/gif, image/jpeg, *; q=.2, */*; q=.2
Connection: keep-alive
Content-Length: 270

--6a8b8002-C--
{"href":"http://slbase.sensing-
labs/com.sl.application.server/V1.0/70B3D580A01064FF/SenlabD/battery_current_level","desc":
{"id":"battery_current_level","name":"Battery current level","unit":"%","is":"UINT_8"},"values":[{"timestamp":"2021-
05-04T22:03:46Z","value":"85"}]}
```

FIGURE 1 – Présentation des fichiers textes

— Les données faisant partie du **Lot 2** étaient déjà prétraitées, elles sont stockées sur un serveur en format csv, sous forme d'un fichier par capteur par jour, elles étaient donc prêtes à utiliser.

6.2 Conversion des archives en csv

Les données collectées du **Lot 1** sont stockées dans des fichiers textes avec beaucoup d'autres informations, pour pouvoir récupérer ces données, on avait besoin de convertir ces archives en des fichiers csv qui faciliteraient un meilleur accès aux données.

Le laboratoire avait à sa disposition un script Python développé par un étudiant stagiaire afin de convertir les fichiers textes en csv, mais le code ne fonctionnait pas. L'une de mes premières tâches c'était de comprendre ses lignes de codes afin de pouvoir apporter les modifications nécessaires pour atteindre notre objectif.
Pour réaliser cette tâche, il a fallu mobiliser des compétences en programmation

Python, plus précisément il fallait avoir la capacité de comprendre un code en Python et la capacité d'écrire un code efficace pour résoudre un problème donné. Les problèmes que j'ai rencontrés pour cette mission étaient au niveau des lignes de codes, certaines fonctions marchaient parfaitement bien et d'autres ne fonctionnaient pas, parfois il y avait des lignes de codes difficile à comprendre car elles n'étaient pas commentées. Pour gérer ces difficultés, j'ai exécuté chaque ligne de code pour voir quels étaient les résultats, détecter les endroits où il y avait des erreurs. En termes de résultats, j'ai réussi à modifier les fonctions, corriger les erreurs pour finalement récupérer les données dans les fichiers textes.
Dans un premier temps, tout marchait très bien, un problème est survenu après plusieurs mois lorsqu'il y a eu un mélange de formats xml et geojson à l'intérieur des fichiers textes, pour cela il fallait modifier le code pour qu'il prenne en compte les deux formats de fichiers. Au final, ce script Python permet de convertir les archives en fichier csv. Pour exploiter ces données, plusieurs solutions sont possibles.
On peut avoir :

- un fichier par jour
- un fichier par capteur
- un fichier par capteur par jour
- un fichier qui contient toutes les données

Les fichiers csv ressemblent au tableau suivant.

ID number	type of measure	date and time	values
70B3D580A0106443	battery	2020-10-13 22:00:53+00:00	91.0000
70B3D580A0106443	window opened/closed	2020-10-13 22:00:53+00:00	1.0000
70B3D580A0103467	battery	2020-10-13 22:01:28+00:00	87.0000
70B3D580A0103467	window opened/closed	2020-10-13 22:01:28+00:00	1.0000
70B3D580A010950E	battery	2020-10-13 22:00:54+00:00	86.0000
...	...	...	...
70B3D580A0109505	temperature	2021-05-24 21:54:53+00:00	24.5625
70B3D580A0109505	temperature	2021-05-24 21:55:53+00:00	24.5625
70B3D580A0109505	temperature	2021-05-24 21:56:53+00:00	24.5625
70B3D580A0109505	temperature	2021-05-24 21:57:53+00:00	24.5625
70B3D580A0109505	temperature	2021-05-24 21:58:53+00:00	24.5625

FIGURE 2 – Représentation des fichiers csv

7 Scatterplot et Heatmap

Objectifs :

i) Visualiser les données collectées.

ii) Controler le fonctionnement des capteurs globalement.

iii) Repérer les endroits et les périodes de temps où les capteurs n'envoient pas de données.

Étant donné qu'on avait à notre disposition une base gigantesque de données, on voulait voir à quoi ressemble les données, de ce fait, on avait affiché les nuages de points et des heatmap pour avoir une idée globale des données collectées.

Pour réaliser cette mission, il fallait maitriser la bibliothèque **Matplotlib** [3] qui est une bibliothèque permettant de faire de la visualisation en Python. Elle dispose de beaucoup de fonctionnalités pour rendre les graphiques lisibles et compréhensibles.
Sur les prochains graphiques, il y a un rectangle en haut à droite qui contient les caractéristiques des capteurs. Il permet de réperer facilement quel capteur à partir de son identifiant, la zone d'installation (bâtiment et étage), le pas de temps d'acquisition des données, le type de mesure effectué, la date d'installation du capteur, les dates éventuelles d'enlèvement pour les travaux et de réinstallation.
Sur les graphiques, on voulait identifier facilement les quatres périodes considérées pour les différents capteurs. De ce fait, il y a une ligne verticale bleu positionnée à la date d'installation des capteurs, une ligne verticale rouge aux dates d'enlèvement et de réinstallation lorsqu'elles existent un rectangle de couleur rouge s'affiche sur la surface limitée par ces deux dates.
Les graphiques suivantes représentent quelques cas de figure des nuages de points des données de température et d'ouverture de fenêtre.

7.1 Scatterplot

Représentation des données sur toute la période d'installation

— Ce capteur a été installé le 23-09-2020, pour des raisons de travaux il a été enlevé le 20-10-2020 et réinstallé le 13-01-2021.

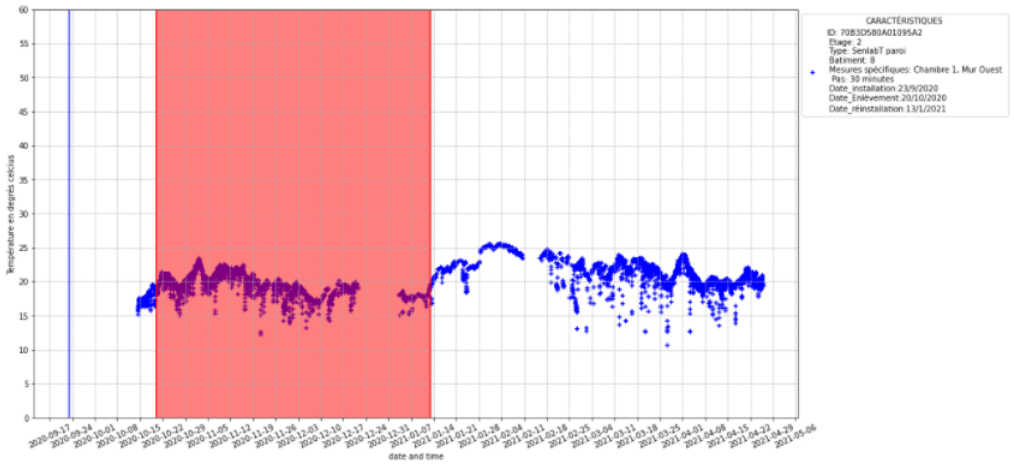

FIGURE 3 – Nuage de points des données de température

— Ce capteur a été installé le 23-09-2020, pour des raisons de travaux il a été enlevé le 23-10-2020, il n'est pas encore réinstallé

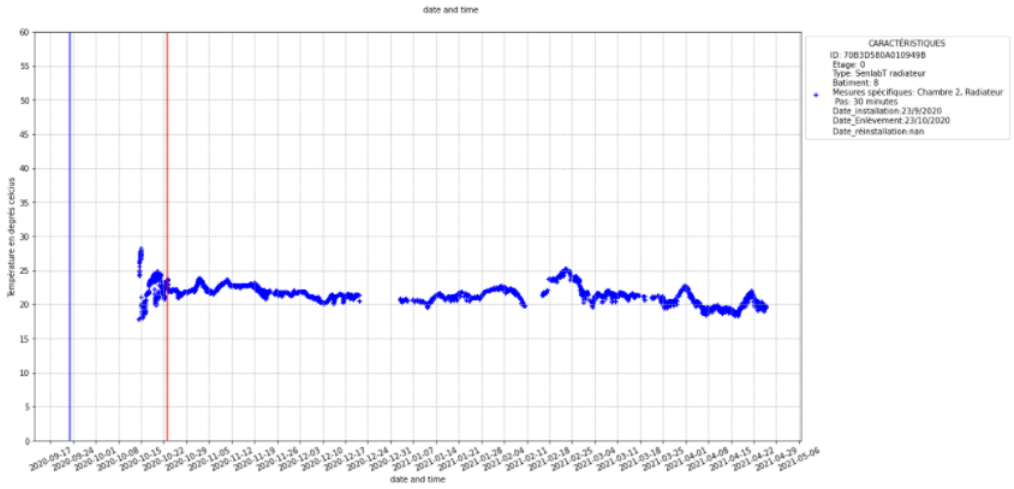

FIGURE 4 – Nuage de points des données de température

— Ce capteur est resté sur place depuis sa date d'installation

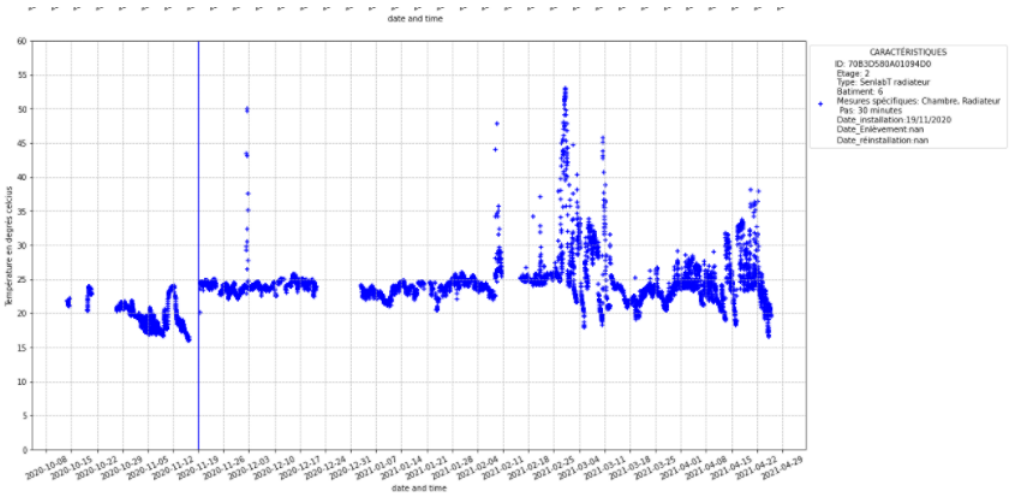

FIGURE 5 – Nuage de points des données de température

— Ce capteur est resté sur place depuis sa date d'installation, il capte des données d'ouverture de fenêtre.

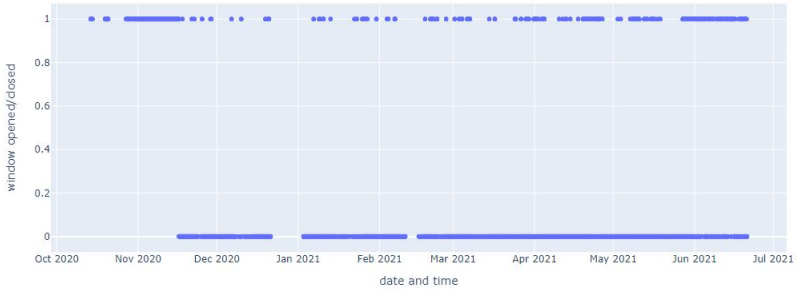

FIGURE 6 – Nuage de points des données d'ouverture de fenêtre

7.2 Heatmap

Capteurs à pas de temps d'acquisition d'une minute

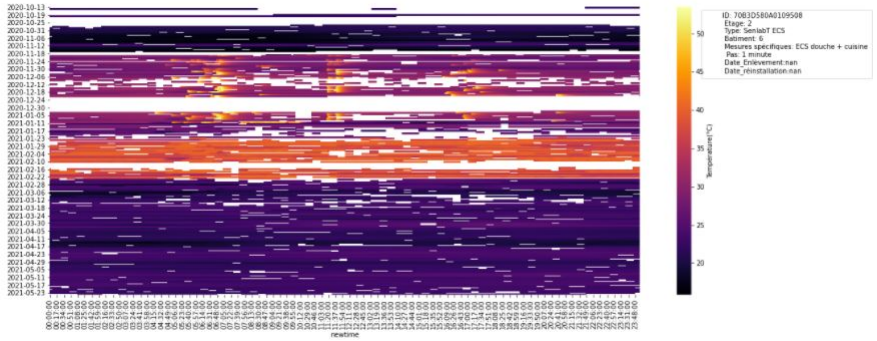

FIGURE 7 – Heatmap

Capteurs à pas de temps d'acquisition de 30 minutes

Pour les capteurs à pas de temps d'acquisition de 30 minutes, on a recalé toutes les mesures sur le premier pas qui apparait et on a utilisé un intervalle de 30 minutes.

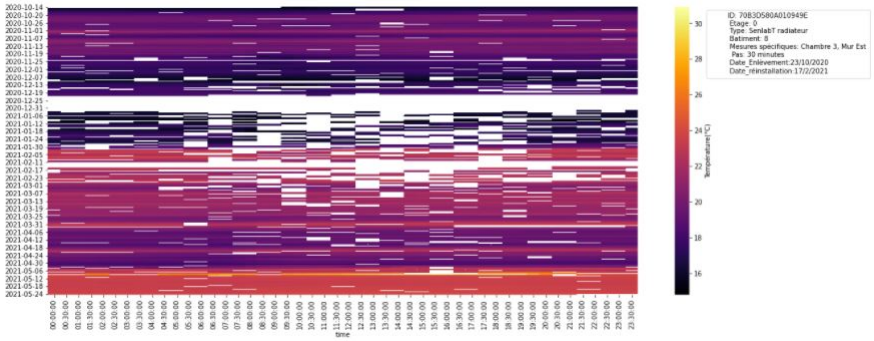

FIGURE 8 – Heatmap

7.3 Statistiques de données manquantes

Objectifs :

i) Évaluer la qualité des données collectées.

ii) Controler le fonctionnement des capteurs suivant leurs types.

iii) Controler le fonctionnement des capteurs par appartement.

Les capteurs installés n'ont pas le même pas de temps d'acquisition, ils ne sont pas installés au même endroit et ils ne sont pas de même type. On voulait savoir s'il y a des capteurs qui perdent des données, et si c'est le cas, on voulait savoir pourquoi. Pour comprendre le fonctionnement du réseau, on a fait une statistique de données manquantes.

Pour ce faire, on a essayé de comparer le nombre de données reçus et le nombre attendu. On rappelle que les capteurs ont un pas de temps d'acquisition qui varie entre 1 minute et 60 minutes. Pour effectuer ce calcul, il faudra faire des considérations par rapport aux types de données.

i) **Données de température**
   Pour un capteur à pas de temps d'une minute, on attend à avoir 1440 points de données par jour. Pour un capteur à pas de temps de 30 minutes, on espère avoir 48 points de données par jour.

ii) **Données d'ouverture de fenêtre**
   Normalement on attend à avoir un point de données par heure, soit 24 par jour.

Dans tous les cas, si le nombre attendu est inférieur au nombre reçu on accepte que le capteur envoie 100 % de données. Le taux de données manquantes est calculé en fonction du nombre de jours dans la période considérée, le nombre de données reçues et le pas de temps d'acquisition. Les difficultés rencontrées pour cette mission sont liées aux caractéristiques des capteurs. Tous les capteurs n'ont pas été installés à la même date, certains ont été enlevés pour des travaux, parmi les capteurs enlevés certains ont été réinstallés. Pour gérer ces difficultés, un script Python a été écrit afin de récupérer pour chaque capteur le nombre de jours et le nombre de données.

La figure suivante représente les quatres périodes considérées pour les analyses de données.

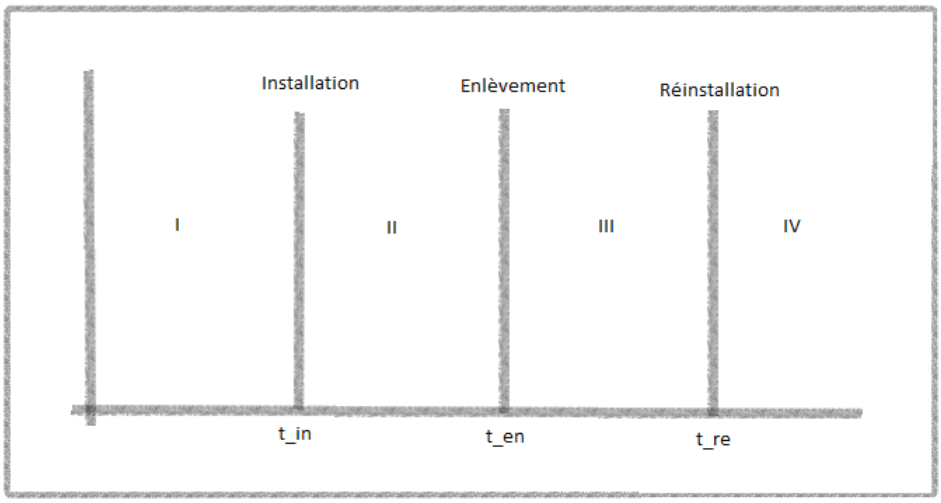

FIGURE 9 – Subdivision des périodes

— I : Avant la date d'installation des capteurs
— II : Entre la date d'installation des capteurs et la date d'enlèvement pour les travaux
— III : Entre la date d'enlèvement pour les travaux et la date de réinstallation
— IV : Après la date de réinstallation

Pour les capteurs qui n'ont pas été enlevés pour des raisons de travaux, la période la plus pertinente est celle qui se trouve après la date de réinstallation, pour les autres capteurs on considère deux périodes pertinentes pour analyser les données, entre la date d'installation et la date d'enlèvement, après la date de réinstallation.

Les deux graphiques suivantes représentent les résultats obtenus pour les statistiques de données manquantes des données du **Lot 1** par type de capteurs et par appartement.

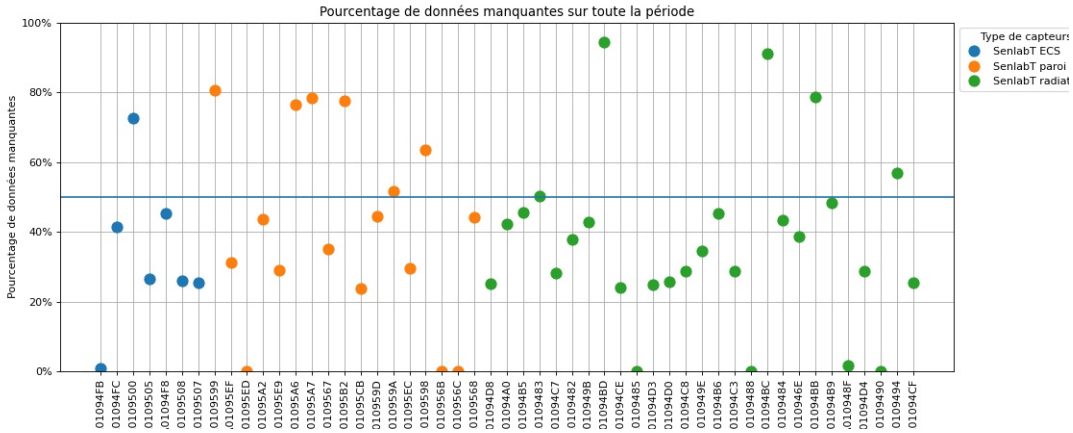

FIGURE 10 – Pourcentage de données manquantes par type de capteurs.

En résumé :

Type de capteurs	**Pourcentage de données manquantes (%)**
SenlabT ECS	40
SenlabT Paroi	51
SenlabT Radiateur	42

Lecture graphique

- Pour les capteurs de type **SenlabT ECS**, 6 sur 7 capteurs, soit 85.71 % ont un taux de données manquantes inférieur à 50%, globalement on a un taux de données manquantes de 40%.
- Pour les capteurs de type **SenlabT Paroi**, 11 sur 17 capteurs, soit 64.71 % des capteurs ont un taux de données manquantes inférieur à 50% globalement on a un taux de données manquantes de 51%.
- Pour les capteurs de type **SenlabT radiateur** 23 sur 27 capteurs soit 85.19 % ont un taux de données manquantes inférieur à 50%, globalement on a un taux de données manquantes de 42%.

Interprétation

On constate que le type de capteur ne permet pas de classer les capteurs, le graphique et le tableau ne permettent pas de savoir quel type de capteur qui fonctionne mieux que d'autres car ils n'ont pas le même pas de temps d'acquisition.

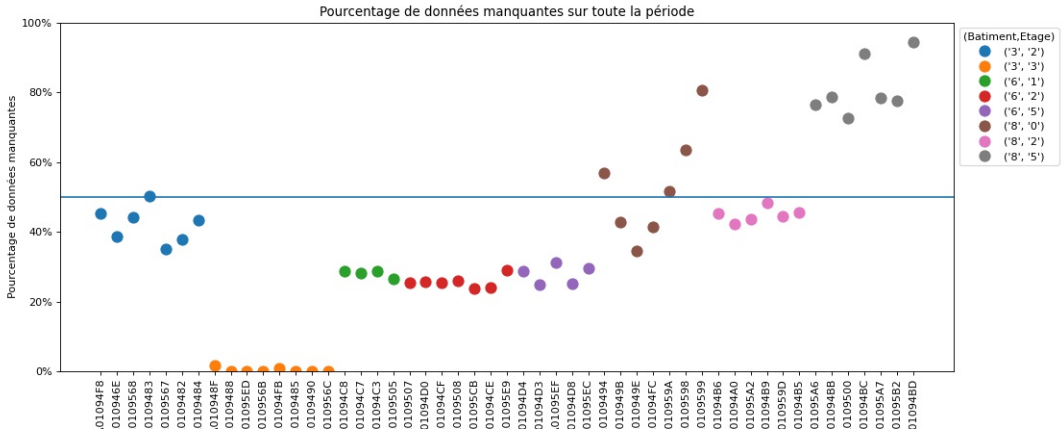

FIGURE 11 – Pourcentage de données manquantes par appartement.

En résumé :

Bâtiment	Etage	Pourcentage de données manquantes (%)
3	2	45
3	3	1
6	1	27
6	2	26
6	5	28
8	0	44
8	2	45
8	5	74

Lecture graphique

- **Bâtiment 3 Etage 2 :** 6 sur 7 capteurs, soit 85.71 % ont un taux de données manquantes inférieur à 50%, globalement on a un taux de données manquantes de 45%.
- **Bâtiment 3 Etage 3 :** Tous les capteurs n'ont quasiment pas de données manquantes, globalement on a un taux de données manquantes de 1%.
- **Bâtiment 6 Etage 1 :** Tous les capteurs ont un taux de données manquantes inférieur à 40%, globalement on a un taux de données manquantes de 27%.
- **Bâtiment 6 Etage 2 :** Tous les capteurs ont un taux de données manquantes inférieur à 40%, globalement on a un taux de données manquantes de 26%.
- **Bâtiment 6 Etage 5 :** Tous les capteurs ont un taux de données manquantes inférieur à 40%, globalement on a un taux de données man-

quantes de 28%.

— **Bâtiment 8 Etage 0 :** 3 sur 7 capteurs, soit 42.86% des capteurs ont un taux de données manquantes inférieur à 50%, globalement on a un taux de données manquantes de 44%.
— **Bâtiment 8 Etage 2 :** Tous les capteurs ont un taux de données manquantes inférieur à 50%, globalement on a un taux de données manquantes de 45%.
— **Bâtiment 8 Etage 5 :** Tous les capteurs ont un taux de données manquantes supérieur à 50%, globalement on a un taux de données manquantes de 74%.

Interprétation

On constate que l'emplacement du capteur est un bon indicateur permettant de classer les capteurs. On peut voir sur les graphiques que les capteurs se trouvant au bâtiment 3 à l'étage 3 fonctionnent mieux que tous les autres. Globalement on constate que certains capteurs perdent des données, on pense que les causes qui expliquent cette perte sont : les éléments métalliques (portes, échafaudages), le nombre de parois que les signaux radio doivent traverser, l'éloignement par rapport à la gateway.

7.4 Lien entre la taille des fichiers et le pourcentage de données reçues

Chaque jour, nous reçevons une alerte e-mail nous informant sur la taille des fichiers reçus, on voulait voir s'il y avait une relation entre le taux de données reçues et la taille des fichiers reçus par jour. En ce sens on a décidé de tracer sur un même graphique les courbes pour voir s'il y a un lien entre eux.

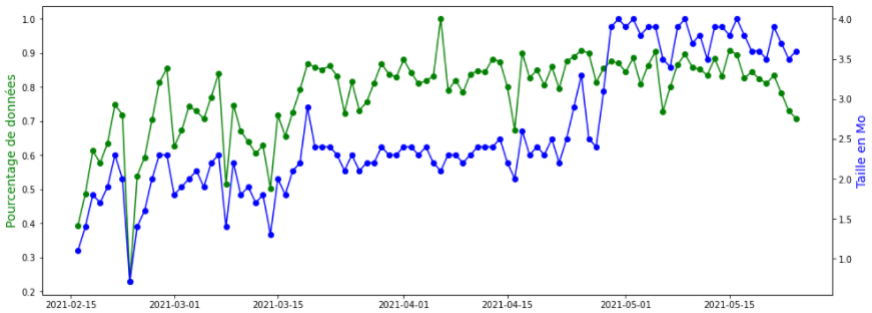

FIGURE 12 – Taille des fichiers / pourcentage de données.

Interprétation

On constate qu'il y a une corrélation entre le taux de données reçues et la taille des fichiers sur toute la période. On a aussi constaté un changement de coefficient de corrélation qui est lié à un changement de format de fichiers en regardant de plus près les fichiers.

8 Développement de l'application de visualisation des données

Dans le cadre de ce projet, on avait promis aux propriétaires des bâtiments s'ils acceptent de fournir leurs données ils auront en retour une application par appartement qui leur permettra de voir leur consommation moyenne d'énergie électrique etc...
On voulait aussi avoir un outil de visualisation qui soit adaptable aux données collectées, c'est pourquoi on a décidé de mettre en place une application de visualisation de données.

8.1 Présentation des librairies utilisées dans l'application

1. **dash :**
   Version 1.20.0
   Lorsqu'on importe cette librairie, elle apporte en même temps plusieurs bibliothèques telles que :
   - **dash_core_components :** Cette bibliothèque renferme un ensemble de composants qui facilite les interactions entre l'utilisateur et l'interface.
   - **dash_html_components :** C'est une bibliothèque qui permet de faire des manipulations sur des composants HTML.
   - **dash_table :** C'est une bibliothèque qui permet de faire des manipulations sur des tableaux de données.
   - **plotly :** Bibliothèques de graphiques interactifs Python
2. **dash_bootstrap_components :** C'est une bibliothèque qui renferme des composants bootstrap, on l'utilise pour la création du design sur les sites web, également pour rendre un site responsive, c'est-à-dire qui peut être adapté à tous les écrans.
3. **pandas**
4. **numpy**
5. **flask :**
   Version 1.1.2
   C'est un framework de développement web en python.
6. **datetime :** C'est une bibliothèque permettant de manipuler des dates.

8.2 Environnement virtuel

Pour éviter qu'il y ait des problèmes de dépendences entre les paquets, Python donne la possibilité de travailler sur des environnements virtuels.
Lorsqu'on crée un environnement virtuel, ça crée un dossier qui contient tous les exécutables nécessaires pour utiliser les paquets avec les versions qu'un projet Python pourrait nécessiter.

8.3 Présentation de la librairie dash

8.3.1 Dash

C'est une librairie qui permet de créer des applications Web de visualisations interactives, elle est gratuite et open source, elle a été créé en juin 2017 par l'équipe de développement de plotly qui est basé sur Flask, plotly.js et react.js.

8.3.2 Composition d'une application Dash

Une application Dash est composée de deux parties, le "Layout" qui décrit à quoi ressemble le tableau de bord, la deuxième partie "callback" qui contrôle la réactivité de l'application.

8.3.3 Tableau de bord

La partie "layout" peut avoir un ou plusieurs composants, nous allons en présenter ceux qui font partie de l'application.

Quelques composants utilisés dans l'application

Composants	**Rôle**
dcc.Checklist	Cases à cocher
dcc.Dropdown	Listes déroulantes
dcc.DatePickerRange	Sélection de dates
dcc.Markdown	Afficher du texte
dcc.Graph	Afficher des graphiques
dcc.Link	Afficher des liens

Chacun de ces composants contiennnent plusieurs arguments tels que : "id","options","value", etc...
Pour accéder aux arguments d'un composant on exécute le code suivant :
help(Nom de la composante)
Ex :
help(dcc.Dropdown)

8.3.4 Réactivité de l'application

Pour rendre une application Dash réactive, on utilise les fonctions de rappels ("callback"). Ce sont des fonctions Python qui sont automatiquement appelées

chaque fois qu'une propriété d'entrée change. Une fonction de rappel ressemble à :

```
@app.callback(
    Output("component-id-to-output-to", "children"),
    Input("component-id-to-listen-to", "valuesIn"),
    State("component-id-to-listen-to", "valuesIn")
)
def callback_name(arg1,arg2) :
    code for callback to execute
    return output_to_children
```

Définition des mots clés

Output : Toute fonction de rappel doit avoir une sortie, elle peut être une page, un graphique, un composant (Liste déroulante, cases à cocher, etc...). Elle est Caractérisée par un "component_id" qui est l'identifiant du composant et un "component_property" qui est une propriété du composant tel que : "options" si on veut afficher les options d'une liste déroulante ou des cases à cocher etc...,"figure" pour afficher des graphiques, "value" pour afficher les choix sélectionnés.

Input : Définitions des variables d'entrée, on a les mêmes caractéristiques par rapport à **Output**.

State : La seule différence entre **State** et **Input** c'est qu'il n'y a pas de mis à jour automatique sur l'application lorsqu'on modifie une propriété d'entrée.

callback_name :
C'est le nom de la fonction, on peut choisir n'importe quel nom pour définir la fonction.

"children" : C'est la propriété du composant qu'on souhaiterait avoir en sortie.

"valuesIn" : C'est la propriété du composant défini comme variable d'entrée.

"component-id-to-output-to" : L'identifiant du composant à afficher.

"component-id-to-listen-to" : L'identifiant du composant de la variable d'entrée.

arg1,... : Listes des arguments de la fonction, s'il y a plusieurs **Input**, il faut écrire les arguments de la fonction de rappel en ordre par rapport à chacun d'eux.

Le graphique suivant explique le fonctionnement des fichiers.

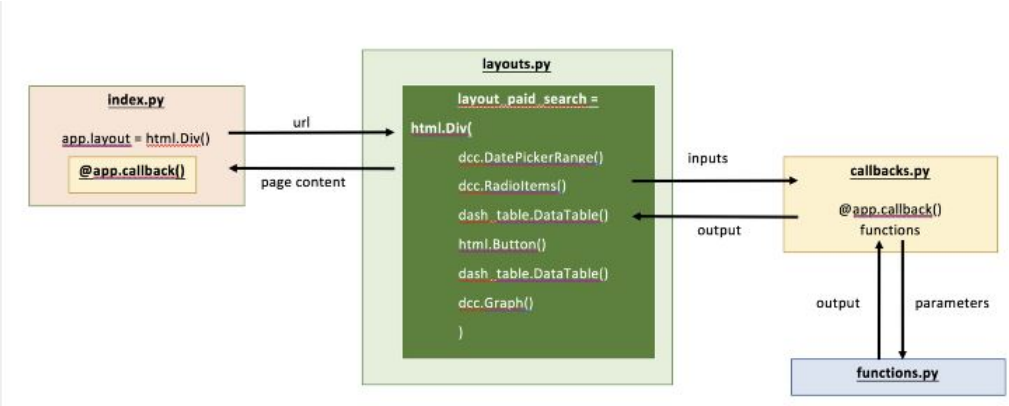

FIGURE 13 – Schéma des fichiers du tableau de bord [1]

8.4 Structure de l'application

Une application Dash peut se présenter de différentes manières :
on peut avoir une application sur une seule page, tout comme elle peut avoir plusieurs pages, c'est le cas pour cette application, elle est stucturée comme suit :
- app.py
- index.py
- datasets
- apps
  __init__.py
  app1.py
  app2.py

Le fichier app.py sert à définir l'instance Dash

```
import dash

app = dash.Dash(__name__, suppress_callback_exceptions=True)
server = app.server
```

FIGURE 14 – Définition de l'instance Dash.[4]

Pour l'exécution de l'application on lance le fichier index.py. Pour cette application on définit les "callback" dans les fichiers app1. py, app2.py sachant que ces derniers ont un accès à l'instance de l'application Dash. Le dossier datasets contient tous les fichiers csv .

8.5 Présentation de l'application

La page d'accueil contient cinq onglets :

1) Energie
2) Usages
3) Environnement intérieur
4) Affichage personnalisée
5) Résumé

Pour les trois premiers onglets, il y a deux listes déroulantes, la première permet de sélectionner "Partie commune" ou "Appartement", la deuxième permet de sélectionner le type de données qui peut être "Chauffage" ou "électricité" ou "gaz".

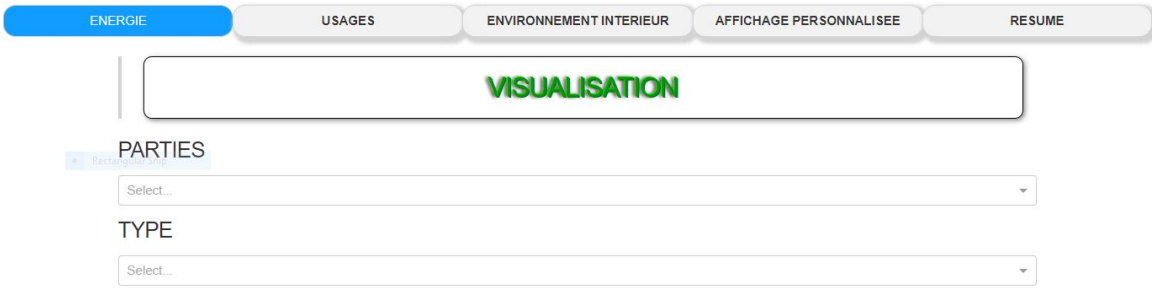

FIGURE 15 – Page d'accueil de l'application de visualisation des données

Une fois qu'on aura sélectionné la partie et le type de données, il y a une page qui s'affiche, sur cette page on pourra afficher deux types de graphiques : la première qui donne une visualisation sur toute la période d'installation des capteurs sous forme d'une courbe ou d'un nuage de points.

Nuage de points

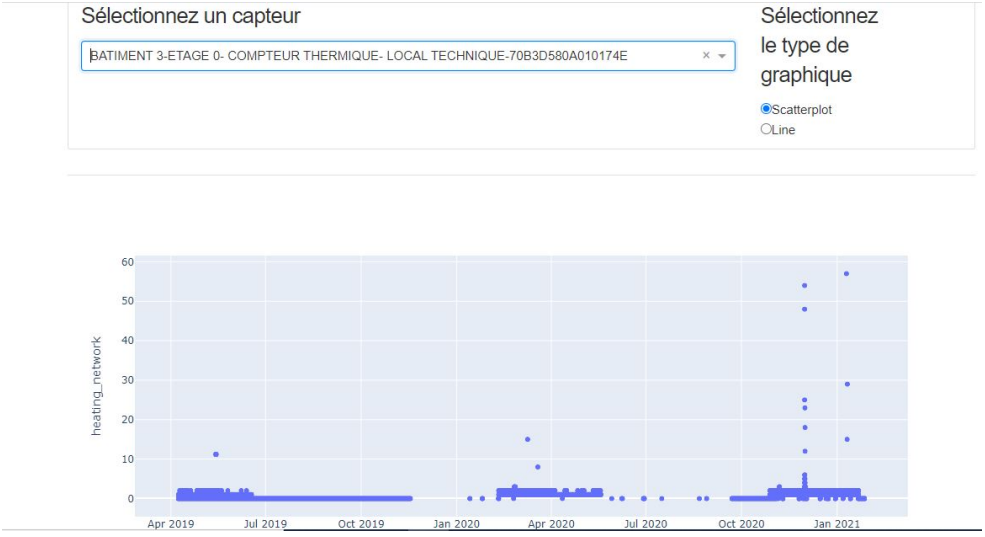

FIGURE 16 – Visualisation sur toute la période d'installation des capteurs

Courbe

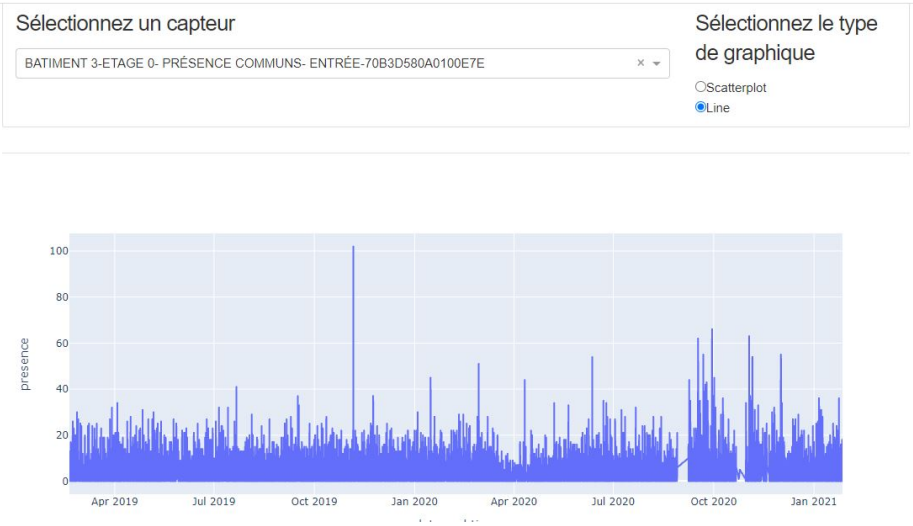

FIGURE 17 – Visualisation sur toute la période d'installation des capteurs

La seconde qui permet d'afficher une ou plusieurs courbes superposées tout en précisant le type de mesures, les bâtiments, les étages, les zones d'installation, les identifiants des capteurs et enfin une date d'entrée et une date de fin.

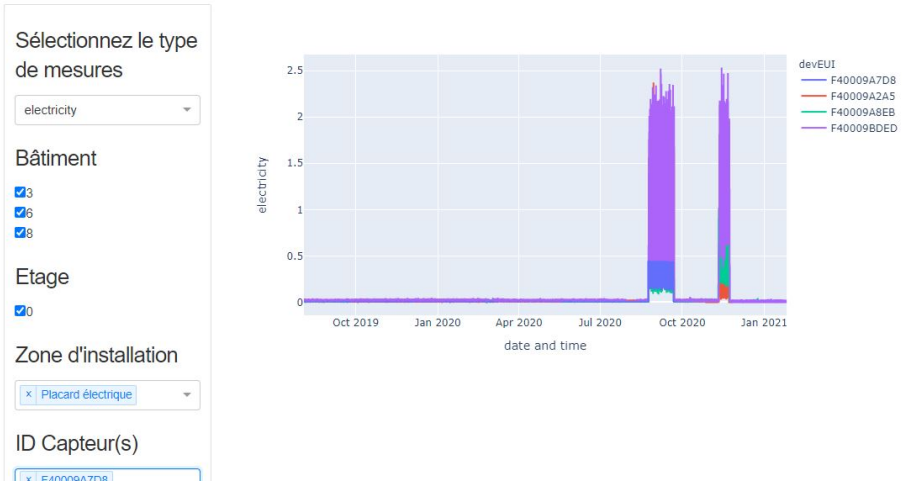

FIGURE 18 – Graphique personnalisée

Le quatrième onglet "Affichage personnalisée" contient deux listes déroulantes, la première permet de sélectionner un capteur parmi tous les capteurs installés pour afficher soit des nuages de points ou une courbe (comme la figure 16). La deuxième qui permet d'afficher des courbes superposées pour n'importe quel type de données(comme la figure 17).

Le dernier onglet "Résumé" contient trois listes déroulantes, la première pour sélectionner le type de données, la deuxième pour sélectionner l'identifiant d'un ou plusieurs capteurs et la dernière pour sélectionner les informations dont on veut avoir sur ces capteurs. Une fois que les champs sont sélectionnés, il y a un tableau qui s'affiche avec les informations.

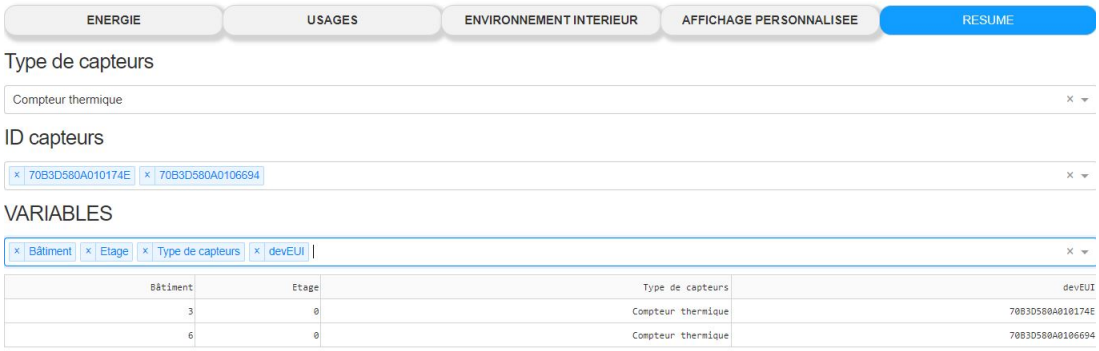

FIGURE 19 – Tableau résumé des capteurs

9 Bilan du travail accompli

9.1 Apport du stage

Ce stage a été très enrichissant pour moi en terme d'apprentissage, il m'a permis de travailler sur des données réelles de volume important. J'ai découvert une nouvelle librairie permettant de visualiser des données avec des graphiques interactifs. Les échanges avec les tuteurs m'ont aidé à développer la capacité de présenter un travail de manière synthétique dans un temps limité.

J'ai apprécié le fait de pouvoir travailler en autonomie, mes tuteurs étaient toujours à ma disposition pour m'orienter sur mes choix et mes méthodes de travail.
J'ai renforcé mes connaissances en programmation Python, plus précisément au niveau des librairies : numpy, pandas, matplotlib, plotly, seaborn, datetime, ces bibliothèques représentent des outils incontournables en analyse de données.

Mes compétences académiques en développement Python m'ont facilité la réalisation des différentes missions qui m'ont été confiées que ce soit au niveau du prétraitement des données et au développement de l'application.

Pour le développement de l'application de visualisation des données, Etant donné que j'avais déjà une première expérience avec la librairie Rshiny, cette librairie permet de créer des applications web pour visualiser des données avec le langage R, ça a été très facile pour moi de comprendre le fonctionnement de la librairie dash qui lui est très similaire.

Ce stage a confirmé mon désir de vouloir m'orienter dans un domaine très proche des mathématiques et de l'informatique.
Fort de cette expérience, j'aimerais beaucoup par la suite soit faire une thèse en Mathématiques appliquées soit travailler dans une entreprise en tant que data analyst.

9.2 Bilan technique

Lors de la première visualisation des données, on s'était rendu compte que le tracé en ligne continu des séries temporelles des capteurs n'était pas adapté car il y avait des données manquantes. Il fallait donc choisir un nuage de points pour à la fois une meilleure représentation et la détection de l'absence de données.

Pour les données d'ouverture de fenêtre, les capteurs doivent envoyer un point de données minimum par heure, lors du calcul des statistiques de données manquantes, on avait supposé que 24 points de données minimum pour une journée correspondaient à 100% de données reçues. On a fait ce choix, car une vérification manuelle serait un travail fastidieux.

L'utilisation de la librairie dash qui utilise Plotly pour afficher les graphiques a été très utile, non seulement pour sa facilité d'utilisation mais aussi elle permet de créer des graphiques interactifs qui permettent d'avoir une meilleure vue, de sélectionner des endroits précis sur le graphique.

Dans le cadre du développement de l'application, lorsqu'on met à jour les données, il faut d'abord lancer un script de prétraitement pour avoir les données nécessaires pour lancer l'application, puis exécuter le code de l'application. On avait procédé ainsi car si on réunit les deux scripts, l'application mettra beaucoup de temps à répondre, une piste d'amélioration serait d'avoir un code qui fait le prétraitement de manière automatique après chaque mis à jour, pour ensuite stocker les données nécessaires sur une base de données qui elle même est liée avec l'application.

10 Conclusion et Perspectives

Pour conclure, ce stage m'a permis de mettre en pratique mes connaissances théoriques et pratiques acquises durant ma formation en Ingénierie Mathématiques et Biostatistiques à l'Université de Paris.

En termes de résultats, on a réalisé la mise en forme des données, le prétraitement de données, la visualisation, une statistique de données manquantes et le développement d'une application de visualisation de données.

On a pu détecter les capteurs qui perdent des données à partir des premiers graphiques (scatterplot et heatmap). On a vu aussi que le type de capteur n'était pas un bon indicateur permettant de comprendre pourquoi il y avait des données manquantes, mais l'appartement auquel le capteur est installé pouvait aider à évoquer des hypothèses.
Avec les statistiques de données manquantes on a constaté que certains capteurs envoyaient plus de données que prévues et certains envoyaient des données qu'elles n'étaient pas censé envoyer.
En comparant le taux de données reçues et la taille des fichiers, on constaté qu'il y avait toujours une corrélation sur la période considérée et un changement de coefficient de corrélation au bout d'un moment qui est lié à un changement de format de fichiers.

En termes de perspectives, on pourra procéder à une analyse de données approfondie, ce qui permettra de voir entres autres s'il y a des corrélations entre les données météorologiques, analyser le système de chauffage et les usages.

On pourra faire du clustering afin de dégager des tendances, de voir s'il y a des relations entre différents profils. on pourra aussi faire du machine learning qui permettra de faire des prédictions.

Sur l'application de visualisation de données on pourra ajouter une page d'accueil qui nécessite un identifiant et un mot de passe pour accéder à l'application, créer une interface pour chaque appartement, on pourra limiter certains utilisateurs par rapport à certaines pages.

11 Bibliographie

Références

[1] C David. "How to Build a Reporting Dashboard using Dash and Plotly". In : (2019). DOI : 10.1109/MCSE.2007.55.

[2] P Fanny. "Les formations pour devenir data analys". In : (2020). URL : https://www.futura-sciences.com/sciences/questions-reponses/ecoles-formations-formations-devenir-data-analyst-14573/.

[3] J. D. Hunter. "Matplotlib : A 2D graphics environment". In : *Computing in Science & Engineering* 9.3 (2007), p. 90-95. DOI : 10.1109/MCSE.2007.55.

[4] Plotly Technologies Inc. *Collaborative data science*. 2025.