DESCRIPTION DES FICHIERS

- Le fichier **Preprocessing.ipynb** contient l'ensemble de fonctions pour traiter les données bruts des fichiers CSV fournies par les enseignants.

- Le fichier **load_data.ipynb** est une version plus compacte de preprocessing avec les fonctions nécessaires pour faire la lecture des données nettoyés et en obtenir des features pour l'apprentissage. JE CONSEILLE VIVEMENT D'UTILISER CE FICHIER PLUTÔT QUE PREPROCESSING
- Data.csv: contient les données du fichier transtion_ecologique traitées et filtrées. Les colonnes innécessaires ont été enlevées. Les questions ouvertes ont été converties en dictionnaires avec comme clé le mot et comme valeur le nombre de répétitions. Les réponses oui/non ont été transformées en 1 pour 'oui', -1 pour 'non" et 0 si la question n'a pas été répondue. Les QCM ont été traitées comme les yes/no sauf que le tag s'incremente a partir de 1.
    - Ordre des colones:
        - Colonne 0:TItre
        - Colonne 1: Categorie de l'utilisateur
        - Colonnes 2-17: Questions
            - yes /no questions: colonnes 4,6,10
            - Questions choix multiples: 12
        - Colonnes 18: Code Postal
- Word_count_by_question.csv: ce fichier contient un dictionnaire par question avec tous les mots qui aparaissent dans ce dernier.
- Word_count_total.csv: il contient la combinaison des tous les dictionnaires par question. C'est un dictionnaire general des données
- Le fichiers city_information.tsv et correspondance-code-insee-code-postal.csv contient des informations utiles sur les communes/villes (population, aire, code zip)

DESCRIPTION DES FONCTIONS

- **read_data**('data.csv',numrical_answers_array): charge le fichier data.csv en indiquant les questions qui ont un nombre comme reponse.
- **read_word_count**('word_count_by_question.csv'): charge soit le fichier word_count_by_question.csv ou soit word_count_total.csv et en genere un dictionnaire ou un array de dictionnaires
- read_dictionary(dictionary_string): lire un string qui represente un dictionnaire et en genere un.
- Sort_dictionnary: trier un dictionnaire et renvoie une liste avec commo premier element le mots avec le plus d;ocurrences.
- **Normalize_counts**: renvoie un dictionnaire des mots ou la valeur et sa frequence dans le texte.

- find_zip_codes_by_town(density_threshold,filename='city_information.tsv'): renvoie deux ensembles de zips codes des villes dont la densité est superieur et inferieur au density_threshold
- **city_village_classifier**(density_threshold,data): fait la classification des entries de la base de données en fonction de sa density_threshold. La convention est 1 pour les grand villes et -1 pour le villages.
- word_count_by_question(data,yes_no_questions): count tous les mots des questions textuels par question et les met dans un array.
- word_count_total(dictionary_array):: combine tous les elements de plusieurs dictionnaires stockes dans un array.
- **get_most_used_words**(data, yes_no_array, word_count_array, number_of_words): elle prend en arguments un array de entries, un array qui indique les reponses aux yes/no questions et aux QCM et le nombre de mots a prendre par question. Elle renvoie l'array d'origine dont les dictionnaires des questions ont été transformés en une liste de taille number_of_words avec le nombre d'ocurrences des mots qui coincident avec word_count_array
- **get_set_features**(data,column): renvoie une colonne de l'array de data.