Projet NF26

1 Introduction

Le but du projet est de manipuler diverses données climatiques issues de plusieurs stations réparties sur le globe. Nous avons été affectés au cas de l'Espagne et nous avons pris la liberté d'utiliser la période de temps de 2010 à 2013 plutôt que celle de 2001 à 2010, car elle regorgeait davantage d'informations.

Les enjeux étaient de stocker, de façon pertinente, les données dans des bases de données NoSQL Cassandra et d'exploiter ces résultats avec le langage Python pour répondre à plusieurs objectifs.

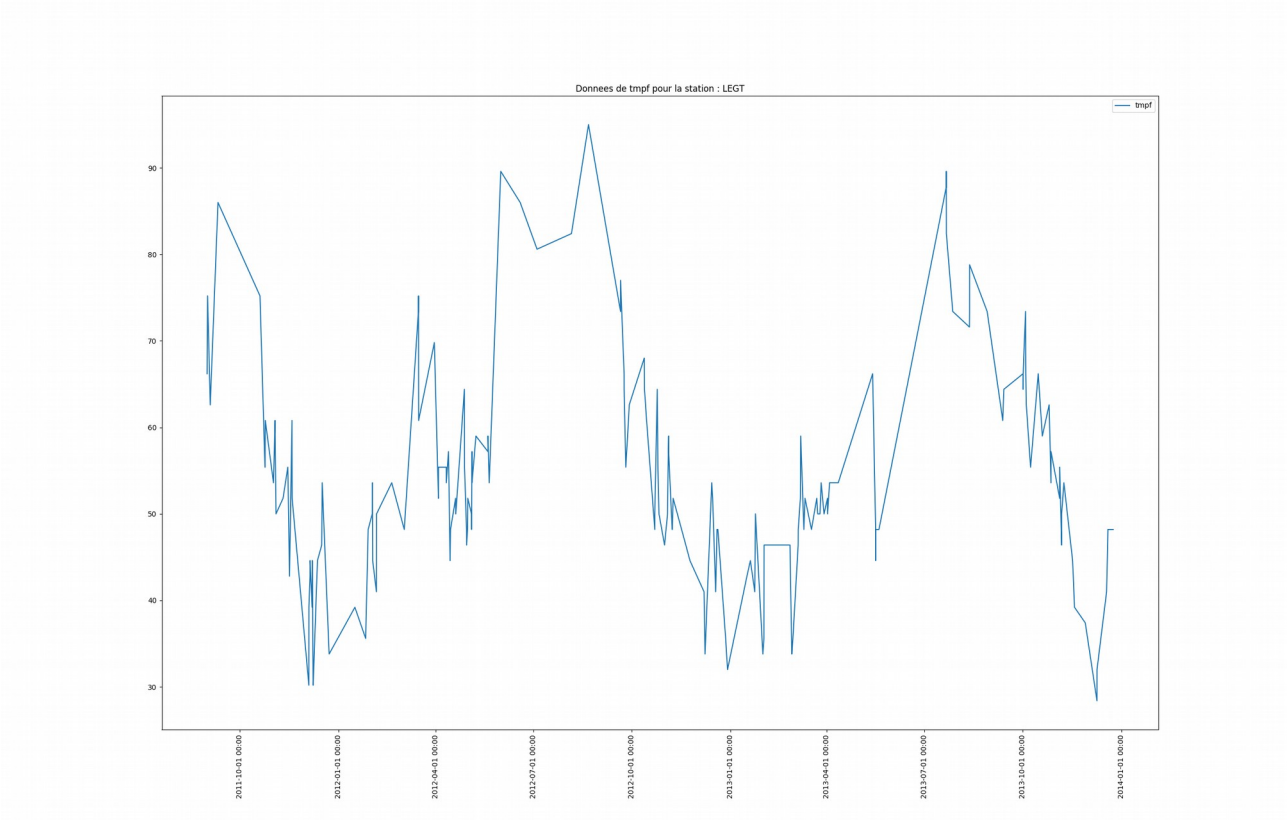
Les graphiques et cartes sont affichés pour une visualisation immédiate et stocké pour une utilisation future.

2 Objectif 1

Le premier objectif fut, pour un point donné, autrement dit, pour l'une des stations de l'Espagne, d'obtenir un historique des données d'un attribut (par exemple l'humidité). Nous avons donc utilisé comme clef de partition la station, permettant ainsi d'obtenir l'ensemble de ses mesures et pouvoir faire l'historique de chacun. Pour différencier chacune de ces mesures, nous nous sommes servi de la date de celles-ci en clef de tri (année, mois, jour, minute, heure) .

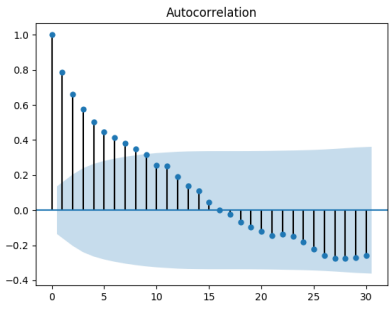
Il est vrai que, dans notre exploitation, nous choisissons une station puis un seul attribut (pour plus de clarté) et que, dans ce cas, la possibilité d'utiliser l'attribut comme clef de partition, la date comme clef de tri et y associer l'ensemble des mesures de chaque station est possible. Or, si suite à une évolution, nous voulons faire, en une fois, l'historique de tous les attributs d'une station cela n'est plus possible, d'où notre choix. De plus, nous stockons des informations non-essentielles comme la longitude ou la latitude. Cela est fait dans le cas d'une autre exploitation de la base.

Pour afficher l'historique, nous stockons les mesures de l'attribut voulu en fonction du temps avec l'objet « panda.Series », puis nous les affichons sous formes de courbes avec « matplotlib.pyplot ». Nous obtenons ce type de courbes :

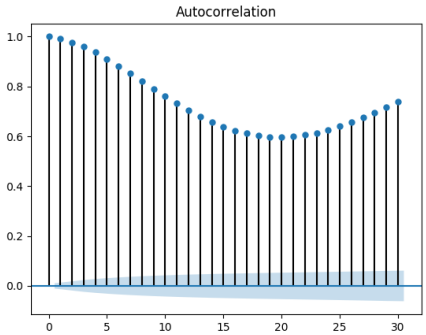


Courbe de la température de la station LEGT

Pour la saisonnalité, nous utilisons « statsmodels.api » qui nous donne ce type de graphique :

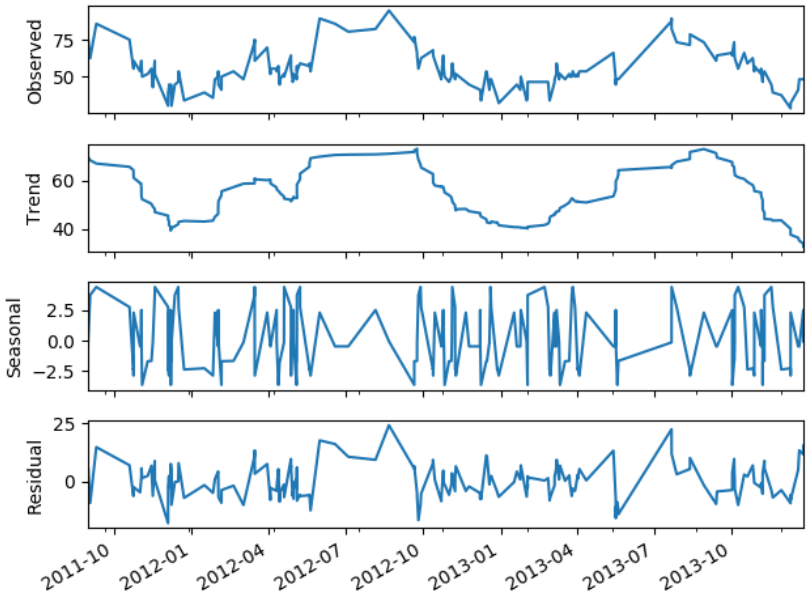


Saisonnalité de la température de la station LEGT



Saisonnalité de la température de la station LEBB (beaucoup plus de données)

Pour les écarts à la saisonnalité, nous utilisons « statsmodels.graphics » et obtenons :

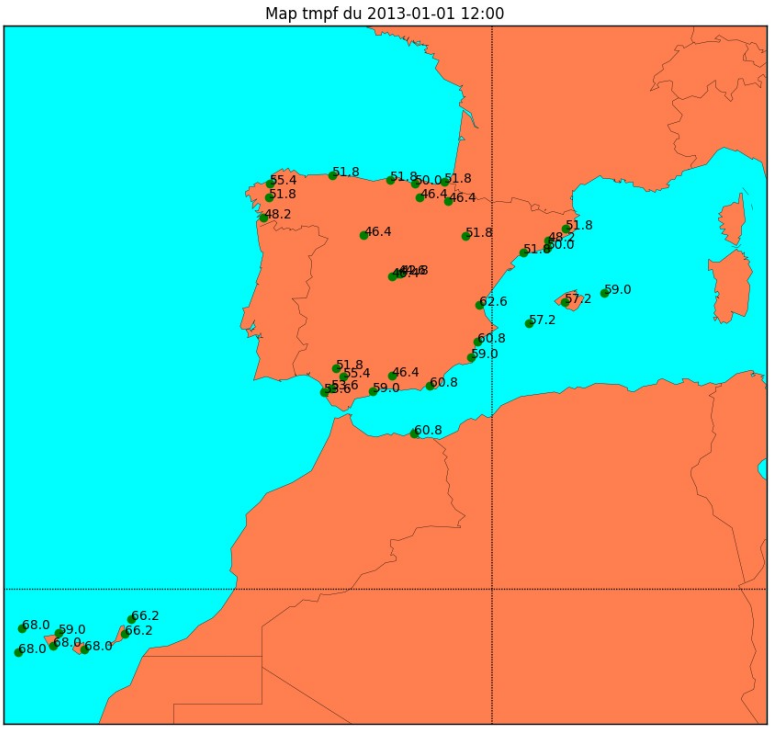


Décomposition de la température de la station LEGT

3 Objectif 2

Le second objectif fut d'obtenir, à un instant donné, une carte affichant les mesures d'un indicateur. La table de l'objectif 1 n'était plus appropriée. Nous avons donc créé une seconde table avec en clef de partition : la date (année, mois, jour, heur, minute), affin d'obtenir facilement l'ensemble des valeurs à un instant donnée. Comme clef de tri, nous aurions pu utiliser la station, mais nous avons préféré utiliser la longitude et la latitude car leur association correspond à une seule et unique station et ces données sont nécessaires pour ensuite pouvoir placer les mesures sur la carte.

Une fois les données récupérées, nous utilisons l'objet « mpl_toolkits.basemap.Basemap » pour obtenir une carte. La carte est centrée sur l'Espagne et les mesures y sont placés aux coordonnées des stations qui les ont récupérées.



Exemple de carte : Température du 2013-01-01 à 12h00

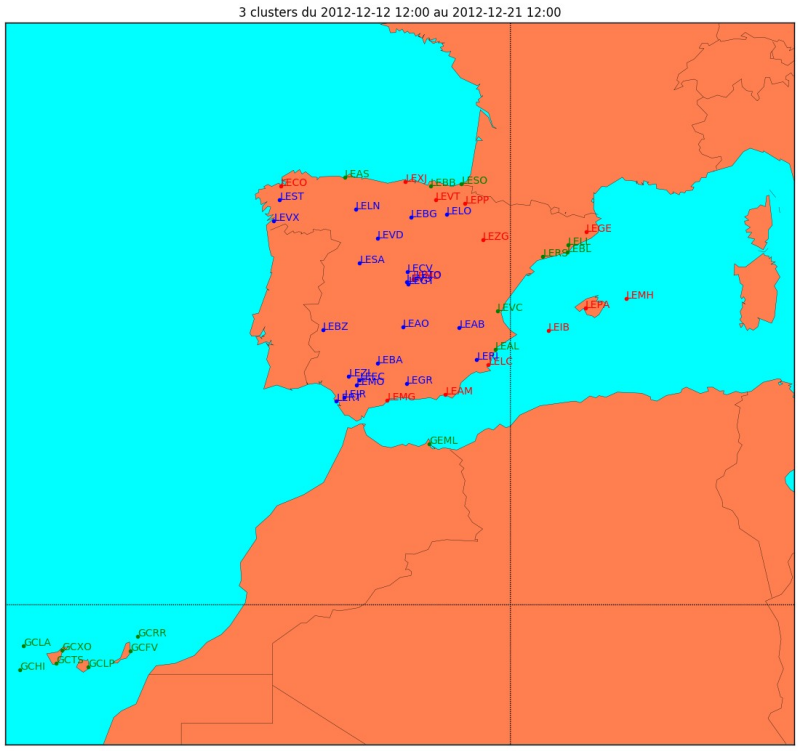
4 Objectif 3

L'objectif 3 consiste à effectuer une clusterisation des différentes stations par rapport aux données météorologiques sur une période de temps demandée. Nous faisons une moyenne de chaque attribut de chaque station sur la période étudiée, en récoltant les données en streaming.

Le nombre de cluster souhaité est demandé à l'utilisateur. Pour clusteriser, dans un premier temps, nous générons les centroÏdes avec des valeurs aléatoires comprises entre le minimum et le maximum des moyennes recueillies pour chaque attribut. Nous clusterisons donc sur N dimensions, N étant le nombre d'attributs. La norme des vecteurs obtenus est comparée à celle de chaque station. De nouveaux centroïdes sont calculés par rapport à la moyenne des stations dont la norme de leur vecteur est la plus proche. Cette opération est effectuée jusqu'à une stabilisation des centroïdes. Si les centroïdes n'ont aucun station associée, les centroïdes sont renouvelées aléatoirement.

Pour effectuer cet objectif, la table de l'objectif 2 a été reprise, car sa clef de partition est le temps. Une itération de requête est effectuée avec une date précise, en étant incrémentée de la plus petite unité de mesure : la minute. Nous aurions pu créer une nouvelle table partitionnée à l'heur ou au jour, mais avons choisi de garder la même table pour un souci de volume de donnée et en préconisant des périodes courtes. Un nouveau partitionnement aurait suivi les mêmes principes.

Une fois clusterisé, les stations sont affichées sur la carte comme dans l'objectif précédent. Leur appartenance à un cluster est visible par la couleur affichée.



3 cluster calculés sur la période du 2012-12-12 à 12h au 2012-12-21 à 12h

5 Architecture du projet

Dans cette partie, nous allons présenter l'utilité des différents composant du projet.

- Le fichier « parameters.py » :

Il permet de stoker la configuration du projet (pays concerné, période de temps étudiée…). Il possède également les informations utiles à tout le projet.

- Le fichier « requierments.txt » :

Ce fichier permet d'obtenir toutes les librairies à installer pour faire fonctionner le projet.

- Le fichier « download_data.py » :

Ce fichier permet de télécharger les données du pays concerné sur la période temps étudiée depuis le site internet où elles sont stockées. Ces informations seront ensuite placées dans des fichiers « csv » par station dans le dossier « out ». Le code est inspiré de celui trouvé sur le site internet gardant les données. Ce fichier n'est utilisé que très rarement, à une initialisation. Il suffit de l'exécuter pour appeler les bonnes fonctions.

- Le fichier « create_table.py » :

Ce fichier crée les différentes tables utilisées durant le projet et les remplie des informations contenues dans tous les fichiers « csv » du dossier « out ». Ce fichier n'est utilisé que très rarement, à une initialisation. Il suffit de l'exécuter pour appeler les bonnes fonctions.

- Le fichier « main.py » :

Ce fichier gère l'exploitation des données. Il est utilisé fréquemment. Son exécution permet d'obtenir une interface dans le terminal pour choisir son objectif, les éléments nécessaires et obtenir les résultats.

- Le dossier « out » :

Ce dossier stocke les graphiques issues de l'exploitation des données. Les fichiers qu'il contient peuvent être consulté plus tard, par l'utilisateur

- Le dossier « data » :

Ce dossier contient tous les fichiers « csv » contenant toutes les informations, sur les différentes stations, qui devront être mises dans Cassandra. Ce dossier n'est pas censé être exploité manuellement par l'utilisateur.