RAPPORT PROJET INFORMATIQUE

Sujet A – Détection d’anomalie

# Utilisations des données.

La première contrainte de ce projet étant le format sous lequel les données sont fournies, il a fallu déterminer une façon efficace de récupérer ces données de manière à ce que les différentes fonctions du programme puissent les lires facilement, sans faire appel aux fonctions de lecture de csv et à un curseur. Il a donc été décidé de récupérer dans une matrice (de type *‘a list list*). A partir de cette liste de liste, et de la deuxième colonne du csv, on distribue les données pour chaque capteur et chaque variable dans une matrice à 3 dimensions qui sera une variable globale appelée numdata. Il est à noter que les dates sont converties en type datetime.datetime pour faciliter les comparaisons.

Calcul humidex.

Le calcul de l’indice humidex est simplement celui de la formule de Heinrich Gustav Magnus-Tetens.

# Interface utilisateur.

Pour la plupart des fonctions requises au programme, il est souhaitable pour l’utilisateur de préciser quelle valeur et quel capteur il faut utiliser, et pour chacun, l’intervalle de temps à étudier. Concernant les dates, il a été décidé de récupérer les bornes de l’intervalle via un input qui vérifie les conditions de type et de mathématiques (début antérieur à la fin). Il a ensuite été envisagé de différencier la nature des données par une programmation conditionnelle en multipliant les if…elif…else (ex : if variable == « température »…). Au-delà d’un encombrement évident, cette solution a été abandonnée au moment de l’écriture de la fonction covariance, qui aurait été trop complexe. Il a été préférable d’écrire de nouvelles fonctions similaires à ask\_dates(), pour demander à l’utilisateur quelle variable et quel capteur il veut utiliser. Les numéros des capteurs étant des entiers, on s’arrange pour que leur indice dans les listes y corresponde. Pour les variables, on écrit une fonction qui prend en argument une chaine de caractère et qui renvoie pour chaque variable son indice dans numdata.

# Gestion des dates.

Le format datetime ayant été choisi, il est ensuite simple de récupérer la liste des dates d’échantillonnage appartenant à l’intervalle renseigné par l’utilisateur avec la fonction interv.

# Calcul des informations statistiques.

Il a d’abord été envisagé d’utiliser des modules pour faire ces calculs. Seulement il a été préférable dans notre implémentation d’adapter les fonctions au sujet : par volonté de rendre la fonction displayStats plus simple au final, il a été décidé de répartir la complexité sur chaque fonction de calcul.

# Calcul de corrélation entre deux variable :

Ce calcul se sert de la covariance et des écarts-types. Celle-ci est nettement simplifiée par l’interface utilisateur.

# Détection d’anomalies.

Pour détecter une anomalie au départ il a été question de faire un intervalle autour de la moyenne et considérer toute valeur en dehors de cet intervalle comme anormale, cependant la précision n'était pas bonne. On a donc décidé de comparer l'écart relatif entre 2 valeurs successives et de considérer que si cet écart est supérieur à 50% alors la deuxième valeur parmi les 2 successives est fausse. Cependant pour laisser le choix à l'utilisateur de choisir la précision il a été nécessaire de rajouter un "input". Cette méthode implique de considérer que la toute première valeur du capteur est correcte. Cette méthode n'était cependant pas applicable à la luminosité car les valeurs sont parfois nulles et l'écart relatif entre 2 valeurs est très régulièrement supérieur à 50%. Le critère a donc été de considérer que la luminosité évolue de façon croissante ou décroissante. On a donc considéré qu'une valeur était anormale si elle n'était pas comprise entre la valeur précédente et la valeur suivante. Néanmoins, cela ne permettait pas de prendre en compte la valeur du maximum de luminosité, ni les éventuels sauts de luminosité qui peuvent s'expliquer par l'allumage d'une lumière. On a donc rajouté une marge de 30 avant et après. Pour ce qui est de la mise en avant des anomalies on a créé 2 listes vides, l'une pour y mettre les valeurs anormales et l'autre pour y mettre les moments où elles sont prises. Une autre liste s'occupe de garder les bonnes valeurs. Ainsi, on met en avant les anomalies en les affichant dans un autre caractère que les valeurs normales grâce à ces listes.

# Affichage.

En utilisant à nouveau les fonctions d’interface utilisateur, on crée des graphiques avec les fonctions du module matplotlib. La fonction anomalie est similaire mais entoure en rouge ce qu’elle a considéré comme étant des anomalies.

# Horaires d’ouverture de bureau.

Pour ce bonus, il a été décidé de calculer les valeurs moyennes des variables bruit, co2 et luminosité à chaque instant sur l’ensemble des capteurs. Ensuite, on détecte et renvoie les moments où ces trois valeurs passent au-delà respectivement en deçà de leur valeur médiane ensemble pour définir les horaires d’ouverture respectivement de fermeture des bureaux.

# GITHUB.

L’ensemble du travail a été effectué via github. Parfois, l’interface graphique de la version web s’est avérée moins contraignante. Cependant la création du fichier et certaines modifications ont été apportées avec l’invite de commande.