Enseignants responsables:
M. N.LAGA et M. P-A. MASSE

Rapport du Projet de

Programmation Informatique

Groupe A: Projet 1

Binôme 34

Rodrigues Simon
Ross Alexandra
Étudiants à l'École des Ingénieurs de la Ville de Paris

01 DÉCEMBRE 2020

Le retour d'expérience sur GitHub

Dans un premier temps, nous avons eu des difficultés à installer Git sur nos ordinateurs. En effet, possédant un Mac, la procédure d'installation était différente. Il était ainsi difficile de comprendre et suivre les explications données aux autres. Après quelques essais, nous sommes parvenus à installer Git (essais sur d'autres ordinateurs par exemple).

Dans un second temps, nous nous sommes créés des comptes GitHub afin de pouvoir partager notre avancement dans la programmation de nos codes: voir les avancements de l'autre, comprendre les codes de l'autre, avoir un regard critique sur les codes et donc peut-être percevoir des erreurs. Installer GitHub Desktop a également permis d'accéder plus facilement à nos dépôts desktop.

Sur le compte AlexandraRoss, dans le respository "Algorithmique_et_Programmation", on y retrouve trois fichiers : l'excel ("pots-32566-EIVP_KM.csv") , "Projet1_PremièrePartie", "Projet1_SecondePartie_Version1" et "Projet1_SecondePartie_VersionFinale" qui rassemblent tous nos codes. Les mêmes informations sont disponibles sur le compte Rodrigues16.
Nos liens GitHub :https://github.com/AlexandraRoss/Algorithmique_et_Programmation
https://github.com/Rodrigues16/algorithmique_programmation

Maîtrise de Python: Choix des Algorithmes

I- Première Partie du projet: Les fonctions nécessaires

Nous avons, au tout début, essayé de comprendre les différentes fonctionnalités des bibliothèques pandas, numpy et matplotlib afin de mieux appréhender le sujet d'informatique.
On a importé le tableau excel afin de pouvoir le manipuler. Par exemple:
df=pd.read_csv((r'/Users/alexandraross/Documents/eivp/informatique/post-32566-EIVP_KM.csv'),sep=';')
print(df)
Les programmes suivants se trouvent dans un fichier "Projet1_PremièrePartie"

- Afficher des courbes montrant l'évolution d'une variable en fonction du temps

Nous avons codé une fonction type pour afficher une courbe montrant l'évolution d'une variable en fonction du temps. En effet, il suffit de rentrer le numéro du capteur voulu (1,2,3,4,5 ou 6) lorsque le programme le demande. Ainsi nous avons obtenu deux fonctions:

- **sep() :** Cette fonction permet de créer des listes pour toutes les variables pour les différents capteurs . En effet, le graphique affiche les 5 courbes du capteur relatives à la température, au CO2, au son, à la luminosité et l'humidité. On demande à l'utilisateur quel capteur il souhaite utiliser à l'aide de k=int(input('Quel capteur:')).

Dans un premier temps nous avons séparé la fonction sep() en fonction de chaque capteur : Les fonctions **sep_capteur1(), sep_capteur2()**, **sep_capteur3()**, **sep_capteur4()**, **sep_capteur5() et sep capteur 6()** affichent respectivement les courbes des capteurs 1, 2, 3, 4, 5 et 6. Le problème avec cette manière de travailler est que, par la suite, le calcul des dérivées dans la *Seconde Partie*

(Partie Projet) est devenu très laborieux. Nous nous sommes très vite perdus et avant donc décidé de procéder autrement.

Dans un second temps, nous avons essayé de faire un programme unique **sep().** Cette seule et unique fonction permet d'alléger les calculs mais aussi de ne pas se perdre dans un nombre trop important de fonctions. Enfin, nous avons créé une fonction **sep_reduite()** qui est expliquée dans la seconde partie du projet dans la recherche d'anomalie.

- **inter(deb,fin):** Cette fonction permet de tracer la courbe entre deux temps , par exemple le début et la fin. Elle permet donc de spécifier l'intervalle de temps que l'on souhaite étudier.

- Afficher les valeurs statistiques sur la courbe

Nous avons ,dans un premier temps, utilisé les fonctions déjà disponibles dans la bibliothèque afin d'avoir une idée des résultats à obtenir. Puis, dans un second temps, nous avons programmé nos propres fonctions.

- **maximum(l):** Cette fonction permet de retourner le maximum, c'est-à-dire la plus grande valeur des éléments d'un tableau. Elle prend comme argument une liste. On crée une variable max à laquelle on affecte le premier élément de la liste. On initialise une variable L=sep_reduite()[0] (la première liste renvoyée par **sep_reduite(),** c'est-à-dire la variable voulue) et une variable T=sep_reduite()[1] (le temps renvoyée par la fonction **sep_reduite()**). On compare chaque élément de la liste (en parcourant le tableau à l'aide d'une boucle for suivi d'un if) à notre variable max. Ainsi, si un élément est supérieur ou égal à max, on l'affecte à max. Enfin, on retourne max.

- **minimum(l):** Cette fonction permet de retourner le minimum, c'est-à-dire la plus petite valeur, des éléments d'un tableau. Elle prend comme argument une liste. On crée une variable min à laquelle on affecte le premier élément de la liste. On initialise une variable L=sep_reduite()[0] (la première liste renvoyée par **sep_reduite(),** c'est-à-dire la variable voulue) et une variable T=sep_reduite()[1] (le temps renvoyée par la fonction **sep_reduite()**). On compare chaque élément de la liste (en parcourant le tableau à l'aide d'une boucle for suivi d'un if) à notre variable min. Ainsi, si un élément est inférieur ou égal à min, on l'affecte à min. Enfin, on retourne min.

- **moyenne(l):** Cette fonction permet de calculer la moyenne. Elle prend comme argument une liste. On crée une variable initiale moyen (étape d'initialisation). On initialise également une variable L=sep_reduite()[0] (la première liste renvoyée par **sep_reduite(),** c'est-à-dire la variable voulue). À l'aide d'une boucle for, on calcule la somme des éléments de la liste. Ensuite, on divise cette somme par la longueur de la liste (c'est-à -dire le nombre d'éléments de la liste). Enfin, on retourne moyen.

- **mediane(l):** Cette fonction permet de calculer la médiane (c'est-à-dire la valeur centrale: c'est la valeur qui partage une série de nombre en deux parties de même effectif). Elle prend comme argument une liste. On trie la liste à l'aide de la fonction **quickSort(l)**. On crée une variable initiale median. On compare différents cas à l'aide de if : si le résultat de la division euclidienne de la longueur de la liste par 2 est nulle (c'est-à-dire que le nombre d'éléments

est un nombre pair) ou si ce n'est pas le cas (c'est-à-dire que le nombre d'éléments est un nombre impair). Enfin, on retourne mediane.

- **variance(l):** Cette fonction permet de calculer la variance. Il existe différentes manières de calculer la variance : la formule de la variance et la formule de König-Huygens. Nous avons essayé de programmer les deux formules et avons décidé de garder celle de la variance. Elle prend comme argument une liste.
  On pose m une variable à laquelle on affecte la fonction **moyenne(l)** et on crée une variable initiale v. À l'aide d'une boucle for, on parcourt la liste de longueur len(l) et l'on calcul la somme des écarts à la moyenne. On retourne cette somme des écarts à la moyenne divisée par len(l). On retourne ainsi la variance.

- **ecart_type(l):** Cette fonction permet de calculer l'écart-type. Elle prend comme argument une liste. L'écart-type est par définition la racine carrée de la variance. On crée une variable var à laquelle on affecte la fonction **variance(l)**. On retourne la racine carrée de var.

Afin de calculer la fonction **mediane(l)** mentionnée au-dessus, nous avons utilisé la fonction **quickSort(arr,bas,haut)**. Pour créer cette fonction, il a fallu créer une fonction **partition(arr,bas,haut)**.

- **partition(arr,bas,haut):** Cette fonction permet de partitionner les valeurs. On utilise une valeur pivot. Toutes les valeurs inférieures à cette valeur pivot vont se mettre dans une liste et les valeurs supérieures vont se mettre dans une autre liste. En effet, cette fonction divise le tableau donné autour du pivot choisi.

- **quickSort(arr,bas,haut)**: Cette fonction reprend la fonction **partition(arr,bas,haut).** Elle trie maintenant de manière récursive chacune des petites listes créées par la fonction **partition(arr,bas,haut).**

- Calculer l'indice "humidex"

Afin de calculer l'indice "humidex", nous avons utilisé la formule trouvée sur Wikipédia. Il était ainsi nécessaire de créer une fonction supplémentaire **rose()** afin de faciliter les calculs.

La formule pour calculer l'humidex est la suivante[2] :

$$\text{Humidex} = T_{\text{air}} + 0.5555\left[6.11 \times e^{5417.7530\left(\frac{1}{273.16} - \frac{1}{273.15+T_{\text{rosee}}}\right)} - 10\right]$$

où

T_{air} est la température de l'air (degré Celsius)
e = 2,71828
T_{rosee} est le point de rosée (degré Celsius)

Sachant que l'humidité (Phi) et la température (T) sont données, on peut calculer la température de rosée (Tr) à l'aide de la formule suivante:

$$T_{\text{r}} = \frac{b\,\alpha(T,\varphi)}{a - \alpha(T,\varphi)} \qquad \frac{a\,T}{b+T} + \ln\varphi$$

avec aplha =

- **rose():** Cette fonction permet de calculer la température de rosée Tr en utilisant notre fonction **sep()** dans l'initialisation des variables utiles à la fonction. On initialise deux valeurs a et b (données sur wikipédia), et une liste vide Tr. On associe à T la colonne de la température et à H la colonne de l'humidité. À l'aide d'une boucle for, on calcule l'alpha pour chaque ligne du tableau et on trouve ainsi la température de rosée pour chaque ligne du tableau.
- **humidex():** Cette fonction permet de calculer l'indice 'humidex". Maintenant que l'on possède la température de rosée Tr grâce à la fonction **rose()**, il est possible d'appliquer la formule de Wikipédia plus facilement.

- Calcul de l'indice de corrélation entre un couple de variables

La formule de la covariance :

cov(X, Y) = (sum (x - mean(X)) * (y - mean(Y))) * 1/(n-1)

avec mean: la moyenne

La formule du coefficient de corrélation:

correlation coefficient = covariance(X, Y) / (stdv(X) * stdv(Y))

avec stdv: la racine carrée de la multiplication de la variance de X par la variance de Y

Ainsi, on crée deux fonctions pour calculer l'indice de corrélation entre un couple de variables:

- **cov(X,Y):** Cette fonction permet de calculer la covariance. Elle prend comme arguments deux listes X et Y.
- **cor(X,Y):** Cette fonction permet de calculer le coefficient de corrélation. Elle utilise ainsi la fonction **cov(X,Y)** définie juste avant. Elle prend comme arguments deux listes X et Y.

Nous avons essayé de convertir les chaînes de caractères en objet date afin de calculer la corrélation. Notre fonction fonctionne pour des petites listes, cependant à une plus grande échelle, notre programme ne fonctionne pas ou du moins prend trop de temps à fonctionner. Le principe était de spécifier le format de la date en chaine de caractère.

II- Seconde Partie du projet: Le Projet 1

"Trouvez-vous des anomalies dans les données, que pouvez-vous en conclure? Proposer et implémenter un algorithme permettant de relever les anomalies automatiquement et de les montrer sur les courbes.
Bonus: Trouvez automatiquement les périodes horaires d'occupations des bureaux."

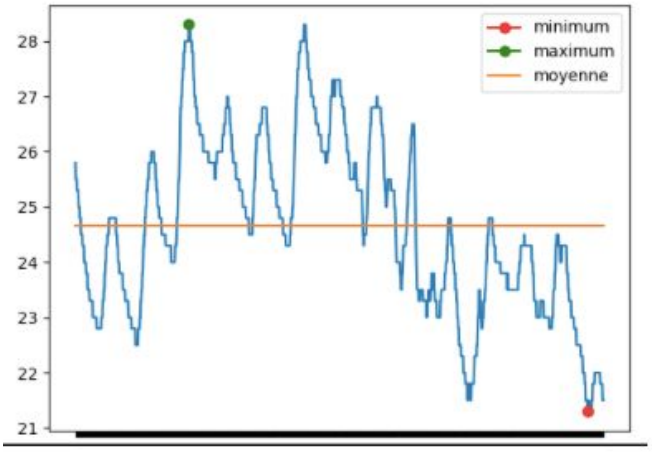
Afin de détecter des anomalies par rapport à l'évolution relative des courbes, il est nécessaire de détecter les variations très importantes, c'est-à-dire les variations incohérentes.
Pour cela, il faut calculer la moyenne et l'écart-type. Ensuite, deux approches sont possibles. La moins précise (c'est-à-dire celle qui ne fonctionnera pas toujours) est de regarder toutes les valeurs

qui sont supérieures ou inférieures à un certain écart-type: elles sont anormales. Une manière plus précise de recherche d'anomalie est de calculer la dérivée de la courbe d'évolution. Nous avons choisi la seconde approche.

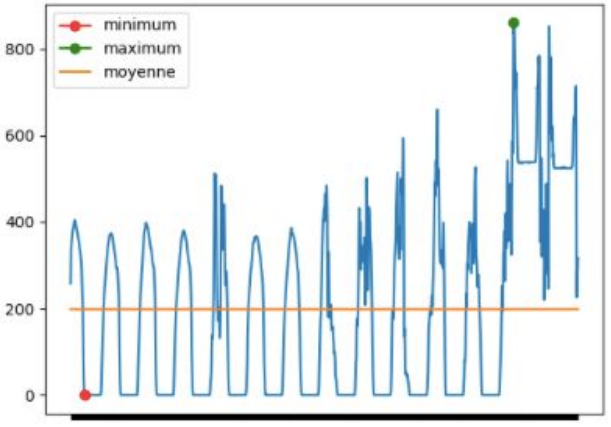
- Choix de l'affichage des courbes

Nous avons décidé d'afficher sur le graphique l'évolution des courbes de toutes les variables pour un seul capteur plutôt que d'afficher l'évolution d'une variable en fonction de tous les capteurs. Ce choix nous paraissait le plus cohérent. Ainsi, nous devrions obtenir un graphique par capteur.

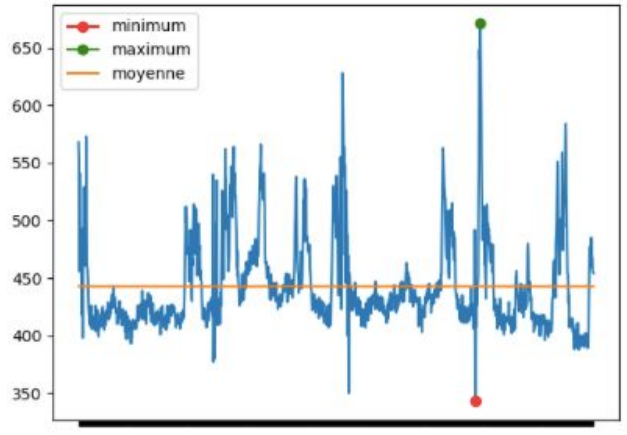
Capteur 1: Température



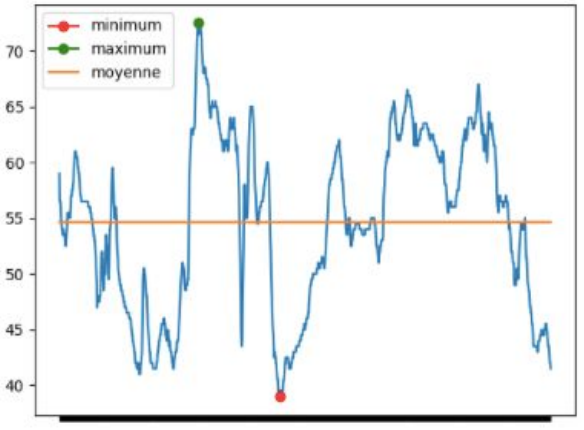
Capteur 2 lum



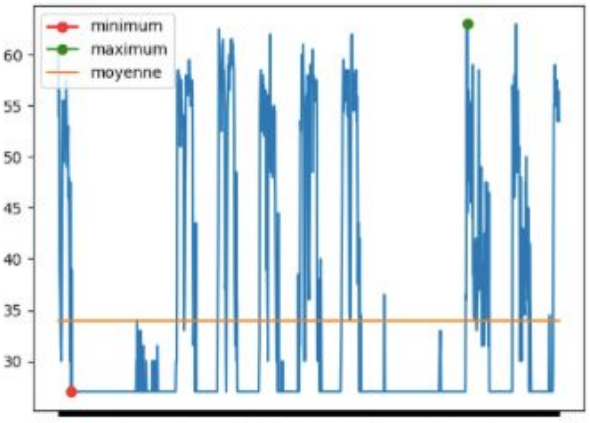
Capteur 3 CO2



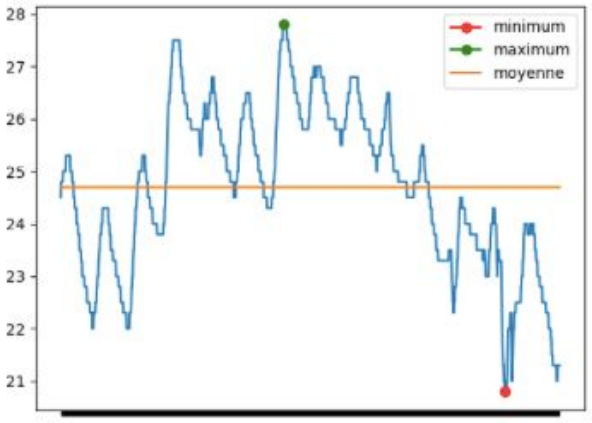
Capteur 4 humidité



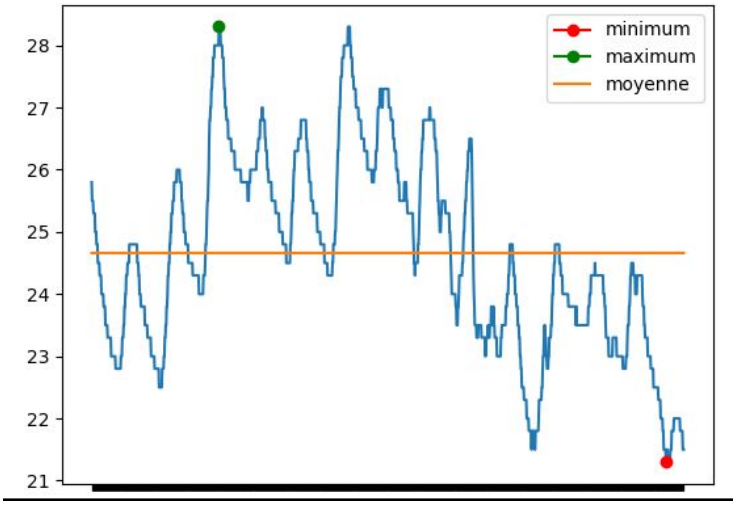
Capteur 5 bruit



Capteur 6 température



- Affichage des valeurs statistiques sur une courbe



Capteur 1: Évolution de la variable de la température et affichage de valeurs statistiques (minimum, maximum et moyenne)

Méthode: Calcul des dérivée des variables en fonction du temps

➢ Premier essai: Nous avons calculé les dérivés de chaque variable pour chaque capteur avec la première version de la fonction **sep()**. C'est-à-dire les fonctions **sep_capteur1(), sep_capteur2(), sep_capteur3(), sep_capteur4() et sep_capteur5(), sep_capteur6().** Nous expliquons cette démarche ci-dessous bien qu'elle ne soit pas la version finale. Nous ne mettons donc pas les résultats obtenus dans cette partie. Ceux-ci sont plus loin dans le rapport dans la partie "Second essai".

Les programmes suivant se trouvent dans un fichier "Projet1_SecondePartie_Version 1"

- Le calcul de la dérivée de la température en fonction du temps

Le programme de calcul de la dérivée de la courbe d'évolution de la température est le même pour chaque capteur sauf lorsque l'on modifie la variable A à laquelle on a associé la fonction **sep_capteur1()** (ou **sep_capteur2()**, **sep_capteur3()**, **sep_capteur4()**, **sep_capteur5()**, **sep_capteur6()** pour respectivement les capteurs 2,3,4,5 et 6) qui reprend la liste des températures du capteur. Pour calculer la dérivée, on calcule la différence entre la valeur de la température à un instant t et celle à un instant t-1.
delta_temperature_capteur1() prend A=sep_capteur1()[0])
delta_temperature_capteur2() prend A=sep_capteur2()[0])
Et ainsi de suite pour les capteurs 3,4,5 et 6.

Le principe du calcul de la dérivée de la courbe d'évolution des autres variables en fonction du temps est le même que celui du calcul de la dérivée de la courbe d'évolution de la température en fonction du temps. Nous ne détaillons donc pas à nouveau.

- Le calcul de la dérivée du bruit en fonction du temps

delta_bruit_capteur1() prend A=sep_capteur1()[2])
delta_bruit_capteur2() prend A=sep_capteur2()[2])
Et ainsi de suite pour les capteurs 3,4,5 et 6.

- Le calcul de la dérivée de l'humidité en fonction du temps

delta_humidité_capteur1() prend A=sep_capteur1()[3])
delta_humidité_capteur2() prend A=sep_capteur2()[3])
Et ainsi de suite pour les capteurs 3,4,5 et 6.

- Le calcul de la dérivée de la luminosité en fonction du temps

delta_luminosité_capteur1() prend A=sep_capteur1()[4])
delta_luminosité_capteur2() prend A=sep_capteur2()[4])
Et ainsi de suite pour les capteurs 3,4,5 et 6.

- Le calcul de la dérivée du CO2 en fonction du temps

delta_CO2_capteur1() prend A=sep_capteur1()[5])
delta_CO2_capteur2() prend A=sep_capteur2()[5])
Et ainsi de suite pour les capteurs 3,4,5 et 6.

➢ Second essai: Nous avons calculé des dérivées de chaque variable pour chaque capteur avec la seconde version de la fonction **sep(),** c'est-à-dire la version finale. Il est maintenant plus simple d'analyser les résultats (en effet, on ne se perd plus dans les nombreuses fonctions). On peut donc, dans cette partie, mettre les résultats obtenus.

Les programmes suivant se trouvent dans un fichier "Projet1_SecondePartie_VersionFinale"

Pour le calcul de la dérivée (la fonction delta) nous avons voulu utiliser les fonctions suivantes : **separation(d)** , **date(d)**, **distance(d)**, **liste(d).** Cependant, en vue des nombreuses données, celles-ci ralentissaient notre programme. Ainsi, il était impossible d'obtenir des résultats, nous avons donc décidé de ne plus nous en servir.

- Le calcul de la dérivée d'une variable en fonction du temps

- **delta():** Le programme de calcul de la dérivée de la courbe d'évolution d'une variable choisie (température, humidité, CO2, luminosité ou bruit) en fonction du temps utilise la fonction **sep()** (c'est elle qui nous permet de choisir quel capteur on veut étudier). Pour calculer la dérivée, on calcule la différence entre la une valeur de la température à un instant t et celle à un instant t-1.

- La recherche d'anomalies

Nous avons également défini de nouvelles fonctions pour trouver les anomalies.

- **quantile()**: Cette fonction calcule le quantile de la liste. On initialise une variable L=delta() et on prend un intervalle de confiance compris entre 0,01 et 0,09.

- **an()**: Cette fonction recherche les anomalies dans la fonction **delta()** à l'aide d'une variable d'initialisation L=delta() et d'une autre variable d'initialisation T=sep_reduite()[1] (la deuxième liste que renvoie la fonction **sep_reduite()**). Cette fonction parcourt les éléments de la liste delta et elle les compare avec les quantiles trouvés (valeurs au-dessus ou en dessous des quantiles). Ces valeurs traduisent une trop grande variation, ce qui pour nous, correspond à une anomalie.

La fonction **sep()** initiale permettait d'afficher toutes les courbes des variables pour un capteur choisi. Or dans cette partie, nous avons uniquement travaillé sur une variable d'un capteur à la fois pour obtenir les anomalies de chaque variable de chaque capteur.

- **sep_reduite()**: Cette fonction permet, comme la fonction **sep()**, de créer des listes pour toutes les variables pour les différents capteurs . Cependant, celle-ci ne permet pas d'afficher les courbes. On demande à l'utilisateur quelle variable il souhaite étudier à l'aide de la fonction input un nombre qui correspond à une variable selon la légende donnée: v=int(input("Choisissez un nombre selon la variable 1=temp,2=lum,3=CO2,4=humidity et 5=noise:",)). On demande également quel capteur il souhaite étudier à l'aide de la variable k=int(input('Quel capteur:')).

- **legende():** Cette fonction renvoie la colonne du tableau de la variable désirée par l'utilisateur.

Les différentes anomalies relevées par nos fonctions pour les 6 capteurs se trouvent sur les pages suivantes dans des tableaux.

On constate de trop grandes variations pour le capteur 1 aux dates:

Anomalies Température	Anomalies de bruit	Anomalies humidité	Anomalies luminosité	Anomalie CO2
['2019-08-14 12:02:52+02:00', '2019-08-20 17:17:31+02:00', '2019-08-20 17:32:31+02:00', '2019-08-20 17:47:31+02:00', '2019-08-20 18:02:27+02:00']	['2019-08-14 08:17:53+02:00', '2019-08-14 10:17:57+02:00', '2019-08-14 12:47:52+02:00', '2019-08-14 18:02:51+02:00', '2019-08-15 12:47:49+02:00', '2019-08-15 13:32:48+02:00', '2019-08-15 19:17:48+02:00', '2019-08-15 19:32:49+02:00', '2019-08-16 10:32:45+02:00', '2019-08-16 13:32:49+02:00', '2019-08-16 14:32:52+02:00', '2019-08-16 18:17:43+02:00', '2019-08-17 08:17:44+02:00', '2019-08-17 10:17:40+02:00', '2019-08-17 13:47:40+02:00', '2019-08-17 19:17:39+02:00', '2019-08-18 13:17:36+02:00', '2019-08-18 17:32:43+02:00', '2019-08-21 11:17:25+02:00', '2019-08-22 16:32:20+02:00', '2019-08-23 12:47:17+02:00', '2019-08-23 13:32:17+02:00', '2019-08-23 18:32:18+02:00', '2019-08-24 08:47:16+02:00', '2019-08-24 12:32:13+02:00', '2019-08-24 13:02:12+02:00', '2019-08-25 12:02:09+02:00', '2019-08-25 13:02:08+02:00']	['2019-08-13 20:32:55+02:00', '2019-08-15 04:17:58+02:00', '2019-08-15 04:32:58+02:00', '2019-08-15 04:48:02+02:00', '2019-08-16 15:32:44+02:00', '2019-08-16 15:47:44+02:00', '2019-08-16 16:17:43+02:00', '2019-08-16 16:47:43+02:00', '2019-08-16 17:02:44+02:00', '2019-08-16 17:47:43+02:00', '2019-08-16 18:02:43+02:00', '2019-08-17 12:02:40+02:00', '2019-08-19 12:02:32+02:00', '2019-08-20 17:17:31+02:00', '2019-08-20 17:32:31+02:00', '2019-08-20 17:47:31+02:00', '2019-08-20 21:17:36+02:00', '2019-08-21 15:17:24+02:00']	['2019-08-15 11:02:49+02:00', '2019-08-15 12:17:48+02:00', '2019-08-15 15:02:48+02:00', '2019-08-19 12:02:32+02:00', '2019-08-20 16:47:28+02:00', '2019-08-21 15:17:24+02:00', '2019-08-23 10:32:17+02:00', '2019-08-23 12:32:16+02:00', '2019-08-23 13:02:17+02:00', '2019-08-23 16:02:20+02:00', '2019-08-23 16:17:20+02:00', '2019-08-23 17:17:24+02:00', '2019-08-23 18:32:18+02:00', '2019-08-23 18:47:16+02:00', '2019-08-24 10:47:23+02:00', '2019-08-24 12:02:16+02:00', '2019-08-24 13:32:13+02:00', '2019-08-24 15:02:12+02:00', '2019-08-24 16:17:12+02:00', '2019-08-24 16:32:18+02:00', '2019-08-24 16:47:12+02:00', '2019-08-24 18:02:12+02:00', '2019-08-24 18:17:14+02:00', '2019-08-24 18:32:12+02:00', '2019-08-25 10:02:10+02:00', '2019-08-25 11:02:09+02:00', '2019-08-25 16:02:08+02:00']	['2019-08-14 09:17:54+02:00', '2019-08-14 12:47:52+02:00', '2019-08-15 12:32:48+02:00', '2019-08-15 13:47:48+02:00', '2019-08-16 10:47:44+02:00', '2019-08-16 13:17:48+02:00', '2019-08-17 10:17:40+02:00', '2019-08-17 14:32:40+02:00', '2019-08-17 17:47:39+02:00', '2019-08-17 18:02:40+02:00', '2019-08-17 18:32:39+02:00', '2019-08-18 13:17:36+02:00', '2019-08-18 14:02:36+02:00', '2019-08-18 15:17:36+02:00', '2019-08-21 09:32:24+02:00', '2019-08-22 09:02:21+02:00', '2019-08-22 09:17:21+02:00', '2019-08-22 09:32:24+02:00', '2019-08-22 10:17:20+02:00', '2019-08-22 10:32:21+02:00', '2019-08-22 10:47:20+02:00', '2019-08-22 14:02:21+02:00', '2019-08-23 17:32:25+02:00', '2019-08-24 09:02:16+02:00', '2019-08-24 10:32:25+02:00', '2019-08-24 13:02:12+02:00', '2019-08-25 13:02:08+02:00']
5 anomalies	28 anomalies	18 anomalies	27 anomalies	27 anomalies

On constate de trop grandes variations pour le capteur 2 aux dates:

Anomalies Température	Anomalies de bruit	Anomalies humidité	Anomalies luminosité	Anomalie CO2
['2019-08-14 12:05:08+02:00']	['2019-08-14 08:20:03+02:00', '2019-08-15 08:04:59+02:00', '2019-08-15 13:04:59+02:00', '2019-08-15 13:34:59+02:00', '2019-08-16 13:04:56+02:00', '2019-08-17 10:19:53+02:00', '2019-08-18 18:04:52+02:00', '2019-08-18 18:19:54+02:00', '2019-08-21 12:04:39+02:00', '2019-08-21 12:49:39+02:00', '2019-08-21 13:19:40+02:00', '2019-08-21 13:34:41+02:00', '2019-08-21 13:49:41+02:00', '2019-08-21 14:04:42+02:00', '2019-08-21 17:49:41+02:00', '2019-08-21 18:04:38+02:00', '2019-08-22 19:04:35+02:00', '2019-08-22 19:19:35+02:00', '2019-08-22 19:34:38+02:00', '2019-08-23 08:49:33+02:00', '2019-08-23 15:34:32+02:00', '2019-08-23 15:49:32+02:00', '2019-08-23 18:49:40+02:00', '2019-08-24 08:49:30+02:00', '2019-08-24 11:04:35+02:00', '2019-08-24 13:49:29+02:00', '2019-08-24 14:19:32+02:00', '2019-08-24 15:04:29+02:00']	['2019-08-11 11:50:13+02:00', '2019-08-12 13:20:09+02:00', '2019-08-13 20:35:04+02:00', '2019-08-13 20:50:04+02:00', '2019-08-15 04:05:02+02:00', '2019-08-15 04:20:03+02:00', '2019-08-15 04:35:04+02:00', '2019-08-15 04:50:05+02:00', '2019-08-16 15:05:04+02:00', '2019-08-16 15:35:02+02:00', '2019-08-16 15:50:03+02:00', '2019-08-16 16:05:04+02:00', '2019-08-16 16:20:04+02:00', '2019-08-16 16:49:55+02:00', '2019-08-16 17:04:55+02:00', '2019-08-16 17:49:55+02:00', '2019-08-16 18:04:55+02:00', '2019-08-16 20:49:55+02:00', '2019-08-17 01:04:53+02:00', '2019-08-17 12:04:52+02:00', '2019-08-20 17:34:42+02:00']	['2019-08-15 12:19:59+02:00', '2019-08-15 15:04:59+02:00', '2019-08-15 15:19:59+02:00', '2019-08-18 13:49:55+02:00', '2019-08-18 14:34:49+02:00', '2019-08-19 15:19:45+02:00', '2019-08-20 12:49:43+02:00', '2019-08-20 14:49:42+02:00', '2019-08-20 16:34:42+02:00', '2019-08-20 16:49:42+02:00', '2019-08-21 15:04:46+02:00', '2019-08-22 16:19:35+02:00', '2019-08-23 13:04:33+02:00', '2019-08-23 14:34:33+02:00', '2019-08-23 15:34:32+02:00', '2019-08-23 16:04:32+02:00', '2019-08-23 17:19:38+02:00', '2019-08-24 10:49:34+02:00', '2019-08-24 12:04:38+02:00', '2019-08-24 13:34:29+02:00', '2019-08-24 14:49:29+02:00', '2019-08-24 15:04:29+02:00', '2019-08-24 15:49:28+02:00', '2019-08-24 16:19:29+02:00', '2019-08-24 16:34:29+02:00', '2019-08-24 16:49:32+02:00', '2019-08-24 18:04:29+02:00', '2019-08-25 11:04:26+02:00']	['2019-08-11 11:50:13+02:00', '2019-08-11 13:05:12+02:00', '2019-08-12 06:20:09+02:00', '2019-08-14 09:20:03+02:00', '2019-08-14 10:20:03+02:00', '2019-08-14 13:50:02+02:00', '2019-08-14 14:05:02+02:00', '2019-08-15 14:04:59+02:00', '2019-08-18 12:34:49+02:00', '2019-08-18 14:19:48+02:00', '2019-08-18 15:19:49+02:00', '2019-08-21 09:19:46+02:00', '2019-08-21 12:34:39+02:00', '2019-08-21 12:49:39+02:00', '2019-08-21 14:04:42+02:00', '2019-08-22 09:04:52+02:00', '2019-08-22 09:19:37+02:00', '2019-08-22 09:34:37+02:00', '2019-08-22 10:34:36+02:00', '2019-08-22 11:04:36+02:00', '2019-08-22 11:19:36+02:00', '2019-08-22 13:04:36+02:00', '2019-08-23 17:34:36+02:00', '2019-08-24 11:19:36+02:00', '2019-08-24 13:04:32+02:00', '2019-08-24 18:19:31+02:00', '2019-08-25 08:49:26+02:00']
1 anomalie	28 anomalies	21 anomalies	28 anomalies	27 anomalies

On constate de trop grandes variations pour le capteur 3 aux dates:

Anomalies Température	Anomalies de bruit	Anomalies humidité	Anomalies luminosité	Anomalie CO2
['2019-08-19 11:35:28+02:00']	['2019-08-11 18:20:52+02:00', '2019-08-11 19:20:51+02:00', '2019-08-11 19:35:52+02:00', '2019-08-14 10:05:42+02:00', '2019-08-14 10:20:42+02:00', '2019-08-14 14:20:44+02:00', '2019-08-14 14:50:42+02:00', '2019-08-15 12:50:38+02:00', '2019-08-17 08:50:38+02:00', '2019-08-17 10:20:31+02:00', '2019-08-17 13:50:30+02:00', '2019-08-18 11:05:26+02:00', '2019-08-18 11:20:27+02:00', '2019-08-18 16:50:25+02:00', '2019-08-21 12:50:18+02:00', '2019-08-21 13:35:16+02:00', '2019-08-21 17:50:13+02:00', '2019-08-22 08:35:20+02:00', '2019-08-22 18:50:10+02:00', '2019-08-23 09:35:08+02:00', '2019-08-23 17:20:09+02:00', '2019-08-23 19:05:15+02:00', '2019-08-23 19:20:19+02:00', '2019-08-24 08:50:07+02:00', '2019-08-24 12:50:03+02:00', '2019-08-24 13:50:03+02:00']	['2019-08-12 13:20:49+02:00', '2019-08-13 20:20:44+02:00', '2019-08-13 20:35:44+02:00', '2019-08-15 04:05:41+02:00', '2019-08-15 04:20:42+02:00', '2019-08-15 04:35:42+02:00', '2019-08-16 15:35:40+02:00', '2019-08-16 15:50:41+02:00', '2019-08-16 16:50:33+02:00', '2019-08-16 17:05:33+02:00', '2019-08-16 17:50:33+02:00', '2019-08-16 18:05:36+02:00', '2019-08-16 21:05:33+02:00', '2019-08-20 17:20:18+02:00', '2019-08-24 10:35:07+02:00']	['2019-08-11 11:50:53+02:00', '2019-08-15 11:05:38+02:00', '2019-08-15 11:20:38+02:00', '2019-08-16 09:50:34+02:00', '2019-08-17 14:05:30+02:00', '2019-08-18 11:50:26+02:00', '2019-08-19 10:50:26+02:00', '2019-08-19 11:20:28+02:00', '2019-08-19 11:50:29+02:00', '2019-08-19 12:05:29+02:00', '2019-08-19 12:20:31+02:00', '2019-08-20 10:35:19+02:00', '2019-08-21 12:20:15+02:00', '2019-08-22 09:50:12+02:00', '2019-08-22 10:35:11+02:00', '2019-08-22 10:50:11+02:00', '2019-08-22 12:35:11+02:00', '2019-08-22 13:20:11+02:00', '2019-08-23 10:50:07+02:00', '2019-08-23 11:05:07+02:00', '2019-08-23 12:35:09+02:00', '2019-08-24 10:20:09+02:00', '2019-08-24 10:50:08+02:00', '2019-08-24 12:05:12+02:00', '2019-08-24 16:35:02+02:00', '2019-08-25 10:05:00+02:00', '2019-08-25 10:20:00+02:00', '2019-08-25 11:05:03+02:00']	['2019-08-11 12:05:53+02:00', '2019-08-11 12:20:56+02:00', '2019-08-11 14:20:53+02:00', '2019-08-11 14:35:55+02:00', '2019-08-11 16:36:02+02:00', '2019-08-11 17:05:52+02:00', '2019-08-15 03:20:39+02:00', '2019-08-15 03:35:39+02:00', '2019-08-15 05:20:45+02:00', '2019-08-15 05:35:46+02:00', '2019-08-15 05:50:47+02:00', '2019-08-15 06:05:48+02:00', '2019-08-18 15:05:29+02:00', '2019-08-18 15:20:26+02:00', '2019-08-18 15:50:25+02:00', '2019-08-18 17:20:27+02:00', '2019-08-18 17:35:28+02:00', '2019-08-18 17:50:28+02:00', '2019-08-18 19:20:34+02:00', '2019-08-18 19:35:37+02:00', '2019-08-18 19:50:35+02:00', '2019-08-18 20:20:25+02:00', '2019-08-22 06:35:13+02:00', '2019-08-22 06:50:14+02:00', '2019-08-22 07:05:14+02:00', '2019-08-22 08:50:20+02:00', '2019-08-22 09:20:12+02:00', '2019-08-24 15:35:02+02:00']
1 anomalie	26 anomalies	15 anomalies	28 anomalies	28 anomalies

On constate de trop grandes variations pour le capteur 4 aux dates:

Anomalies Température	Anomalies de bruit	Anomalies humidité	Anomalies luminosité	Anomalie CO2
Aucune anomalie détectée	['2019-08-14 08:07:23+02:00', '2019-08-14 14:52:21+02:00', '2019-08-14 15:22:22+02:00', '2019-08-14 15:37:21+02:00', '2019-08-15 08:07:19+02:00', '2019-08-15 19:07:21+02:00', '2019-08-15 19:22:18+02:00', '2019-08-17 09:37:23+02:00', '2019-08-17 17:22:11+02:00', '2019-08-18 08:07:09+02:00', '2019-08-18 19:37:16+02:00', '2019-08-21 18:06:57+02:00', '2019-08-22 09:36:54+02:00', '2019-08-22 11:51:55+02:00', '2019-08-22 13:21:54+02:00', '2019-08-22 15:21:54+02:00', '2019-08-22 19:21:56+02:00', '2019-08-22 19:36:53+02:00', '2019-08-23 12:51:50+02:00', '2019-08-23 15:21:53+02:00', '2019-08-23 15:36:50+02:00', '2019-08-23 17:51:55+02:00', '2019-08-23 18:51:57+02:00', '2019-08-24 08:51:47+02:00', '2019-08-24 17:21:46+02:00', '2019-08-24 17:36:46+02:00']	['2019-08-11 11:52:33+02:00', '2019-08-12 13:22:29+02:00', '2019-08-12 23:22:28+02:00', '2019-08-13 20:22:25+02:00', '2019-08-13 20:37:24+02:00', '2019-08-13 20:52:25+02:00', '2019-08-15 04:07:22+02:00', '2019-08-15 04:22:23+02:00', '2019-08-15 04:37:23+02:00', '2019-08-16 14:52:19+02:00', '2019-08-16 15:37:22+02:00', '2019-08-16 15:52:23+02:00', '2019-08-16 16:52:14+02:00', '2019-08-16 17:07:18+02:00', '2019-08-16 17:52:15+02:00', '2019-08-16 18:07:15+02:00', '2019-08-16 18:37:14+02:00', '2019-08-16 20:52:14+02:00', '2019-08-16 21:07:14+02:00', '2019-08-17 00:52:14+02:00', '2019-08-17 12:22:12+02:00', '2019-08-20 17:22:01+02:00']	['2019-08-19 10:52:08+02:00', '2019-08-19 11:52:12+02:00', '2019-08-19 12:07:12+02:00', '2019-08-19 14:07:04+02:00', '2019-08-19 14:37:05+02:00', '2019-08-19 14:52:04+02:00', '2019-08-20 14:37:01+02:00', '2019-08-20 14:52:01+02:00', '2019-08-21 11:51:58+02:00', '2019-08-21 13:36:59+02:00', '2019-08-21 15:07:04+02:00', '2019-08-22 12:21:54+02:00', '2019-08-22 12:36:54+02:00', '2019-08-22 13:21:54+02:00', '2019-08-22 14:21:53+02:00', '2019-08-22 14:51:54+02:00', '2019-08-23 12:36:51+02:00', '2019-08-23 13:06:50+02:00', '2019-08-23 13:36:50+02:00', '2019-08-23 16:06:50+02:00', '2019-08-24 10:51:52+02:00', '2019-08-24 12:06:56+02:00', '2019-08-24 13:36:47+02:00', '2019-08-24 15:21:47+02:00', '2019-08-24 16:36:46+02:00', '2019-08-25 10:06:44+02:00', '2019-08-25 11:06:44+02:00']	['2019-08-11 11:52:33+02:00', '2019-08-11 12:52:34+02:00', '2019-08-11 16:37:36+02:00', '2019-08-11 17:07:37+02:00', '2019-08-14 08:07:23+02:00', '2019-08-14 13:52:21+02:00', '2019-08-16 10:07:15+02:00', '2019-08-17 10:22:12+02:00', '2019-08-18 11:52:09+02:00', '2019-08-18 12:37:08+02:00', '2019-08-18 15:22:08+02:00', '2019-08-21 08:51:58+02:00', '2019-08-22 09:07:05+02:00', '2019-08-22 09:21:55+02:00', '2019-08-22 10:36:55+02:00', '2019-08-22 10:51:54+02:00', '2019-08-22 11:06:54+02:00', '2019-08-22 14:21:53+02:00', '2019-08-22 14:36:54+02:00', '2019-08-23 08:36:51+02:00', '2019-08-23 17:36:53+02:00', '2019-08-24 09:21:47+02:00', '2019-08-24 11:36:57+02:00', '2019-08-24 15:51:47+02:00', '2019-08-24 17:21:46+02:00', '2019-08-24 17:36:46+02:00', '2019-08-24 18:21:50+02:00', '2019-08-24 18:36:46+02:00']
0 anomalie	26 anomalies	22 anomalies	27 anomalies	28 anomalies

On constate de trop grandes variations pour le capteur 5 aux dates:

Anomalies Température	Anomalies de bruit	Anomalies humidité	Anomalies luminosité	Anomalie CO2
['2019-08-11 11:45:45+02:00', '2019-08-17 12:30:24+02:00', '2019-08-17 14:30:27+02:00']	['2019-08-11 18:30:44+02:00', '2019-08-14 08:45:35+02:00', '2019-08-14 12:45:43+02:00', '2019-08-14 19:45:33+02:00', '2019-08-15 08:45:32+02:00', '2019-08-15 12:30:34+02:00', '2019-08-15 18:45:30+02:00', '2019-08-15 19:00:33+02:00', '2019-08-16 08:45:28+02:00', '2019-08-16 13:30:27+02:00', '2019-08-16 19:45:26+02:00', '2019-08-17 08:15:31+02:00', '2019-08-17 10:30:25+02:00', '2019-08-17 11:15:25+02:00', '2019-08-17 18:00:23+02:00', '2019-08-18 13:45:20+02:00', '2019-08-21 09:15:13+02:00', '2019-08-21 12:30:10+02:00', '2019-08-21 15:45:18+02:00', '2019-08-23 19:15:12+02:00', '2019-08-24 08:45:03+02:00', '2019-08-24 12:59:59+02:00', '2019-08-24 13:15:02+02:00', '2019-08-24 13:44:59+02:00']	['2019-08-11 11:45:45+02:00', '2019-08-12 13:15:41+02:00', '2019-08-12 21:45:40+02:00', '2019-08-13 20:15:37+02:00', '2019-08-13 20:30:37+02:00', '2019-08-16 15:00:33+02:00', '2019-08-16 15:30:34+02:00', '2019-08-16 15:45:34+02:00', '2019-08-16 17:00:27+02:00', '2019-08-16 17:45:27+02:00', '2019-08-16 18:00:26+02:00', '2019-08-17 01:00:29+02:00', '2019-08-17 14:15:23+02:00', '2019-08-23 14:45:05+02:00']	['2019-08-15 10:45:31+02:00', '2019-08-15 11:30:31+02:00', '2019-08-15 11:45:34+02:00', '2019-08-15 12:15:31+02:00', '2019-08-15 12:30:34+02:00', '2019-08-15 15:15:34+02:00', '2019-08-15 16:30:30+02:00', '2019-08-18 13:45:20+02:00', '2019-08-18 14:30:21+02:00', '2019-08-19 10:45:20+02:00', '2019-08-19 15:15:16+02:00', '2019-08-20 16:45:12+02:00', '2019-08-23 15:30:03+02:00', '2019-08-23 17:15:05+02:00', '2019-08-24 12:00:11+02:00', '2019-08-24 13:15:02+02:00', '2019-08-24 13:29:59+02:00', '2019-08-24 14:44:59+02:00', '2019-08-24 14:59:59+02:00', '2019-08-24 16:14:59+02:00', '2019-08-24 16:44:58+02:00', '2019-08-24 17:59:59+02:00', '2019-08-24 18:29:59+02:00', '2019-08-25 10:59:57+02:00']	['2019-08-11 11:45:45+02:00', '2019-08-11 12:15:45+02:00', '2019-08-11 12:45:50+02:00', '2019-08-12 04:00:43+02:00', '2019-08-14 16:15:34+02:00', '2019-08-15 08:45:32+02:00', '2019-08-15 11:45:34+02:00', '2019-08-15 13:45:31+02:00', '2019-08-15 18:00:30+02:00', '2019-08-16 09:00:28+02:00', '2019-08-16 11:45:27+02:00', '2019-08-17 13:45:23+02:00', '2019-08-17 15:45:24+02:00', '2019-08-18 15:00:20+02:00', '2019-08-18 16:15:20+02:00', '2019-08-18 21:15:19+02:00', '2019-08-21 09:00:10+02:00', '2019-08-21 10:00:10+02:00', '2019-08-21 11:30:10+02:00', '2019-08-24 11:45:08+02:00', '2019-08-24 12:59:59+02:00', '2019-08-25 08:59:56+02:00']
3 anomalies	24 anomalies	14 anomalies	24 anomalies	22 anomalies

<u>On constate de trop grandes variations pour le capteur 6 aux dates:</u>

Anomalies Température	Anomalies de bruit	Anomalies humidité	Anomalies luminosité	Anomalie CO2
['2019-08-23 17:30:59+02:00', '2019-08-23 20:45:55+02:00', '2019-08-23 21:00:59+02:00', '2019-08-23 21:15:56+02:00', '2019-08-24 03:30:55+02:00', '2019-08-24 03:45:54+02:00']	['2019-08-11 13:01:41+02:00', '2019-08-11 18:16:40+02:00', '2019-08-11 18:31:41+02:00', '2019-08-14 18:16:29+02:00', '2019-08-14 18:46:31+02:00', '2019-08-15 13:31:34+02:00', '2019-08-16 08:46:24+02:00', '2019-08-16 13:01:22+02:00', '2019-08-16 14:01:23+02:00', '2019-08-16 18:16:22+02:00', '2019-08-18 16:46:15+02:00', '2019-08-18 19:16:14+02:00', '2019-08-21 16:46:04+02:00', '2019-08-22 09:16:01+02:00', '2019-08-22 09:31:01+02:00', '2019-08-22 13:01:01+02:00', '2019-08-22 13:16:00+02:00', '2019-08-22 16:01:06+02:00', '2019-08-22 17:16:00+02:00', '2019-08-23 10:45:57+02:00', '2019-08-23 14:45:57+02:00', '2019-08-23 17:45:57+02:00', '2019-08-23 18:30:56+02:00', '2019-08-24 17:30:53+02:00', '2019-08-25 06:30:51+02:00', '2019-08-25 08:45:50+02:00']	['2019-08-11 11:46:42+02:00', '2019-08-13 20:31:33+02:00', '2019-08-16 15:46:22+02:00', '2019-08-16 16:01:23+02:00', '2019-08-16 17:16:23+02:00', '2019-08-16 18:01:22+02:00', '2019-08-16 18:16:22+02:00', '2019-08-16 21:16:21+02:00', '2019-08-20 17:16:08+02:00', '2019-08-20 17:31:08+02:00', '2019-08-23 17:15:56+02:00', '2019-08-23 22:45:56+02:00', '2019-08-24 03:45:54+02:00']	['2019-08-15 12:16:30+02:00', '2019-08-15 12:31:31+02:00', '2019-08-15 15:01:26+02:00', '2019-08-15 15:16:26+02:00', '2019-08-18 14:31:19+02:00', '2019-08-20 16:46:08+02:00', '2019-08-21 13:46:05+02:00', '2019-08-22 16:16:07+02:00', '2019-08-23 10:30:58+02:00', '2019-08-23 13:01:00+02:00', '2019-08-23 13:45:57+02:00', '2019-08-23 15:30:56+02:00', '2019-08-23 16:15:56+02:00', '2019-08-23 17:15:56+02:00', '2019-08-24 10:45:53+02:00', '2019-08-24 12:00:57+02:00', '2019-08-24 13:15:53+02:00', '2019-08-24 13:30:53+02:00', '2019-08-24 14:45:53+02:00', '2019-08-24 15:00:53+02:00', '2019-08-24 15:30:53+02:00', '2019-08-24 16:15:53+02:00', '2019-08-24 16:30:53+02:00', '2019-08-24 16:45:52+02:00', '2019-08-24 18:00:55+02:00', '2019-08-24 18:30:56+02:00', '2019-08-25 11:00:52+02:00']	['2019-08-11 11:46:42+02:00', '2019-08-14 09:31:31+02:00', '2019-08-14 10:16:31+02:00', '2019-08-14 17:46:30+02:00', '2019-08-15 14:01:39+02:00', '2019-08-15 14:31:30+02:00', '2019-08-15 15:31:27+02:00', '2019-08-15 17:16:26+02:00', '2019-08-16 10:46:23+02:00', '2019-08-16 13:16:26+02:00', '2019-08-17 10:16:19+02:00', '2019-08-17 12:31:19+02:00', '2019-08-17 12:46:19+02:00', '2019-08-17 13:46:19+02:00', '2019-08-18 12:31:15+02:00', '2019-08-18 14:46:15+02:00', '2019-08-18 15:16:15+02:00', '2019-08-21 13:01:04+02:00', '2019-08-22 08:46:01+02:00', '2019-08-22 09:16:01+02:00', '2019-08-22 10:46:01+02:00', '2019-08-22 15:16:04+02:00', '2019-08-24 13:00:53+02:00', '2019-08-24 17:45:54+02:00', '2019-08-24 18:30:56+02:00', '2019-08-25 08:45:50+02:00']
6 anomalies	26 anomalies	13 anomalies	27 anomalies	27 anomalies

Chaque capteur relève des valeurs mesurées au sein d'un bâtiment de bureau. Ces valeurs sont, comme on le constate dans les tableaux ci-dessus, celles de la mesure de la température ambiante, de l'humidité relative, du niveau sonore, du niveau lumineux et de la quantité de CO2. Il y a 6 capteurs qui mesurent ces valeurs au sein du bâtiment. Les données sont comprises entre le 11 Août 2019 et le 25 Août 2019, soit 15 jours. Chaque capteur relève des données toutes les 15 minutes.

Ainsi, les trop grandes variations, que l'on a recensées à l'aide de la dérivée de la courbe d'évolution des différentes variables des différents capteurs (fonction **delta()**) et de la fonction **an()**),

ne sont pas nécessairement de vraies discontinuités puisque la relève de données n'est faite que toutes les 15 minutes. Cet intervalle de temps laisse la possibilité à de grandes variations, comme par exemple pour le bruit.

On constate que lorsque qu'on relève une anomalie de bruit, le capteur relève régulièrement une élévation ou diminution de la quantité de CO2 dans la pièce (au même instant ou 15 minutes après): nous pensons que celles-ci sont dues aux individus qui sortent ou qui entrent dans une pièce.

On remarque également peu d'anomalies de température: nous pensons que ceci est dû au fait que la température d'une pièce ne varie pas aussi rapidement en l'espace de 15 minutes.

Nous n'avons pas eu le temps de faire le bonus : de trouver automatiquement les périodes horaires d'occupations des bureaux.