Algorithmique Programmation

Note de synthèse

Introduction

Dans le cas de ce projet, il est question d'analyser des données d'un bâtiment générées par six capteurs, avec pour objectifs de trouver des anomalies dans les données en question en proposant un algorithme qui devrait le faire automatiquement.

De plus, ce projet nous permettra d'améliorer nos compétences en langage python et nous donnera une autre vue sur comment utiliser l'outil de gestion et partage de code informatique (Github).

Dans les lignes qui suivent nous allons détailler de fond en comble notre travail en soulignant ;

- Notre maîtrise de Github ;
- Notre maîtrise du langage python ;
- Notre maîtrise de l'algorithme et notre capacité de répondre à un problème à l'aide d'un programme Python.

1. Maîtrise de GitHub

Étant un espace collaboratif, github nous a permis de stocker et de partager les différents codes que nous avons créés. Dans un premier temps nous avons créé un "repository" et utilisé les commit pour pouvoir apporter des modifications durant l'implémentation de notre algorithme. Par ailleurs, nous avons rencontré certaines difficultés dans l'utilisation de l'invite de commande et compte tenu de notre mauvaise maîtrise de cette dernière, nous nous sommes tournés vers l'utilisation de github online en créant un compte GitHub.com.

2. Maîtrise du langage python

Pour ce qui concerne cette partie nous avons utilisé le logiciel **spyder** de python et d'importer les bibliothèques "**pandas**" et "**matplotlib**".

En étant débutant en langage de programmation python, **pandas** était à nos yeux une bibliothèque moins complexe nous permettant de manipuler et d'analyser les données du fichier csv qui nous a été fourni.

Grâce à cette bibliothèque nous avons dans un premier temps lu le fichier csv et le diviser facilement (avec quelques lignes de code) en six dataframes correspondants aux six (6) capteurs. Quant à matplotlib, elle nous a été utile dans la visualisation graphique des données à travers la fonction pyplot ().

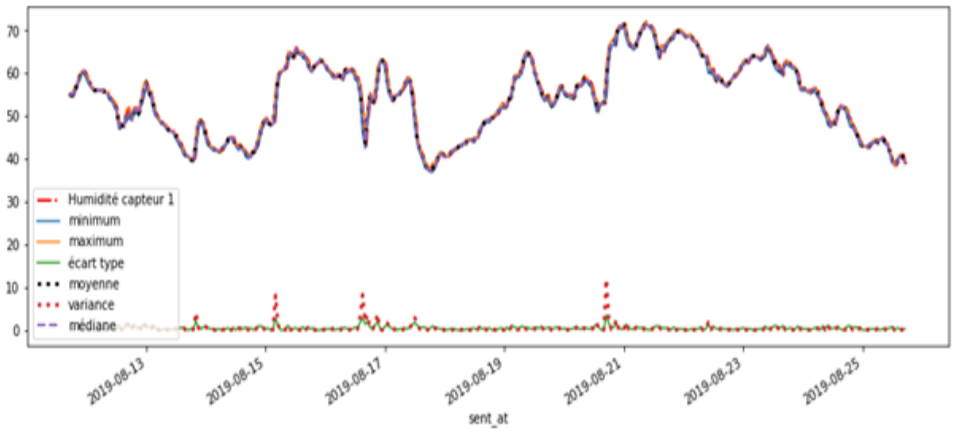
Par ailleurs, nous avons utilisé les “built-in functions” de pandas tels que : mean(), max(), min(), std(), var(), corr() pour calculer les variables statistiques respectivement la moyenne, le maximum, le minimum, l’écart-type, la variance et l’indice de corrélation. Ces “built-in functions” nous ont permis de construire (implémenter) nos fonctions pour afficher à la fois, les courbes montrant l’évolution des variables, des variables statistiques et de l’indice “Humidex” en fonction du temps. Nous pouvons citer par exemple : **cap_1t()**, **ind_cb4()**, **idh_cap6()**. La fonction **cap_1t ()**, implémentée de la manière que les fonctions **cap 2t(), cap_1n(), cap_3l(), cap_4h(), cap_6c().....cap_5t(),** affiche les courbes de la variable ‘température’ et des variables statistiques correspondantes en fonction du temps pour le capteur un (1). La fonction **ind_cb4()** calcule l’indice de corrélation entre les variables 'bruit' et ‘humidité’. Et la dernière **idh_cap6()** calcule l’indice ‘Humidex’ pour le capteur six (6).

Exemple : Fonction pour afficher le graphe de l'humidité pour le capteur 1:

```
def cap_1h():
    plt.figure(figsize=(15,5))
    u['humidity'].plot(label='Humidité capteur 1',alpha=2,lw=2,ls='-.',c='red')
    u['humidity'].resample('h').min().plot(alpha=2,label='minimum')
    u['humidity'].resample('h').max().plot(alpha=2,label='maximum')
    u['humidity'].resample('h').std().plot(alpha=2,label='écart type')
    u['humidity'].resample('h').mean().plot(lw=3,ls=':',alpha=2,label='moyenne',c='black')
    u['humidity'].resample('h').var().plot(lw=3, ls=':',alpha=2,label='variance')
    u['humidity'].resample('h').median().plot(lw=2, ls='--',alpha=2,label='median')
    plt.legend()
    plt.show()
```

- figsize=(x,y)) : permet de dimensionner le graph, longueur des abscisses = x et longueur des ordonnées= y;
- u [‘nom variable’]: permet d’indexer la colonne variable;

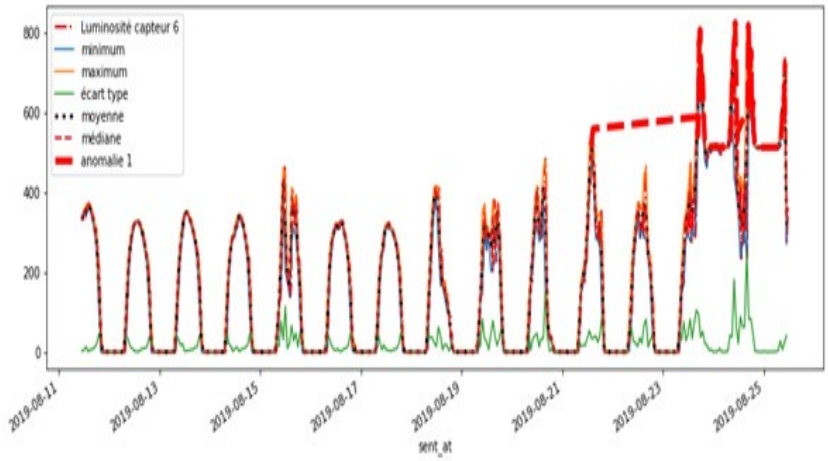
- resample('h') : permet de regrouper les données selon une fréquence. Dans notre c'est par heure ;
- label : permet de donner le nom de la courbe;
- c = 'couleur de la courbe': spécifie la couleur de la courbe;
- alpha : donne l'intensité de la couleur;
- lw (line width) : permet d'indiquer l'épaisseur de la courbe ;
- ls (line style) : pour le style de la courbe.



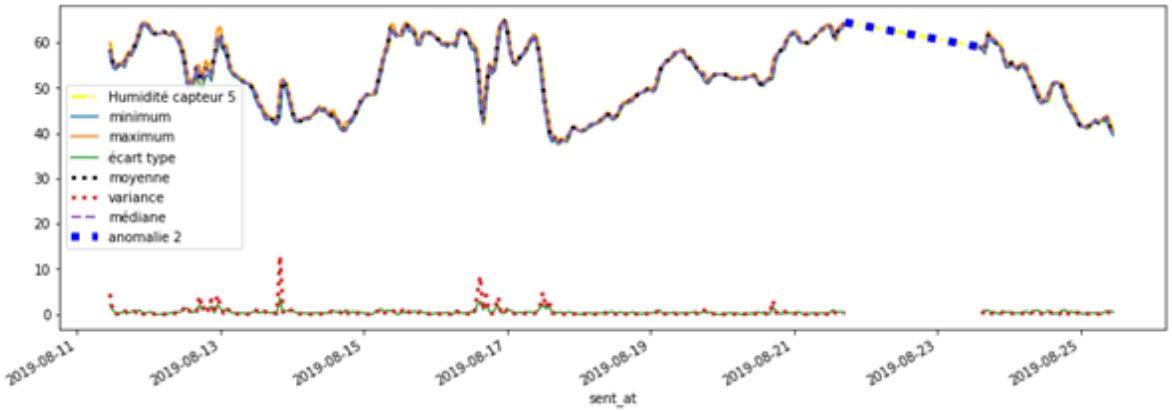
De plus, au niveau de la détection d'anomalies, nous sommes passés par la méthode de détection des valeurs aberrantes (OUTLIERS). C'est une méthode qui consiste à détecter toutes les valeurs extrêmes ou anomalies qui peuvent être décrites comme données qui s'écartent trop du reste des observations, c'est-à-dire qu'elles se distinguent nettement des autres. Ces *Outliers* se distinguaient pour la variable "lum" (luminosité).

```
limite=s['lum'].quantile(.90)
anomalie = s[s['lum']>limite]
anomalie['lum'].plot(label='anomalie 1',c='red',ls='--',lw=5)
```

Pour la variable "lum" toujours, on a constaté pour les 6 capteurs que du 23 au 25 août, les données n'étaient pas nulles entre 20h30 et 07h45, cela est anormal car, le bâtiment de bureau n'est pas censé être éclairé pendant cette période.



On s'est aussi rendu compte à travers la visualisation de toutes les variables qu'il y avait des données manquantes au niveau du capteur 5 du 21 août à 17h au 23 août à 14h. Ces anomalies ont tout simplement été identifiées sur les courbes par des traits interrompus à travers la fonction **pyplot()** qui permet bien évidemment de visualiser des données.



3. Maîtrise de l'algorithme et notre capacité de répondre à un problème à l'aide d'un programme Python

Cette section consiste à donner en détail l'implémentation des différents algorithmes dans la résolution de la thématiques proposées, nous procéderons donc de cette manière.

- ***Affichage des courbes des variables et des variables statistiques fonction du temps.*** Pour pouvoir afficher les différentes variables en fonction du temps, nous avons d'abord implémenté trente (30) fonctions correspondant aux six (6) capteurs et aux cinq

variables (température, luminosité, etc. …) de sorte que chacune d'elles génère la courbe de chaque variable en fonction du temps. Pour que sur cette même courbe s'affiche les variables statistiques on a introduit dans chaque fonction les différentes ***"built-in functions"*** qui permettent de les calculer et ensuite utiliser **pyplot** pour le visionnage.

Pour terminer cette partie nous avons implémenté quelques lignes de code qui permet à un utilisateur quelconque d'entrer la variable (dans le code il se fait avec des chiffres qui représentent chaque variable, exemple : 1= bruit) qu'il veut visualiser.

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

Nous avons utilisé la fonction ***corr()*** de pandas pour calculer l'indice de corrélation par paires de toutes les variables.

```
def ind_ca1():
    plt.figure(figsize=(15,5))
    u['noise'].plot(label='Bruit capteur 1',alpha=2)
    u['temp'].plot(label='Température capteur 1',alpha=2)
    ic=str(u['noise'].corr(u['temp']))
    print("L'indice de corrélation est :",ic)
    plt.legend()
    plt.show()
```

Ensuite afficher les courbes des deux variables tout en indiquant l'indice de corrélation.

- ***Calcul de l'indice de l'humidex.***

Nous avons ajouté une colonne "Indice Humidex" à chaque dataframes correspondants aux différents capteurs. Ensuite, stocker à chaque ligne les valeurs de l'indice "Humidex" obtenues par la formule suivante :

$$\text{Humidex} = T + \frac{5}{9}(6{,}112 \times 10^{7{,}5\frac{T}{237{,}7+T}} \times \frac{H}{100} - 10)$$

T = température et H = Humidit

- ***Détection des anomalies***

Pour détecter les anomalies nous avons d'abord effectué une brève étude des différentes courbes, durant cette étude nous avons détecté un premier problème avec les données qui manquaient (une discontinuité dans le traçage de la courbe) pour toutes les variables (température, humidité, bruit, etc. ...) du capteur cinq (5). Deuxièmement, nous avons aussi observé que pour tous les capteurs, les valeurs de la variable luminosité étaient parfois très grandes (valeurs inattendues) et pour les trois derniers jours (23/08/2019 au

25/08/2019), elles ne revenaient pas à zéro (une manière de dire que la lumière ne s'est pas éteinte pour ces jours durant la nuit)

En effet, pour l'anomalie du capteur cinq (5), l'algorithme consiste à indiquer sur la courbe l'existence de valeurs manquantes. En ce qui concerne la variable luminosité, on implémente un algorithme détectant les valeurs inattendues, et les valeurs non nulles pendant les nuits du 23 au 25 août et qui les affichent sur la courbe de cette variable.

Conclusion

Pour terminer, malgré certaines difficultés rencontrées sur Github nous avons pu appréhender son importance et chercher à savoir comment il fonctionne, car en tant que futur ingénieur il sera très bénéfique pour échanger et collaborer dans le monde professionnel sans oublier la partie programmation de ce projet nous a permis de connaître certaines bibliothèques (comme pandas) et fonctions et faire de bons choix d'implémentation des algorithmes dans un langage de programmation (dans notre cas python).