**Détection des émotions**

**Introduction :**

Le but de ce travail est de réaliser un programme qui reçoit un texte puis détecte les émotions présentes dans le texte. Pour le réaliser, on va utiliser une base de données qui des phrases classées selon 13 émotions. Et on va utiliser ces tweets pour apprendre à un modèle de classifier des phrases aléatoires selon ces émotions.

Le fait d’analyser et de comprendre des phrases relève du domaine du traitement du langage naturel : *Natural Language Processing (NLP)*.

Python dispose de plusieurs librairies qui aident à analyser les données textuelles. Dans ce projet, on va utiliser NLTK : *Natural Language Toolkit.*

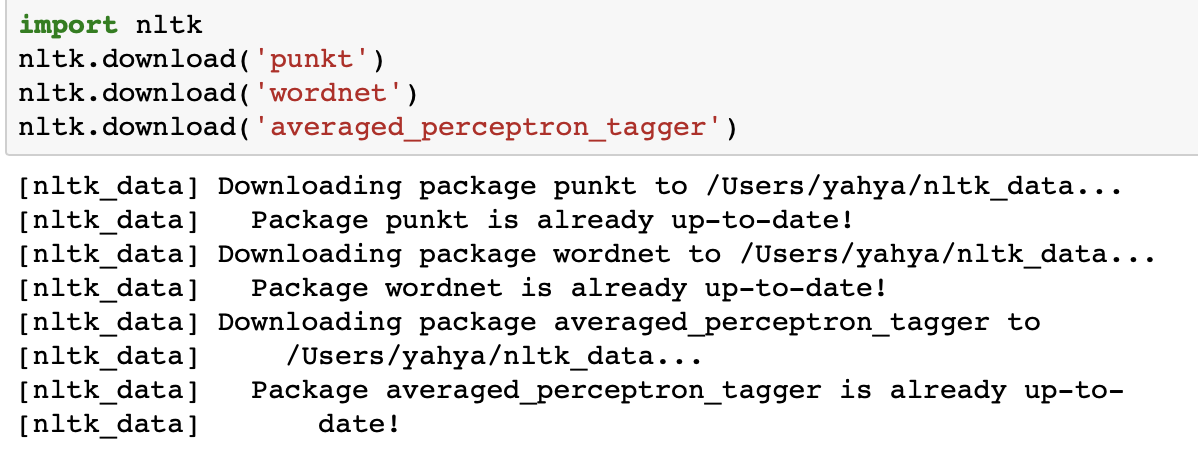
Le fichier jupyter est nommé Emotion.ipynb.

Le fichier python est nommé Emotion.py.

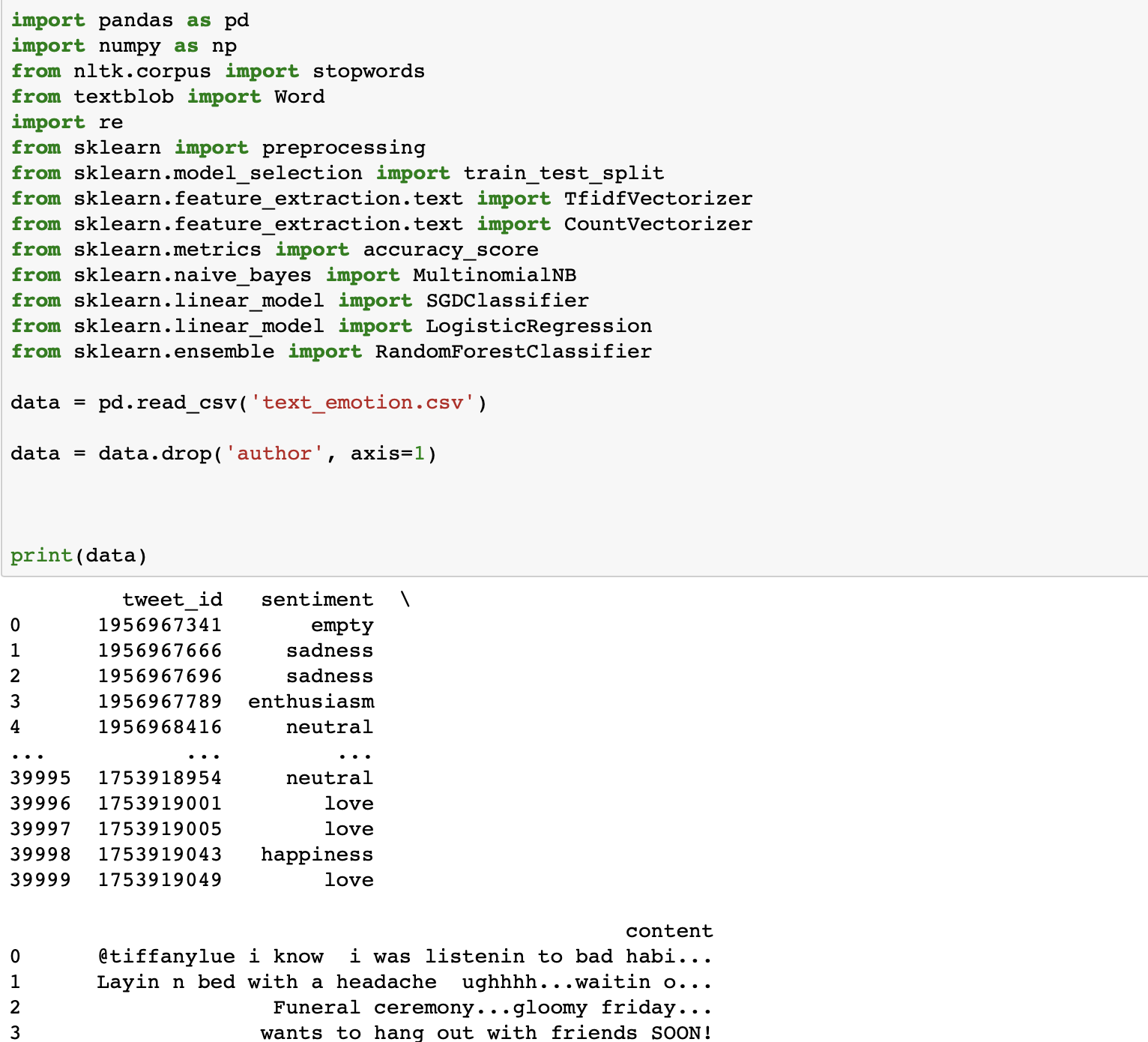
**Installation de NLTK et préparation des données :**

On installe NLTK par la commande suivante  : pip install nltk.

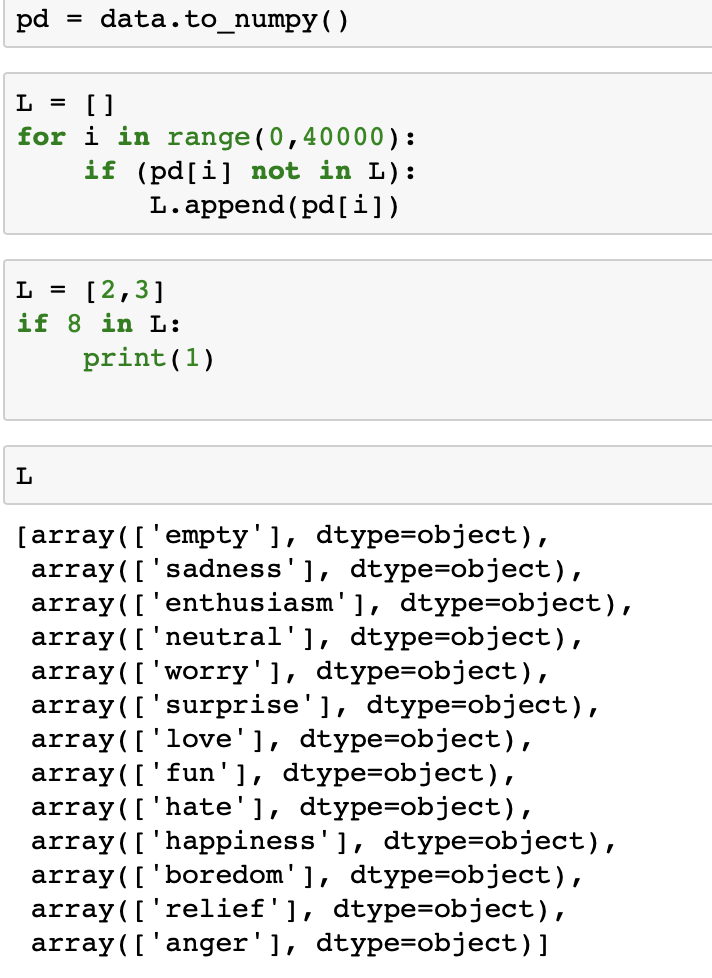
On importe NTLK et télécharge ces modules.



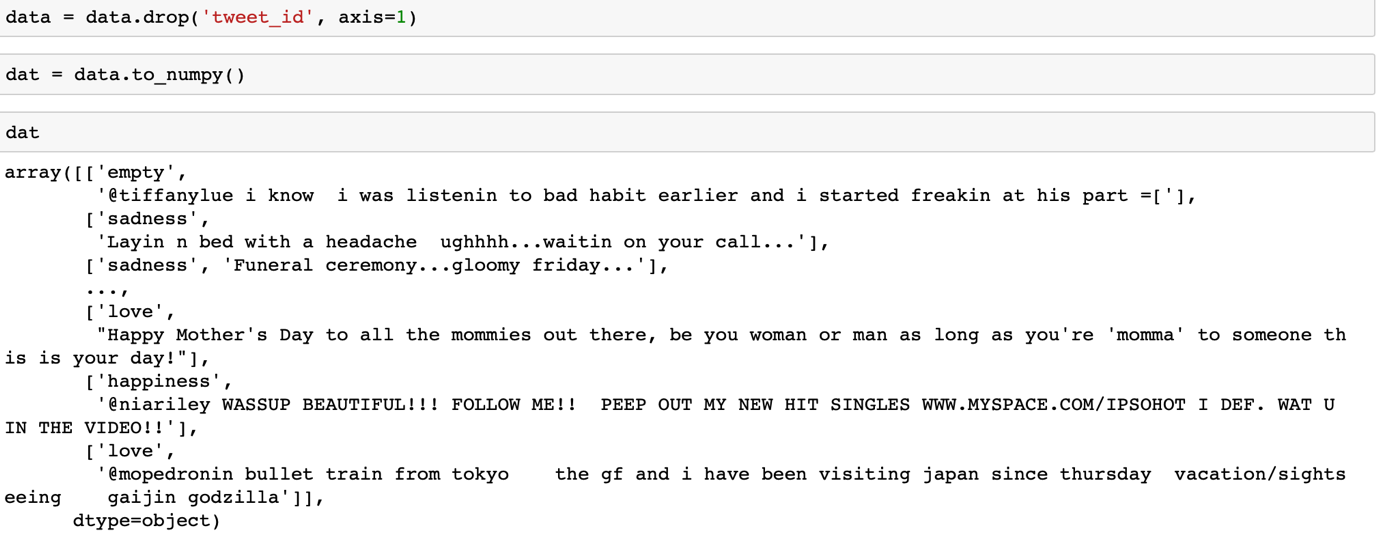
On importe le fichier csv où se trouve les données :



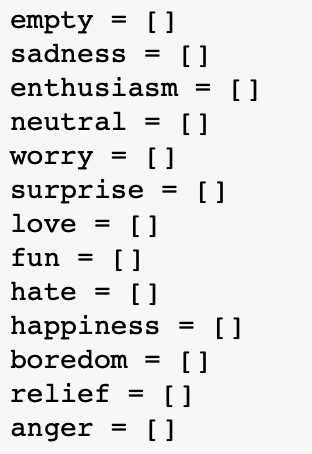
On affiche les émotions qu’on utilise :



On transforme les données en liste de listes.



Puis on les séparent selon les émotions :

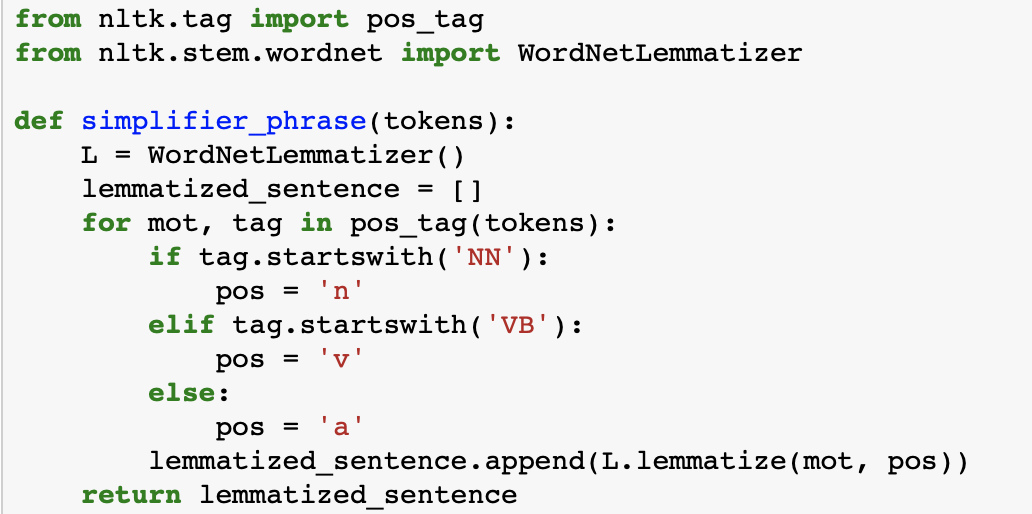




Après avoir séparés les données en 13 listes de listes, on prépare chacune de ces listes.

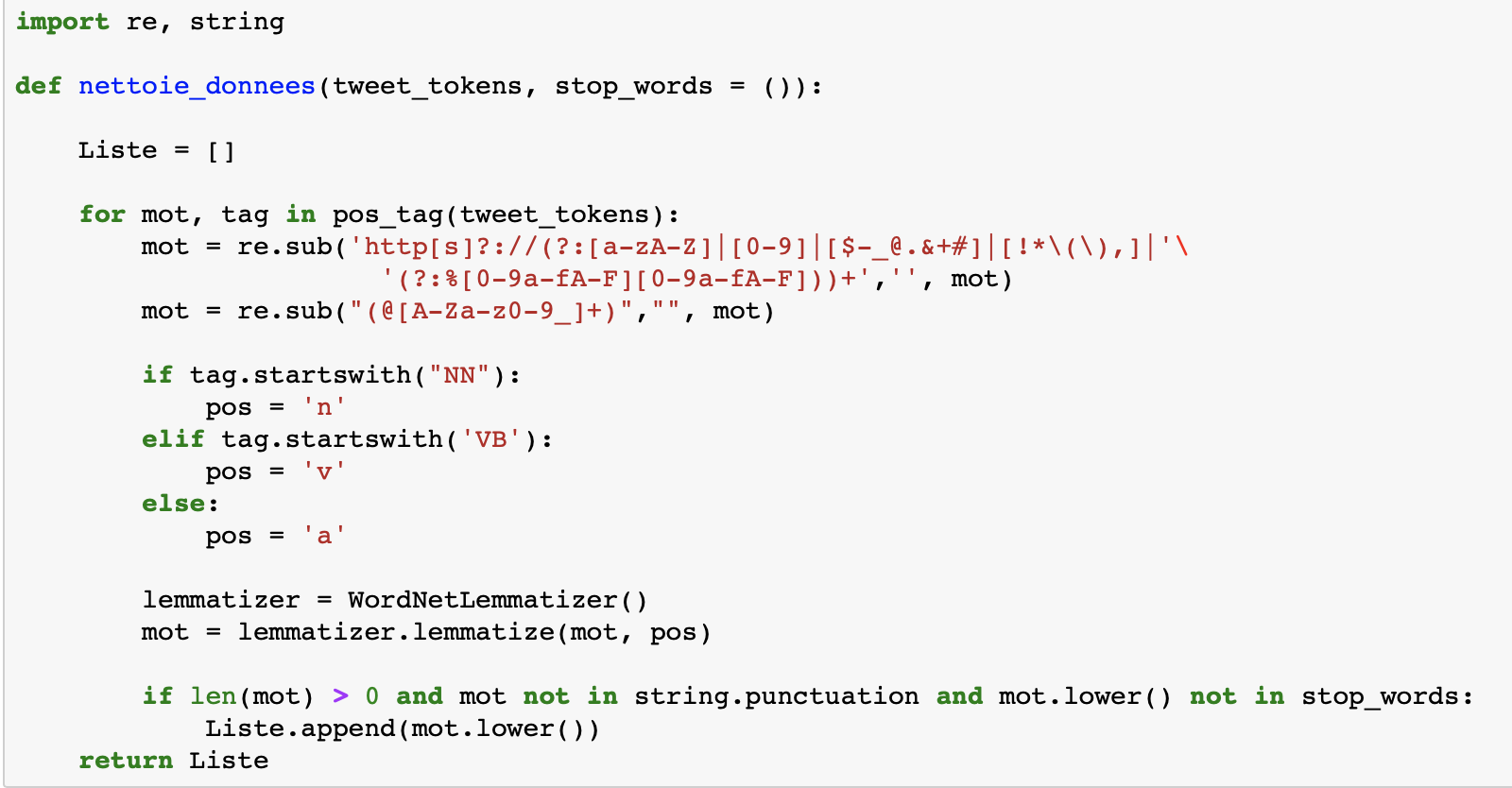
Le module *WordNetLemmatizer* de NTLK dispose de la fonction *lemmatize* qui reçoit un mot et sa position (si nom, verbe ou adjectif) et retourne sa forme basique.

On crée alors la fonction simplifier phrase quireçoit une phrase déjà séparé puis rend tous les mots de la phrase en leurs forme basique :

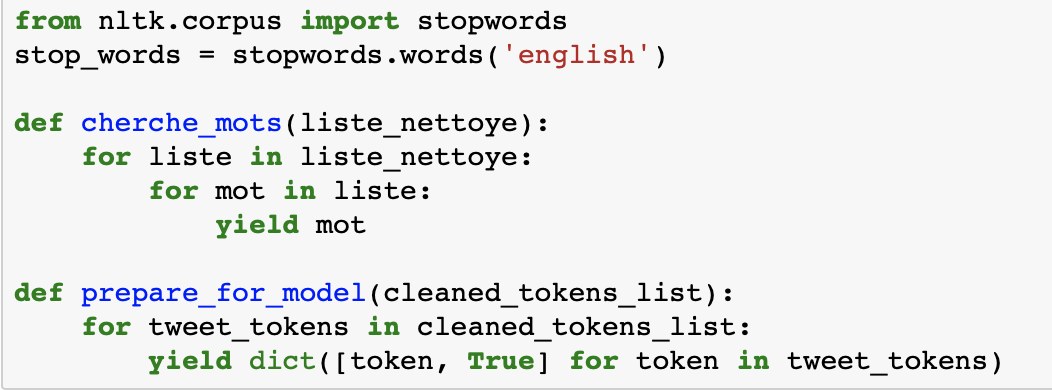


Après on nettoie nos données, on enlevant les hyperliens, les noms d’utilisateurs Twitter précédés par @, les ponctuations, les caractères spéciaux ou les mots inutiles ( par exemple « and » ou « or » ….).

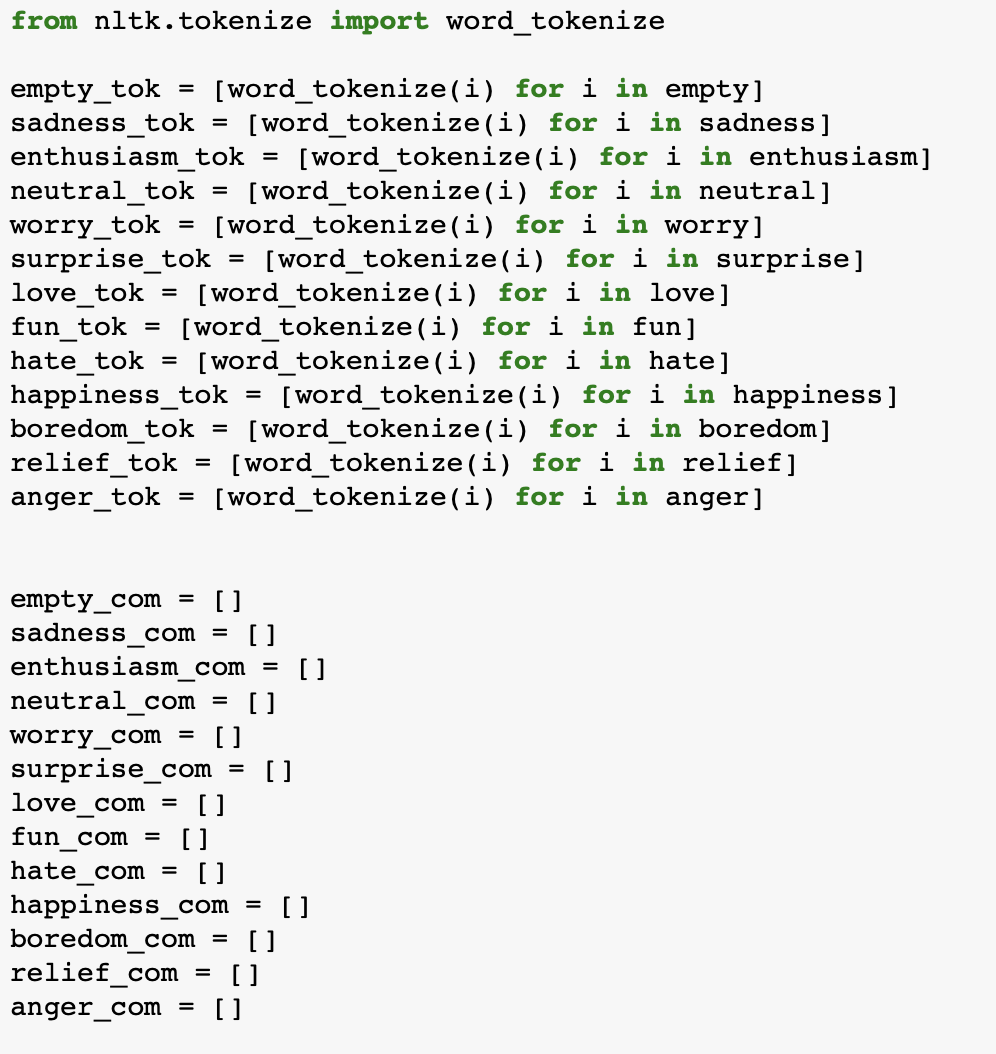
Pour optimiser, on crée une fonction qui à la fois nettoie les données et rend les mots en leur forme basique :



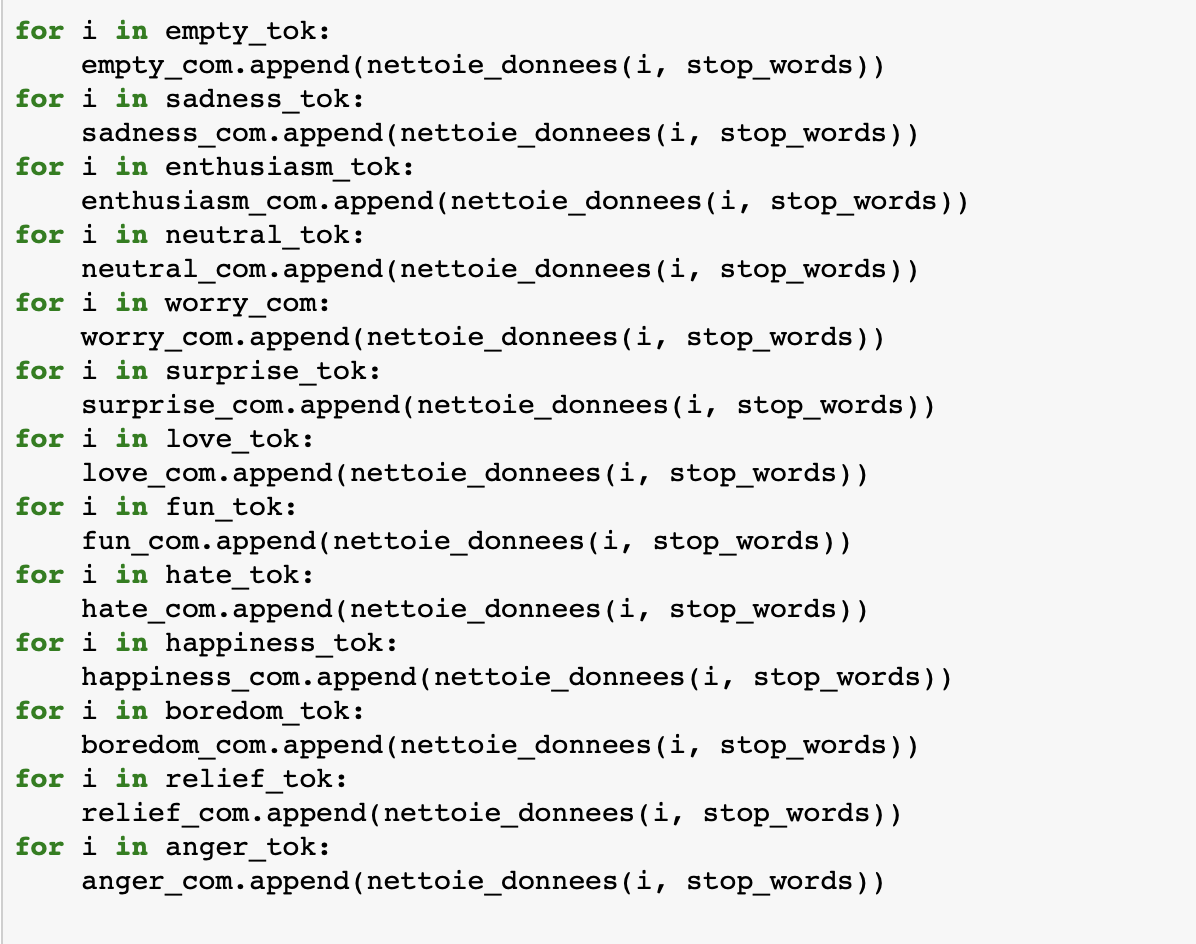
On prépare les données en les transformant en des dictionnaires en préparation à l’utilisation du classifieur de Bayes.

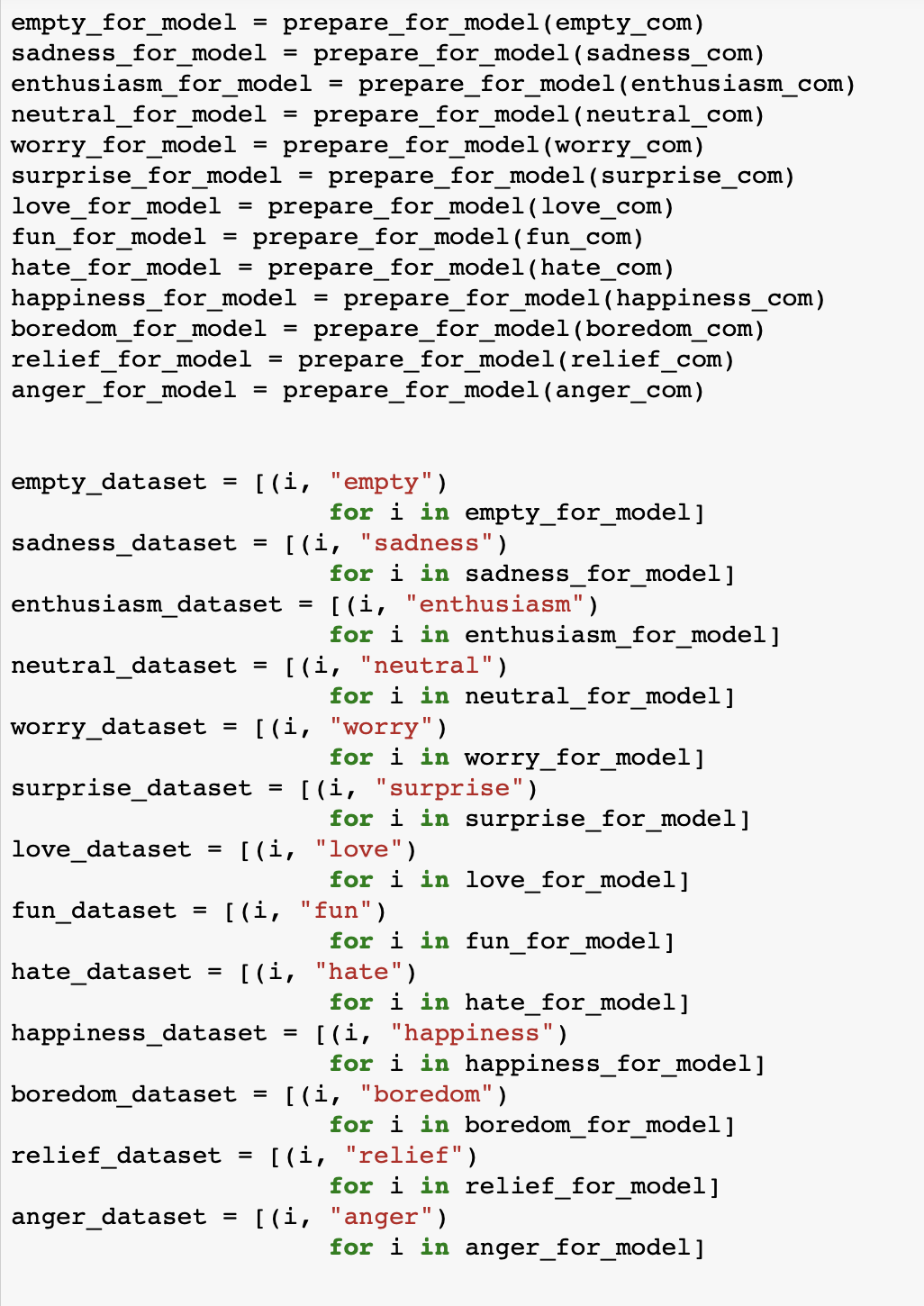


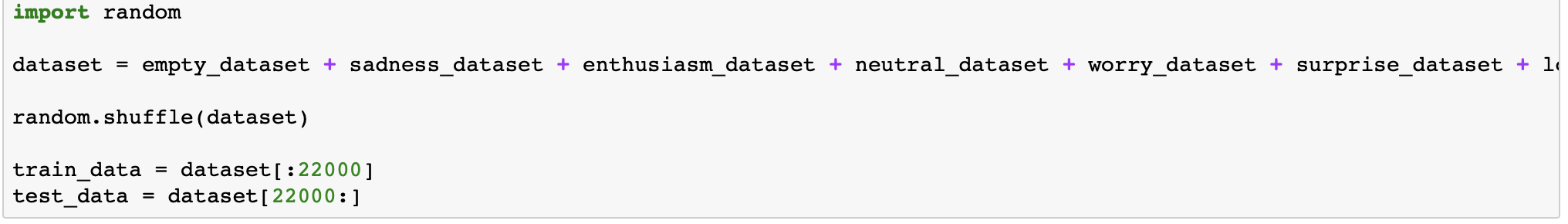
Après on réalise le script, on commence par séparer chaque phrase mot par mot en utilisant la fonction *tokenize* :



Après on nettoie les données pour chaque liste :



Puis, on prépare chaque liste au classifieur : 

On mélange les différentes dataset : 

Puis on importe le classifieur :