I. Présentation du projet

Le gestionnaire du réseau de transport d'électricité (RTE) a pour mission de distribuer une alimentation électrique à tous ses clients. Cela inclue le transport mais aussi la surveillance de la consommation. Il faut en effet assurer à chaque instant l'équilibre production-consommation.

L'énergie électrique est produite par de nombreuses centrales électrique de nature différentes (nucléaire, hydraulique, éolienne, etc.). Mais toutes fournissent du courant électrique alternatif qui, contrairement au courant continu, ne se stocke pas.
Si la production instantanée n'est pas suffisante, il faut alors démarrer des centrales dites « centrales d'appoint ».
Si la production est trop importante, il faut alors espérer que nos voisins (Italie, Espagne, …) aient besoin d'énergie et nous la rachète, elle est perdue sinon !

Prévoir la consommation électrique journalière est donc très important pour RTE.

Comme vous avez étudié l'algorithme des k plus proches voisins dans le chapitre précédent, vous vous proposez d'aider RTE. Vous allez écrire un logiciel en python qui essaiera de prédire la consommation électrique en fonction des jours de l'année et de la température.

II. Jeux de données

Un ensemble de mesures est disponible dans le fichier « pic-journalier-consommation.csv ». Les données présentent la consommation journalière Française et les températures relevées pour la période de janvier 2012 à janvier 2022. Le détail de chaque champ est :

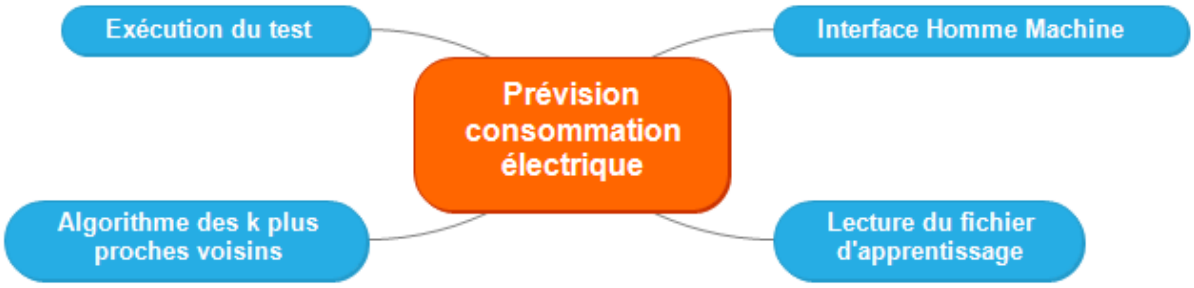
- La date : format "aaaa-mm-jj"
- La consommation : en Méga Watt
- La température moyenne : en degré Celsius
- La température de référence : en degré Celsius

Un autre fichier est fourni « jeuTests.csv » avec 10 mesures réelles qui vous permettra de tester votre programme.

Remarque : La température de référence est une température calculée par moyennage des relevés des années précédentes. C'est donc la température attendue pour un jour de l'année

III. Cahier des charges

Le projet "Prévision consommation électrique" nécessite quatre entités :



III-1 L'interface Homme Machine (IHM)

On utilisera la bibliothèque Tkinter. L'IHM devra permettre à l'utilisateur :

- De choisir le nombre de voisins (k)
- D'entrer les paramètres date, température moyenne, température de référence du point à vérifier
- De lancer le test par appui sur un bouton
- D'afficher la consommation prévue

III-2 Lecture du fichier d'apprentissage

Cette partie du projet va lire le fichier csv afin de créer une liste de tous les enregistrements. Chaque élément de cette liste contiendra :

- Un tuple (Numéro du jour, Température moyenne, température de référence)
- La consommation électrique

Trois fonctions seront nécessaires :

1. Fonction **lecture** qui prend en paramètre le nom du fichier csv à lire et qui renvoie un tableau
2. Fonction **numeroJour** qui prend en paramètre la date (au format texte "aaaa-mm-jj") et qui renvoie un entier correspondant au numéro du jour correspondant (entre 1 et 365)
3. Fonction **estBissextile** qui prend en paramètre l'année à vérifier et qui renvoie True si elle est bissextile, False sinon (on pourra s'aider avec https://fr.wikipedia.org/wiki/Année_bissextile)

Remarque : Pour simplifier le programme, si l'année est bissextile, on numérotera les 30 et 31 décembre avec le même numéro 365.

III-3 Algorithme des k plus proches voisins

Cette partie du projet va calculer la distance entre un point à vérifier et tous les enregistrements. Ces distances seront stockées dans une nouvelle liste qu'il faudra trier afin de ne garder que les k premiers termes.
Trois fonctions seront nécessaires :

1. Fonction **distance** qui prend en paramètre 2 tuples (Numéro du jour, Température moyenne, température de référence) et qui renvoie un nombre réel représentant la distance euclidienne (*voir remarque ci-après*)
2. Fonction **kPlusProches :**
   Paramètres d'entrée :
   - Un tuple (Numéro du jour, Température moyenne, température de référence) correspondant au point à vérifier
   - Une liste représentative du jeu de données original où chaque ligne contient un tuple (Numéro du jour, Température moyenne, Température de référence) et la consommation électrique

   Paramètres de sortie :
   - Une liste triée en fonction de la distance dont les éléments sont des tuples (distance, indice du tableau original)
3. Fonction **PuissanceMoyenne :**
   Calcule et renvoie la puissance électrique moyenne des k plus proches voisins.
   Paramètre d'entrée :
   Une liste triée en fonction de la distance dont les éléments sont des tuples (distance, indice du tableau original)
   Paramètre de sortie :
   Puissance électrique moyenne des k plus proches voisins

Remarques : Précisions pour le calcul de distance

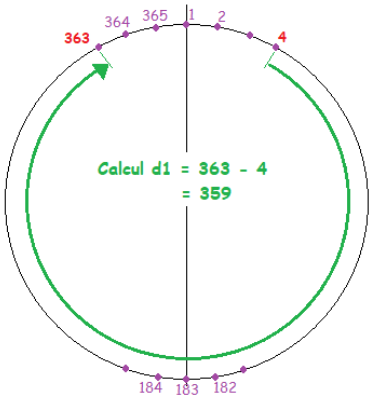
Calcul d'une distance euclidienne entre deux points $M(x_1,y_1,z_1)$ et $N(x_2,y_2,z_2)$:

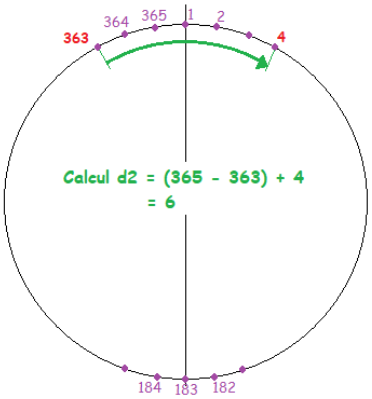
$$MN = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2}$$

Dans votre fonction, on pourra se passer du calcul de la racine carré (très couteuse en calcul) car les distances ne sont calculées que pour être classées dans l'ordre croissant.

La distance entre les numéros des jours est un peu particulière. En effet, entre le 31 décembre (jour 365) et le 1er janvier (jour 1), la distance n'est que de 1…
Ci-dessous un exemple du calcul de la distance entre le jour 363 et le jour 4 :

Le plus simple informatiquement est probablement de calculer les deux distances d1 et d2 puis de ne garder que la plus petite des deux.
On pourra remarquer que :
$d_2 = 365 - d_1$





III-4 Exécution du test

Il sera nécessaire d'écrire une fonction **lancerTest** qui sera exécutée lors de l'appui sur le bouton de l'IHM. Cette fonction aura pour rôles :

- Récupérer les diverses entrées de l'utilisateur (nombre de voisins, paramètres du point à vérifier) puis de les mettre en forme
- Lancer la lecture du fichier d'apprentissage
- Lancer l'algorithme de k plus proches voisins
- Calculer la consommation électrique prévue en effectuant la moyenne des consommations des k proches voisins
- Afficher le résultat sur l'IHM

IV. Travail en équipe

Votre équipe doit analyser correctement le cahier des charges afin de vous répartir l'écriture des différentes parties. Vous serez très attentifs sur le respect du cahier des charges (paramètres d'entrées et de sortie de chaque fonction) car vous risquez d'avoir des surprises lors de l'intégration finale !

Remarque importante : Le programme doit être commenté et les fonctions documentées de façon précise et concise. (On évitera un commentaire à toutes les lignes ...)

V. Pour aller plus loin (les plus motivés…)

Votre projet pourrait être enrichi de manière très intéressante en proposant deux nouvelles fonctionnalités :

- Plutôt que de calculer la consommation pour un seul point de mesure, votre programme pourrait lire le fichier « jeuTest.csv » et appliquer l'algorithme pour l'ensemble des points y figurant.
- Vous utilisez le calcul de distance euclidienne mais il existe un autre type de calcul, la distance de Manhattan. Votre programme pourrait proposer à l'utilisateur l'une ou l'autre de ces deux méthodes.

Calcul d'une distance de Manhattan entre deux points M(x_1,y_1,z_1) et N(x_2,y_2,z_2) :

$$MN = |x_1 - x_2| + |y_1 - y_2| + |z_1 - z_2|$$