École nationale supérieure d'informatique et d'analyse des systèmes - ENSIAS

Génie de Data (GD)

Détection des Communautés dans les Réseaux Sociaux

Projet de Text Mining

Réalisé par :
AIT BOUAZZA Zaynab
EL ASLA Mohamed Amine

Encadré par :
Pr.Y TABII

Année academique : 2024 - 2025

Remerciement

Tout d'abord, nous exprimons notre profonde gratitude envers Dieu le Tout-Puissant, source de toute force et de patience, qui nous a guidés tout au long de ce parcours et nous a accordé les ressources nécessaires pour mener à bien ce projet.

Nous souhaitons également exprimer notre sincère reconnaissance à M. Y. Tabii pour son soutien inestimable et son accompagnement attentif, qui ont été des piliers essentiels dans la réalisation de ce travail.

Nos remerciements s'adressent également à Mme Sanaa El Fkih, qui a généreusement accepté de siéger en tant que membre du jury pour évaluer et soutenir notre projet.

Résumé

Ce projet vise à analyser un grand ensemble de tweets issus de la plateforme Twitter afin de détecter automatiquement des communautés d'utilisateurs. Ces communautés sont formées en fonction d'intérêts communs ou de comportements similaires, et leur identification est essentielle pour comprendre les dynamiques sociales et les tendances qui émergent sur la plateforme.

Dans un premier temps, les données sont prétraitées à travers diverses étapes de nettoyage, incluant la suppression des éléments indésirables tels que les URLs, les mentions, les hashtags et les caractères spéciaux. Ensuite, une personnalisation des stopwords est effectuée pour mieux adapter le texte aux spécificités des réseaux sociaux.

Une fois ces transformations réalisées, des techniques de clustering sont appliquées pour regrouper les utilisateurs en communautés thématiques, notamment KMeans, DBSCAN et Agglomerative. Ces algorithmes permettent d'identifier des groupes d'utilisateurs partageant des intérêts ou des comportements similaires, ce qui facilite la détection de communautés sur la plateforme. Le projet utilise des outils techniques tels que Python, NLTK, Scikit-learn, Kaggle et Google Colab pour entraîner les modèles, ajuster les hyperparamètres et analyser les données à grande échelle.

Mots-clés : Text Mining, Clustering, Communautés d'utilisateurs, Twitter, KMeans, DBSCAN, Agglomerative Clustering, NLTK, Tokenisation, Stopwords, Lemmatisation

Abstract

This project aims to analyze a large set of tweets from the Twitter platform to automatically detect user communities. These communities are formed based on common interests or similar behaviors, and their identification is essential for understanding the social dynamics and trends that emerge on the platform.

Initially, the data is preprocessed through various cleaning steps, including the removal of undesirable elements such as URLs, mentions, hashtags, and special characters. Next, a customization of stopwords is performed to better adapt the text to the specificities of social media.

Once these transformations are applied, clustering techniques are used to group users into thematic communities, including KMeans, DBSCAN, and Agglomerative clustering. These algorithms help identify groups of users sharing similar interests or behaviors, making it easier to detect communities on the platform. The project uses technical tools such as Python, NLTK, Scikit-learn, Kaggle, and Google Colab to train models, adjust hyperparameters, and analyze large-scale data.

Keywords : Text Mining, Clustering, User Communities, Twitter, KMeans, DBSCAN, Agglomerative Clustering, NLTK, Tokenization, Stopwords, Lemmatization.

Table des matières

Table des figures

Introduction Générale

Dans le contexte actuel, les réseaux sociaux comme Twitter génèrent une quantité massive de données chaque jour, offrant une mine d'informations sur les opinions, les comportements et les interactions des utilisateurs. Cependant, l'exploitation de ces données pour en extraire des informations significatives reste un défi majeur, en particulier en ce qui concerne l'identification des communautés d'utilisateurs. Ces communautés, basées sur des intérêts communs ou des comportements similaires, peuvent être cruciales pour comprendre les dynamiques sociales et les tendances qui émergent sur la plateforme.

Le projet que nous présentons se concentre sur l'utilisation de techniques de text mining et de clustering pour analyser un grand ensemble de tweets, afin de détecter automatiquement des communautés d'utilisateurs sur Twitter. L'objectif est de structurer les données non seulement à travers des analyses de texte, mais aussi en appliquant des méthodes de clustering pour regrouper les utilisateurs selon leurs affinités thématiques.

Ce projet est structuré en plusieurs sections clés. Dans le premier chapitre intitulé "Cadre général du projet", nous posons les bases théoriques et techniques du projet. Le deuxième chapitre est dédié au prétraitement des données, où nous détaillons les étapes de nettoyage et de transformation des tweets avant d'entamer l'analyse. Enfin, le troisième chapitre aborde le clustering, où nous appliquons des modèles pour identifier et regrouper les communautés d'utilisateurs en fonction de leurs comportements et intérêts.

Chapitre 1

Cadre général du projet

1.1 Introduction

Ce premier chapitre vise à poser les bases de notre projet de text mining, qui consiste à analyser un large volume de tweets afin de déceler des communautés d'intérêts et d'opinions partagées. En introduisant les objectifs, la problématique, le périmètre, ainsi que les données utilisées, ce chapitre établit un cadre clair pour l'étude et prépare le terrain pour l'analyse des résultats à venir.

1.2 Le Constat :

À l'ère du numérique, les réseaux sociaux sont devenus un vecteur majeur d'échange et de partage d'informations, avec des millions de personnes exprimant leurs opinions, intérêts, et émotions chaque jour. Twitter (renommé X), en particulier, est une plateforme privilégiée pour la diffusion d'opinions publiques sur des sujets variés, allant des actualités politiques aux tendances culturelles et aux avis sur des produits ou services. Ces plateformes créent ainsi une opportunité unique pour observer et analyser les interactions humaines à grande échelle.

Cependant, la masse de données générée en continu par les utilisateurs rend leur analyse complexe. Les informations y sont souvent non structurées et brèves, rendant difficile l'identification de groupes homogènes ou de communautés qui partagent des centres d'intérêt ou des opinions communes. Face à cet afflux de données, des techniques de text mining sont devenues nécessaires pour structurer ces informations et en extraire des connaissances utiles.

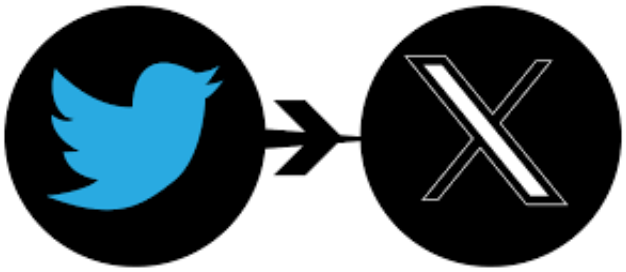

Figure 1.1 – Twitter Logo (renommé X)

1.3 La Problématique :

Dans ce contexte, l'identification automatique de communautés d'utilisateurs sur Twitter représente un défi méthodologique et technique majeur. Bien que les techniques de text mining permettent de structurer partiellement le contenu généré, leur mise en œuvre pour détecter des groupes d'intérêts homogènes reste complexe en raison de plusieurs contraintes.

D'une part, les tweets, par leur format limité en caractères, ne fournissent que des indices partiels sur les sujets abordés, rendant difficile l'identification de thèmes de manière robuste. En outre, le langage utilisé sur les réseaux sociaux est souvent informel et varié, incluant fautes, abréviations, émoticônes et hashtags. Ces spécificités textuelles rendent le traitement des données par des méthodes traditionnelles de traitement du langage naturel (NLP) plus ardu et nécessitent des adaptations spécifiques.

D'autre part, l'identification de communautés d'utilisateurs va au-delà de la simple structuration textuelle : elle implique le regroupement des utilisateurs selon des similarités thématiques et comportementales plus profondes. Cette tâche exige des techniques de clustering avancées, capables de détecter des modèles cachés et d'organiser les vastes volumes de données en groupes exploitables et significatifs.

La problématique se pose donc comme suit : **comment appliquer des techniques de text mining et de clustering pour identifier efficacement des communautés d'utilisateurs sur Twitter, en tenant compte des particularités linguistiques et des caractéristiques spécifiques de la plateforme ?**

1.4 Les Objectifs :

Pour répondre à la problématique identifiée, ce projet vise à développer une méthode permettant d'identifier et d'analyser des communautés d'utilisateurs sur Twitter en se basant leurs centres d'intérêt communs. Les objectifs principaux et secondaires sont détaillés comme suit :

1. **Objectif Principal :**
   - Détecter des communautés d'utilisateurs sur Twitter en regroupant les tweets par thématique, de manière à obtenir une vision structurée et exploitable de l'opinion publique.

2. **Objectifs Secondaires :**
   - **Prétraitement des Données :** Nettoyer et transformer les tweets pour faciliter l'analyse textuelle, en prenant en compte les spécificités de la plateforme, comme les abréviations, les émojis, et les hashtags.
   - **Détection des Thèmes :** Identifier les sujets ou thématiques principaux dans les tweets à l'aide de techniques de text mining, comme le topic modeling, afin de regrouper les tweets par intérêt commun.
   - **Application d'un Modèle de Clustering :** Utiliser des techniques de clustering pour regrouper les utilisateurs en communautés distinctes, en tenant compte des thèmes abordés dans leurs tweets.
   - **Évaluation des Résultats :** Mesurer la performance des modèles de clustering en utilisant des métriques pertinentes pour garantir la pertinence et l'efficacité de l'approche.

1.4.1 Planification du Projet

La planification du projet est une étape essentielle pour assurer son bon déroulement et sa réussite. Dans cette section, nous présentons la planification détaillée du projet à l'aide d'un diagramme de Gantt. L'objectif de cette planification est de définir les tâches spécifiques nécessaires à la réalisation du projet, d'estimer la durée de chaque tâche, de séquencer leur exécution de manière logique, et d'identifier les dépendances entre les différentes tâches.

La planification du projet a été réalisée en plusieurs étapes :

1. **Identification des tâches** : Nous avons identifié toutes les tâches nécessaires à la réalisation du projet, en tenant compte des objectifs définis. Cette étape critique permet de s'assurer que toutes les activités requises sont prises en compte et correctement planifiées.

2. **Estimation de la durée** : Pour chaque tâche, nous avons estimé la durée nécessaire à son accomplissement, en prenant en compte les ressources disponibles et les contraintes du projet. Une estimation précise de la durée est essentielle pour établir un calendrier réaliste et gérable.

3. **Séquencement des tâches** : Les tâches ont été séquencées de manière logique, en tenant compte des dépendances entre elles. Cela nous permet de déterminer l'ordre dans lequel les tâches doivent être exécutées, garantissant ainsi une progression fluide et efficace du projet.

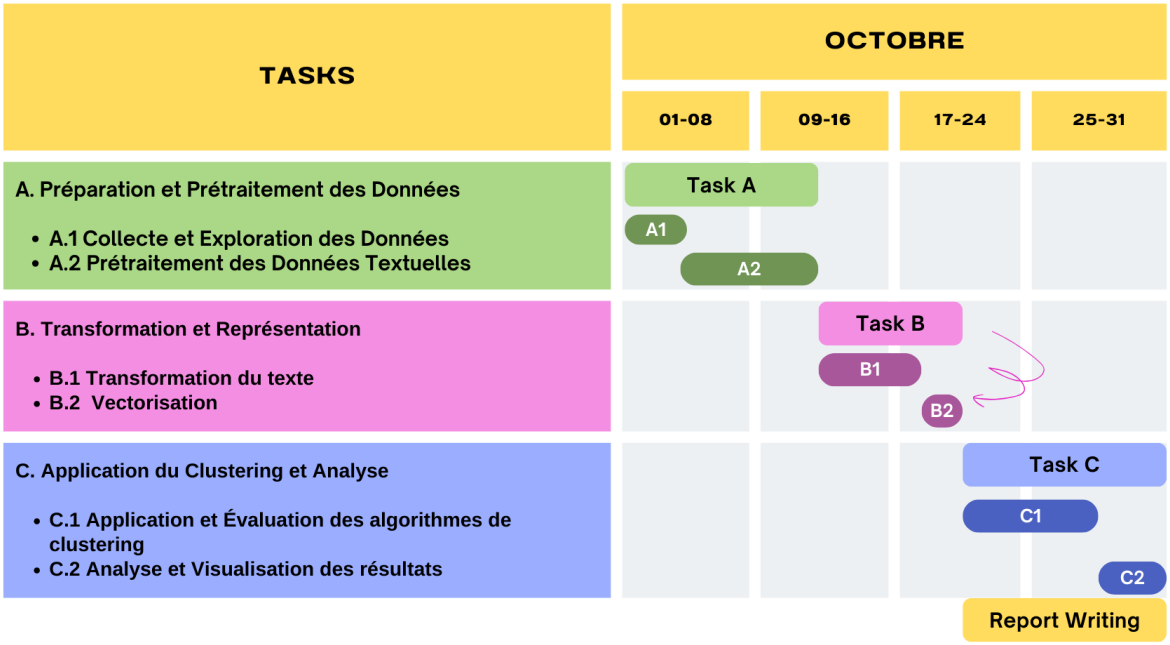

FIGURE 1.2 – Diagramme de Gantt

1.5 Le Périmètre :

Ce projet de text mining appliqué aux données de Twitter vise à identifier et analyser des communautés d'utilisateurs, en respectant les contraintes techniques et les objectifs définis. Les principaux aspects du périmètre sont les suivants :

- **Types de Données Utilisées :** Le projet est centré sur les données textuelles issues de tweets, essentielles pour le text mining. Les autres formats de données (images, vidéos) ne sont pas pris en compte.
- **Étendue du Prétraitement des Données :** Le prétraitement inclut le nettoyage des données textuelles (suppression de caractères spéciaux, normalisation de mots, gestion des hashtags et émojis..). Ce périmètre exclut toute optimisation avancée pour d'autres tâches de NLP non liées à au clustering.
- **Modèles d'Analyse de Clustering :** La méthodologie inclut l'implémentation de techniques de clustering pour segmenter les utilisateurs en groupes homogènes. Seuls les algorithmes de machine learning classiques (ex : K-means, DBSCAN) sont utilisés, sans recourir à des réseaux de neurones ou techniques de deep learning nécessitant des ressources intensives.
- **Limites de l'Analyse des Communautés :** L'analyse des communautés se limite à la similarité des thèmes exprimés dans les tweets, sans étudier les interactions sociales (par exemple, réseaux de follow ou graphes d'interactions).
- **Outils et Technologies Utilisés :** Les outils se limitent aux bibliothèques open-source de Python telles que NLTK, Scikit-learn, et Pandas, pour le traitement de texte, l'analyse de sentiment, et le clustering. Les solutions nécessitant des ressources matérielles élevées (ex : GPU, Cloud) sont hors périmètre.

1.6 Présentation des Données :

Pour ce projet, nous avons utilisé le dataset Sentiment140, un jeu de données populaire pour l'analyse sur Twitter. Ce dataset se compose de 1,6 million de tweets collectés à partir de la plateforme Twitter, chacun accompagné de métadonnées permettant de mener une analyse approfondie des thèmes abordés. Les principales caractéristiques du dataset sont les suivantes :

1. **Description du Dataset :**
   - **Source :** Les tweets proviennent de la plateforme Twitter et sont pré-étiquetés en fonction de leur polarité.
   - **Volume de Données :** Le dataset contient 1,6 million de tweets, ce qui permet d'obtenir des résultats statistiquement significatifs et de faire émerger des tendances générales dans l'expression des opinions.

2. **Colonnes Principales :**
   - **target :** Indique la polarité du tweet (0 pour négatif et 4 pour positif). Ce champ est utilisé comme étiquette pour les modèles d'analyse de sentiment.
   - **ids :** Identifiant unique du tweet, utile pour le suivi et l'identification spécifique de chaque observation.
   - **date :** Date et heure de publication du tweet, permettant de comprendre la temporalité des sujets abordés et des sentiments exprimés.

— **flag :** Requête ou mot-clé utilisé lors de la recherche de tweets. Lorsque ce champ est vide, il est marqué par "NO_QUERY".
— **user :** Nom d'utilisateur Twitter de l'auteur du tweet. Ce champ permet de différencier les utilisateurs, bien que l'analyse reste anonyme.
— **text :** Contenu textuel du tweet, qui est l'élément principal utilisé pour les analyses de sentiment et de clustering. Le texte des tweets est souvent informel et inclut des éléments spécifiques à Twitter (hashtags, mentions, abréviations, etc.).

3. **Propriétés et Spécificités des Données :**
— **Langue :** Tous les tweets sont en anglais, ce qui simplifie le traitement linguistique en évitant d'avoir à gérer plusieurs langues.
— **Longueur des Textes :** Les tweets étant limités en nombre de caractères, les messages sont généralement courts et directs, mais peuvent inclure des abréviations, des hashtags, et des émojis.
— **Non-structuration :** Les tweets contiennent souvent un texte non structuré, avec des expressions informelles et des fautes d'orthographe. Cela nécessite un prétraitement minutieux pour transformer ces données en format utilisable par les modèles de machine learning.

https ://www.kaggle.com/datasets/kazanova/sentiment140

Ce dataset présente ainsi une base riche en informations pour mener une étude de text mining sur Twitter, notamment en ce qui concerne la détection de communautés.

1.7 Conclusion

Ce premier chapitre a permis de présenter les éléments fondamentaux du projet en définissant la problématique, les objectifs et le périmètre d'analyse. En nous appuyant sur le dataset Sentiment140, nous avons établi une base solide pour mener une étude approfondie sur la détection de communautés sur Twitter.

Chapitre 2

Prétraitement des Données

2.1 Introduction :

Pour que les données textuelles issues des tweets puissent être utilisées efficacement dans les modèles de text mining, il est essentiel de suivre plusieurs étapes de prétraitement. Ce chapitre détaille le processus de préparation des données pour extraire des informations exploitables et atteindre les objectifs du projet.

2.2 Outils Techniques :

2.2.1 Python :

Python est un langage de programmation polyvalent et très populaire dans le domaine de l'analyse de données et du machine learning. Sa syntaxe simple et lisible facilite l'écriture et la compréhension du code, le rendant accessible aux débutants tout en offrant une puissance considérable pour les experts. Python dispose d'un large éventail de bibliothèques spécialisées : Numpy et Pandas pour la manipulation de données, Matplotlib et Seaborn pour la visualisation, ainsi que Scikit-learn pour les modèles de machine learning. Son écosystème en fait un choix privilégié pour des tâches variées, allant du nettoyage des données et de l'analyse exploratoire à la modélisation prédictive et au traitement du langage naturel.

Figure 2.1 – Logo de Python

2.2.2 NLTK :

NLTK (Natural Language Toolkit) est une bibliothèque Python essentielle pour le traitement et l'analyse du langage naturel (NLP). Elle propose une collection complète d'outils pour effectuer des tâches variées, comme la tokenisation, la suppression de mots vides (stopwords), la racinisation (stemming) et la lemmatisation. NLTK permet également de manipuler de grandes quantités de texte, facilitant ainsi l'analyse linguistique et la création de modèles de machine learning pour des applications en text mining.

2.2.3 Visual Studio Code :

Visual Studio Code (VS Code) est un éditeur de code open-source, léger et polyvalent, développé par Microsoft. Conçu pour supporter de nombreux langages de programmation, dont Python, il propose des fonctionnalités avancées comme l'autocomplétion, le débogage intégré, et la gestion des extensions, permettant aux utilisateurs de personnaliser leur environnement de développement. Grâce à ses fonctionnalités de contrôle de version Git, son intégration facile avec des outils de développement cloud, et son interface intuitive, VS Code est très apprécié des développeurs pour son efficacité et sa flexibilité.

FIGURE 2.2 – Logo de Visual Studio Code

2.3 Analyse Exploratoire des Données (EDA) :

L'analyse exploratoire des données (EDA) est essentielle pour comprendre la structure, les tendances et les relations présentes dans notre dataset. Dans cette section, nous examinerons divers aspects des données du Sentiment140 dataset afin d'en extraire des informations significatives.

2.3.1 Aperçu des Données :

Le Sentiment140 dataset contient 1,6 million de tweets annotés avec des sentiments. Chaque tweet est associé à plusieurs attributs, dont l'ID, la date, le texte du tweet, l'utilisateur, et la polarité (0 pour négatif, 4 pour positif).

Afin de mieux comprendre la structure des données, nous avons exploré un aperçu des premières lignes du dataset en affichant les cinq premières entrées via **df.head()**. Cela permet de visualiser les différentes colonnes et leurs types de données pour s'assurer qu'elles correspondent aux attentes.

```
print(df.head())
```

```
Mounted at /content/drive
   polarity          id                          date     query  \
0         0  1467810369  Mon Apr 06 22:19:45 PDT 2009  NO_QUERY
1         0  1467810672  Mon Apr 06 22:19:49 PDT 2009  NO_QUERY
2         0  1467810917  Mon Apr 06 22:19:53 PDT 2009  NO_QUERY
3         0  1467811184  Mon Apr 06 22:19:57 PDT 2009  NO_QUERY
4         0  1467811193  Mon Apr 06 22:19:57 PDT 2009  NO_QUERY

              user                                                text
0  _TheSpecialOne_  @switchfoot http://twitpic.com/2y1zl - Awww, t...
1    scotthamilton  is upset that he can't update his Facebook by ...
2         mattycus  @Kenichan I dived many times for the ball. Man...
3          ElleCTF    my whole body feels itchy and like its on fire
4           Karoli  @nationwideclass no, it's not behaving at all....
```

FIGURE 2.3 – Premières lignes du dataset

En parallèle, l'affichage de **df.info()** fournit des informations utiles sur chaque colonne, notamment le type de données et le nombre de valeurs non-nulles pour détecter la présence éventuelle de valeurs manquantes ou de doublons. Cette inspection préliminaire aide à confirmer la cohérence et la complétude des données, garantissant qu'elles sont bien structurées pour les étapes de nettoyage et d'analyse ultérieures.

```
df_sampled.info()
```

```
Mounted at /content/drive
<class 'pandas.core.frame.DataFrame'>
Index: 160000 entries, 541200 to 427964
Data columns (total 6 columns):
 #   Column    Non-Null Count   Dtype 
---  ------    --------------   ----- 
 0   polarity  160000 non-null  int64 
 1   id        160000 non-null  int64 
 2   date      160000 non-null  object
 3   query     160000 non-null  object
 4   user      160000 non-null  object
 5   text      160000 non-null  object
```

FIGURE 2.4 – Informations sur le dataset

2.3.2 Longueur des Tweets :

La longueur d'un tweet est mesurée ici par le nombre de caractères dans le texte. L'analyse de cette longueur fournit des informations utiles sur la structure typique des messages et aide à adapter les étapes de prétraitement.

Pour mieux comprendre cette distribution, nous avons visualisé la fréquence des différentes longueurs de tweets à l'aide d'un histogramme.

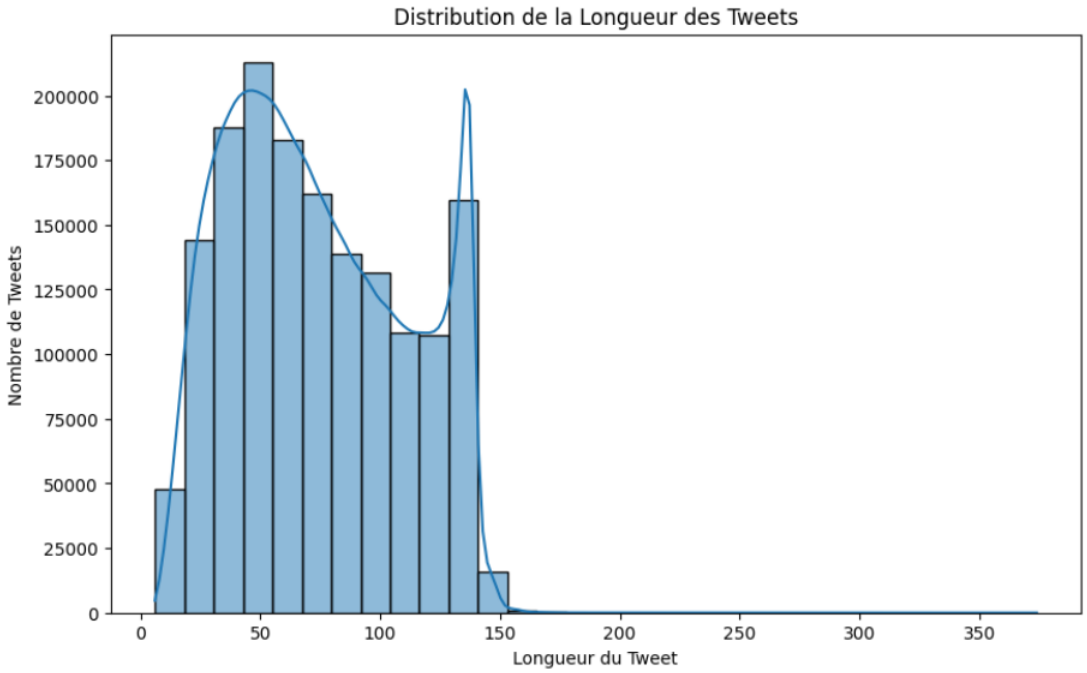

FIGURE 2.5 – Distribution de la longueur des tweets

— La majorité des tweets ont une longueur située entre 0 et 150 caractères, ce qui reflète une préférence pour les messages courts et concis.
— Un pic notable apparaît autour de 50 caractères, indiquant une tendance à formuler des idées simples et directes dans un format court.
— Une baisse marquée autour de 140-150 caractères, reflétant probablement l'ancienne limite de 140 caractères imposée par Twitter.

2.3.3 Analyse des Utilisateurs et de l'Activité :

L'analyse des utilisateurs les plus actifs sur Twitter peut révéler des profils influents ou des comptes à forte activité. Dans ce contexte, nous avons identifié les utilisateurs les plus actifs en comptant le nombre de tweets publiés par chaque utilisateur et en extrayant les 10 utilisateurs ayant posté le plus grand nombre de tweets.

Pour visualiser ces données, un barplot a été utilisé pour montrer le nombre de tweets de chaque utilisateur dans ce groupe.

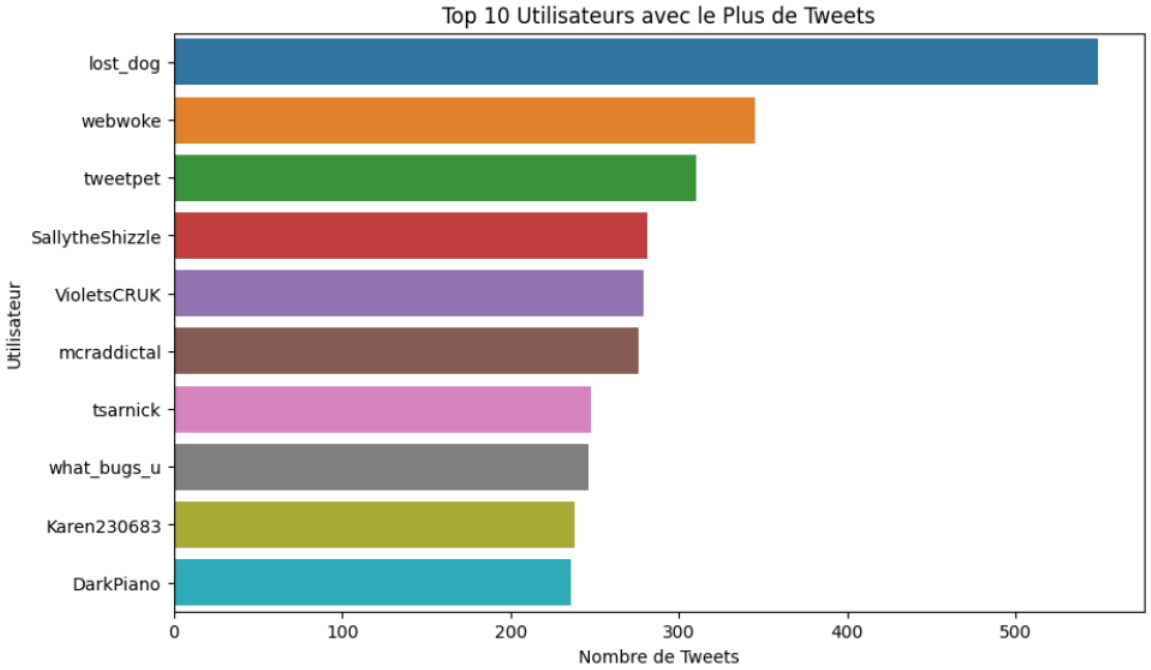

FIGURE 2.6 – Top 10 des utilisateurs ayant le plus de tweets

Ces utilisateurs, par leur fréquence de publication, peuvent influencer la distribution globale des thèmes dans le dataset. Cette concentration d'activité peut également indiquer la présence de comptes automatisés ou de profils spécialisés dans des sujets spécifiques, ce qui pourrait introduire un certain biais dans les données.

2.3.4 Analyse de Hashtags

Les hashtags jouent un rôle essentiel dans l'expression de sujets et de tendances spécifiques sur Twitter. Leur analyse permet d'obtenir des informations sur les sujets d'intérêt et les tendances populaires au sein des communautés d'utilisateurs. Dans cette section, nous présentons l'analyse des hashtags les plus fréquents dans notre dataset, ainsi que les observations issues de cette analyse.

Nous avons extrait les hashtags présents dans les tweets afin de déterminer les thèmes récurrents et les sujets d'actualité. L'extraction a été réalisée en identifiant les mots précédés du symbole # dans le texte des tweets.

Les résultats montrent que le hashtag #followfriday est le plus fréquemment utilisé, avec une fréquence de plus de 2000 occurrences. D'autres hashtags populaires incluent #fb, #squarespace, et #FF. Ces hashtags reflètent les interactions sociales et les campagnes de promotion souvent présentes sur la plateforme.

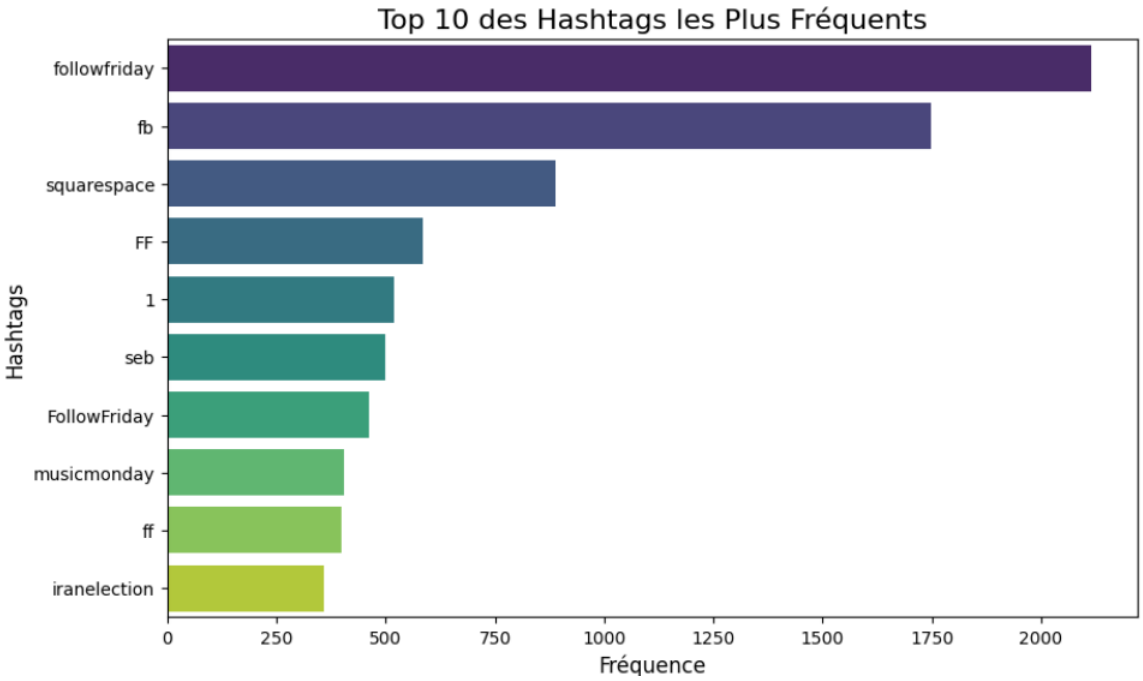

FIGURE 2.7 – Top 10 des Hashtags les Plus Fréquents

2.4 Nettoyage des Données :

Le nettoyage des données est une étape cruciale pour garantir la qualité et la pertinence des informations avant toute analyse ou modélisation.

2.4.1 Suppression de Colonnes Indésirables :

Certaines colonnes du Sentiment140 dataset sont inutiles pour notre analyse et modélisation, et leur suppression permet de réduire la complexité des données :

- **ID :** Cette colonne est un identifiant unique pour chaque tweet mais n'apporte aucune valeur analytique directe.
- **Target :** Cette colonne indique le sentiment associé à chaque tweet, mais comme notre analyse se concentre sur la détection de communautés sans analyse de sentiment, cette information est également omise.
- **Date** : Bien que la colonne Date puisse être utile dans une analyse temporelle pour observer des tendances au fil du temps, elle n'est pas exploitée dans ce projet et sera donc supprimée.
- **Flag :** Cette colonne indique les requêtes associées aux tweets, mais comme toutes les valeurs sont égales à "NO_QUERY", elle n'apporte aucune information pertinente pour l'analyse.

Après cette étape, seules les colonnes Target, User, et Text sont conservées pour la suite de l'analyse.

2.4.2 Élimination des Caractères Indésirables :

Le texte des tweets contient souvent des caractères indésirables ou des éléments superflus qui peuvent nuire à la qualité de l'analyse textuelle. Les étapes suivantes sont appliquées pour nettoyer le texte :

— **URLs et mentions :** Les liens web (URLs) et les mentions d'autres utilisateurs (commençant par "@") sont supprimés, car ils ne fournissent généralement pas d'informations pertinentes pour l'identification de communautés et peuvent introduire du bruit dans les données.
— **Émoticônes et symboles spéciaux :** Bien que certains symboles et émoticônes puissent indiquer des émotions, ils nécessitent des traitements supplémentaires pour être exploitables. Ils sont donc éliminés afin de simplifier le texte, se concentrant sur le contenu verbal pour l'analyse de communautés.
— **Ponctuation et caractères spéciaux :** La ponctuation excessive et les caractères spéciaux (tels que "&", "%", "#;; etc.) sont supprimés pour homogénéiser le texte. Cette étape permet de simplifier l'analyse, de réduire le bruit dans les donnée.
— **Extraction des hashtags :** Les hashtags, qui commencent par le caractère "#", peuvent contenir des informations utiles sur le sujet ou le thème du tweet. Pour conserver ces informations tout en simplifiant le texte, les mots des hashtags sont extraits et le symbole # est supprimé. Ainsi, les mots-clés sont intégrés au texte comme des termes normaux, facilitant l'analyse des centres d'intérêt des utilisateurs et la détection de communautés thématiques.

Ces étapes de nettoyage préparent le dataset pour la phase suivante, où les textes pourront être transformés et représentés de manière appropriée pour les techniques de text mining et de clustering.

2.5 Transformation du Texte :

La transformation du texte est une étape essentielle qui consiste à convertir les données textuelles brutes en une forme exploitable par les modèles de text mining et d'analyse de données. Cette transformation permet d'uniformiser le texte et de faciliter son interprétation pour les algorithmes d'apprentissage automatique.

2.5.1 Mise en minuscule :

La première étape consiste à convertir tout le texte en minuscules. Cette normalisation permet de traiter les mots de manière uniforme, sans distinguer les majuscules des minuscules. Par exemple, "Happy" et "happy" seront considérés comme le même mot, ce qui réduit les variations inutiles et simplifie l'analyse.

2.5.2 Tokenisation :

La tokenisation consiste à diviser chaque tweet en unités de mots appelées tokens, ce qui permet d'analyser le texte mot par mot et de faciliter les étapes de nettoyage et de transformation qui suivent. Pour cette étape, nous avons utilisé la fonction word_tokenize de la bibliothèque NLTK (Natural Language Toolkit), une méthode fiable et performante pour segmenter le texte en mots.

2.5.3 Suppression des mots vides (stopwords) :

Les mots vides (ou stopwords) sont des mots très fréquents qui n'apportent généralement pas de valeur analytique directe. Dans le contexte des tweets, les mots vides incluent à la fois des mots courants en anglais (comme "and", "the", "is") et des termes spécifiques aux réseaux sociaux (par exemple, "lol", "u", "rt") qui n'ajoutent pas de contenu sémantique pertinent. En les supprimant, nous réduisons le bruit dans les données, ce qui permet de concentrer l'analyse sur les mots porteurs de sens.

Dans un premier temps, nous avons utilisé la bibliothèque nltk pour obtenir une liste de mots vides (stopwords) standard. Cependant, après avoir visualisé les 30 mots les plus fréquents, nous avons constaté que de nombreux mots informels spécifiques aux réseaux sociaux apparaissaient régulièrement. Par exemple, des abréviations courantes telles que "im" pour "I'm", "u" pour "you", "gotta" pour "got to", et "cant" pour "can't", ainsi que des acronymes comme "lol" (laughing out loud) et des termes comme "rt" (pour retweet) et "amp" (pour and après certains liens) étaient présents, mais ne contribuaient pas à l'analyse des contenus.

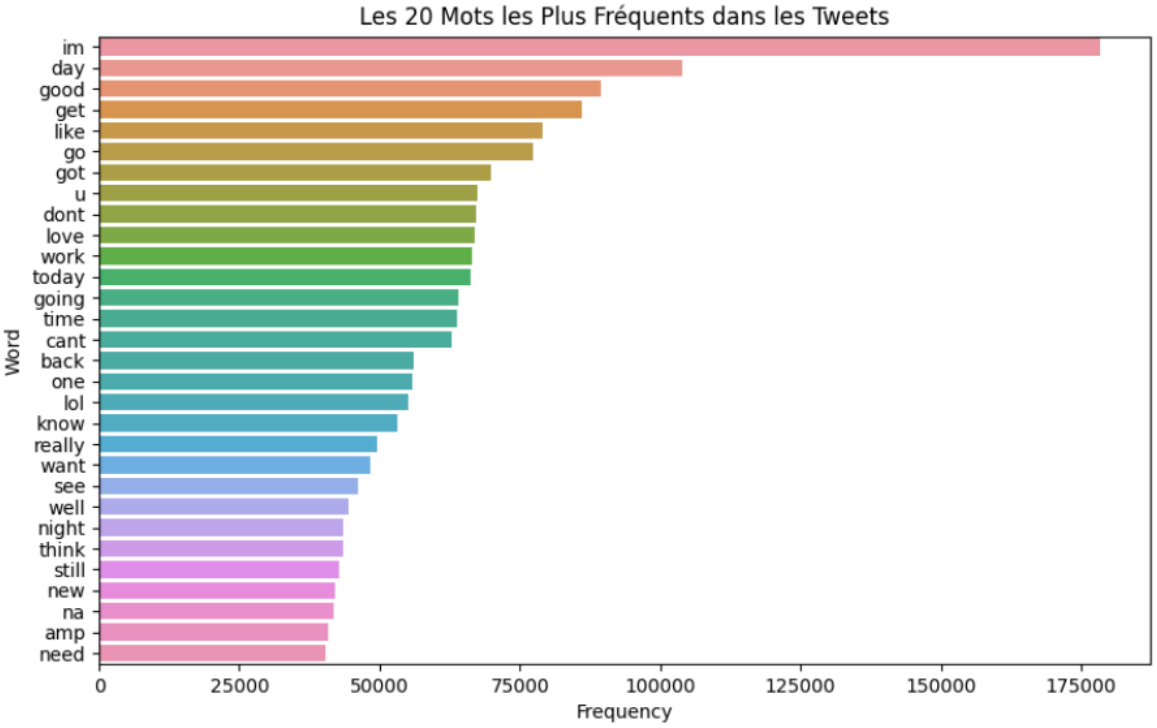

FIGURE 2.8 – Les 30 mots les plus fréquents

Afin d'améliorer la pertinence du prétraitement des textes, nous avons donc choisi de définir notre propre liste de mots vides, spécifiquement adaptée à l'analyse des tweets. Cette liste comprend des termes informels et des abréviations souvent utilisées sur les réseaux sociaux, mais qui ne fournissent aucune information utile pour l'analyse des communautés d'intérêts et des comportements des utilisateurs.

Liste des mots vides personnalisés : ['im', 'dont', 'cant', 'gotta', 'wanna', 'gonna', 'aint', 'lol', 'lmao', 'u', 'ur', 'idk', 'dm', 'like', 'go', 'day', 'good', 'great', 'really', 'time', 'today', 'back', 'one', 'see', 'omg', 'hey', 'hi', 'hello', 'nah', 'yeah', 'yep', 'nope', 'pls', 'haha', 'hehe', 'thanks', 'thank', 'ok', 'okay', 'alright', 'amp', 'rt', 'oh', 'ah', 'uh', 'heyre', 'shes', 'hes', 'thm', 'wat', 'wut', 'hoo', 'thats', 'dis', 'dat', 'thse', 'wuz', 'wer', 'bein', 'havent', 'havnt', 'hav', 'haz', 'doin', 'dosent', 'didnt', 'cuz', 'bcuz', 'becuz', 'coz', 'bcoz', 'get', 'got', 'shoulda', 'becoz', 'til', 'till', 'az', 'whl', 'whle', 'abt', 'bout', 'btwn', 'thru', 'durin', 'abv', 'frm', 'ovr', 'wen', 'hw', 'othr', 'shld', 'shouldnt', 'id', 'tbh', 'bff', 'btw', 'smh', 'fomo', 'thx', 'lmk', 'fml', 'plz', 'fyi', 'na']

Après avoir appliqué cette suppression de mots vides personnalisée, nous avons visualisé à nouveau les 30 mots les plus fréquents dans le dataset, et nous avons constaté une amélioration dans la qualité des mots restants, ce qui permet une analyse plus ciblée des communautés et des sujets traités sur Twitter.

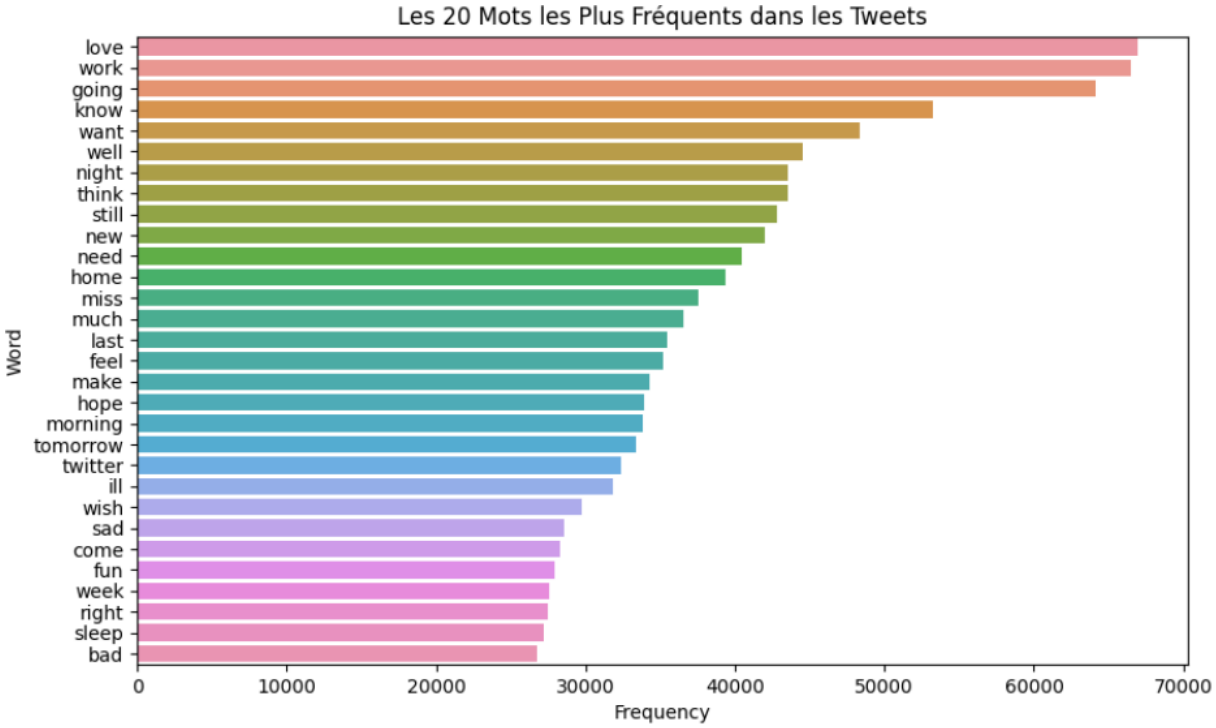

FIGURE 2.9 – 30 mots les plus fréquents après personnalisation des stopwords

2.5.4 Lemmatisation :

La lemmatisation consiste à ramener chaque mot à son lemme, c'est-à-dire sa forme de base ou canonique, en tenant compte de son contexte grammatical. Contrairement à la racinisation (ou stemming), qui se contente de supprimer les suffixes pour obtenir la racine, la lemmatisation analyse le mot dans son contexte pour le transformer en sa forme correcte. Par exemple, "better" est transformé en "good" et "running" en "run". Cette technique est donc souvent plus précise, car elle s'appuie sur des règles linguistiques pour interpréter le sens du mot.

Dans notre projet, nous avons utilisé WordNetLemmatizer de la bibliothèque NLTK, un outil qui exploite la base de données linguistique WordNet pour réaliser la lemmatisation en fonction du type grammatical du mot (nom, verbe, adjectif, etc.). Cette approche permet d'obtenir des mots dans leur forme sémantique correcte, améliorant la qualité de la représentation textuelle pour la détection de communautés.

Pour chaque mot du tweet, WordNetLemmatizer est appliqué en spécifiant sa catégorie grammaticale, lorsque possible. Par exemple, un mot comme "cats" (noms pluriels) sera transformé en "cat", et "ran" (verbe au passé) sera transformé en "run", rendant ainsi le texte plus homogène et pertinent pour les étapes de modélisation.

2.6 Resultat du Netoyage et de Transformation :

Dans cette section, nous présentons les résultats du processus de nettoyage et de transformation appliqué aux tweets. Nous avons mis en évidence les tweets avant et après le nettoyage, ainsi que les effets de la personnalisation des stopwords sur la qualité du texte final.

```
                                                text  \
0  @switchfoot http://twitpic.com/2y1zl - Awww, t...
1  is upset that he can't update his Facebook by ...
2  @Kenichan I dived many times for the ball. Man...
3    my whole body feels itchy and like its on fire
4  @nationwideclass no, it's not behaving at all....

                                      cleaned_text
0     thats bummer shoulda got david carr third day
1  upset cant update facebook texting might cry r...
2    dived many time ball managed save rest go bound
3                    whole body feel itchy like fire
4                           behaving im mad cant see
```

FIGURE 2.10 – Tweet Avant et Après le Nettoyage et la Transformation

Ce nettoyage permet de supprimer les mentions (@user), les hashtags, ainsi que les caractères spéciaux, rendant ainsi le texte plus propre et prêt pour l'analyse suivante. La transformation, quant à elle, permet d'éliminer les stopwords standards et à d'appliquer une lemmatisation, afin de réduire le texte à ses racines tout en préservant le sens des mots.

Avant la personnalisation des stopwords, certains mots informels fréquemment utilisés sur Twitter, comme "im", "cant", "thats", etc., étaient encore présents dans le texte après nettoyage. Cela alourdissait le contenu sans apporter de valeur à l'analyse.

	text	cleaned_text	hashtags
0	@switchfoot http://twitpic.com/2y1zl - Awww, t...	bummer david carr third	[]
1	is upset that he can't update his Facebook by ...	upset update facebook texting might cry result...	[]
2	@Kenichan I dived many times for the ball. Man...	dived many time ball managed save rest bound	[]
3	my whole body feels itchy and like its on fire	whole body feel itchy fire	[]
4	@nationwideclass no, it's not behaving at all....	behaving mad	[]

FIGURE 2.11 – Tweet avec Personnalisation des Stopwords

La personnalisation des stopwords a permis de supprimer des termes informels et des abréviations courantes sur les réseaux sociaux. Cela réduit le bruit dans les données et met en lumière les termes essentiels pour l'analyse.

2.7 Représentation des Textes :

La représentation des textes est une étape cruciale pour transformer les tweets, qui sont sous forme de texte brut, en un format exploitable par les algorithmes de clustering. Les techniques de représentation permettent de convertir les mots en vecteurs numériques, conservant ainsi leur signification et leurs relations entre eux. Nous avons principalement exploré les méthodes suivantes pour représenter les tweets dans notre projet :

2.7.1 Représentation par Sac de Mots (Bag of Words - BoW) :

La méthode du Sac de Mots (BoW) est une approche simple mais efficace pour représenter des textes. Elle consiste à créer un dictionnaire de tous les mots présents dans le corpus, puis à compter le nombre de fois que chaque mot apparaît dans chaque tweet. Bien que cette méthode ne conserve pas l'ordre des mots, elle fournit une première représentation des tweets en capturant la fréquence des mots.

Pour notre projet, nous avons utilisé un vecteur pour chaque tweet, où chaque dimension représente un mot du dictionnaire, et la valeur indique sa fréquence d'apparition dans le tweet. Cependant, cette méthode peut produire des vecteurs très larges et creux (sparse), surtout avec un grand corpus, rendant les calculs moins efficaces.

2.7.2 TF-IDF (Term Frequency - Inverse Document Frequency) :

Le modèle TF-IDF affine la méthode BoW en pondérant les mots en fonction de leur importance dans le corpus. Plutôt que de se baser uniquement sur la fréquence d'apparition, cette approche diminue le poids des mots très fréquents (comme les articles ou mots communs) et augmente celui des mots spécifiques. La formule est la suivante :

$$\text{TF-IDF}(t, d) = \text{TF}(t, d) \times \text{IDF}(t)$$

où :

— **TF** (Term Frequency) correspond à la fréquence d'un mot dans un document spécifique.

— **IDF** (Inverse Document Frequency) est calculé en fonction de l'inverse de la fréquence de documents contenant ce mot dans tout le corpus.

Avec TF-IDF, les mots rares mais pertinents reçoivent un poids plus élevé, ce qui aide à distinguer les tweets contenant des mots porteurs de sens. Cette représentation améliore la pertinence des textes pour l'analyse, notamment pour l'identification de communautés ayant des intérêts communs.

2.8 Conclusion :

Ce chapitre a présenté en détail les différentes étapes de nettoyage et de transformation appliquées, depuis la suppression des éléments indésirables jusqu'à la personnalisation des stopwords. Ces transformations permettent de simplifier le texte, de réduire le bruit dans les données et d'améliorer la cohérence des informations, rendant ainsi les tweets prêts pour l'analyse.

Chapitre 3

Modèle de Clustering

3.1 Introduction :

Dans ce chapitre, nous abordons le processus de création et d'application d'un modèle de clustering pour identifier des communautés d'utilisateurs partageant des intérêts ou opinions similaires. Après le nettoyage et la transformation des données textuelles, le clustering permet de regrouper les tweets en fonction de leurs similarités sémantiques.

3.2 Outils Techniques :

Pour le développement et l'application des modèles de clustering dans notre projet, nous avons sélectionné plusieurs outils techniques et plateformes qui simplifient l'entraînement des modèles et l'expérimentation de diverses méthodes. En complément de **Python** et **Visual Studio Code**, nous avons intégré des bibliothèques et des environnements spécialement adaptés aux nos besoins. Voici un aperçu des principaux outils utilisés :

3.2.1 Scikit-learn :

Scikit-learn est une bibliothèque essentielle pour le machine learning en Python, largement utilisée pour ses fonctionnalités complètes et sa facilité d'utilisation. Elle propose un large éventail d'algorithmes d'apprentissage automatique, comme le clustering, la classification, et la régression, ainsi que des outils pour la réduction de la dimensionnalité et l'évaluation des performances des modèles. Conçue pour être accessible aux débutants tout en offrant des fonctionnalités avancées pour les experts, Scikit-learn est optimisée pour des opérations rapides et efficaces sur de grands ensembles de données.

FIGURE 3.1 – Logo de Scikit-learn

3.2.2 Kaggle :

Kaggle est une plateforme en ligne qui permet aux data scientists et aux passionnés d'intelligence artificielle de collaborer, partager des jeux de données, et participer à des compétitions de machine learning. Elle offre un environnement cloud prêt à l'emploi, avec accès à des notebooks Python et R ainsi qu'à des ressources gratuites pour l'entraînement de modèles. Grâce à une grande variété de jeux de données publics et à une communauté active, Kaggle facilite l'apprentissage pratique et la mise en œuvre de projets de data science.

FIGURE 3.2 – Logo de Kaggle

3.2.3 Google Colab :

Google Colab est une plateforme gratuite basée sur le cloud qui permet d'écrire et d'exécuter du code Python directement dans un navigateur. Elle offre un environnement interactif, avec un accès facile aux bibliothèques populaires telles que TensorFlow, Keras, et Scikit-learn, ce qui en fait un outil précieux pour le développement et l'expérimentation en data science et machine learning. Colab permet également d'utiliser des GPU et TPU gratuitement, ce qui est un avantage important pour l'entraînement de modèles lourds. En plus de sa facilité d'utilisation et de sa compatibilité avec GitHub, Google Colab permet de partager facilement des notebooks avec d'autres utilisateurs, favorisant ainsi la collaboration et le partage de connaissances.

FIGURE 3.3 – Logo de Google Colab

3.3 Limitations et Obstacles rencontrés

Lors de l'application du modèle de clustering, plusieurs limitations ont été rencontrées. Tout d'abord, le bruit dans les données et la présence de mots polysémiques ont rendu l'analyse plus complexe. Les tweets étant souvent informels et contenant des termes ambigus, il est difficile de déterminer un seul sens pour certains mots, ce qui affecte la précision du modèle.

Ensuite, le volume des données s'est avéré un défi majeur. Au départ, nous avons tenté d'entraîner les modèles à travers un pipeline local en ajustant les hyperparamètres. Cependant, la capacité de nos ordinateurs locaux était insuffisante pour traiter l'ensemble des données, en raison de la mémoire limitée et des ressources de calcul.

Nous avons ensuite essayé de travailler sur des plateformes cloud comme Google Colab et Kaggle, qui offrent des CPU et GPU, mais ces plateformes n'ont pas donné de résultats satisfaisants en raison de la taille extrêmement volumineuse du dataset.

Finalement, après plusieurs essais infructueux, nous avons réduit la taille de notre dataset à 10 000 tweets, ce qui a permis de former un modèle plus performant et d'obtenir des résultats exploitables. Malgré ces ajustements, la gestion de grands volumes de données demeure un obstacle dans la mise en œuvre de solutions efficaces à grande échelle.

3.4 Choix du Modèle de Clustering :

Le choix du modèle de clustering est crucial pour identifier des communautés pertinentes dans les données textuelles. Plusieurs algorithmes peuvent être utilisés, chacun ayant des caractéristiques et avantages spécifiques pour les données textuelles. Voici les modèles de clustering envisagés :

3.4.1 K-means :

L'algorithme K-means vise à partitionner un ensemble de données en un nombre prédéfini de clusters **k**, en minimisant la distance entre chaque point de données et le centre (centroïde) de son cluster associé. Chaque cluster est représenté par son centroïde, et l'algorithme optimise progressivement ces centroïdes pour obtenir une meilleure homogénéité entre les points d'un même cluster.

3.4.1.1 Étapes de l'Algorithme :

1. **Initialisation des Centroïdes** : K-means commence par choisir aléatoirement kk centroïdes.
2. **Affectation des Points** : Chaque point est assigné au cluster dont le centroïde est le plus proche, selon une mesure de distance (généralement la distance euclidienne).
3. **Mise à Jour des Centroïdes** : Les centroïdes sont recalculés en prenant la moyenne des points assignés à chaque cluster.
4. **Répétition** : Les étapes d'affectation et de mise à jour sont répétées jusqu'à ce que les centroïdes n'évoluent plus significativement ou qu'un nombre d'itérations maximum soit atteint.

3.4.1.2 Sélection du Nombre Optimal de Clusters :

Le nombre de clusters k est un paramètre clé dans l'algorithme K-means. Pour déterminer ce nombre optimal, nous utilisons deux méthodes :

— **Méthode du coude** : Cette technique consiste à tracer le graphique de l'inertie (ou somme des distances au carré entre chaque point et le centroïde de son cluster) en fonction de k. Le point où la réduction de l'inertie ralentit (formant un "coude") indique le nombre optimal de clusters.
— **Indice de Silhouette** : L'indice de silhouette mesure la cohérence interne des clusters. Un score de silhouette élevé indique que les points sont bien regroupés dans leurs clusters respectifs et bien séparés des autres clusters.

3.4.1.3 Avantages et Limitations de K-means

— **Avantages :** K-means est rapide, scalable et efficace pour des volumes importants de données. Il est également facile à interpréter, ce qui permet d'analyser les thèmes sous-jacents dans chaque cluster.

— **Limitations :** L'algorithme suppose des clusters de forme sphérique, ce qui peut ne pas être optimal pour des données de forme complexe. De plus, K-means est sensible aux valeurs initiales des centroïdes, ce qui peut mener à des résultats différents selon l'initialisation.

3.4.2 DBSCAN (Density-Based Spatial Clustering of Applications with Noise) :

DBSCAN est une méthode de clustering basée sur la densité, particulièrement utile pour détecter des clusters de formes variées et identifier les points considérés comme du bruit. Contrairement à K-means, DBSCAN n'exige pas de spécifier le nombre de clusters à l'avance, ce qui le rend adapté aux données non homogènes telles que les tweets.

DBSCAN regroupe les points qui sont densément connectés entre eux en fonction de deux paramètres :

— **Epsilon (ϵ) :** Distance maximale entre deux points pour qu'ils soient considérés comme voisins.

— **MinPts** : Nombre minimal de points requis pour qu'un point soit qualifié de "point central" d'un cluster.

3.4.2.1 Étapes de l'Algorithme :

L'algorithme DBSCAN suit les étapes suivantes :

1. **Définition des Points de Base** : Les points avec au moins MinPts voisins dans un rayon de ϵ sont considérés comme des points de base.
2. **Expansion des Clusters** : Chaque point de base étend son cluster en incluant les points qui sont voisins dans un rayon de ϵ.
3. **Identification des Points Frontaliers et de Bruit** : Les points situés dans le voisinage de plusieurs clusters sont des points frontaliers. Les points isolés (sans assez de voisins) sont considérés comme du bruit et ne sont pas assignés à un cluster.

3.4.2.2 Avantages et Limitations de DBSCAN

— **Avantages :** DBSCAN est efficace pour détecter des clusters de formes variées et gère bien les valeurs aberrantes en les marquant comme du bruit. Il a l'avantage de ne pas nécessiter de spécifier le nombre de clusters à l'avance, car il les détermine automatiquement en fonction de la densité des points.

— **Limitations :** DBSCAN est sensible aux choix des paramètres ϵ (rayon de recherche) et MinPts (nombre minimal de voisins), ce qui peut affecter les résultats. Il est également moins performant pour les clusters de densités variables, où des paramètres globaux ne capturent pas bien les différentes structures des données.

3.4.2.3 Choix des Paramètres :

La sélection des valeurs pour ϵ et MinPts est cruciale pour obtenir des clusters de qualité. Voici des méthodes pour les ajuster :

- **k-distance Graphique :** En traçant la distance des k-plus proches voisins pour chaque point, il est possible d'estimer ϵ en cherchant le point d'inflexion (où la distance commence à augmenter rapidement).
- **Essais et Erreurs :** En testant différentes valeurs pour ϵ et MinPts, et en évaluant les résultats visuellement ou avec des indicateurs comme la silhouette, on peut optimiser ces paramètres.

3.4.3 Modèle de Clustering : Clustering Hiérarchique Agglomératif :

Le clustering hiérarchique agglomératif est une méthode de clustering qui construit une hiérarchie de clusters en fusionnant progressivement des données ou des sous-groupes de données en fonction de leur similarité. Contrairement aux méthodes de clustering basées sur la densité ou les partitions comme K-means et DBSCAN, le clustering agglomératif ne nécessite pas de spécifier le nombre de clusters à l'avance et produit un arbre de regroupement (ou dendrogramme) qui visualise les relations de similarité entre les données.

3.4.3.1 Principe du Clustering Hiérarchique Agglomératif :

L'algorithme suit un processus ascendant (bottom-up) :

- **Initialisation :** Chaque point est considéré comme un cluster individuel.
- **Fusion de Clusters :** Les clusters les plus proches les uns des autres sont fusionnés successivement jusqu'à ce qu'il ne reste plus qu'un seul cluster englobant tous les points.
- **Construction d'un Dendrogramme :** Le processus de fusion est représenté visuellement dans un dendrogramme, qui illustre la similarité et la distance entre les clusters.

La distance entre les clusters est mesurée de plusieurs façons, dont les principales sont :

- **Lien simple (single linkage) :** Distance minimale entre les points de deux clusters.
- **Lien complet (complete linkage) :** Distance maximale entre les points de deux clusters.
- **Lien moyen (average linkage) :** Moyenne des distances entre tous les points de deux clusters.
- **Lien du centre de gravité :** Distance entre les centroïdes des clusters.

3.4.3.2 Avantages et Limitations du Clustering Hiérarchique Agglomératif

- **Avantages :** Le clustering hiérarchique agglomératif n'exige pas de spécifier le nombre de clusters à l'avance, ce qui est utile lorsqu'on ignore la structure des données. Il permet également une visualisation des relations hiérarchiques entre les données via un dendrogramme, aidant à explorer les regroupements possibles. Cette méthode est adaptée aux ensembles de données de petite et moyenne taille.

— **Limitations :** Le clustering hiérarchique agglomératif est limité par sa complexité computationnelle élevée, surtout pour les grands ensembles de données. Il ne permet pas de réajustement une fois que les points sont fusionnés, ce qui peut mener à des clusters sous-optimaux. Enfin, il est sensible à la méthode de liaison choisie, ce qui peut affecter la formation des clusters.

3.5 Application des Modèles et Analyse :

3.5.1 Clustering par KMeans :

3.5.2 Choix du Nombre de Clusters :

Lors de l'entraînement des modèles de clustering, le choix du nombre optimal de clusters k est essentiel pour obtenir des groupes cohérents et significatifs. Pour déterminer ce nombre, nous avons utilisé la **méthode du coude**, qui est une approche visuelle et mathématique pour évaluer la qualité de la segmentation des données.

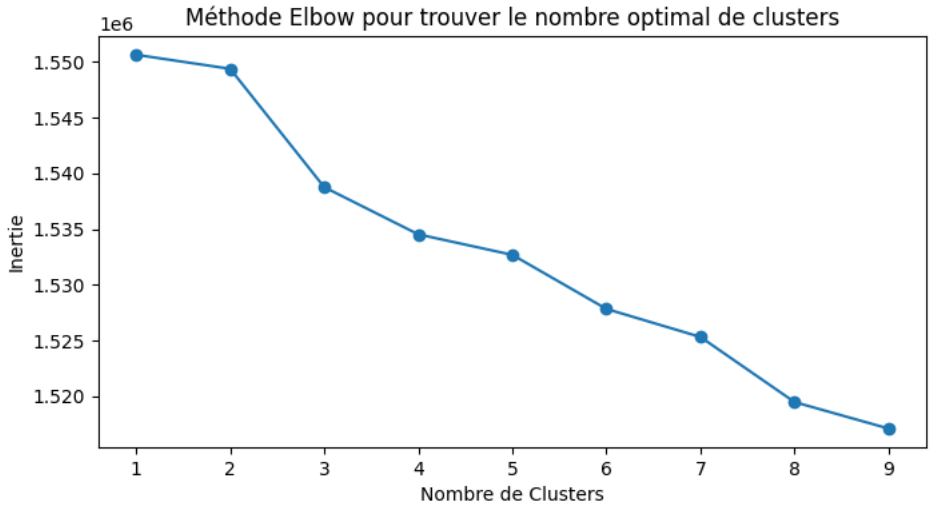

FIGURE 3.4 – Méthode Elbow pour trouver le nombre optimal de clusters

Le graphique de la méthode du coude montre la diminution de l'inertie au fur et à mesure que k augmente. Le nombre optimal de clusters est situé au point où la courbe forme un "coude", c'est-à-dire le point où la réduction de l'inertie commence à ralentir. Dans notre cas, la Figure 3.4 montre que le point de coude se situe autour de $k = 4$, ce qui justifie le choix de $k = 4$ pour le modèle KMeans.

3.5.2.1 Visualisation UMAP :

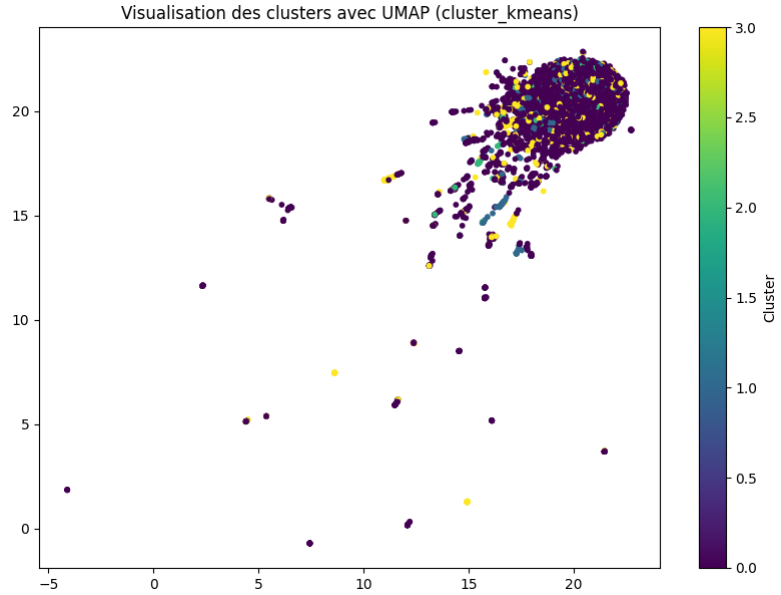

FIGURE 3.5 – Visualisation des clusters KMeans avec UMAP

Bien que le nombre de clusters soit fixé à 4, la Figure 3.5 montre que les clusters ne sont pas clairement séparés. On observe un chevauchement significatif entre les clusters, ce qui peut indiquer que KMeans a du mal à bien séparer des clusters lorsqu'ils ne sont pas de forme sphérique.

3.5.2.2 Graphe de Silhouette :

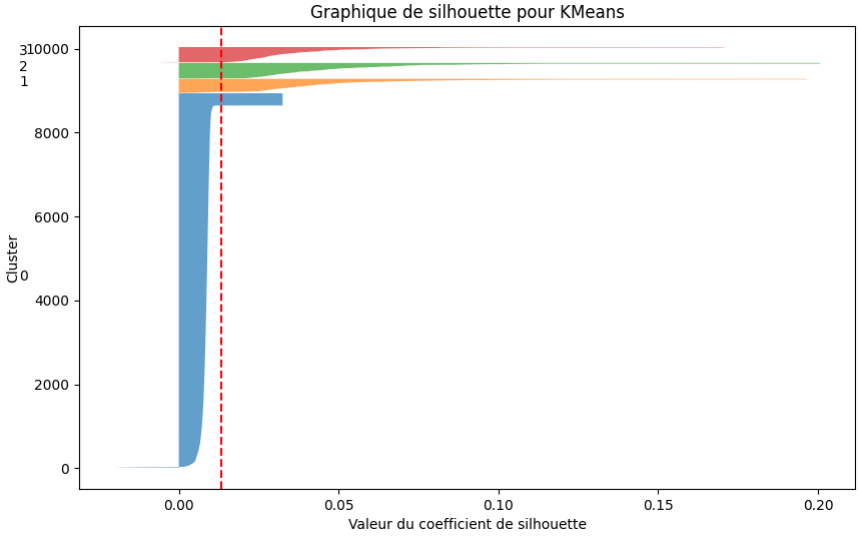

FIGURE 3.6 – Graphe de silhouette pour le modèle KMeans

La Figure 3.6 révèle que la majorité des points ont un score de silhouette proche de 0, voire négatif. Cela confirme que les clusters formés par KMeans ont une cohésion faible et que certains points se situent aux frontières des clusters.

3.5.3 Clustering par DBSCAN :

3.5.3.1 Visualisation UMAP :

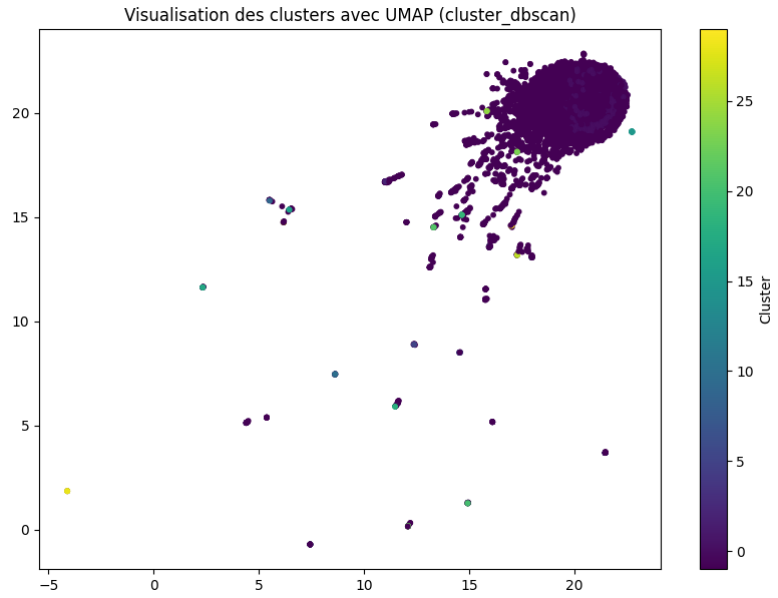

FIGURE 3.7 – Visualisation des clusters DBSCAN avec UMAP

La Figure 3.7 montre des clusters bien définis et des points isolés marqués comme du bruit. Cela montre que DBSCAN est efficace pour identifier des clusters de formes variées et gérer les outliers.

3.5.3.2 Graphe de Silhouette :

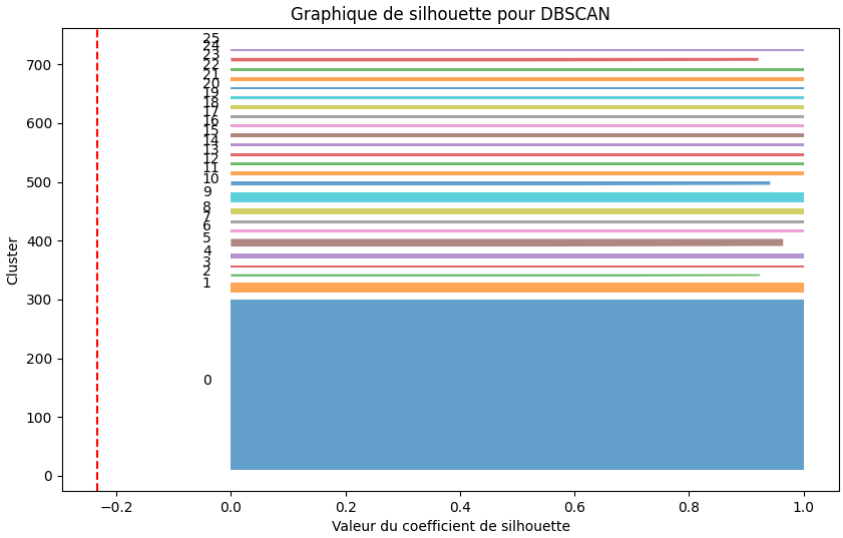

FIGURE 3.8 – Graphe de silhouette pour le modèle DBSCAN

La Figure 3.8 montre des scores de silhouette variés, allant de négatifs à positifs. Cela indique que DBSCAN, bien qu'efficace pour détecter des clusters non linéaires, présente une qualité inégale des clusters.

3.5.4 Clustering par Agglomerative Clustering :

3.5.4.1 Visualisation UMAP :

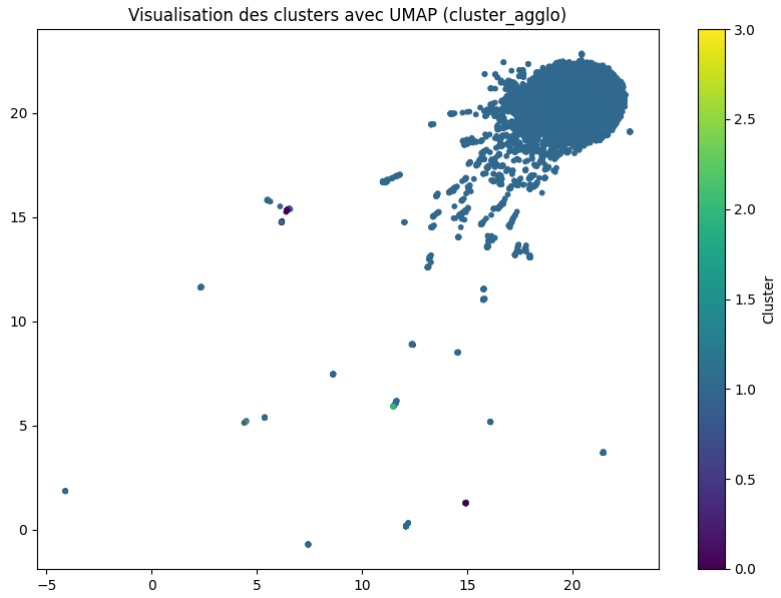

FIGURE 3.9 – Visualisation des clusters Agglomerative Clustering avec UMAP

La Figure 3.9 montre des clusters qui sont relativement distincts, mais pas aussi bien séparés que ceux de DBSCAN. L'Agglomerative Clustering, bien qu'il respecte le nombre de clusters fixé à 4, montre des chevauchements.

3.5.4.2 Graphe de Silhouette :

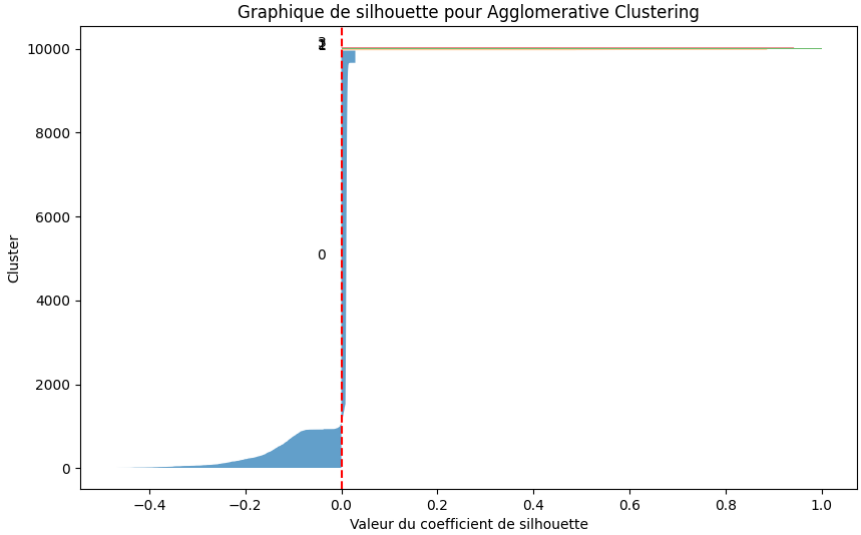

FIGURE 3.10 – Graphe de silhouette pour le modèle Agglomerative Clustering

La Figure 3.10 montre des scores de silhouette bas, similaires à ceux de KMeans, indiquant une faible cohésion intra-cluster et une séparation imparfaite des clusters.

3.5.5 Comparaison et Analyse des Résultats :

3.5.5.1 Matrice de Confusion entre KMeans et Agglomerative Clustering :

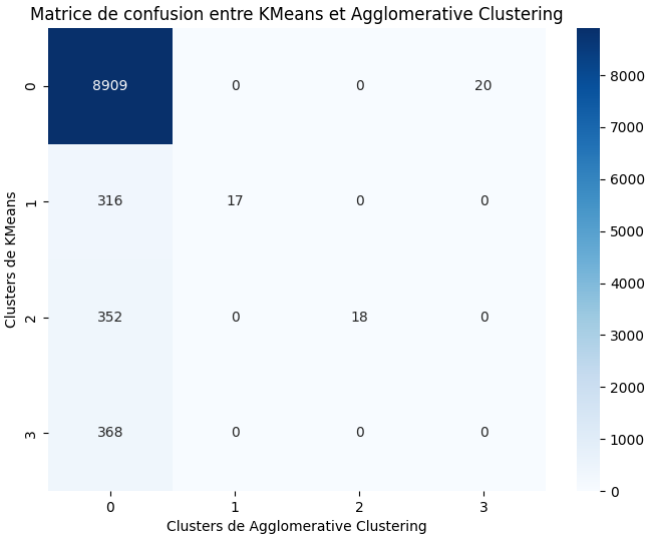

FIGURE 3.11 – Matrice de confusion entre KMeans et Agglomerative Clustering

La Figure 3.11 montre que KMeans et Agglomerative Clustering partagent des similarités dans l'assignation des points, mais des différences notables subsistent, ce qui indique que les deux modèles segmentent les données différemment.

3.5.5.2 Matrice de Confusion entre KMeans et DBSCAN :

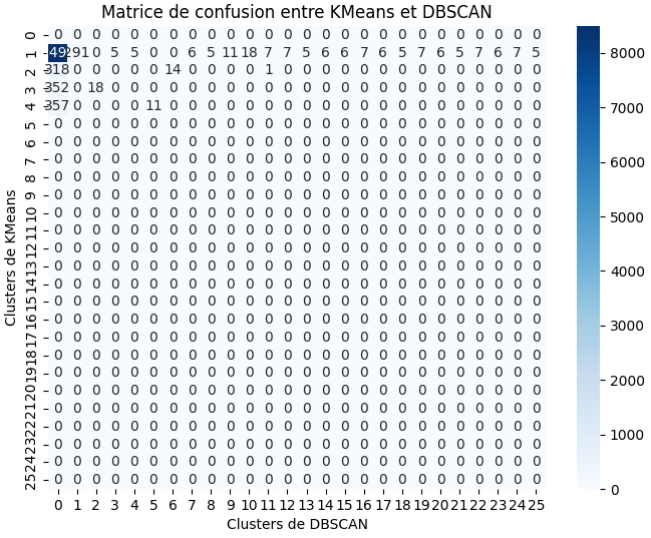

FIGURE 3.12 – Matrice de confusion entre KMeans et DBSCAN

La Figure 3.12 montre que DBSCAN identifie de nombreux points comme du bruit, ce qui le distingue de KMeans. Cela illustre la capacité de DBSCAN à ne pas forcer l'assignation des points à des clusters lorsqu'ils ne répondent pas aux critères de densité.

3.5.5.3 Matrice de Confusion entre Agglomerative Clustering et DBSCAN :

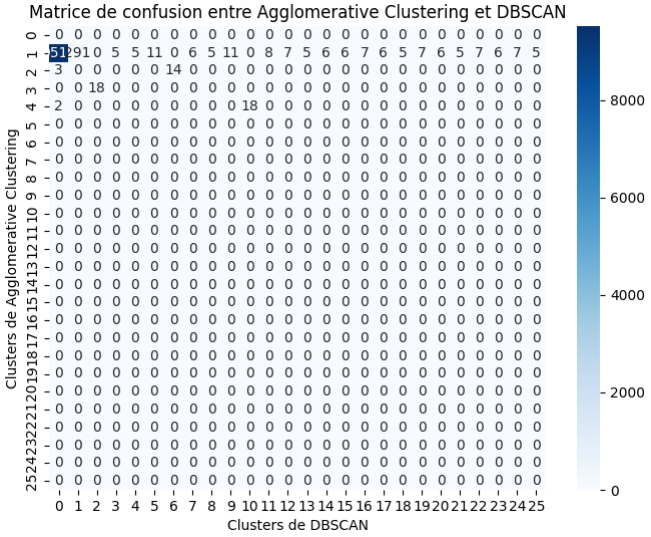

FIGURE 3.13 – Matrice de confusion entre Agglomerative Clustering et DBSCAN

La Figure 3.13 montre que l'Agglomerative Clustering et DBSCAN diffèrent considérablement dans la répartition des points, DBSCAN identifiant beaucoup de points comme du bruit.

3.6 Analyse des Résultats :

Les visualisations montrent des différences significatives entre les modèles :

— **KMeans** : Les clusters sont peu distincts, avec des chevauchements importants et un score de silhouette bas, indiquant une séparation imparfaite des clusters.
— **DBSCAN** : Se distingue par sa capacité à identifier des clusters de formes variées et à gérer les outliers, bien que la qualité des clusters soit inégale, comme le montrent les scores de silhouette.
— **Agglomerative Clustering** : Bien qu'il respecte la contrainte de 4 clusters, il présente des chevauchements similaires à KMeans et une faible cohésion intra-cluster.

3.7 Conclusion

Parmi les modèles analysés, **DBSCAN** est le plus performant pour gérer des structures de clusters variées et identifier les outliers. Bien que les scores de silhouette varient, il offre une bonne séparation des clusters, contrairement à **KMeans** et **Agglomerative Clustering**, qui montrent des performances similaires mais des limites dans la séparation des clusters.

Conclusion Générale

Ce projet a permis de mettre en lumière l'importance de l'analyse des données issues des réseaux sociaux, en particulier sur Twitter, pour détecter des communautés d'utilisateurs ayant des intérêts ou des comportements similaires. L'utilisation de techniques de text mining et de clustering a permis de nettoyer et de transformer les tweets, facilitant ainsi l'identification de ces communautés thématiques. Après un processus de prétraitement rigoureux, incluant la suppression des éléments indésirables, la personnalisation des stopwords, et la lemmatisation des mots, des algorithmes de clustering tels que KMeans, DBSCAN, et Agglomerative Clustering ont été appliqués. Parmi ces modèles, DBSCAN s'est révélé être le plus adapté pour gérer des structures de clusters variées, notamment en identifiant les outliers.

Les résultats obtenus montrent que l'application du clustering permet de mieux comprendre les dynamiques sociales sur Twitter et d'extraire des informations pertinentes des discussions en ligne. Cependant, des défis demeurent, notamment concernant le traitement de grandes quantités de données et l'amélioration continue des modèles pour une meilleure séparation des clusters.

Les perspectives de ce projet offrent plusieurs pistes d'amélioration et d'extension. Tout d'abord, l'intégration de modèles de clustering plus avancés, comme le deep clustering ou le clustering basé sur des graphes, pourrait améliorer la précision de l'identification des communautés. En outre, l'enrichissement du prétraitement des données avec des techniques d'analyse des émotions ou des sentiments pourrait également permettre de détecter des communautés en fonction de la tonalité des discussions. Enfin, pour traiter des volumes de données plus importants, l'optimisation des modèles et l'utilisation de solutions distribuées ou de plateformes cloud seraient nécessaires pour garantir la scalabilité du projet.

Bibliographie et Webographie

— A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*, Prentice Hall, 1988.
— B. Liu, *Sentiment Analysis and Opinion Mining*, Morgan & Claypool Publishers, 2012.
— P. N. Tan, M. Steinbach, and V. Kumar, *Introduction to Data Mining*, Pearson Addison-Wesley, 2005.
— Scikit-learn Documentation, *Scikit-learn : Machine Learning in Python*, 2024. Retrieved from https://scikit-learn.org/stable/.
— S. Bird, E. Loper, and E. Klein, *Natural Language Processing with Python*, O'Reilly Media, 2009.
— `NLTK Documentation`, *Natural Language Toolkit (NLTK)*, 2024. Retrieved from https://www.nltk.org/.
— `Kaggle`, *Kaggle : Your Home for Data Science*, 2024. Retrieved from https://www.kaggle.com/.
— `Google Colab`, *Google Colab : Free Cloud-based Python Notebooks*, 2024. Retrieved from https://colab.research.google.com/.