Mini Projet :
Prédiction l'aération des pièces sur une maison

Laura MALLET
Sophie WINUM

Sommaire

I. Introduction générale

Le secteur du bâtiment représente aujourd'hui près de 44 % de l'énergie consommée en France et émet plus de 123 millions de tonnes de CO_2. Il est ainsi un secteur clé dans la lutte contre le réchauffement climatique et la transition énergétique. C'est pourquoi, les bâtiments sont de plus en plus pensés intelligemment avec l'aide de capteurs afin d'optimiser l'efficacité énergétique et de réduire la consommation énergétique.

II. Présentation de l'habitat intelligent

Le projet Expe-smarthouse a été initié en 2018 afin de récolter des données d'une maison connectée de 120 m² occupée par 5 personnes en Savoie. Ce projet donne accès à environ 340 points de mesure pour les scientifiques associée à différents paramètres comme la température, l'humidité, la détection de mouvement ou encore la position d'ouverture de chaque porte et fenêtre.

III. Problématique

Nous allons, ainsi, pouvoir analyser certaines de ces données afin de prédire l'influence de l'aération (ouverture des fenêtres) sur la maison. En effet, il paraît important de conserver un air intérieur d'une bonne qualité pour évacuer les mauvaises odeurs, pour réguler l'humidité à moindres frais de l'air intérieur et donc pour éviter l'accumulation de composés nocifs (bactéries/moisissures/ substances chimiques polluantes). De plus, une pièce bien aérée permet d'améliorer grandement le confort des occupants surtout en été où la chaleur peut être désagréable.
Enfin, un air ni trop sec, ni trop humide est plus facile à chauffer ou à climatiser, ce qui permet de faire des économies d'énergie.

Dans le cadre de ce projet, nous allons ainsi analyser:

1) L'influence de la température extérieure sur la fréquence et la durée pendant lesquelles les fenêtres sont ouvertes. Les occupants auront, par exemple, tendance à ne pas ouvrir les fenêtres longtemps quand il fera froid en hiver.
2) L'impact de cette ouverture de fenêtres sur la température intérieure, le bruit, l'humidité et sur le taux de CO_2 dans deux pièces de la maison:
    - La chambre 1
    - La cuisine
3) Réaliser un modèle de prédiction de l'ouverture de la fenêtre à l'aide d'un arbre de décision, un outil de machine learning supervisée et en analyser les paramètres.

La localisation des pièces et des fenêtres est donnée en *Annexe 1.*

IV. Extraction de données

1. Introduction

Les données que nous utilisons sont récoltées grâce à différents capteurs situés dans chaque pièce de la maison (voir *Annexe 2*). Ces données peuvent être gérées grâce au logiciel Influxdb. Dans le cadre de notre projet, elles seront extraites sur un fichier csv à l'aide de Python afin de les exploiter. Il sera, en effet, intéressant d'analyser ces différentes données entre elles au cours de l'année.

Afin d'évaluer l'impact de l'ouverture sur différents indicateurs, nous avons besoin des séries de données suivantes :

❖ Cuisine
➔ Détecteur Ouverture Fenêtre Evier/Ouverture (885)
➔ Qualité Air RDC/CO2 (3887)
➔ Qualité Air RDC/Température (3889)
➔ Qualité Air RDC/Humidité (3892)
➔ Qualité Air RDC/Bruit (3888)

❖ Chambre 1
➔ Détecteur Ouverture Fenêtre Chambre 1/Ouverture (4296)
➔ Qualité Air ch1/CO2 (5484)
➔ Qualité Air ch1/Température (3917)
➔ Qualité Air ch1/Humidité (3918)

❖ Extérieur
➔ Qualité Air Extérieur/Température (3894)
➔ Qualité Air Extérieur/Humidité (3897)

2. Codage

Comme expliqué précédemment, en utilisant *Python*, on commence par extraire une série de données (ex: qualité Air RDC/CO2) puis on l'importe dans un fichier csv. On effectue la même opération pour les autres séries. Pour ce faire, nous avons utilisé la bibliothèque *Panda* de *Python* afin de lire le fichier csv et de convertir les données en data frame qui sont des structures plus facilement modulables que les listes.

Nous avons ensuite observé trois problèmes majeurs:

- Les données n'étaient pas récupérées par les capteurs en même temps pour les différentes séries. Il fallait donc rééchantillonner sur une heure les données de chaque série pour pouvoir les comparer entre elles.
    - Pour cela, on a d'abord uniformiser le format de la date en ne gardant que le jour et l'heure (hms): 2022-01-30T16:09:51Z,1536.0 → 2022-01-30T16:09:51
    - En utilisant la fonction *resample* de la bibliothèque *Panda*, on a effectué la somme sur une heure des données des séries correspondant à l'ouverture des fenêtres afin d'avoir le nombre d'ouverture par heure. On a effectué de même avec les autres séries en appliquant cette fois-ci une moyenne sur une heure afin d'avoir la mesure moyenne par heure de la température, de l'humidité, du taux de CO_2 et du bruit.

  Pour pouvoir les comparer plus facilement, on crée un nouveau fichier *csv* qui regroupe toutes nos séries de données dans un seul fichier.

- Malheureusement, les séries correspondant à l'ouverture des fenêtres n'avaient pas les données de la date actuelle, les mesures s'arrêtant à la veille contrairement aux autres séries. Il a donc fallu supprimer toutes les données correspondant au jour actuel dans les autres séries par homogénéité dans l'excel final.

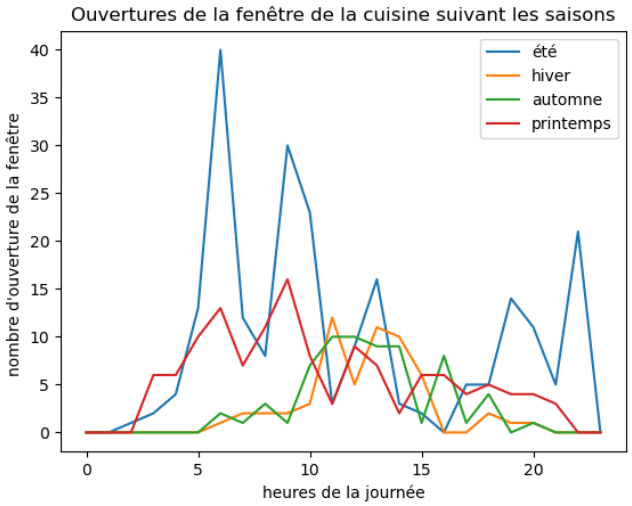
- Il y a également des jours où il n'y a pas du tout de données (NaN), sûrement dues à des défauts ou des pannes de capteurs. Nous avons donc supprimé ces jours-là, afin de ne pas avoir de fausses interprétations.

L'ensemble du code pour l'extraction des données est en *Annexe 3.*

V. Résultats et interprétations

1. Temporalité de l'ouverture des fenêtres

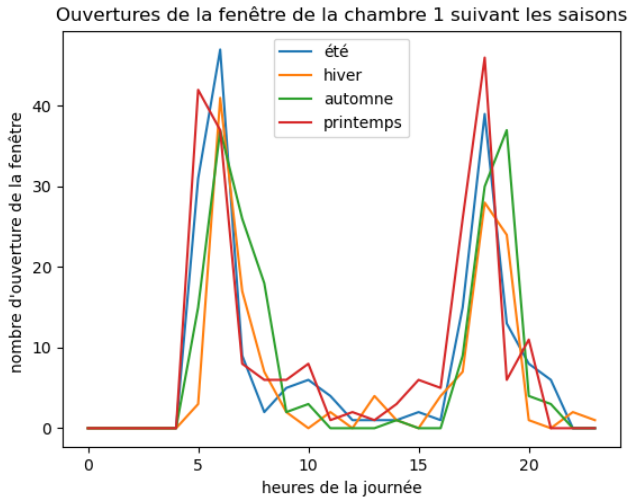
Nous avons réalisé un code simple (*Annexe 4*) afin de savoir les heures où il y a le plus d'ouverture suivant les saisons.
Pour la cuisine :



Dans la cuisine, la fenêtre est principalement ouverte entre midi et deux afin de chasser les odeurs causées par la préparation du repas. Il en est de même parfois pour le petit-déjeuner ou le repas du soir.

En été et printemps, la fenêtre est souvent ouverte, pour le confort de l'habitation. Plutôt tôt le matin et tard le soir, cela permet de laisser passer un peu de fraîcheur dans les pièces. Les habitants évitent d'ouvrir l'après-midi après 15h, là où le ressenti est le plus chaud. En hiver et en automne, au contraire, il fait trop froid pour aérer vers 6h et 22h.

Pour la chambre 1 :

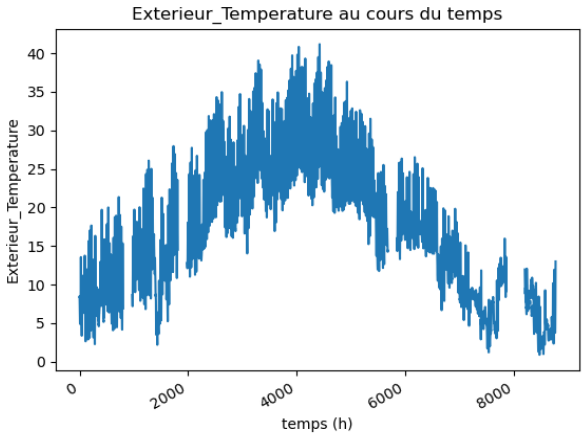


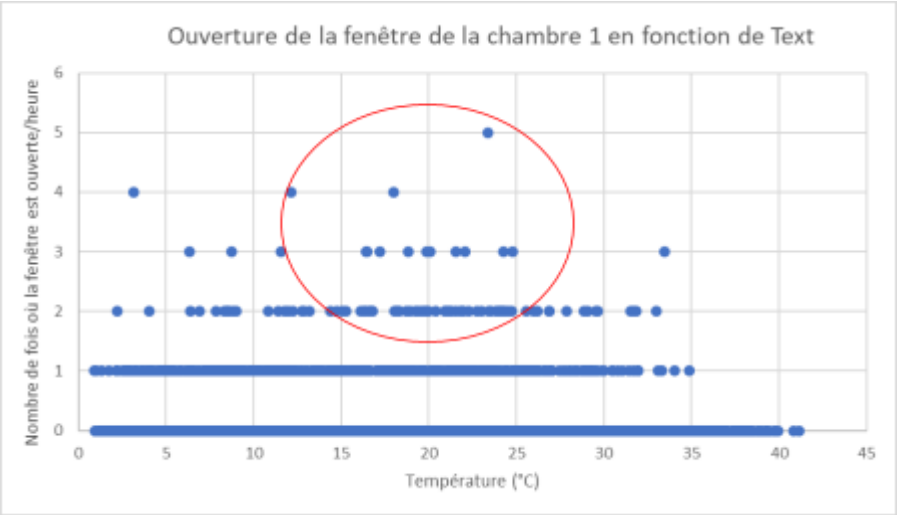
Dans la chambre 1, on remarque qu'en majeure partie, la fenêtre de la chambre est ouverte afin d'aérer la pièce le matin aux alentours de 6h quand la personne se lève et le soir vers 18h. C'est le cas pour toutes les saisons.

2. Influence de l'extérieur sur l'ouverture des fenêtres

Dans cette partie, on étudiera l'influence sur la **chambre n°1.**

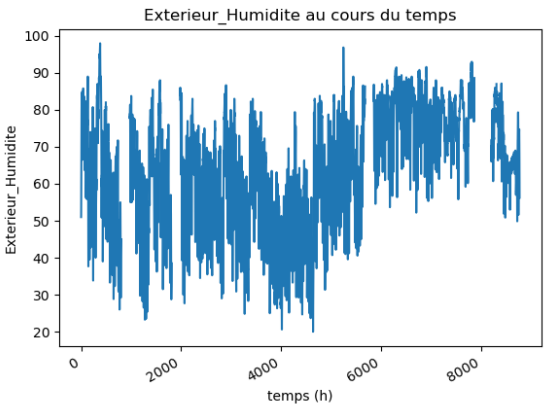
2.1. Influence de la température extérieure

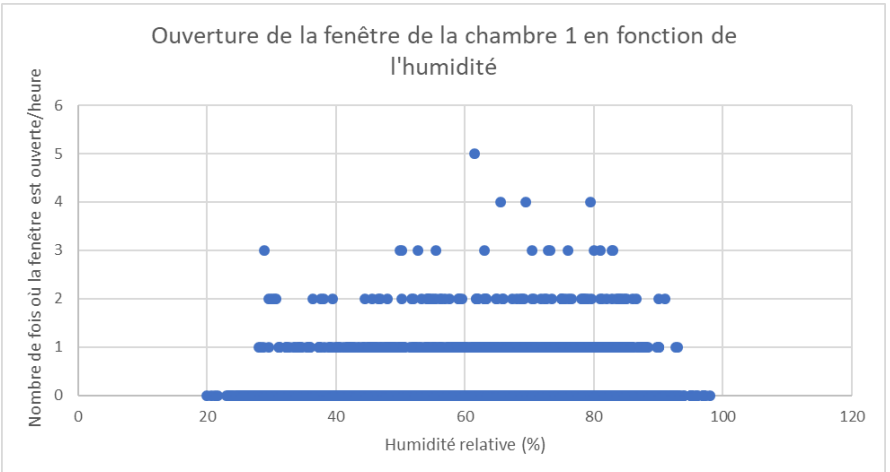




Quand il fait très chaud (supérieure à 35°C), la fenêtre n'est pas du tout ouverte, afin de ne pas faire rentrer la chaleur.
Un maximum d'ouverture par heure a été détecté quand la température est moyenne (entre 15 et 25°C). Cela paraît logique puisqu'on va laisser plus longtemps les fenêtres ouvertes pour des températures douces. S'il fait très froid, on n'ouvrira pas longtemps pour ne pas laisser passer la fraîcheur et inversement pour les températures très hautes.

2.2. Influence de l'humidité extérieure





Quand l'humidité est la plus importante (> 65%), le nombre d'ouverture détectée par heure est le plus grand. Ainsi on aère plus longtemps en temps humide entre 60 et 80% pour assécher un peu son habitation et éviter toutes moisissures et bactéries.
Quand l'humidité extérieure est trop basse (20%) ou au contraire trop haute (95%), la fenêtre n'est jamais ouverte justement pour ne pas trop assécher la maison ou au contraire la rendre trop humide.

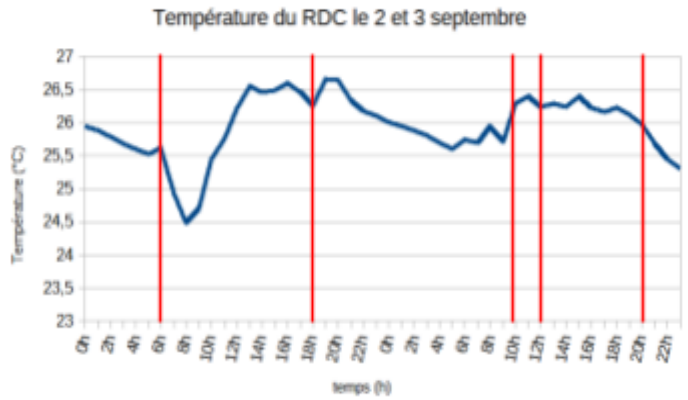
3. Influence de l'ouverture sur l'intérieur des pièces

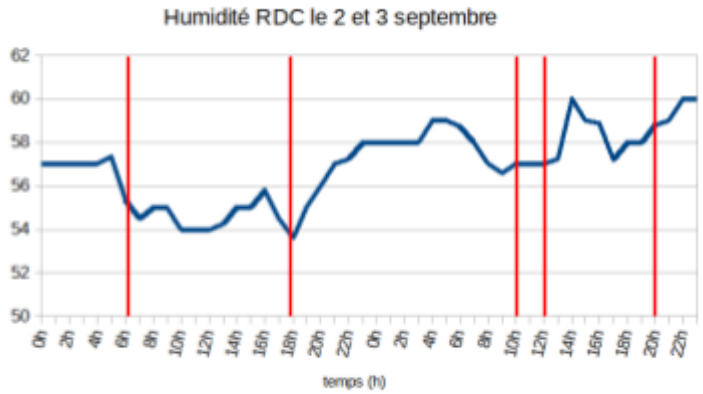
Nous avons, ensuite, étudié l'influence de l'ouverture des fenêtres sur les paramètres de la cuisine et de la chambre n°1. Pour cela, nous avons pris 2 jours dans l'année : le 02 et 03 septembre 2022 à partir de 0h. Les fenêtres ont été souvent ouvertes ces jours-ci ce qui permet de mieux étudier l'influence de l'aération.

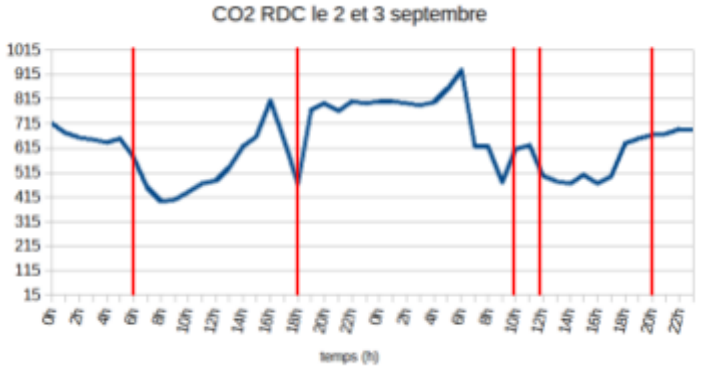
En *Annexe 5*, se trouve aussi les mêmes graphiques pour d'autres jours en hiver (25 et 26 février 2022).

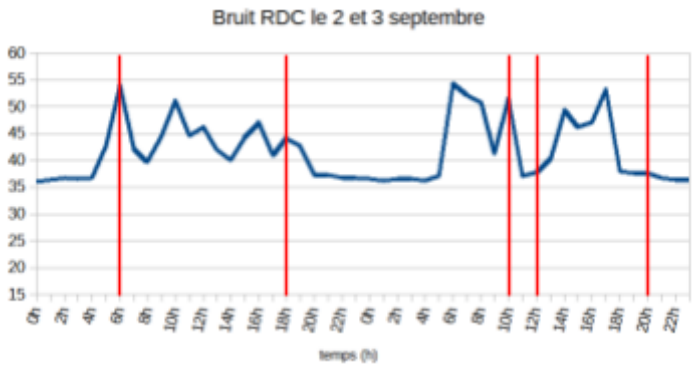
3.1. Cuisine

02/09/2022 : ouvertures de la fenêtre de la cuisine à 6h et 18h
03/09/2022 : ouvertures de la fenêtre de la cuisine à 10h, 12h et 20h.









Fin d'été, l'ouverture de la fenêtre tôt le matin et tard le soir permet une diminution de la température. Ici la température a chuté de 1°C. En effet, les températures extérieures sont plus basses qu'en journée et plus basses que la température intérieure de la pièce ce qui permet de la rafraîchir.

L'aération n'a pas un impact visible sur l'humidité sur ce graphique. Cela peut s'expliquer par le fait que l'humidité de la cuisine dépend aussi de l'humidité extérieure et d'autres paramètres qui influent plus grandement sur celle-ci. A noter, selon l'Observatoire de la qualité de l'air intérieur, le taux d'humidité intérieur ne doit pas dépasser les 40 à 60%, ce qui est bien le cas ici.

Aérer une pièce permet de diminuer le pourcentage de CO_2 dans une pièce théoriquement. Dans la cuisine, cette diminution est peu visible sur le graphique car elle dépend également du nombre de personnes qui restent dans la cuisine avant et après ouverture de fenêtres et qui vont émettre du CO_2.

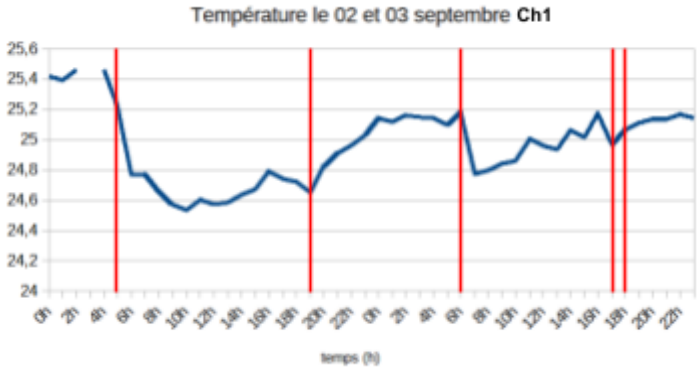
Le bruit n'est pas forcément lié à l'ouverture de la fenêtre. En effet, le bruit est lié à plein d'autres causes, comme le nombre de personnes dans la pièce par exemple. La courbe nous donne l'impression du contraire: le bruit diminue quand on ouvre la fenêtre de la cuisine. Cela est peut être dû au fait que le bruit se disperse plus et se propage dehors au lieu d'être seulement dans la cuisine.

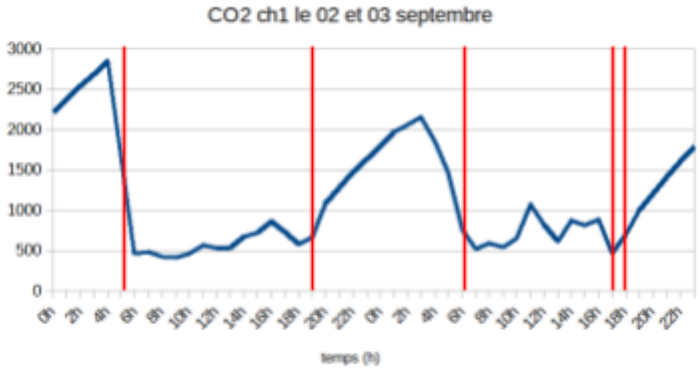
Finalement, on peut dire que dans cette pièce, l'ouverture de la fenêtre n'influence pas clairement les différents paramètres trackés. C'est pourquoi, on va également étudier la chambre 1, qui se trouve au premier étage.

3.2. Chambre 1

On étudie maintenant les paramètres de température, humidité et CO_2 de la chambre 1, pour les mêmes jours étudiés dans la cuisine.

Soit le 02/09 : ouvertures de la fenêtre à 5h et 19h
Et le 03/09 : ouvertures de la fenêtre à 6h, 17h et 18hx2





L'ouverture de la fenêtre de la chambre a un grand impact sur la température intérieure. En effet, la température intérieure décroît considérablement après l'ouverture le matin, la température extérieure étant plus faible à cet horaire. En revanche, pour aérer la chambre avant de dormir, l'ouverture de fenêtre en début de soirée a tendance à augmenter la température intérieure car il fait encore chaud dehors.

Pour l'humidité dans la chambre, cette dernière diminue efficacement grâce à une aération matinale. Elle est la plus haute pendant la nuit car la respiration des occupants qui dorment crée une humidité naturelle, qui se condense au contact des murs froids. C'est pourquoi, il est important d'aérer le matin. Les valeurs d'humidité atteintes restent raisonnables pour les habitants.

La quantité de CO_2 est la plus haute la nuit quand la personne est présente dans sa chambre. Cependant, on peut voir que dès que cette dernière aère le matin, cette quantité chute drastiquement. L'aération permet donc bien de diminuer le taux de CO_2 dans l'air des pièces. Cependant dès qu'une personne entre dans la pièce, il en sort que celle-ci émet globalement plus de CO_2 que l'aération n'en enlève.

En conclusion, les différentes ouvertures de la fenêtre de la chambre 1 influencent plus visiblement les paramètres choisis comparé à la cuisine. Cependant, d'autres paramètres sont très influents comme la température et l'humidité extérieure que nous avons étudié auparavant ou encore la présence de personnes dans les pièces, c'est pourquoi il nous est impossible de voir clairement l'impact de l'aération sur l'air intérieur. Il faudrait pour cela, avoir accès à d'autres données comme la détection de mouvement et ajouter aux graphiques au cours du temps les paramètres extérieurs (humidité, température).

VI. Prédiction de l'ouverture à l'aide d'un arbre de décision

Maintenant que nous avons vu que l'ouverture des fenêtres influe sur les différents paramètres, nous avons voulu prédire l'ouverture de la fenêtre de la chambre 1 (le label) en fonction des données d'entrée (aussi appelées des features), à savoir la température, l'humidité et le taux de CO_2 dans la chambre 1. L'ouverture de la fenêtre étant aussi corrélée à la température extérieure ainsi qu'à

l'humidité extérieure, on tient également compte de ces paramètres. Cette prédiction s'effectue grâce à une machine learning supervisée.

Nous utilisons un arbre de décision. La machine learning va apprendre à prédire grâce à notre base de données qui se découpera en deux parties :

- Une partie des données est utilisée pour la **phase d'entraînement**. La machine learning va apprendre à calculer le nombre d'ouverture par heure de la fenêtre grâce aux features citées ci-dessus, le but étant de se rapprocher de la bonne valeur de ce nombre d'ouverture qu'elle connaît déjà. Elle comprendra alors quelles caractéristiques ont plus ou moins d'impact sur cette prédiction. La phase d'entraînement nous permet de construire un modèle de décision.
- La seconde est la **phase de test**, qui consiste à valider le modèle précédemment établi. Le but est de le tester sur le reste des données afin d'obtenir une valeur estimée du nombre d'ouverture par heure et de la comparer avec celle théoriquement trouvée grâce aux capteurs. Si les deux valeurs sont proches, notre modèle est pertinent. Pour le savoir, nous utilisons des critères de précision: f_score, accuracy, precision.

Souvent, plus il y a de données dans la phase d'apprentissage, plus la machine learning peut se rapprocher de résultats cohérents. On utilise donc 70% des données pour la phase d'entraînement et 30% pour la phase de test qui correspond au meilleur ratio pour l'arbre de décision.

1. Modélisation

Nous avons mis dans un fichier CSV les données utiles pour la construction de l'arbre.
Les features sont donc : Exterieur_Temperature, Exterieur_Humidite, Ch1_Temperature, Ch1_Humidite.
Le label est : Ch1_Ouverture. Le label prend 5 valeurs qui correspondent au nombre de fois où la fenêtre à été ouverte dans l'heure : 0, 1, 2, 3, 4. Quand le label est à 0, la fenêtre est fermée, ce qui correspond à la plupart des cas. Il faut savoir qu'en moyenne l'ouverture de la fenêtre est d'environ une demi-heure.

Le code est donné en *Annexe 6*.

2. Résultats

2.1. Arbre de décision

Sans prendre de maximum de profondeur, on obtient un arbre de décision très complexe :

2.1.1. Paramètres de précision

Précision	F-Score	Global accuracy (= exactitude)
87,69 %	89,42 %	91,74 %

De premier abord, on remarque que les paramètres de précision sont bons. Si les paramètres sont si bons, c'est parce que la taille des classes de label sont très déséquilibrées. En effet, le nombre d'ouverture par heure différent de zéro dans les données du label est très faible. Ainsi, si la machine learning se trompe et met trop de zéro, la précision globale restera bonne.

L'accuracy globale ne permet, ici, pas de bien voir si notre arbre de décision est pertinent. C'est pourquoi, on étudie aussi l'accuracy moyenne ainsi que les accuracy pour les différentes valeurs des labels.

Matrice des critères en fonction des valeurs des labels:

	precision	recall	f1-score	support
fenetre fermée	0.93	0.92	0.93	2181
fenetre ouverte 1fois	0.12	0.15	0.13	140
fenetre ouverte 2fois	0.00	0.00	0.00	21
fenetre ouverte 3fois	0.00	0.00	0.00	6
fenetre ouverte 4fois	0.00	0.00	0.00	2
accuracy			0.86	2350
macro avg	0.21	0.21	0.21	2350
weighted avg	0.87	0.86	0.87	2350

On voit que l'accuracy par feature est très élevée pour la porte fermée, c'est-à-dire pour les 0, comme dit plus haut. De plus, on voit que pour les autres features, l'accuracy est très faible, voire nulle. L'accuracy moyenne (86 %) est plus faible que l'accuracy globale (91,74 %), ce qui est plus cohérent.

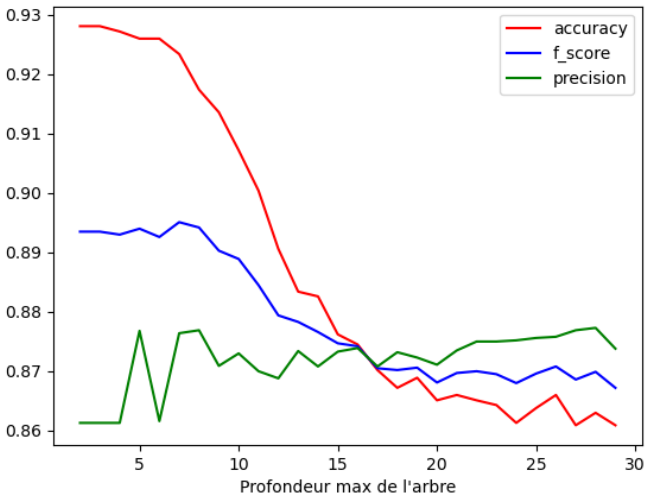
2.1.2. Importance des features

Features	Importance
Ch1_Humidite	0.2364
bbxco2_ch1	0.2235
Exterieur_Humidite	0.1943
Exterieur_Temperature	0.185
Ch1_Temperature	0.1608

On voit que la feature la plus importante est l'humidité dans la chambre. Cela veut dire que c'est l'humidité qui est le plus sensible à l'ouverture de la fenêtre dans la chambre.

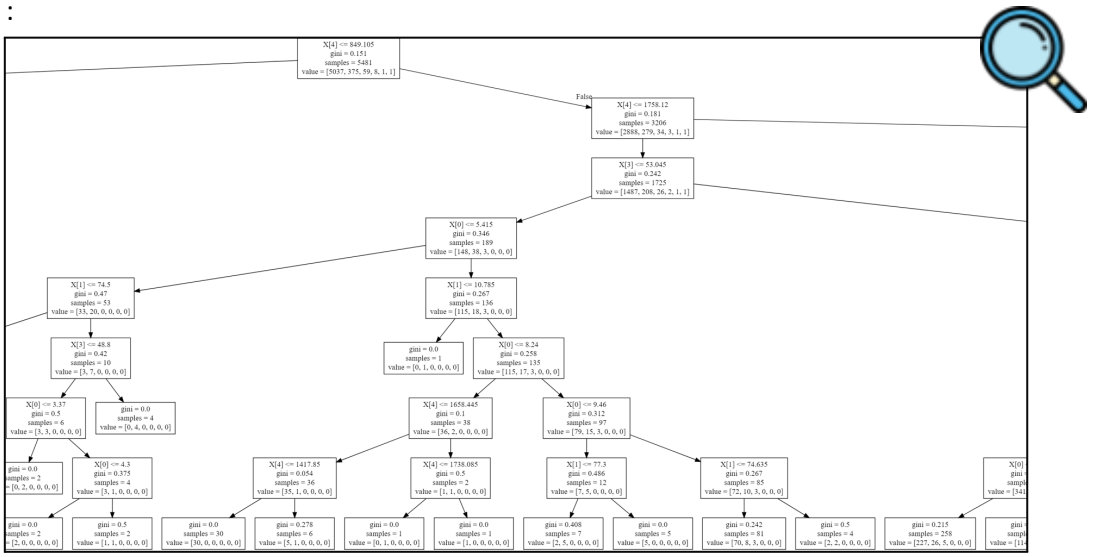
2.2. Modification de la profondeur de l'arbre et son influence sur l'exactitude du modèle

Dans le premier modèle, on a obtenu une profondeur d'arbre de 30. On va maintenant faire varier la profondeur de l'arbre pour voir quelle est la profondeur optimale pour avoir la meilleure exactitude.



Pour la même raison que précédemment, l'exactitude est très bonne pour des petites profondeurs d'arbre. En effet, dans le fichier de départ, le nombre de labels différents de zéro est très faible. Ainsi quand les profondeurs sont plus petites, le machine learning va mettre plus de zéro, et donc l'exactitude va être plus élevée.

On prend une profondeur 8 pour avoir un arbre de décision plus facile à lire. Et on obtient l'arbre suivant :



Pour mieux voir comment se construit l'arbre, voir l'*Annexe 8*.

Précision	F-Score	Global accuracy
87,69 %	89,42 %	91,74 %

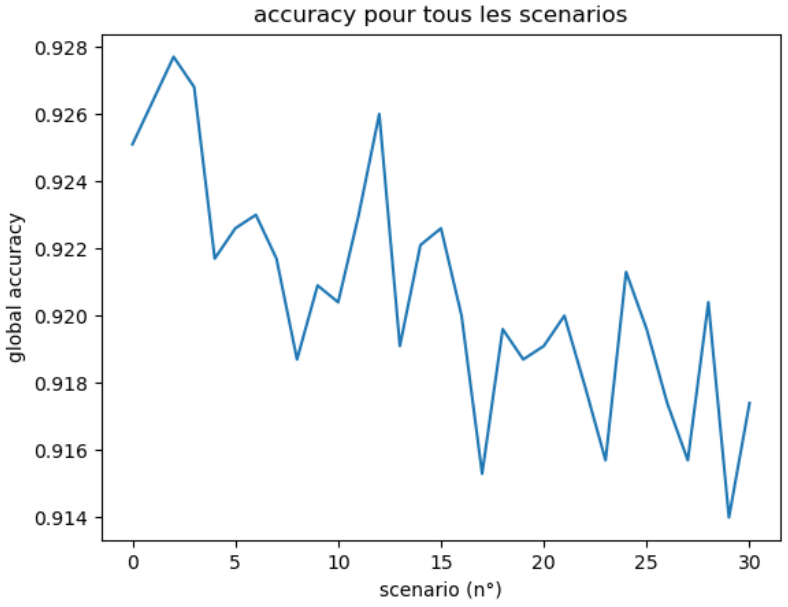
Matrice des critères en fonction des valeurs des labels:

	precision	recall	f1-score	support
fenetre fermée	0.93	0.98	0.96	2181
fenetre ouverte 1fois	0.21	0.06	0.10	140
fenetre ouverte 2fois	0.00	0.00	0.00	21
fenetre ouverte 3fois	0.00	0.00	0.00	6
fenetre ouverte 4fois	0.00	0.00	0.00	2
accuracy			0.92	2350
macro avg	0.23	0.21	0.21	2350
weighted avg	0.88	0.92	0.89	2350

Comme expliqué plus haut, l'accuracy par feature pour la porte fermée est donc plus élevée avec une profondeur plus faible (93%). Il en est de même pour l'accuracy moyenne (92% contre 86% avant). Les autres features ont, elles, une accuracy encore plus faible qu'avant.
De plus, la profondeur ne change pas la classification des features: l'humidité intérieure reste le paramètre le plus influent sur le nombre d'ouverture de la fenêtre par heure.

2.3. Exactitude en fonction de l'utilisation des capteurs

On fait différents scénarios avec l'utilisation de différents capteurs et on regarde l'exactitude. Tous les scénarios sont en *Annexe 7.* On obtient le graphe suivant :



Finalement, le scénario qui obtient la meilleure exactitude est le scénario n°3 ('Ch1_Temperature').
On peut aussi noter le scénario ('Ch1_Temperature', 'Ch1_Humidite').

VII. Conclusion

En conclusion, dans un premier temps, on a pu voir que l'aération était très utilisée dans la cuisine après manger et dans la chambre au réveil et le soir. Cette aération dépend assez significativement de l'environnement extérieur notamment de la température et de l'humidité. Quand ces dernières sont trop extrêmes, les fenêtres resteront fermées.

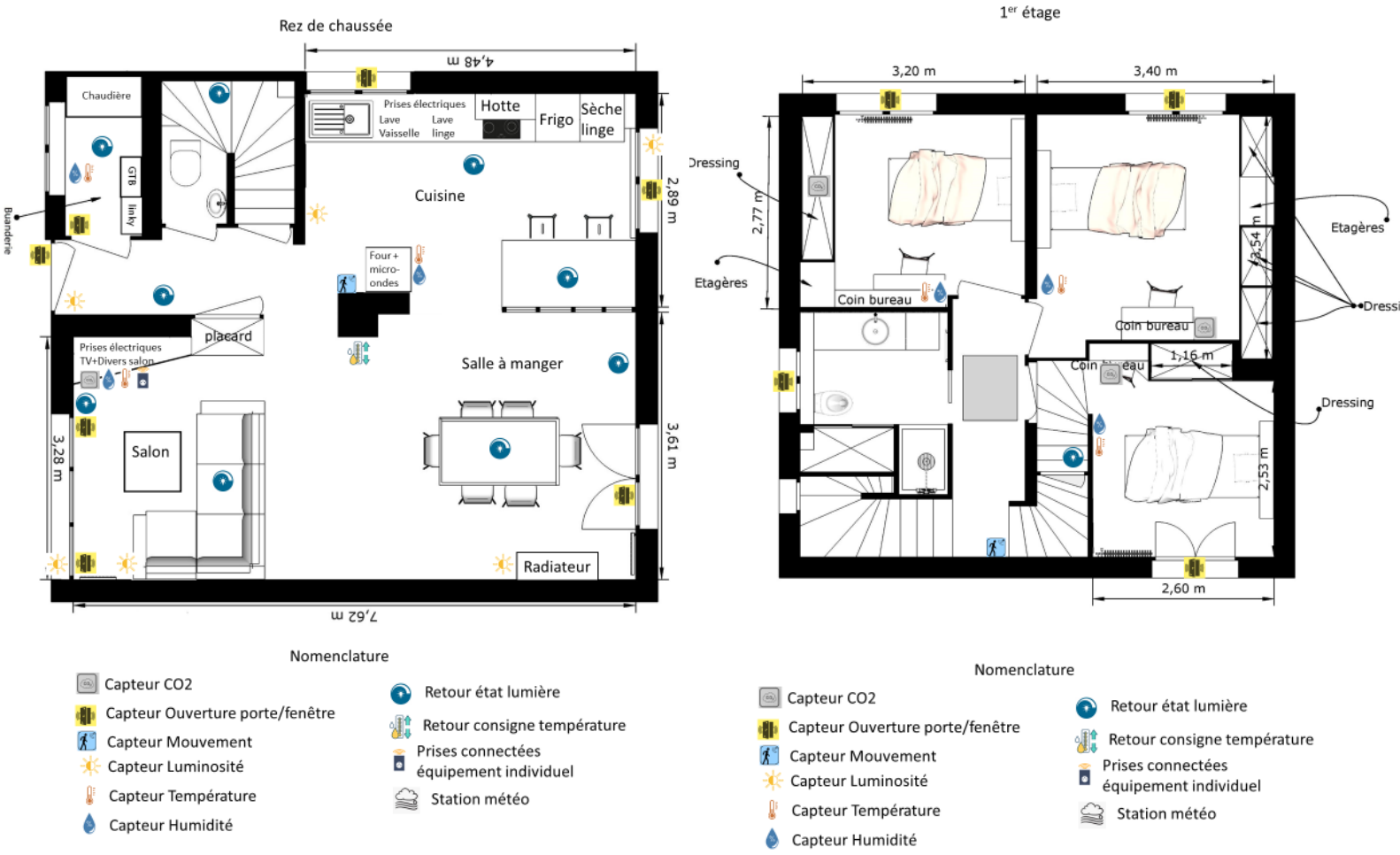
Nous avons également observé l'impact de l'aération sur différents paramètres comme la température, l'humidité et le taux de CO_2 dans la pièce. D'autres facteurs influencent ces paramètres comme la présence de personnes dans la pièce pour le CO_2 ce qui ne rend pas évident l'analyse. Globalement, il en ressort que aérer une pièce permet de diminuer le taux de CO_2 et de modifier en fonction de l'air extérieur l'humidité et la température de la pièce.

Grâce à l'arbre de décision, on a pu voir que l'aération dépend considérablement de l'humidité intérieure. Si les pièces sont trop humides, les occupants vont plus ouvrir les fenêtres. Cependant, cet arbre ne permet pas de prédire correctement le nombre d'ouverture par heure quand celui-ci est non nul. Le modèle est donc à perfectionner. Par exemple, il faudrait rajouter les autres paramètres influents comme données d'entrée et aussi collecter plus de données pour que la machine learning apprenne mieux. Si cette solution ne fonctionne pas, c'est que la corrélation entre nos entrées et notre sortie n'est pas évidente. Il faudrait aussi peut-être prendre mieux en compte le temps d'ouverture de la fenêtre, qu'on a actuellement approximé à moins d'une heure mais qui en pratique n'est pas toujours le cas.

VIII. Annexes

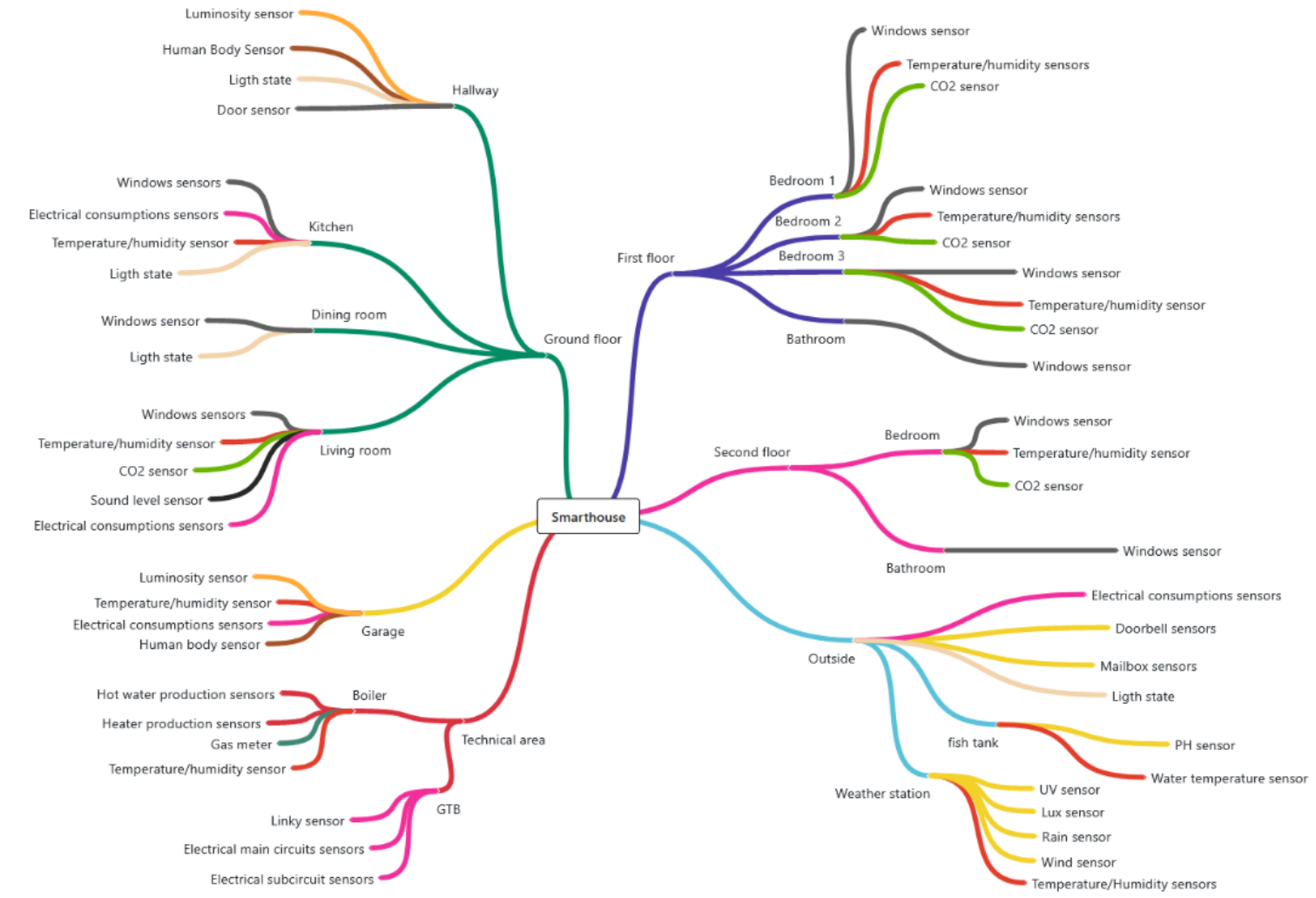
Annexe 1

Plan de la maison pour le rez-de-chaussée (cuisine) et pour le premier étage (chambre 1 et 2)



Annexe 2

Liste des différents capteurs situés dans la maison



Annexe 3

Code pour extraire les données et les important dans un seul fichier .csv

```
from http.server import BaseHTTPRequestHandler, HTTPServer
import urllib
import time
from influxdb import InfluxDBClient
import sys
import csv
import pandas as pd
import matplotlib.pyplot as plt
import time
from datetime import datetime

##########################
#    SCRIPT SETTINGS
##########################
# Set the port where you want the bridge service to run
PORT_NUMBER = 1234
# InfluxDB Server parameters
INFLUXDB_SERVER_IP = '82.65.155.71'
INFLUXDB_SERVER_PORT = 8086
INFLUXDB_USERNAME = 'eleves'
INFLUXDB_PASSWORD = 'SmarthouseG2Elab'
INFLUXDB_DB_NAME = 'jeedom'
##########################

client = InfluxDBClient(INFLUXDB_SERVER_IP, INFLUXDB_SERVER_PORT, INFLUXDB_USERNAME,
INFLUXDB_PASSWORD,
                        INFLUXDB_DB_NAME, ssl=True, verify_ssl=False)

print(client.get_list_database())

client.switch_database('jeedom')

identifiant = [4291, 4296, 885, 3887, 3888, 3889, 3892, 3894, 3897, 3911, 3914, 3915, 3917,
3918, 5484]
names = ['Ch2_Ouverture', 'Ch1_Ouverture', 'RDC_Ouverture', 'RDC_CO2', 'RDC_Bruit',
'RDC_Temperature', 'RDC_Humidite','Exterieur_Temperature', 'Exterieur_Humidite',
'Ch2_Temperature', 'Ch2_Humidite', 'Ch2_CO2', 'Ch1_Temperature', 'Ch1_Humidite',
'bbxco2_ch1']

# fonction qui enlève les caractères après le temps dans la colonne time du dataframe
def date_str_to_datetime(datetime_str):
   if "." in datetime_str:  # par exemple : 1999-03-25T14:21:20.435423Z -->
1999-03-25T14:21:20
       datetime_str = datetime_str.split(".")[0]  # on prend uniquement la chaine de
caractère avant le point
   else:
       datetime_str = datetime_str.split("Z")[0]  # par exemple : 1999-03-25T14:21:20Z →
1999-03-25T14:21:20
   return datetime.fromtimestamp(time.mktime(time.strptime(datetime_str,
'%Y-%m-%dT%H:%M:%S')))

# Initialisation
df_final = pd.DataFrame()  # création du nouvel dataframe qui nous permettra de le mettre
dans un CVS final
liste_pas_bon = []


min_longueur = 0
for n in identifiant:
   name = str(names[identifiant.index(n)])
   print("ind : ", n, " name : ", name)

   # Extraction des données du site
```

```
    datasheet = client.query(f'SELECT "value" FROM "jeedom"."autogen"."' + str(n) + '" WHERE time > now() - 365d')
    exported_data = list(datasheet.get_points())
    header_list = list(exported_data[0].keys())

     # Ecriture des données dans différents csv
    with open("Datas/RAW/dataraw" + str(n) + ".csv", "w", newline='') as fp:
        writer = csv.writer(fp, dialect='excel')
        #  print(header_list[1:])
        value_header = header_list[1]
        offset = sum(c.isalpha() for c in value_header)
        # print(offset)
        # header_list[1:] = sorted(header_list[1:], key=lambda x: int(x[offset:]))
        header_list[1:] = ['value']
        writer.writerow(header_list)
        for line in exported_data:
        writer.writerow([line[kn] for kn in header_list])

    # Mise en forme des fichiers :
    data = pd.read_csv("Datas/RAW/dataraw" + str(n) + ".csv", sep=',')
    data['time'] = data['time'].apply(date_str_to_datetime)
    data['time'] = pd.to_datetime(data['time'], format='%Y-%m-%dT%H:%M:%S')
    data = data.drop_duplicates()  # supprime les lignes où le datetime est le même
    data.rename(columns={'value': name}, inplace=True)  # on change le nom de de la colonne

    # *** les fichiers d'ouverture de fenêtre ne prennent pas en compte le dernier jour et ne sont donc pas sur le même nombre d'heures que les autres. Il faudra donc couper les autres séries pour qu'ils soient de la même taille que les fichiers ouvertures afin de pouvoir les comparer

    # Resample des données sur une heure :
    if n in [4291, 4296, 885]:
        data = data.resample('60T', on='time').sum()  # somme sur les ouvertures
    else:
        data = data.resample('60T', on='time').mean()  # moyenne avec les autres séries

    # crée des nouveaux fichiers CVS avec les bonnes données
    data.to_csv(f"Datas/RAW/dataraw{n}_resample.csv", sep=',')

    # Ajout dans un fichier csv commun :
    if len(data) == 8761:
        L = []
        index_def = data.index
        for i in range(len(data)):
            L.append(data[name][i])

        df_final['index_def'] = index_def
        df_final[name] = L

    else:
        liste_pas_bon.append([name, len(data)])

df_final.set_index("index_def")
df_final.to_csv("Datas/RAW/dataraw.csv", sep=',')  # crée des nouveaux fichiers CVS avec les bonnes données
print(liste_pas_bon)
```

Annexe 4

Code pour étudier la temporalité de l'ouverture des fenêtres

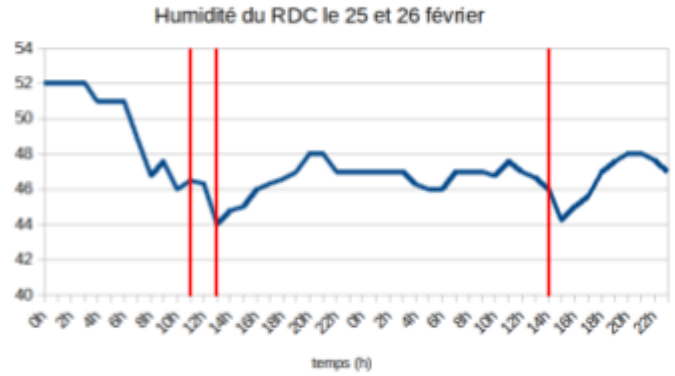
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from datetime import datetime

data = pd.read_csv('Datas/Dtree/data_destree.csv', sep=',')

index_def = data['index_def']
ouverture_ch1 = data['Ch1_Ouverture'] #250399


printemps = [4, 5, 6]
ete = [7, 8, 9]
automne = [10, 11, 12]
hiver = [1, 2, 3]

# --------- Ouverture Chambre 1 ------------
heure_printemps = np.zeros(24)
heure_ete = np.zeros(24)
heure_automne = np.zeros(24)
heure_hiver = np.zeros(24)


for k in range(len(index_def)):
   if int(ouverture_ch1[k]) != 0:
       time = pd.to_datetime(index_def[k], format='%Y-%m-%dT%H:%M:%S')
       if int(time.month) in printemps:
           heure_printemps[int(time.hour)] += ouverture_ch1[k]
       if int(time.month) in ete:
           heure_ete[int(time.hour)] += ouverture_ch1[k]
       if int(time.month) in automne:
           heure_automne[int(time.hour)] += ouverture_ch1[k]
       if int(time.month) in hiver:
           heure_hiver[int(time.hour)] += ouverture_ch1[k]


plt.figure()
plt.plot(range(24), heure_ete, label='été')
plt.plot(range(24), heure_hiver, label='hiver')
plt.plot(range(24), heure_automne, label='automne')
plt.plot(range(24), heure_printemps, label='printemps')
plt.xlabel('heures de la journée')
plt.ylabel("nombre d'ouverture de la fenêtre ")
plt.title("Ouvertures de la fenêtre de la chambre 1 suivant les saisons")
plt.legend()


# --------- Ouverture Cuisine ------------
ouverture_rdc = data['RDC_Ouverture']

heure_printemps_RDC = np.zeros(24)
heure_ete_RDC = np.zeros(24)
heure_automne_RDC = np.zeros(24)
heure_hiver_RDC = np.zeros(24)
```

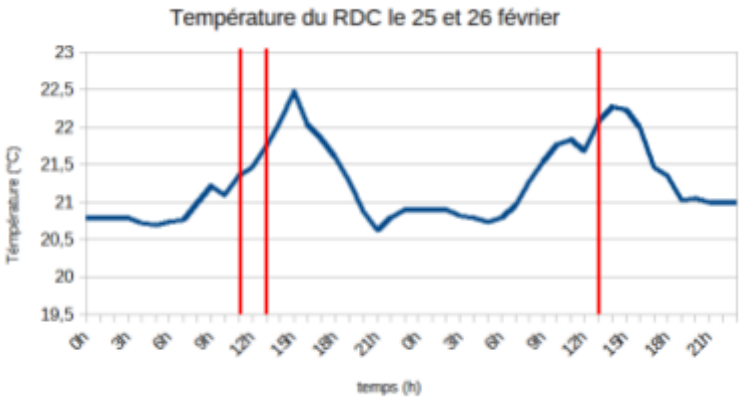
```
for k in range(len(index_def)):
   if int(ouverture_rdc[k]) != 0:
        time = pd.to_datetime(index_def[k], format='%Y-%m-%dT%H:%M:%S')
        if int(time.month) in printemps:
            heure_printemps_RDC[int(time.hour)] += ouverture_rdc[k]
        if int(time.month) in ete:
            heure_ete_RDC[int(time.hour)] += ouverture_rdc[k]
        if int(time.month) in automne:
            heure_automne_RDC[int(time.hour)] += ouverture_rdc[k]
        if int(time.month) in hiver:
            heure_hiver_RDC[int(time.hour)] += ouverture_rdc[k]

plt.figure()
plt.plot(range(24), heure_ete_RDC, label='été')
plt.plot(range(24), heure_hiver_RDC, label='hiver')
plt.plot(range(24), heure_automne_RDC, label='automne')
plt.plot(range(24), heure_printemps_RDC, label='printemps')
plt.xlabel('heures de la journée')
plt.ylabel("nombre d'ouverture de la fenêtre ")
plt.title("Ouvertures de la fenêtre de la cuisine suivant les saisons")
plt.legend()
plt.show()
```

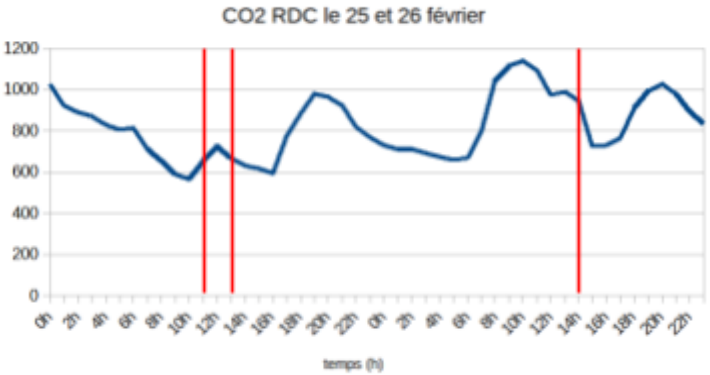
Annexe 5

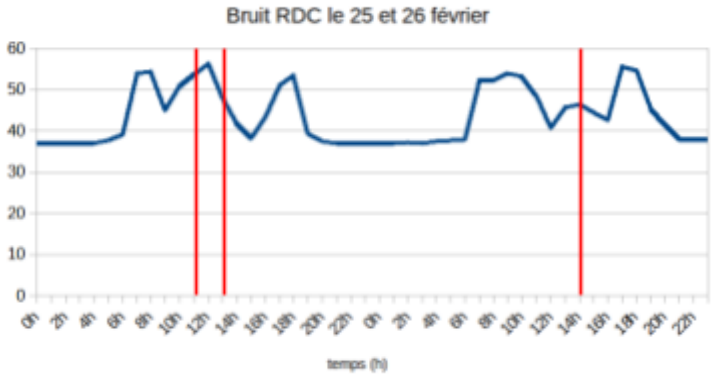
Paramètres de la cuisine suivant les différentes ouvertures pour le 25 et 26 février

le 25/02 : ouverture ch1 à 7h, 20h
le 26/02 : ouverture ch1 8h, 14h, 19h









Annexe 6

Code pour le Decision Tree :

```
data = pd.read_csv('Datas/Dtree/data_destree.csv', sep=',')

# Features
temp_ext = data['Exterieur_Temperature']
hum_ext = data['Exterieur_Humidite']
temp_ch1 = data['Ch1_Temperature']
hum_ch1 = data['Ch1_Humidite']
C02_ch1 = data['bbxco2_ch1']

Features = ['Exterieur_Temperature', 'Exterieur_Humidite', 'Ch1_Temperature',
'Ch1_Humidite', 'bbxco2_ch1']

ouverture_ch1 = data['Ch1_Ouverture']  # label

# ---- création de l'arbre de décision  ---
x_train, x_test, y_train, y_test = train_test_split(data[Features],
data[['Ch1_Ouverture']],
                                                    test_size=0.3, random_state=0)
clf = tree.DecisionTreeClassifier(random_state=0, max_depth=8)
clf = clf.fit(x_train, y_train)
y_pred = clf.predict(x_test)
with open("Datas/Dtree/tree_ouverture.dot", 'w') as f:
    f = tree.export_graphviz(clf, out_file=f)

importance = clf.feature_importances_
print('Importance des features : \n')
for i in range(len(importance)):
    print(f'{Features[i]}, Importance: {round(importance[i], 4)}')

print('\n')
print('Precision: ', round(metrics.precision_score(y_test, y_pred,
average="weighted"), 4))
print('F-Score: ', round(metrics.f1_score(y_test, y_pred, average="weighted"), 4))
print('Accuracy: ', round(metrics.accuracy_score(y_test, y_pred), 4))

print("\nMatrice des critères en fonction des valeurs des labels: ")
print(metrics.classification_report(y_test, y_pred, target_names=['fenetre fermée',
'fenetre ouverte 1fois', 'fenetre ouverte 2fois','fenetre ouverte 3fois', 'fenetre
ouverte 4fois']))


# ---- précision en fonction de la profondeur maximale ---
accuracy = []
f_score = []
precision = []
max_depth = 30

for i in range(2, max_depth):
    clf = tree.DecisionTreeClassifier(random_state=0, max_depth=i)
    clf = clf.fit(x_train, y_train)
    y_pred3 = clf.predict(x_test)

    accur = round(metrics.accuracy_score(y_test, y_pred3), 4)
    f1 = round(metrics.f1_score(y_test, y_pred3, average="weighted"), 4)
    prec = round(metrics.precision_score(y_test, y_pred3, average="weighted"), 4)
```

```
    accuracy.append(accur)
    precision.append(prec)
    f_score.append(f1)

plt.plot(range(2, max_depth), accuracy, label="accuracy", color="red")
plt.plot(range(2, max_depth), f_score, label="f_score", color="blue")
plt.plot(range(2, max_depth), precision, label="precision", color="green")
plt.xlabel("Profondeur max de l'arbre")
plt.legend()
plt.show()

#  ------Etude de différents scénarios avec différentes séries ----
comb = []

for i in range(len(Features) + 1):
    comb_element = [list(k) for k in combinations(Features, i)]
    for k in range(len(comb_element)):
        comb.append(comb_element[k])

comb = comb[1:]  # on enlève le premier élement car il s'agit de la combinaison
vide

accuracy_scenarios = []

for combination in comb:
    x_train4, x_test4, y_train4, y_test4 = train_test_split(data[combination],
data[['Ch1_Ouverture']], test_size=0.3, random_state=0)
    clf = tree.DecisionTreeClassifier(random_state=0, max_depth=8)
    clf = clf.fit(x_train4, y_train4)
    y_pred4 = clf.predict(x_test4)
    accuracy = round(metrics.accuracy_score(y_test4, y_pred4), 4)
    accuracy_scenarios.append(accuracy)
    print(comb.index(combination) + 1, combination, "--> accuracy: ", accuracy)

plt.plot(range(len(comb)), accuracy_scenarios)
plt.xlabel("scenario (n°)")
plt.ylabel("accuracy")
plt.title('accuracy pour tous les scenarios')
plt.show()

print(comb)
```

Annexe 7

Différents scénarios

1 : [['Exterieur_Temperature'],
2 : ['Exterieur_Humidite'],
3 : ['Ch1_Temperature']
4 : ['Ch1_Humidite'],
5 : ['bbxco2_ch1']
6 : ['Exterieur_Temperature', 'Exterieur_Humidite']
7 : ['Exterieur_Temperature', 'Ch1_Temperature'],
8 : ['Exterieur_Temperature', 'Ch1_Humidite'],
9 : ['Exterieur_Temperature', 'bbxco2_ch1'],
10 : ['Exterieur_Humidite', 'Ch1_Temperature'],
11 : ['Exterieur_Humidite', 'Ch1_Humidite'],
12 : ['Exterieur_Humidite', 'bbxco2_ch1'],
13 : ['Ch1_Temperature', 'Ch1_Humidite']
14 : ['Ch1_Temperature', 'bbxco2_ch1'],
15 : ['Ch1_Humidite', 'bbxco2_ch1'],
16 : ['Exterieur_Temperature', 'Exterieur_Humidite', 'Ch1_Temperature'],
17 : ['Exterieur_Temperature', 'Exterieur_Humidite', 'Ch1_Humidite'],
18 : ['Exterieur_Temperature', 'Exterieur_Humidite', 'bbxco2_ch1'],
19 : ['Exterieur_Temperature', 'Ch1_Temperature', 'Ch1_Humidite'],
20 : ['Exterieur_Temperature', 'Ch1_Temperature', 'bbxco2_ch1'],
21 : ['Exterieur_Temperature', 'Ch1_Humidite', 'bbxco2_ch1'],
22 : ['Exterieur_Humidite', 'Ch1_Temperature', 'Ch1_Humidite'],
23 : ['Exterieur_Humidite', 'Ch1_Temperature', 'bbxco2_ch1']
24 : ['Exterieur_Humidite', 'Ch1_Humidite', 'bbxco2_ch1']
25 : ['Ch1_Temperature', 'Ch1_Humidite', 'bbxco2_ch1'],
26 : ['Exterieur_Temperature', 'Exterieur_Humidite', 'Ch1_Temperature', 'Ch1_Humidite'],
27 : ['Exterieur_Temperature', 'Exterieur_Humidite', 'Ch1_Temperature', 'bbxco2_ch1'],
28 : ['Exterieur_Temperature', 'Exterieur_Humidite', 'Ch1_Humidite', 'bbxco2_ch1'],
29 :['Exterieur_Temperature', 'Ch1_Temperature', 'Ch1_Humidite', 'bbxco2_ch1'],
30 : ['Exterieur_Humidite', 'Ch1_Temperature', 'Ch1_Humidite', 'bbxco2_ch1'],
31 : ['Exterieur_Temperature', 'Exterieur_Humidite', 'Ch1_Temperature', 'Ch1_Humidite', 'bbxco2_ch1']]

Annexe 8

Arbre de décision avec une profondeur de 8